Each header block may be designed to fit within a single data packet.

[0496] After header generation is complete (e.g., using simplified header generation process 4000), the secure data parser may enter the data partitioning phase using simplified data splitting process 4010. Each incoming data packet or data block in the stream is encrypted using the split encryption key, K, at step 4012. At step 4014, share integrity information (e.g., a hash H) may be computed on the resulting ciphertext from step 4012. For example, a SHA-256 hash may be computed. At step 4106, the data packet or data block may then be partitioned into two or more data shares using one of the data splitting algorithms described above in accordance with the present invention. In some embodiments, the data packet or data block may be split so that each data share contains a substantially random distribution of the encrypted data packet or data block. The integrity information (e.g., hash H) may then be appended to each data share. An optional post-authentication tag (e.g., MAC) may also be computed and appended to each data share in some embodiments.

[0497] Each data share may include metadata, which may be necessary to permit correct reconstruction of the data blocks or data packets. This information may be included in the share header. The metadata may include such information as cryptographic key shares, key identities, share nonces, signatures/MAC values, and integrity blocks. In order to maximize bandwidth efficiency, the metadata may be stored in a compact binary format.

IPR2025-01200
IBM v. Security First Innovations, LLC
IBM's Exhibit 1010, Vol. 4
EX1010, Page 2225

[0498] For example, in some embodiments, the share header includes a cleartext header chunk, which is not encrypted and may include such elements as the Shamir key share, per-session nonce, per-share nonce, key identifiers (e.g., a workgroup key identifier and a post-authentication key identifier). The share header may also include an encrypted header chunk, which is encrypted with the split encryption key. An integrity header chunk, which may include integrity checks for any number of the previous blocks (e.g., the previous two blocks) may also be included in the header. Any other suitable values or information may also be included in the share header.

[0499] As shown in illustrative share format 4100 of FIGURE 41, header block 4102 may be associated with two or more output blocks 4104. Each header block, such as header block 4102, may be designed to fit within a single network data packet. In some embodiments, after header block 4102 is transmitted from a first location to a second location, the output blocks may then be transmitted. Alternatively, header block 4102 and output blocks 4104 may be transmitted at the same time in parallel. The transmission may occur over one or more similar or dissimilar communications paths.

[0500] Each output block may include data portion 4106 and integrity/authenticity portion 4108. As described above, each data share may be secured using a share integrity portion including share integrity information (e.g., a SHA-256 hash) of the encrypted, pre-partitioned data. To verify the integrity of the outputs blocks at recovery time, the secure data parser may compare the share integrity blocks of each share and then invert the split algorithm. The hash of the

recovered data may then be verified against the share hash.

[0501] Although some common applications of the secure data parser are described above, it should be clearly understood that the present invention may be integrated with any network application in order to increase security, fault-tolerance, anonymity, or any suitable combination of the foregoing.

[0502] Additionally, other combinations, additions, substitutions and modifications will be apparent to the skilled artisan in view of the disclosure herein. Accordingly, the present invention is not intended to be limited by the reaction of the preferred embodiments but is to be defined by a reference to the appended claims.

What is Claimed is:

1. A method for securing a transmission of data blocks in a data stream, the method comprising:

encrypting each data block in the data stream with an encryption key;

distributing portions of the encryption key into at least two share headers;

distributing data units from the encrypted data blocks into at least two data shares, wherein each of the at least two data shares respectfully contains a substantially random distribution of a respective subset of the data units; and

transmitting the at least two data shares and the at least two share headers to a remote location over at least one communications path, whereby the data stream is restorable from at least two data shares of the at least two data shares and from at least two share headers of the at least two share headers.

2. The method of claim 1 further comprising:

generating integrity information for each encrypted data block in the data stream; and

transmitting the integrity information to the remote location.

3. The method of claim 2 wherein the integrity information is transmitted with the at least two data shares.

EX1010, Page 2228

4. The method of claim 2 wherein the integrity information is selected from the group consisting of a hash, a MAC signature, and a digital signature.

5. The method of claim 1 wherein distributing portions of the encryption key comprises distributing portions of the encryption key using a secret sharing algorithm.

6. The method of claim 5 wherein the secret sharing algorithm is selected from the group consisting of Shamir and Blakely.

7. The method of claim 1 wherein transmitting the at least two data shares and the at least two share headers comprises transmitting the at least two data shares and the at least two share headers on a single communications path in a serial transmission.

8. The method of claim 1 wherein transmitting the at least two data shares and the at least two share headers comprises transmitting the at least two data shares and the at least two share headers over multiple communications paths in parallel.

9. The method of claim 1 wherein the transmitting is performed using a file-sharing application.

10. The method of claim 1 wherein the transmitting is performed using a voting or polling application.

11. The method of claim 1 wherein the transmitting is performed using a voice over IP (VoIP) application.

12. The method of claim 1 wherein the data blocks are selected from the group consisting of network data packets, network voice packets, and file system data blocks.

13. The method of claim 1 further comprising:

appending redundancy information to the at least two data shares; and

transmitting the redundancy information to the remote location.

14. The method of claim 1 further comprising encrypting the encryption key with a workgroup key before distributing the encryption key.

15. The method of claim 14 wherein encrypting the encryption key with the workgroup key comprises encrypting the encryption key using an AES key wrap function.

EX1010, Page 2230

Figure 1

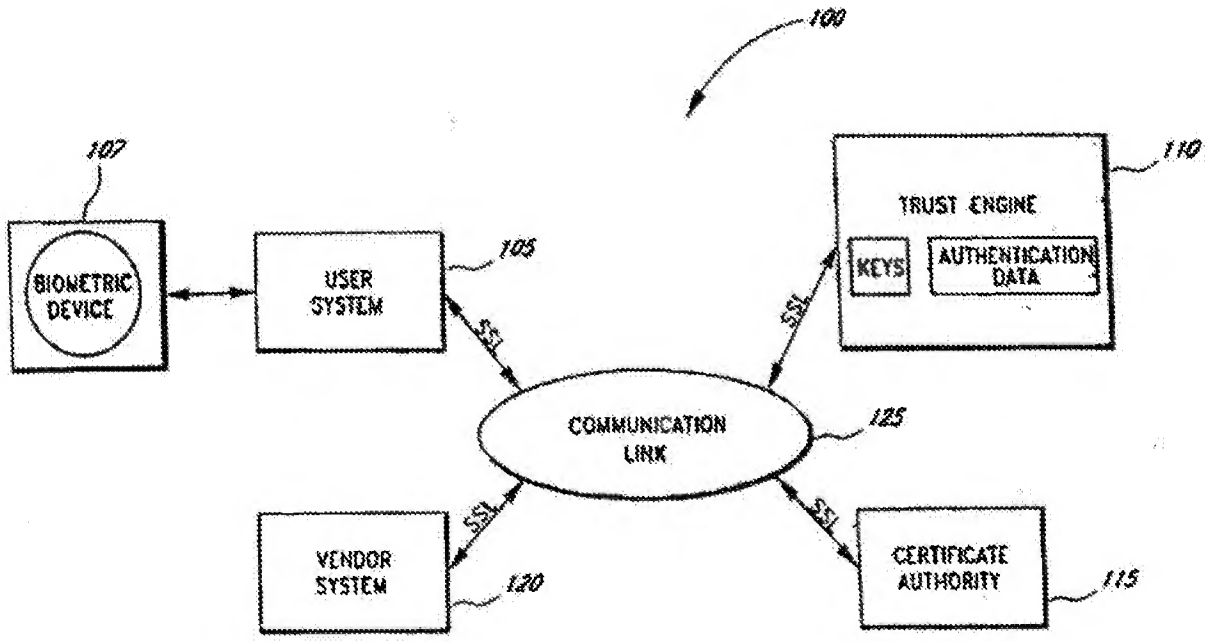

100
107
BIOMETRIC DEVICE
USER SYSTEM
105
SSL
TRUST ENGINE
KEYS
AUTHENTICATION DATA
110
SSL
COMMUNICATION LINK
125
SSL
SSL
VENDOR SYSTEM
120
CERTIFICATE AUTHORITY
115

Figure 2

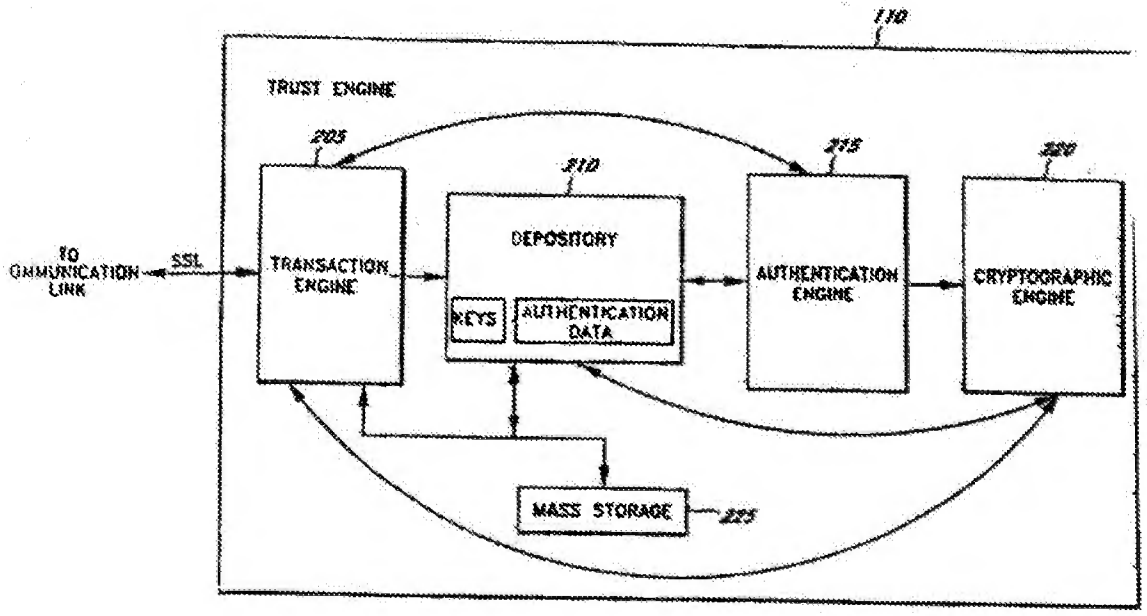

110
TRUST ENGINE
205
210
215
220
TO COMMUNICATION LINK
SSL
TRANSACTION ENGINE
DEPOSITORY
KEYS
AUTHENTICATION DATA
AUTHENTICATION ENGINE
CRYPTOGRAPHIC ENGINE
MASS STORAGE
225

Figure 3

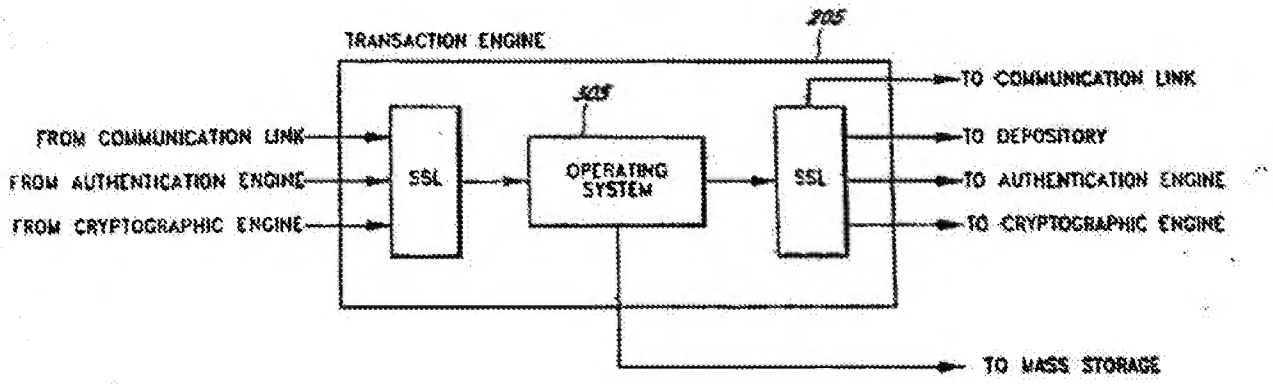

TRANSACTION ENGINE
205
FROM COMMUNICATION LINK
FROM AUTHENTICATION ENGINE
FROM CRYPTOGRAPHIC ENGINE
SSL
OPERATING SYSTEM
SSL
TO COMMUNICATION LINK
TO DEPOSITORY
TO AUTHENTICATION ENGINE
TO CRYPTOGRAPHIC ENGINE
TO MASS STORAGE

Figure 4

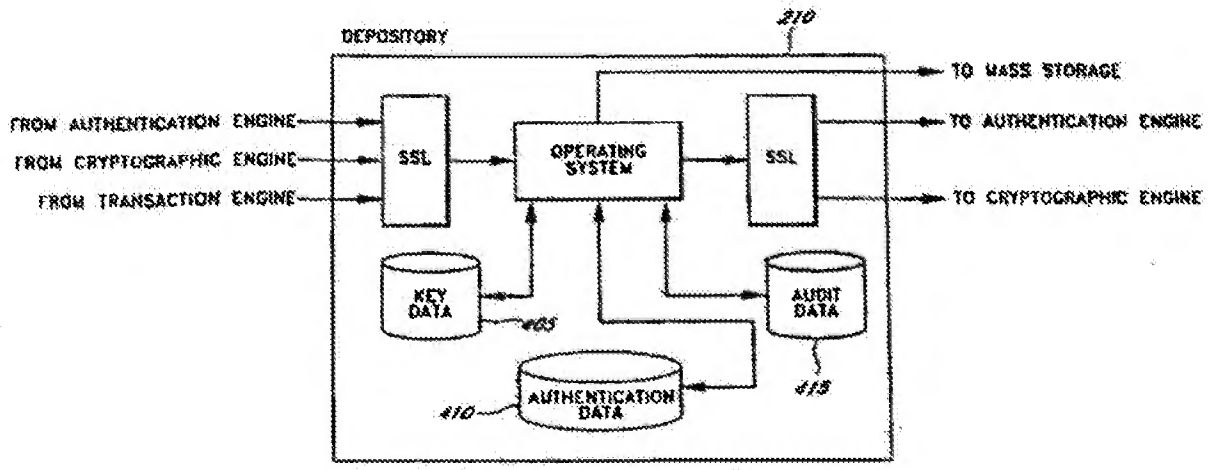

DEPOSITORY
210
FROM AUTHENTICATION ENGINE
FROM CRYPTOGRAPHIC ENGINE
FROM TRANSACTION ENGINE
SSL
OPERATING SYSTEM
SSL
TO MASS STORAGE
TO AUTHENTICATION ENGINE
TO CRYPTOGRAPHIC ENGINE
KEY DATA
405
AUDIT DATA
415
AUTHENTICATION DATA
410

Figure 5

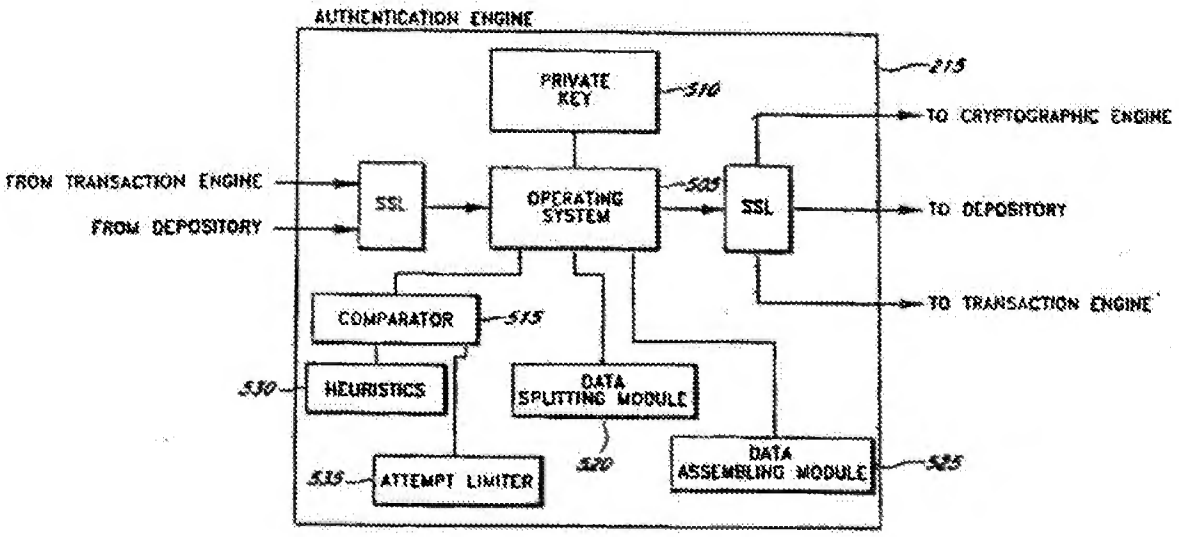

AUTHENTICATION ENGINE
PRIVATE KEY
510
215
TO CRYPTOGRAPHIC ENGINE
FROM TRANSACTION ENGINE
FROM DEPOSITORY
SSL
OPERATING SYSTEM
505
SSL
TO DEPOSITORY
TO TRANSACTION ENGINE
COMPARATOR
515
530
HEURISTICS
DATA SPLITTING MODULE
520
DATA ASSEMBLING MODULE
525
535
ATTEMPT LIMITER

Figure 6

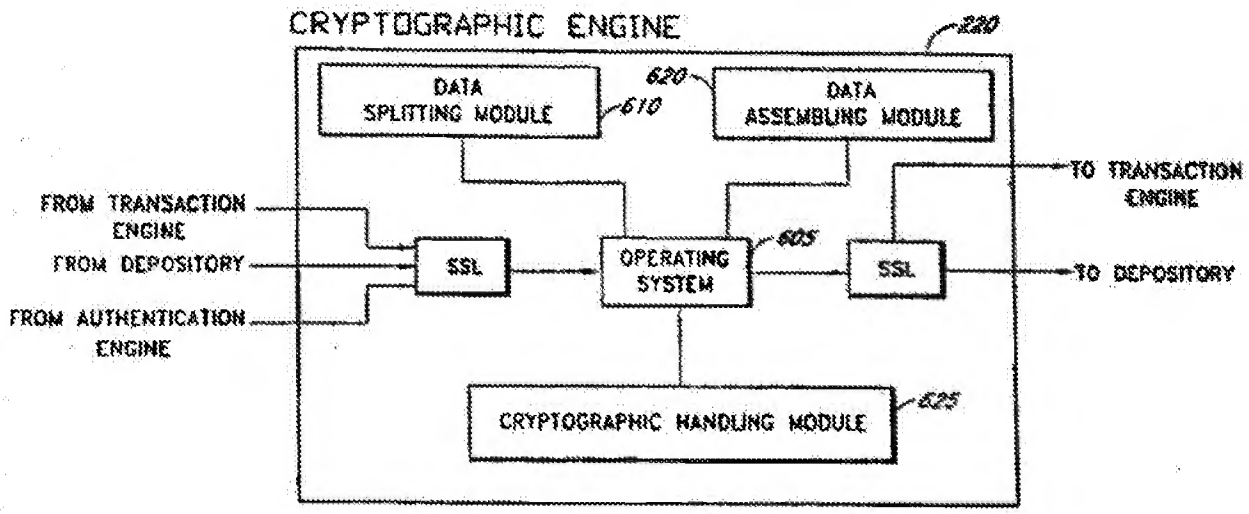

CRYPTOGRAPHIC ENGINE
620
DATA SPLITTING MODULE
610
620
DATA ASSEMBLING MODULE
TO TRANSACTION ENGINE
FROM TRANSACTION ENGINE
FROM DEPOSITORY
FROM AUTHENTICATION ENGINE
SSL
OPERATING SYSTEM
605
SSL
TO DEPOSITORY
CRYPTOGRAPHIC HANDLING MODULE
625

Figure 7

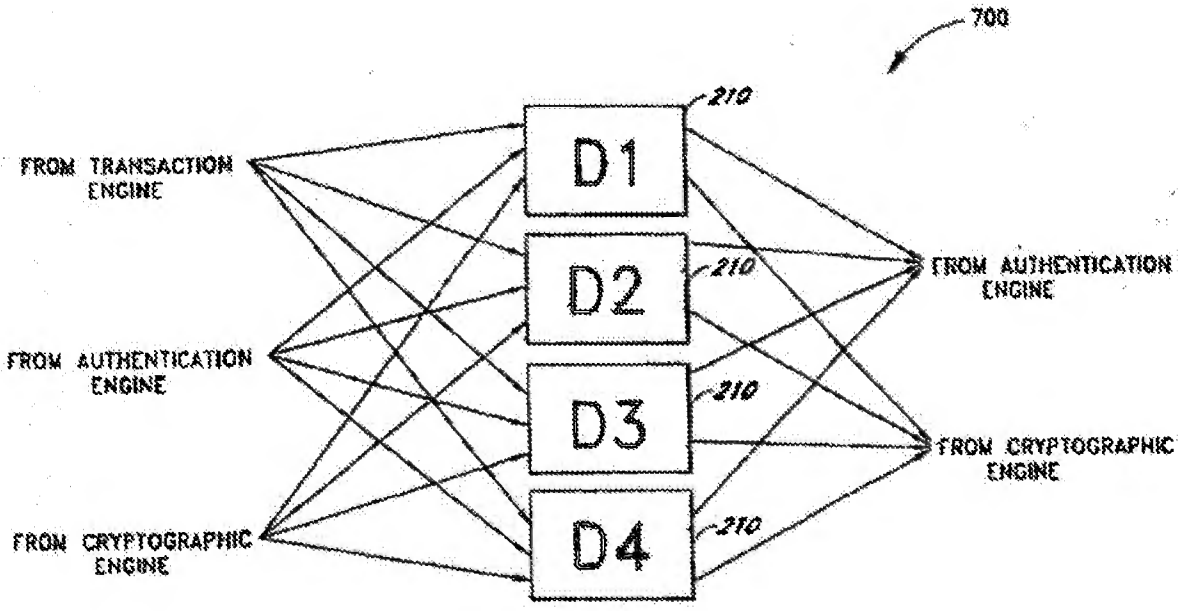

700
210
D1
D2
D3
D4
210
210
210
FROM TRANSACTION ENGINE
FROM AUTHENTICATION ENGINE
FROM CRYPTOGRAPHIC ENGINE
FROM AUTHENTICATION ENGINE
FROM CRYPTOGRAPHIC ENGINE

Figure 8

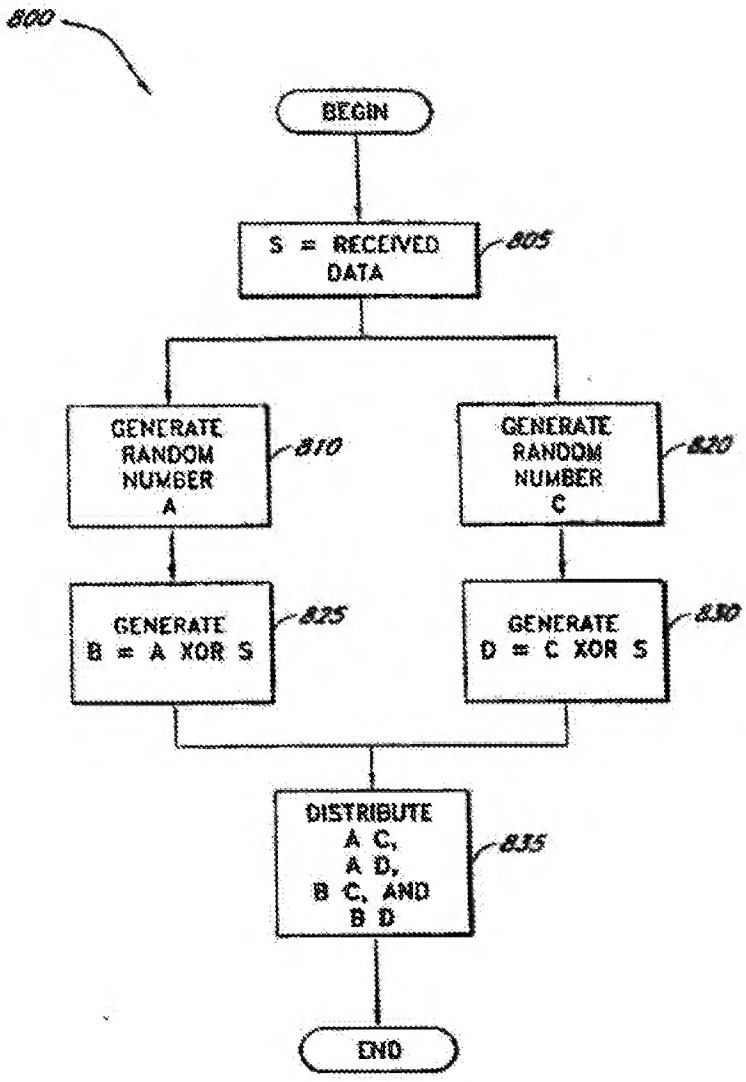

800
BEGIN
S = RECEIVED DATA
805
GENERATE RANDOM NUMBER A
810
GENERATE RANDOM NUMBER C
820
GENERATE B = A XOR S
825
GENERATE D = C XOR S
830
DISTRIBUTE A C, A D, B C, AND B D
835
END

Figure 9, Panel A



ENROLLMENT DATA FLOW			
SEND	RECEIVE	SSL	ACTION
905 USER	TRANSACTION ENGINE(TE)	1/2	TRANSMIT ENROLLMENT AUTHENTICATION DATA (B) AND THE USER ID (UID) ENCRYPTED WITH THE PUBLIC KEY OF THE AUTHENTICATION ENGINE (AE) AS (PUB_AE(UID,B))
915 TE	AE	FULL	FORWARD TRANSMISSION
920			AE DECRYPTS AND SPLITS FORWARDED DATA
925 AE	THE Xth DEPOSITORY (DX)	FULL	STORE RESPECTIVE PORTION OF DATA
WHEN DIGITAL CERTIFICATE REQUESTED			
930 AE	CRYPTOGRAPHIC ENGINE (CE)	FULL	REQUEST KEY GENERATION
935			CE GENERATES AND SPLITS KEY
945 CE	TE	FULL	TRANSMIT REQUEST FOR DIGITAL CERTIFICATE
950 TE	CERTIFICATION AUTHORITY (CA)	1/2	TRANSMIT REQUEST
955 CA	TE	1/2	TRANSMIT DIGITAL CERTIFICATE
960 TE	USER	1/2	TRANSMIT DIGITAL CERTIFICATE
TE	MS	FULL	STORE DIGITAL CERTIFICATE
965 CE	DX	FULL	STORE RESPECTIVE PORTION OF KEY

Figure 9, Panel B

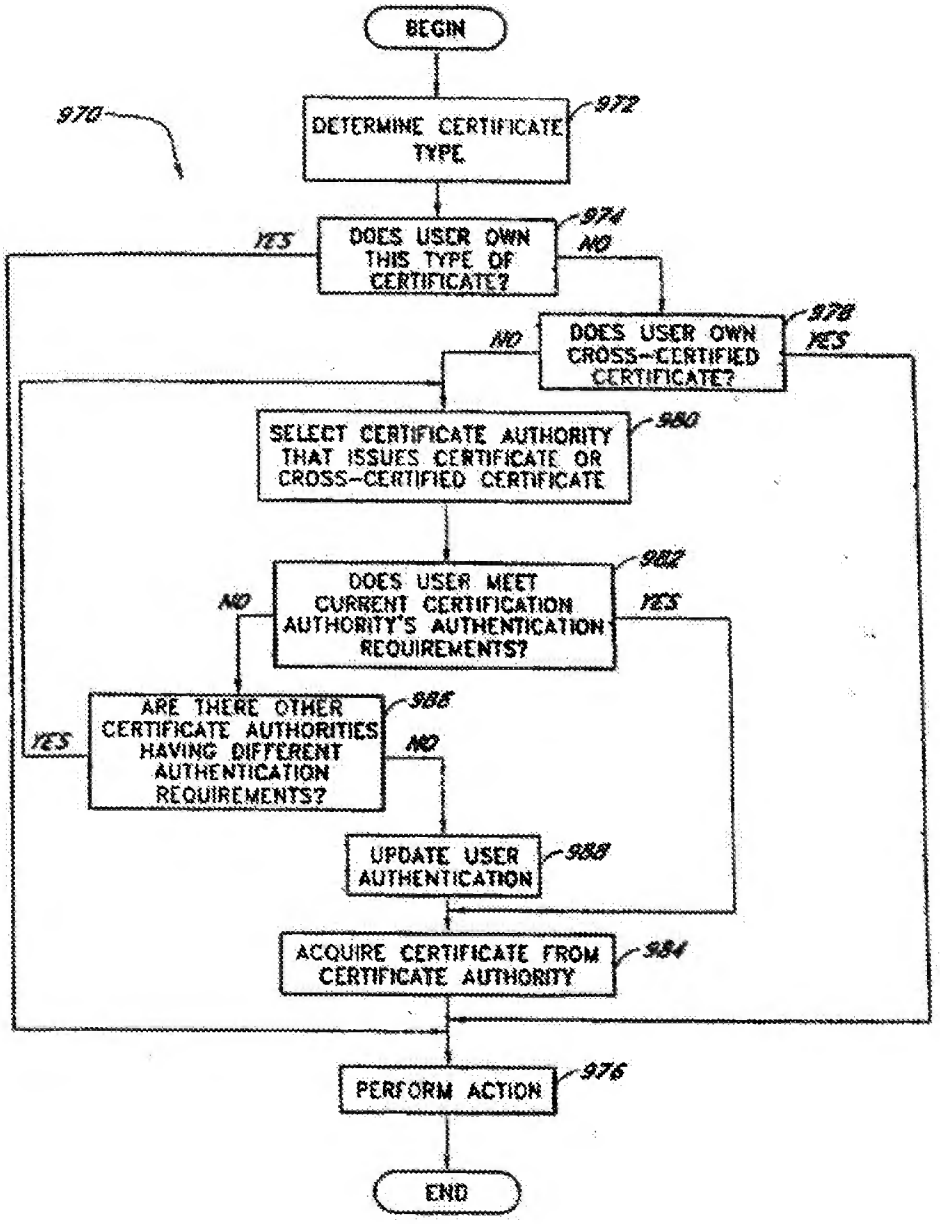


Figure 10



AUTHENTICATION DATA FLOW

	SEND	RECEIVE	SSL	ACTION
1005	USER	VENDOR	1/2	TRANSACTION OCCURS, SUCH AS SELECTING PURCHASE
1010	VENDOR	USER	1/2	TRANSMIT TRANSACTION ID (TID) AND AUTHENTICATION REQUEST (AR)
				AUTHENTICATION DATA (B') IS GATHERED FROM USER
1015	USER	TE	1/2	TRANSMIT TID AND B' WRAPPED IN THE PUBLIC KEY OF THE AUTHENTICATION ENGINE (AE), AS (PUB_AE(TID, B'))
1020	TE	AE	FULL	FORWARD TRANSMISSION
				ENROLLMENT AUTHENTICATION DATA (B) IS REQUESTED AND GATHERED
1025	VENDOR	TRANSACTION ENGINE (TE)	FULL	TRANSMITS TID, AR
1030	TE	MASS STORAGE(MS)	FULL	CREATE RECORD IN DATABASE
1035	TE	THE Xth DEPOSITORY(DX)	FULL	UID, TID
1040	DX	AE	FULL	TRANSMIT THE TID AND THE PORTION OF THE AUTHENTICATION DATA STORED AT ENROLLMENT (BX) AS (PUB_AE(TID, BX))
1045				AE ASSEMBLES B AND COMPARES TO B'
1050	AE	TE	FULL	TID, THE FILLED IN AR
	TE	VENDOR	FULL	TID, YES/NO
1055	TE	USER	1/2	TID, CONFIRMATION MESSAGE

Figure 11

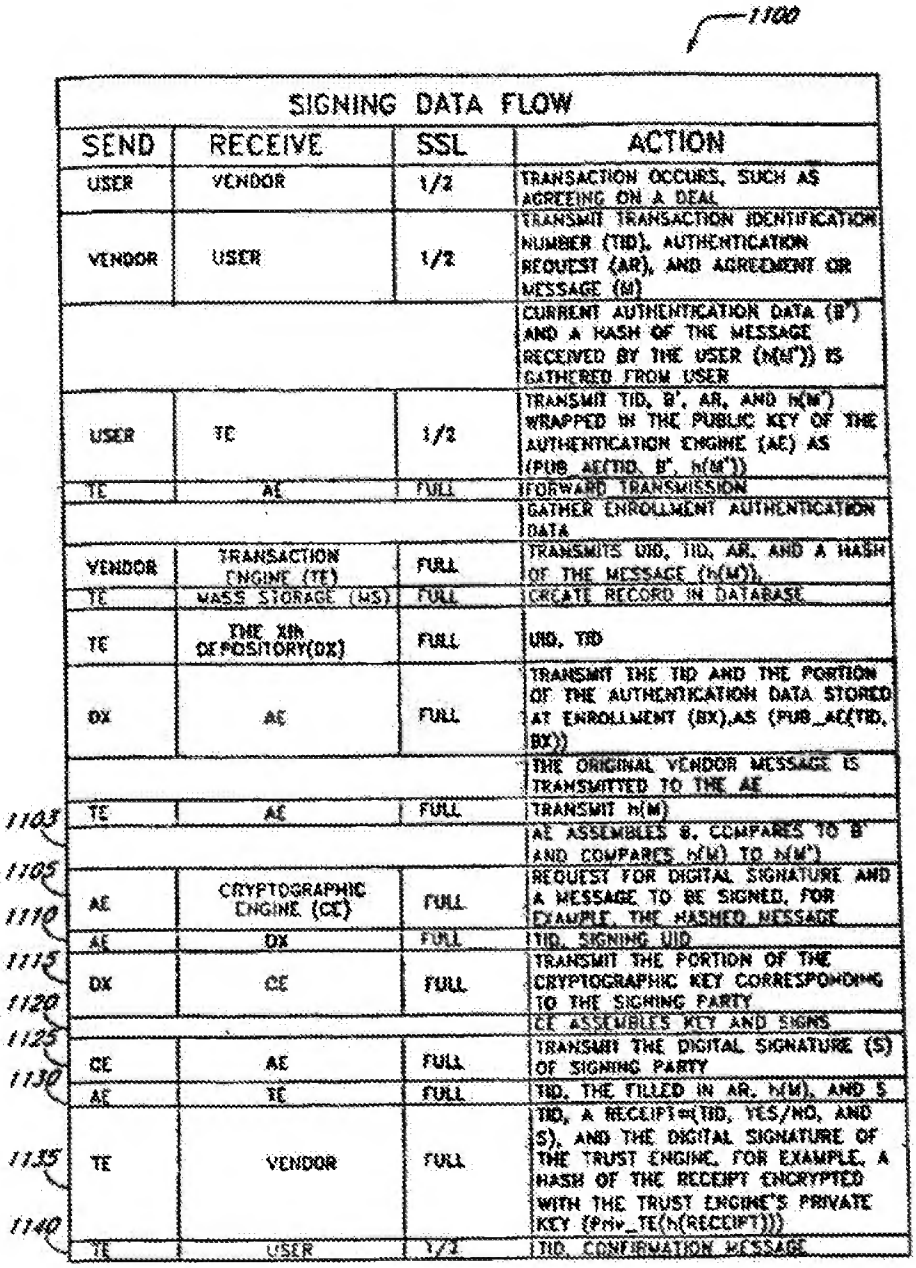

1100
SIGNING DATA FLOW
SEND
RECEIVE
SSL
ACTION
USER
VENDOR
1/2
TRANSACTION OCCURS, SUCH AS AGREEING ON A DEAL
VENDOR
USER
1/2
TRANSMIT TRANSACTION IDENTIFICATION NUMBER (TID), AUTHENTICATION REQUEST (AR), AND AGREEMENT OR MESSAGE (M)
CURRENT AUTHENTICATION DATA (B') AND A HASH OF THE MESSAGE RECEIVED BY THE USER (h(M')) IS GATHERED FROM USER
USER
TE
1/2
TRANSMIT TID, B', AR, AND h(M') WRAPPED IN THE PUBLIC KEY OF THE AUTHENTICATION ENGINE (AE) AS (PUB_AE(TID, B', h(M'))
TE
AE
FULL
FORWARD TRANSMISSION
GATHER ENROLLMENT AUTHENTICATION DATA
VENDOR
TRANSACTION ENGINE (TE)
FULL
TRANSMITS UID, TID, AR, AND A HASH OF THE MESSAGE (h(M)),
TE
MASS STORAGE (MS)
FULL
CREATE RECORD IN DATABASE
TE
THE Xth DEPOSITORY(DX)
FULL
UID, TID
DX
AE
FULL
TRANSMIT THE TID AND THE PORTION OF THE AUTHENTICATION DATA STORED AT ENROLLMENT (BX),AS (PUB_AE(TID, BX))
THE ORIGINAL VENDOR MESSAGE IS TRANSMITTED TO THE AE
1103
TE
AE
FULL
TRANSMIT h(M)
AE ASSEMBLES B, COMPARES TO B' AND COMPARES h(M) TO h(M')
1105
AE
CRYPTOGRAPHIC ENGINE (CE)
FULL
REQUEST FOR DIGITAL SIGNATURE AND A MESSAGE TO BE SIGNED, FOR EXAMPLE, THE HASHED MESSAGE
1110
AE
DX
FULL
TID, SIGNING UID
1115
DX
CE
FULL
TRANSMIT THE PORTION OF THE CRYPTOGRAPHIC KEY CORRESPONDING TO THE SIGNING PARTY
1120
CE ASSEMBLES KEY AND SIGNS
1125
CE
AE
FULL
TRANSMIT THE DIGITAL SIGNATURE (S) OF SIGNING PARTY
1130
AE
TE
FULL
TID, THE FILLED IN AR, h(M), AND S
1135
TE
VENDOR
FULL
TID, A RECEIPT=(TID, YES/NO, AND S), AND THE DIGITAL SIGNATURE OF THE TRUST ENGINE, FOR EXAMPLE, A HASH OF THE RECEIPT ENCRYPTED WITH THE TRUST ENGINE'S PRIVATE KEY (Priv_TE(h(RECEIPT)))
1140
TE
USER
1/2
TID, CONFIRMATION MESSAGE

Figure 12

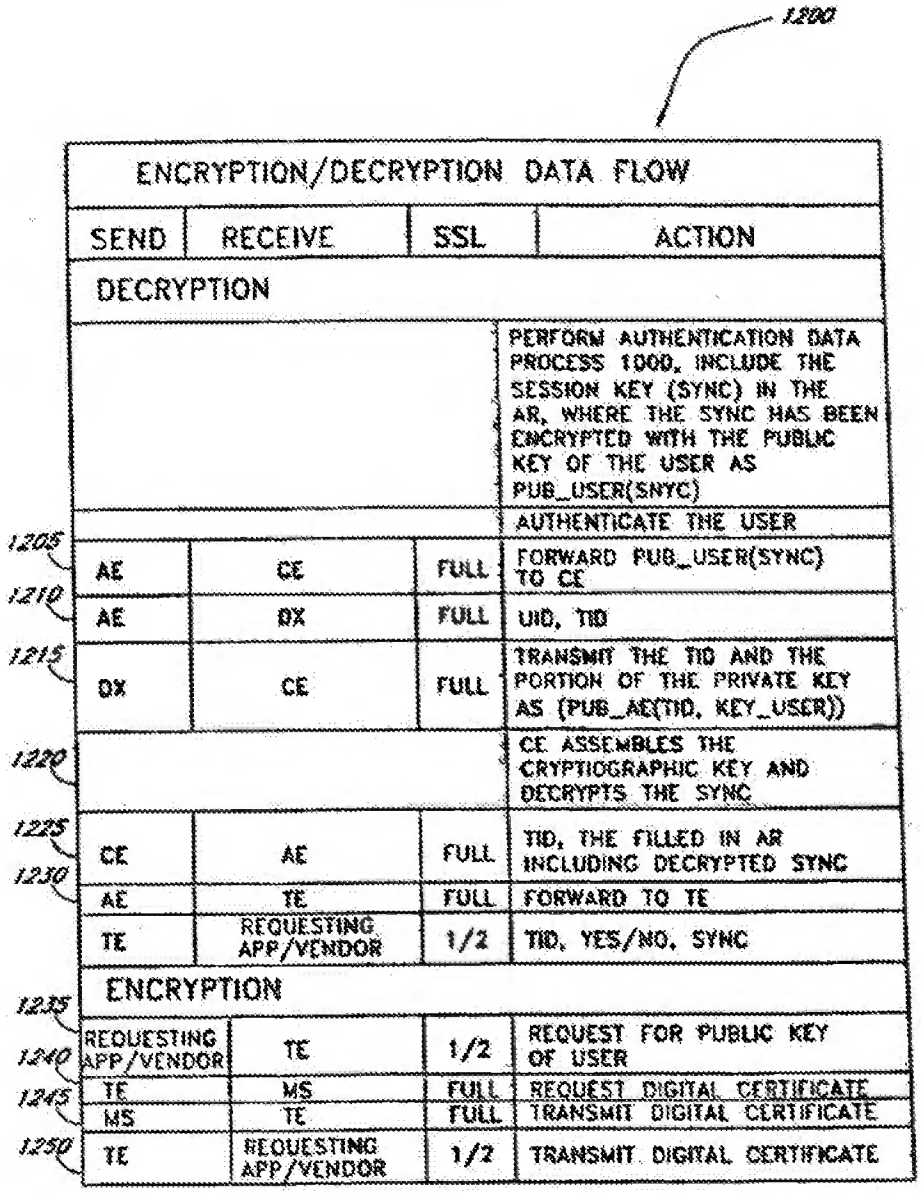

1200
ENCRYPTION/DECRYPTION DATA FLOW
SEND
RECEIVE
SSL
ACTION
DECRYPTION
PERFORM AUTHENTICATION DATA PROCESS 1000, INCLUDE THE SESSION KEY (SYNC) IN THE AR, WHERE THE SYNC HAS BEEN ENCRYPTED WITH THE PUBLIC KEY OF THE USER AS PUB_USER(SNYC)
AUTHENTICATE THE USER
1205
AE
CE
FULL
FORWARD PUB_USER(SYNC) TO CE
1210
AE
DX
FULL
UID, TID
1215
DX
CE
FULL
TRANSMIT THE TID AND THE PORTION OF THE PRIVATE KEY AS (PUB_AE(TID, KEY_USER))
1220
CE ASSEMBLES THE CRYPTIOGRAPHIC KEY AND DECRYPTS THE SYNC
1225
CE
AE
FULL
TID, THE FILLED IN AR INCLUDING DECRYPTED SYNC
1230
AE
TE
FULL
FORWARD TO TE
TE
REQUESTING APP/VENDOR
1/2
TID, YES/NO, SYNC
ENCRYPTION
1235
REQUESTING APP/VENDOR
TE
1/2
REQUEST FOR PUBLIC KEY OF USER
1240
TE
MS
FULL
REQUEST DIGITAL CERTIFICATE
1245
MS
TE
FULL
TRANSMIT DIGITAL CERTIFICATE
1250
TE
REQUESTING APP/VENDOR
1/2
TRANSMIT DIGITAL CERTIFICATE

Figure 13

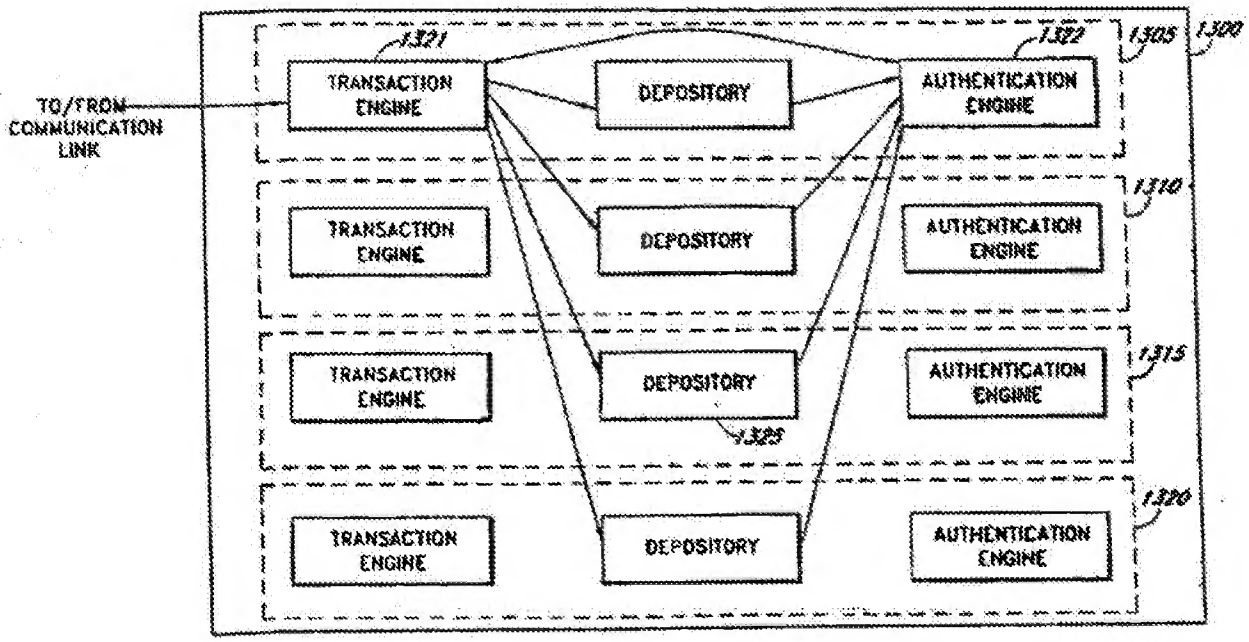

TO/FROM COMMUNICATION LINK
1321
TRANSACTION ENGINE
DEPOSITORY
1322
AUTHENTICATION ENGINE
1305
1300
1310
TRANSACTION ENGINE
DEPOSITORY
AUTHENTICATION ENGINE
1315
TRANSACTION ENGINE
DEPOSITORY
1325
AUTHENTICATION ENGINE
1320
TRANSACTION ENGINE
DEPOSITORY
AUTHENTICATION ENGINE

Figure 14

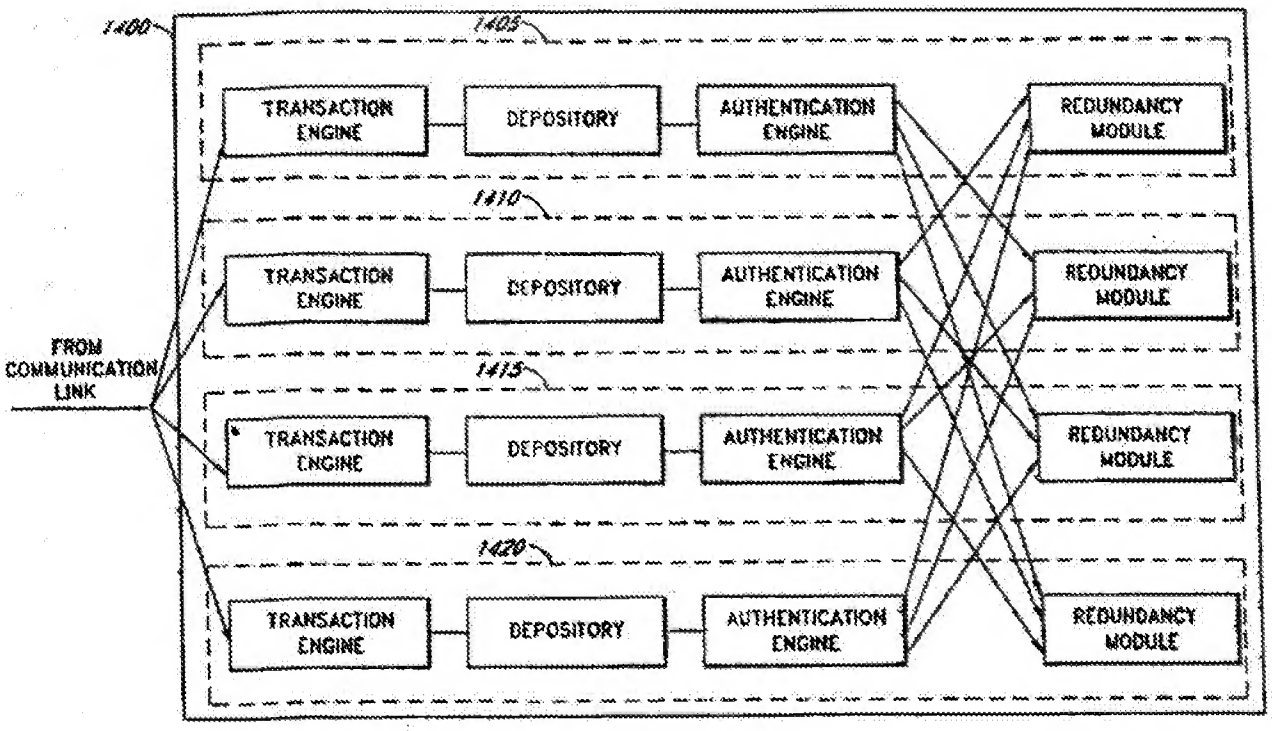

1400
1405
TRANSACTION ENGINE
DEPOSITORY
AUTHENTICATION ENGINE
REDUNDANCY MODULE
1410
TRANSACTION ENGINE
DEPOSITORY
AUTHENTICATION ENGINE
REDUNDANCY MODULE
FROM COMMUNICATION LINK
1415
TRANSACTION ENGINE
DEPOSITORY
AUTHENTICATION ENGINE
REDUNDANCY MODULE
1420
TRANSACTION ENGINE
DEPOSITORY
AUTHENTICATION ENGINE
REDUNDANCY MODULE

Figure 15

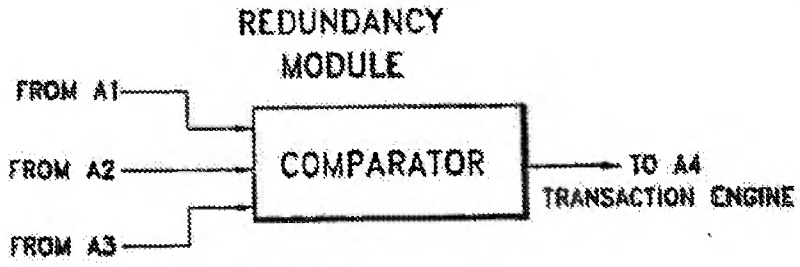

REDUNDANCY
MODULE
FROM A1
FROM A2
FROM A3
COMPARATOR
TO A4
TRANSACTION ENGINE

Figure 16

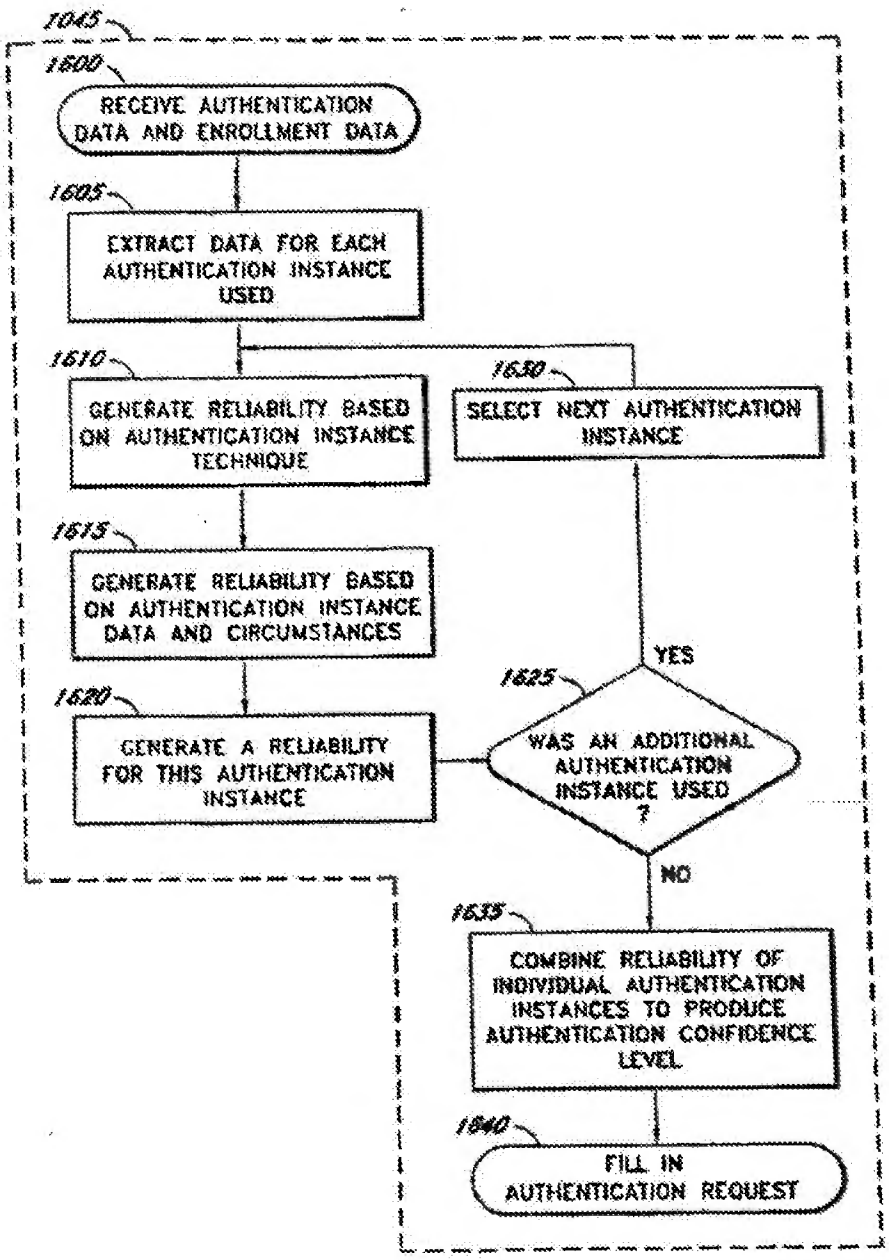

1645
1600
RECEIVE AUTHENTICATION DATA AND ENROLLMENT DATA
1605
EXTRACT DATA FOR EACH AUTHENTICATION INSTANCE USED
1610
GENERATE RELIABILITY BASED ON AUTHENTICATION INSTANCE TECHNIQUE
1630
SELECT NEXT AUTHENTICATION INSTANCE
1615
GENERATE RELIABILITY BASED ON AUTHENTICATION INSTANCE DATA AND CIRCUMSTANCES
1620
GENERATE A RELIABILITY FOR THIS AUTHENTICATION INSTANCE
1625
WAS AN ADDITIONAL AUTHENTICATION INSTANCE USED ?
YES
NO
1635
COMBINE RELIABILITY OF INDIVIDUAL AUTHENTICATION INSTANCES TO PRODUCE AUTHENTICATION CONFIDENCE LEVEL
1640
FILL IN AUTHENTICATION REQUEST

Figure 17

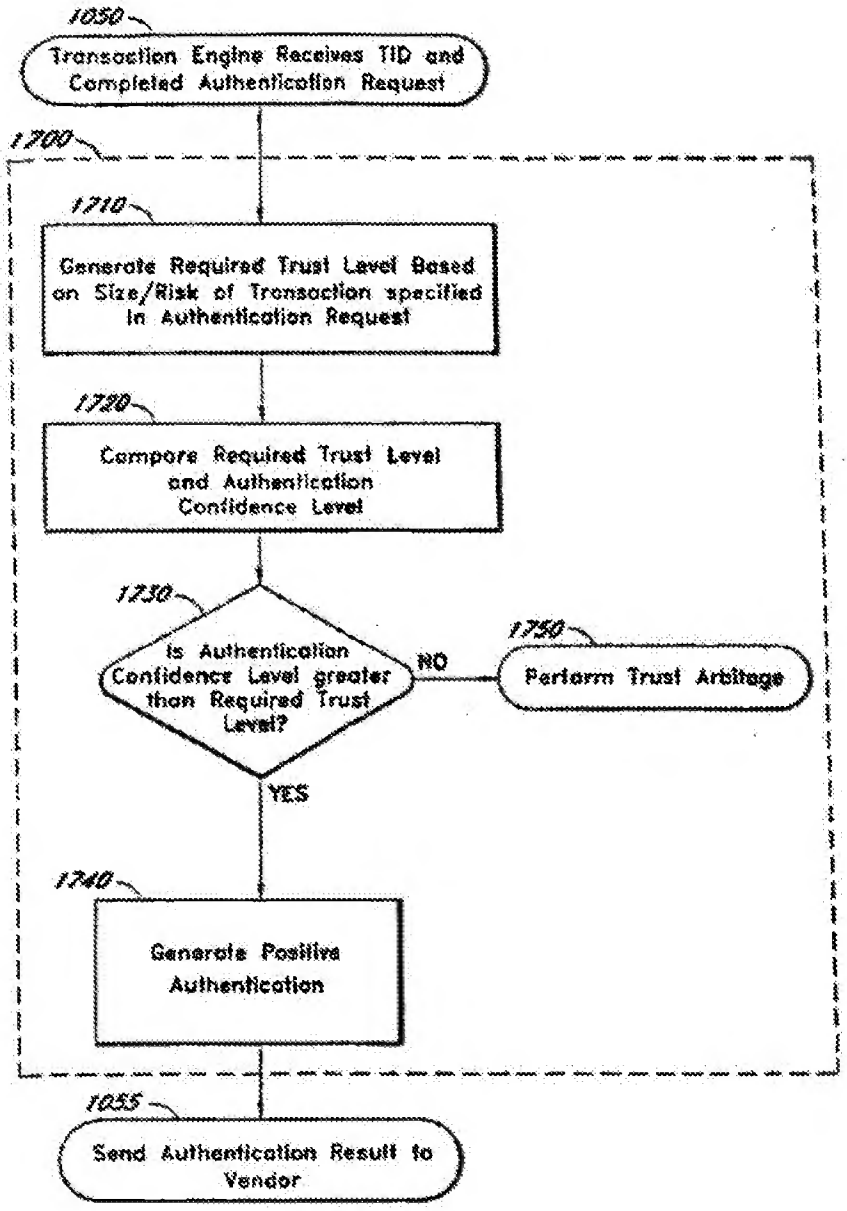

1050
Transaction Engine Receives TID and Completed Authentication Request
1700
1710
Generate Required Trust Level Based on Size/Risk of Transaction specified in Authentication Request
1720
Compare Required Trust Level and Authentication Confidence Level
1730
Is Authentication Confidence Level greater than Required Trust Level?
NO
1750
Perform Trust Arbitage
YES
1740
Generate Positive Authentication
1055
Send Authentication Result to Vendor

Figure 18

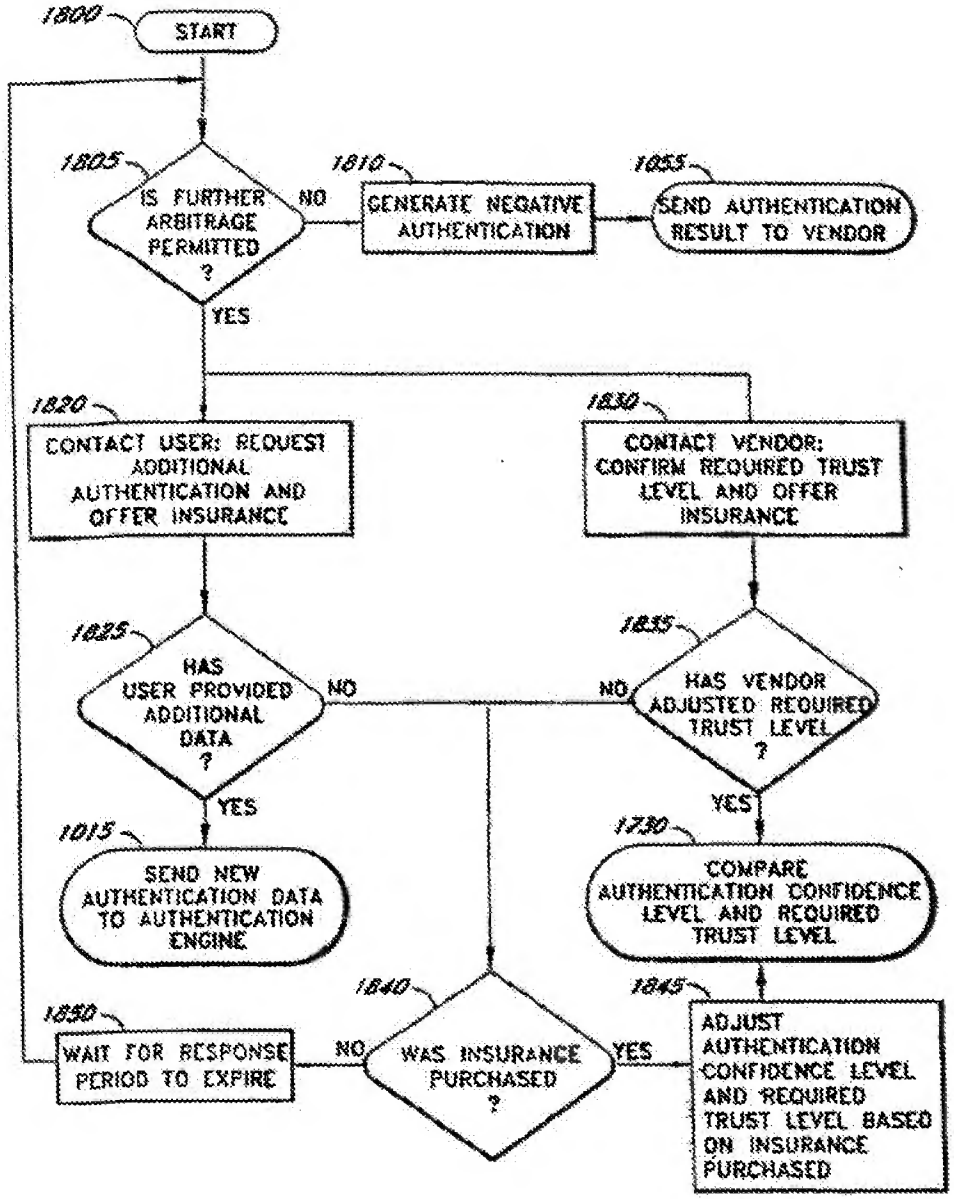

1800
START
1805
IS FURTHER ARBITRAGE PERMITTED ?
NO
1810
GENERATE NEGATIVE AUTHENTICATION
1855
SEND AUTHENTICATION RESULT TO VENDOR
YES
1820
CONTACT USER: REQUEST ADDITIONAL AUTHENTICATION AND OFFER INSURANCE
1830
CONTACT VENDOR: CONFIRM REQUIRED TRUST LEVEL AND OFFER INSURANCE
1825
HAS USER PROVIDED ADDITIONAL DATA ?
NO
1835
HAS VENDOR ADJUSTED REQUIRED TRUST LEVEL ?
NO
YES
YES
1015
SEND NEW AUTHENTICATION DATA TO AUTHENTICATION ENGINE
1730
COMPARE AUTHENTICATION CONFIDENCE LEVEL AND REQUIRED TRUST LEVEL
1840
WAS INSURANCE PURCHASED ?
NO
YES
1845
ADJUST AUTHENTICATION CONFIDENCE LEVEL AND REQUIRED TRUST LEVEL BASED ON INSURANCE PURCHASED
1850
WAIT FOR RESPONSE PERIOD TO EXPIRE

Figure 19

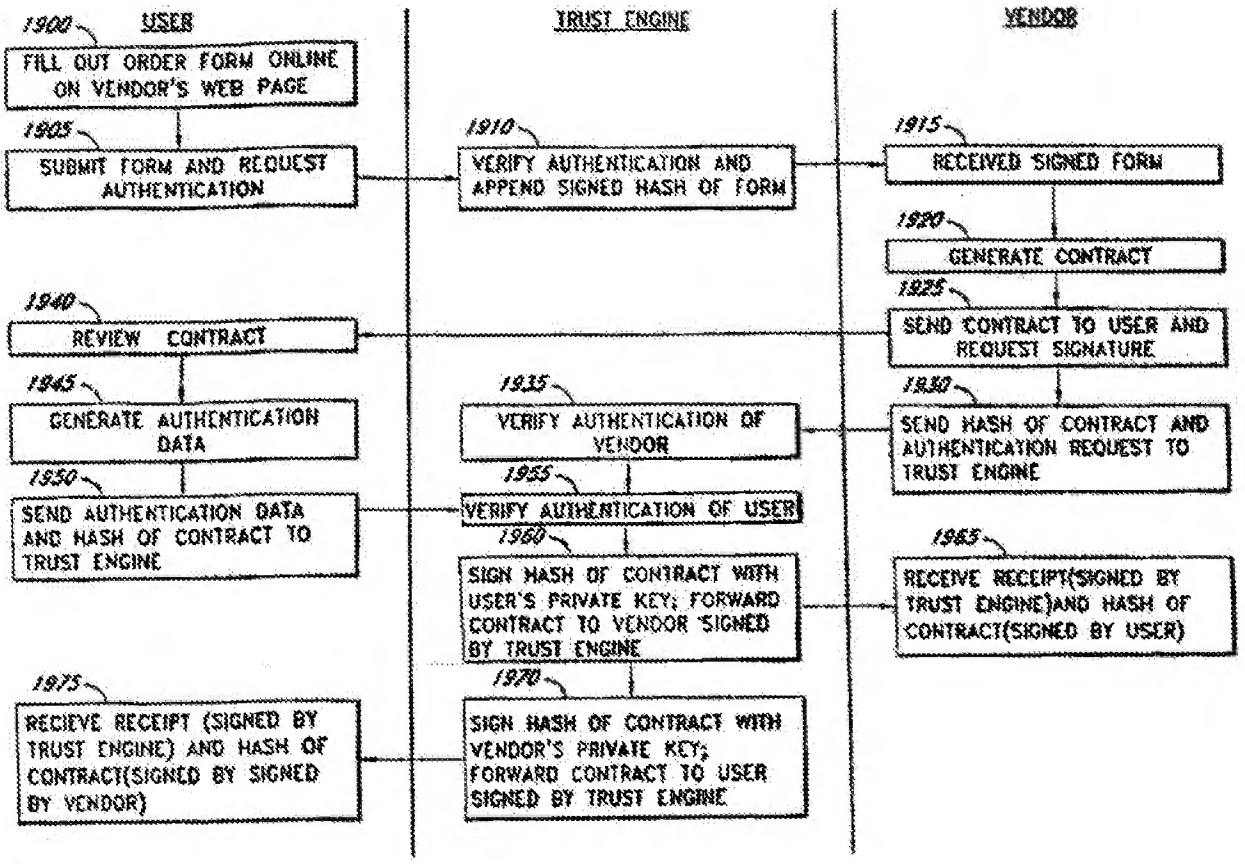

1900
USER
FILL OUT ORDER FORM ONLINE ON VENDOR'S WEB PAGE
1905
SUBMIT FORM AND REQUEST AUTHENTICATION
1910
TRUST ENGINE
VERIFY AUTHENTICATION AND APPEND SIGNED HASH OF FORM
1915
VENDOR
RECEIVED SIGNED FORM
1920
GENERATE CONTRACT
1925
SEND CONTRACT TO USER AND REQUEST SIGNATURE
1930
SEND HASH OF CONTRACT AND AUTHENTICATION REQUEST TO TRUST ENGINE
1935
VERIFY AUTHENTICATION OF VENDOR
1940
REVIEW CONTRACT
1945
GENERATE AUTHENTICATION DATA
1950
SEND AUTHENTICATION DATA AND HASH OF CONTRACT TO TRUST ENGINE
1955
VERIFY AUTHENTICATION OF USER
1960
SIGN HASH OF CONTRACT WITH USER'S PRIVATE KEY; FORWARD CONTRACT TO VENDOR SIGNED BY TRUST ENGINE
1965
RECEIVE RECEIPT(SIGNED BY TRUST ENGINE)AND HASH OF CONTRACT(SIGNED BY USER)
1970
SIGN HASH OF CONTRACT WITH VENDOR'S PRIVATE KEY; FORWARD CONTRACT TO USER SIGNED BY TRUST ENGINE
1975
RECIEVE RECEIPT (SIGNED BY TRUST ENGINE) AND HASH OF CONTRACT(SIGNED BY SIGNED BY VENDOR)

Figure 20

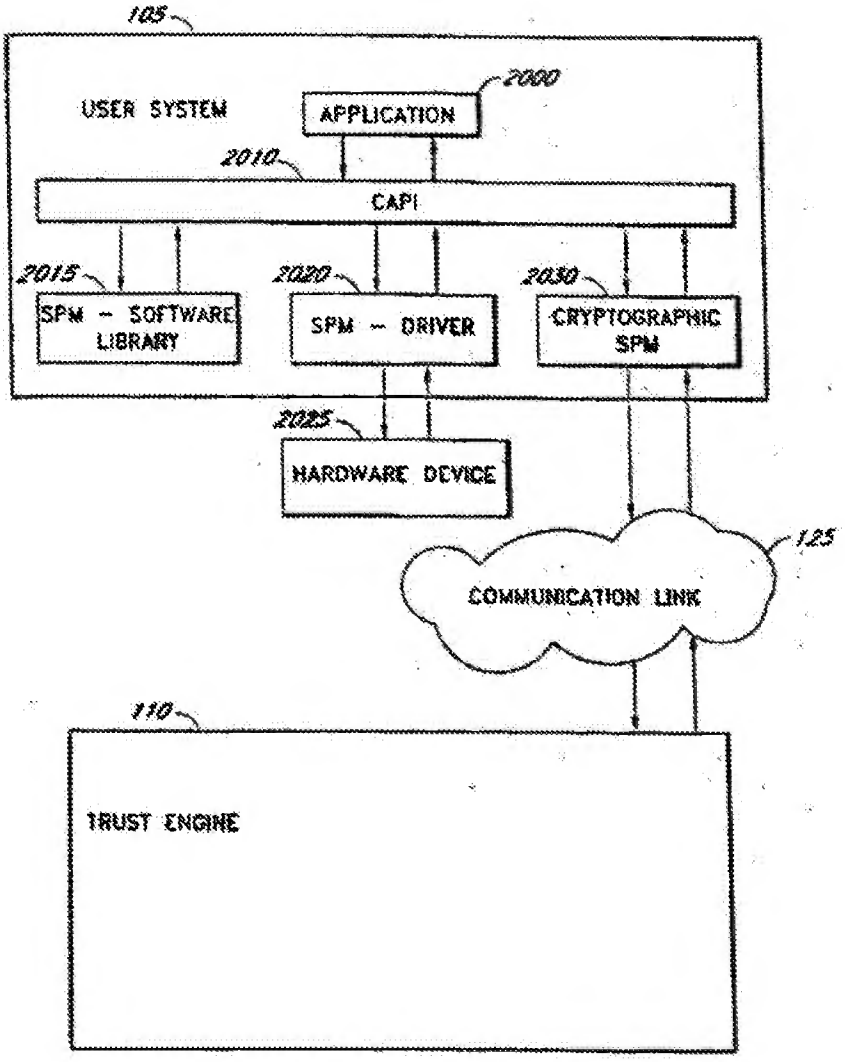

105
USER SYSTEM
APPLICATION
2000
2010
CAPI
2015
SPM – SOFTWARE LIBRARY
2020
SPM – DRIVER
2030
CRYPTOGRAPHIC SPM
2025
HARDWARE DEVICE
125
COMMUNICATION LINK
110
TRUST ENGINE

Figure 21

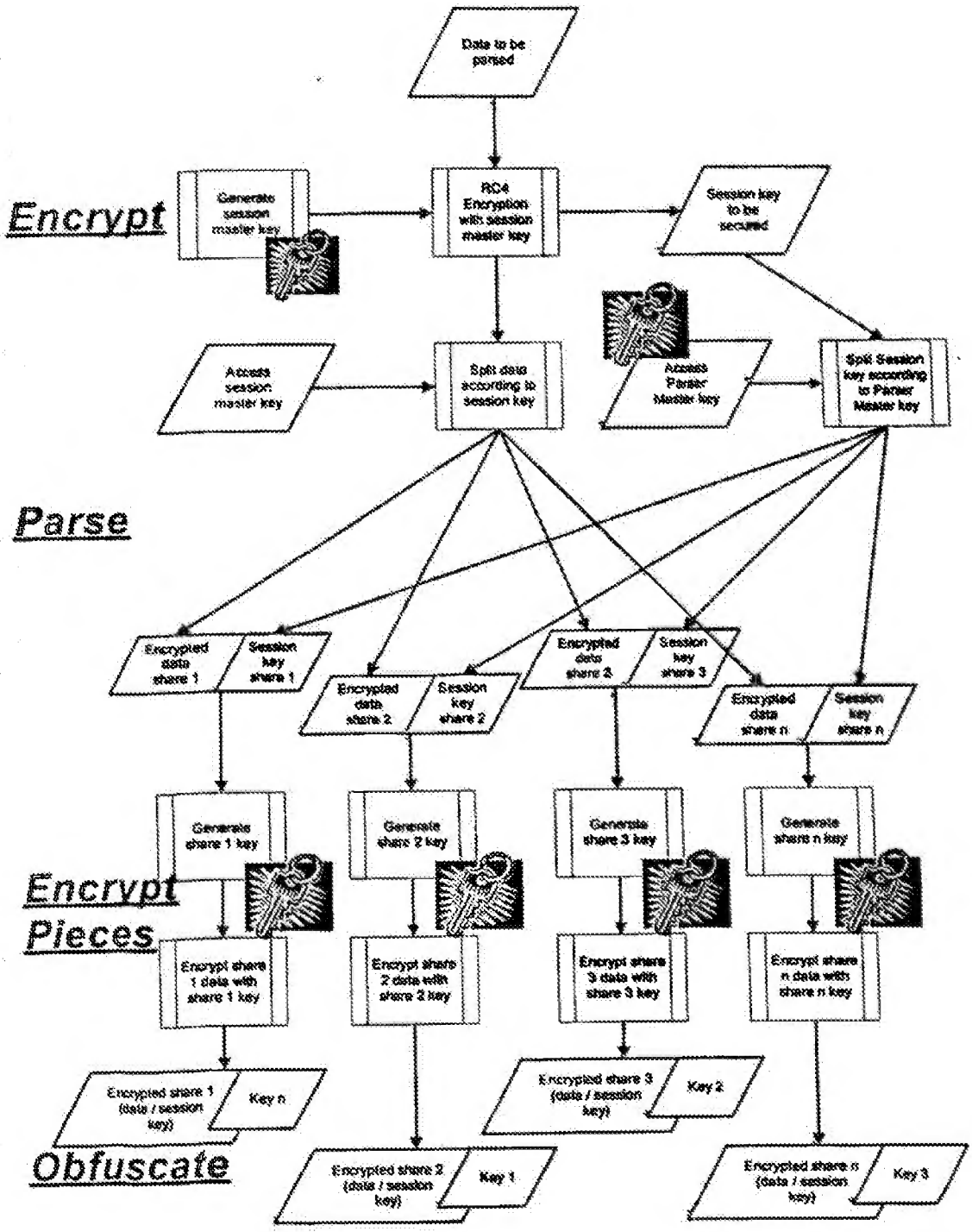

Data to be parsed
Encrypt
Generate session master key
RC4 Encryption with session master key
Session key to be secured
Access session master key
Split data according to session key
Access Parser Master key
Split Session key according to Parser Master key
Parse
Encrypted data share 1
Session key share 1
Encrypted data share 2
Session key share 2
Encrypted data share 3
Session key share 3
Encrypted data share n
Session key share n
Encrypt Pieces
Generate share 1 key
Generate share 2 key
Generate share 3 key
Generate share n key
Encrypt share 1 data with share 1 key
Encrypt share 2 data with share 2 key
Encrypt share 3 data with share 3 key
Encrypt share n data with share n key
Encrypted share 1 (data / session key)
Key n
Encrypted share 3 (data / session key)
Key 2
Obfuscate
Encrypted share 2 (data / session key)
Key 1
Encrypted share n (data / session key)
Key 3

Figure 22

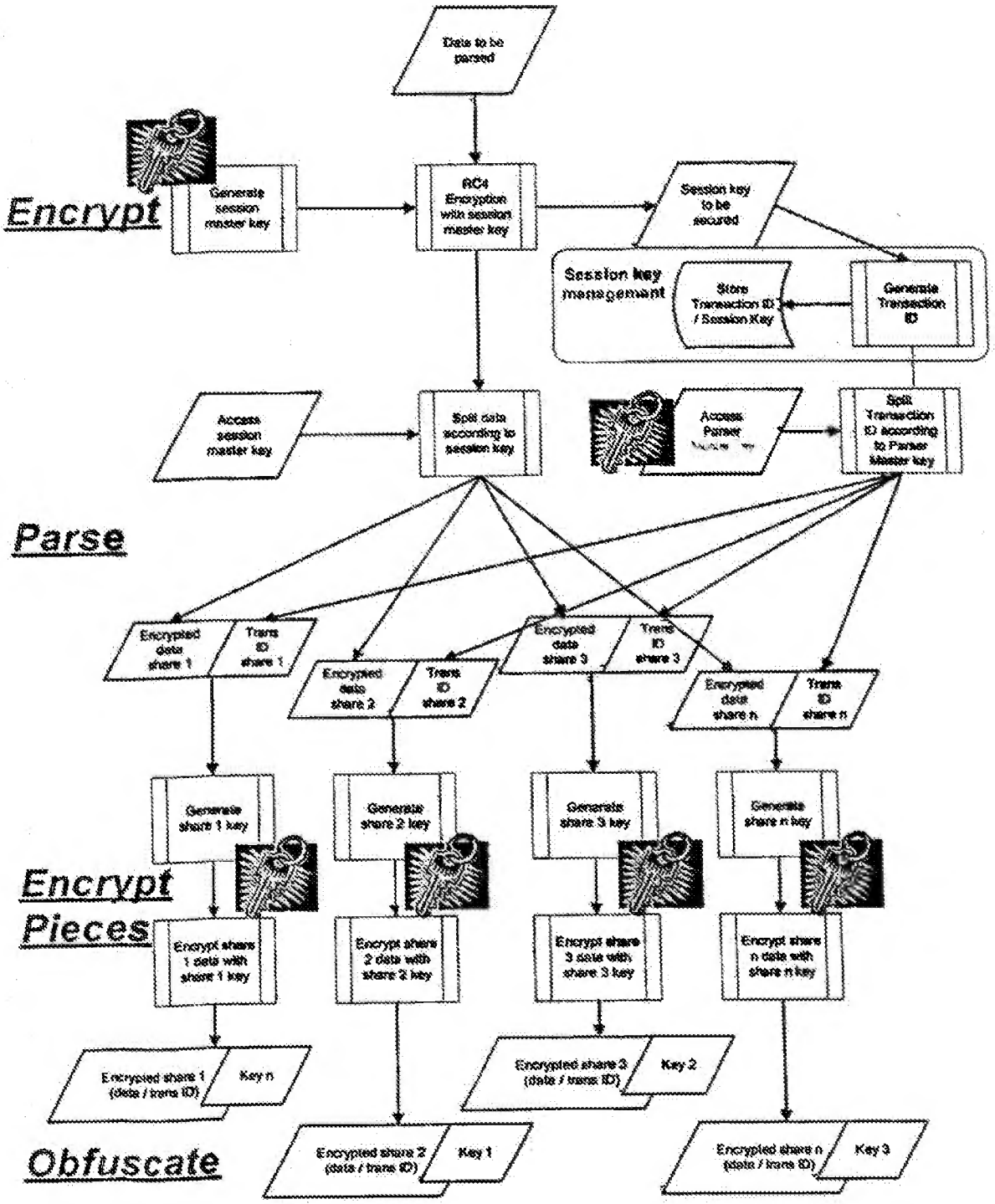

Data to be parsed
Encrypt
Generate session master key
RC4 Encryption with session master key
Session key to be secured
Session key management
Store Transaction ID / Session Key
Generate Transaction ID
Access session master key
Split data according to session key
Access Parser
Split Transaction ID according to Parser Master key
Parse
Encrypted data share 1
Trans ID share 1
Encrypted data share 2
Trans ID share 2
Encrypted data share 3
Trans ID share 3
Encrypted data share n
Trans ID share n
Generate share 1 key
Generate share 2 key
Generate share 3 key
Generate share n key
Encrypt Pieces
Encrypt share 1 data with share 1 key
Encrypt share 2 data with share 2 key
Encrypt share 3 data with share 3 key
Encrypt share n data with share n key
Encrypted share 1 (data / trans ID)
Key n
Encrypted share 3 (data / trans ID)
Key 2
Obfuscate
Encrypted share 2 (data / trans ID)
Key 1
Encrypted share n (data / trans ID)
Key 3

Figure 23

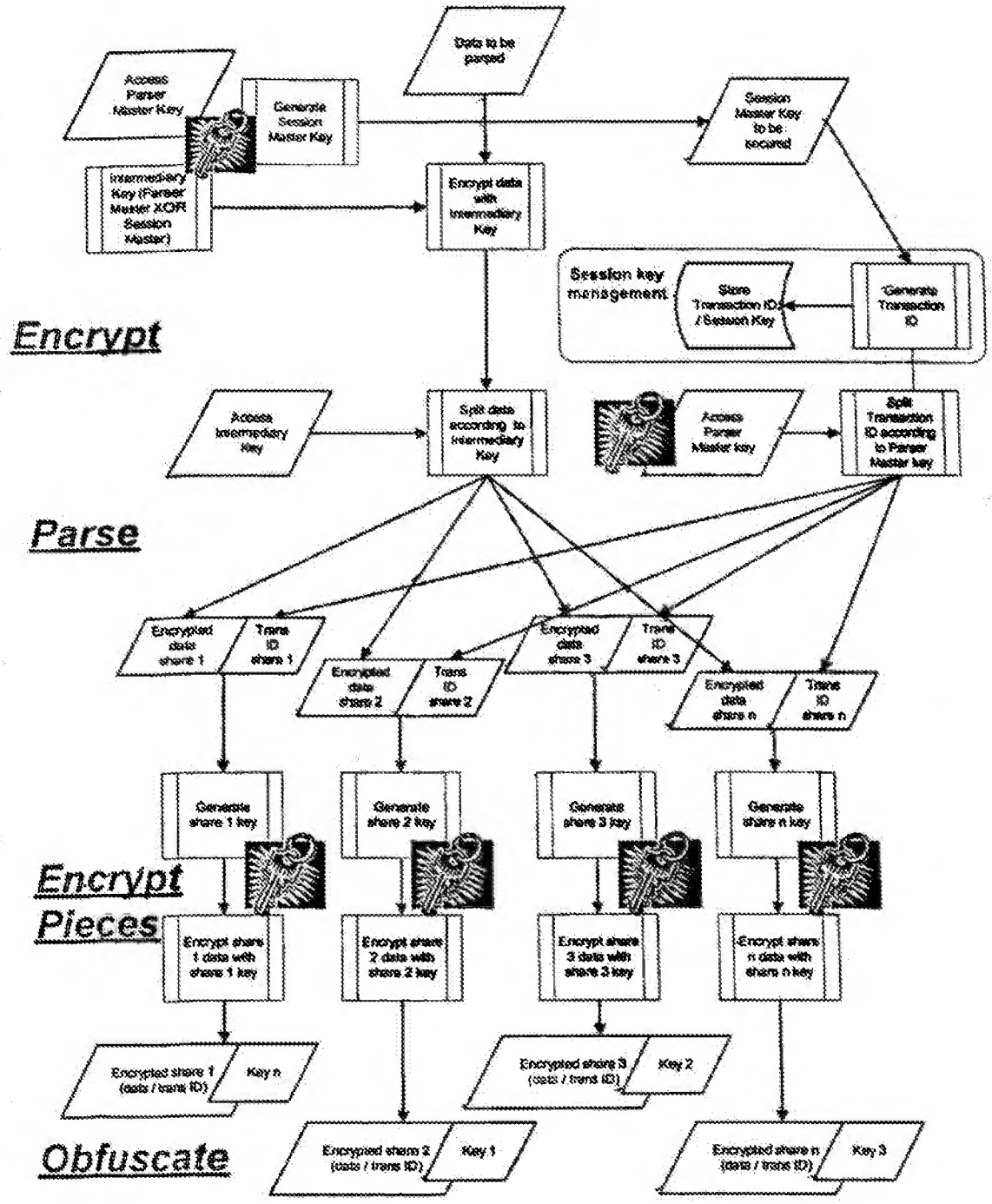

Data to be parsed
Access Parser Master Key
Generate Session Master Key
Session Master Key to be secured
Intermediary Key (Parser Master XOR Session Master)
Encrypt data with Intermediary Key
Session key management
Store Transaction ID / Session Key
Generate Transaction ID
Encrypt
Access Intermediary Key
Split data according to Intermediary Key
Access Parser Master key
Split Transaction ID according to Parser Master key
Parse
Encrypted data share 1
Trans ID share 1
Encrypted data share 2
Trans ID share 2
Encrypted data share 3
Trans ID share 3
Encrypted data share n
Trans ID share n
Generate share 1 key
Generate share 2 key
Generate share 3 key
Generate share n key
Encrypt Pieces
Encrypt share 1 data with share 1 key
Encrypt share 2 data with share 2 key
Encrypt share 3 data with share 3 key
Encrypt share n data with share n key
Encrypted share 1 (data / trans ID)
Key n
Encrypted share 3 (data / trans ID)
Key 2
Obfuscate
Encrypted share 2 (data / trans ID)
Key 1
Encrypted share n (data / trans ID)
Key 3

Figure 24

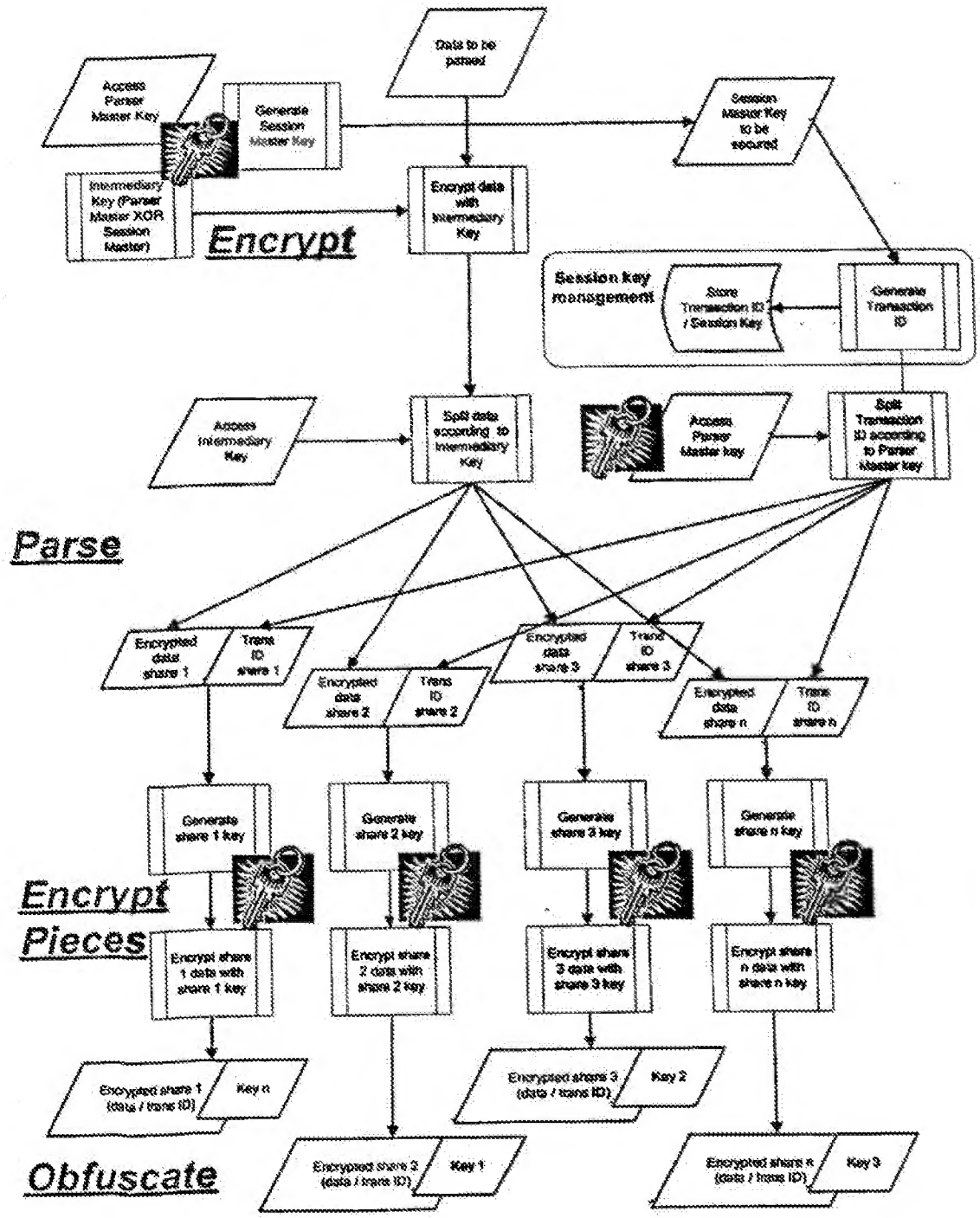
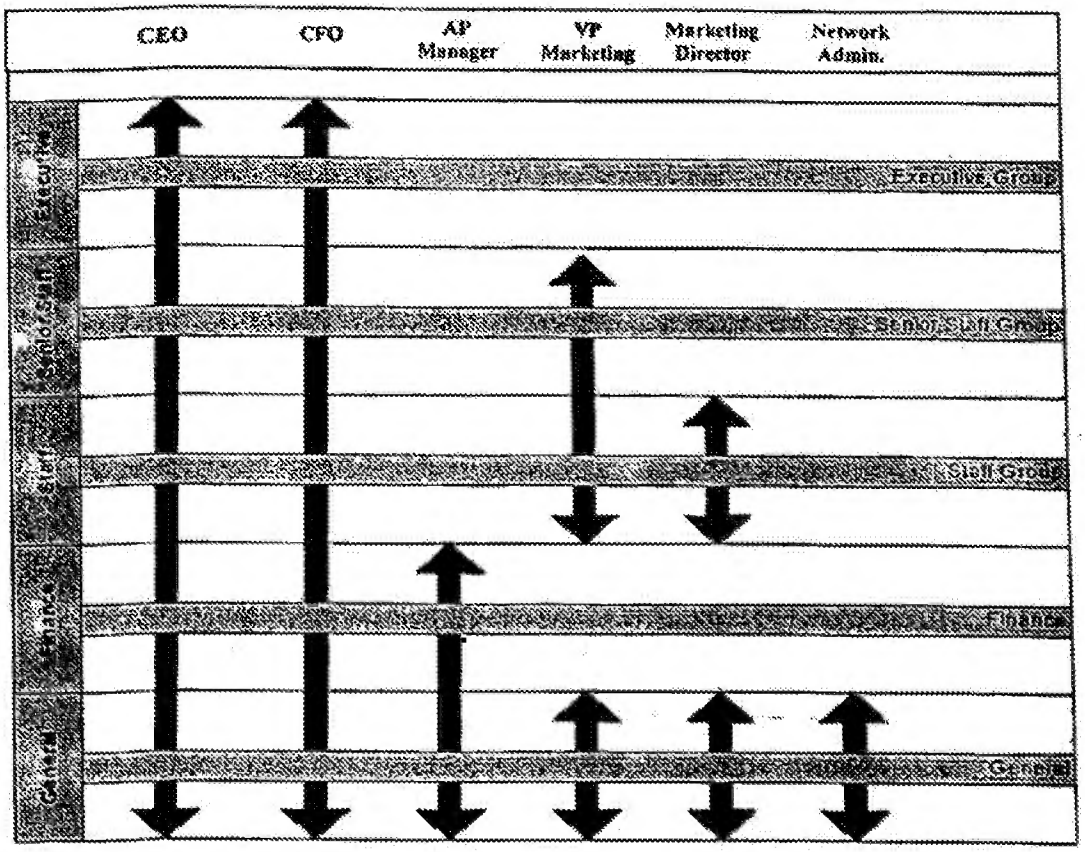

Data to be parsed
Access Parser Master Key
Generate Session Master Key
Session Master Key to be secured
Intermediary Key (Parser Master XOR Session Master)
Encrypt data with Intermediary Key
Encrypt
Session key management
Store Transaction ID / Session Key
Generate Transaction ID
Access Intermediary Key
Split data according to Intermediary Key
Access Parser Master key
Split Transaction ID according to Parser Master key
Parse
Encrypted data share 1
Trans ID share 1
Encrypted data share 2
Trans ID share 2
Encrypted data share 3
Trans ID share 3
Encrypted data share n
Trans ID share n
Generate share 1 key
Generate share 2 key
Generate share 3 key
Generate share n key
Encrypt Pieces
Encrypt share 1 data with share 1 key
Encrypt share 2 data with share 2 key
Encrypt share 3 data with share 3 key
Encrypt share n data with share n key
Encrypted share 1 (data / trans ID)
Key n
Encrypted share 3 (data / trans ID)
Key 2
Encrypted share 2 (data / trans ID)
Key 1
Encrypted share n (data / trans ID)
Key 3
Obfuscate

Figure 25


CEO
CFO
AP Manager
VP Marketing
Marketing Director
Network Admin.
Executive Group
Senior Staff Group
Staff Group
Finance
General

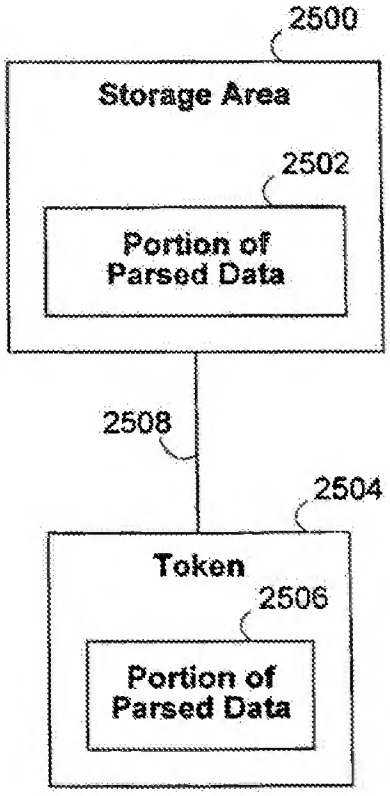

2500
Storage Area
2502
Portion of Parsed Data
2508
2504
Token
2506
Portion of Parsed Data

Figure 26

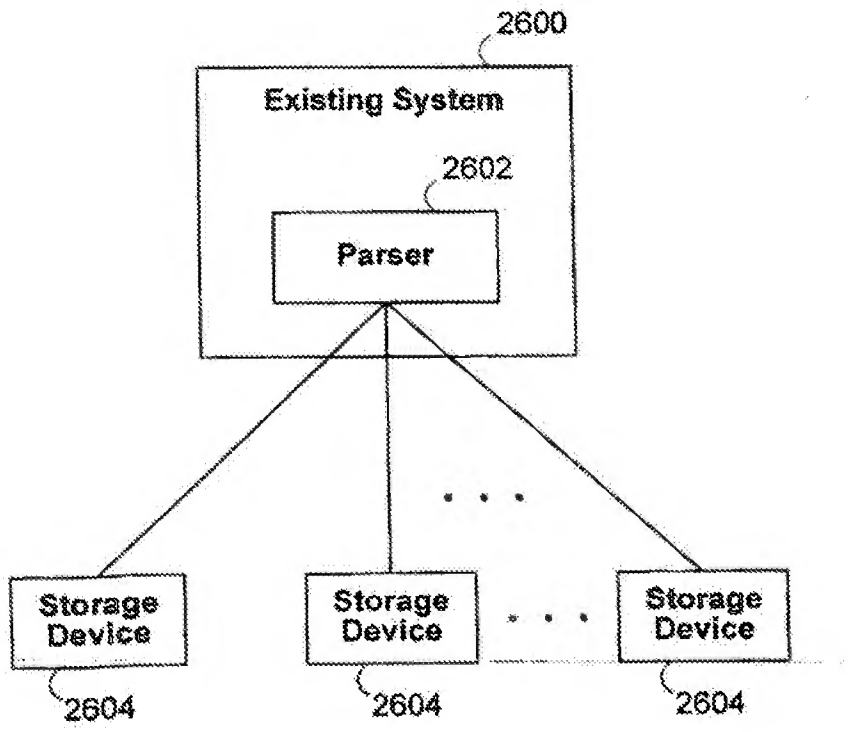

2600
Existing System
2602
Parser
Storage Device
2604
Storage Device
2604
Storage Device
2604

Figure 27

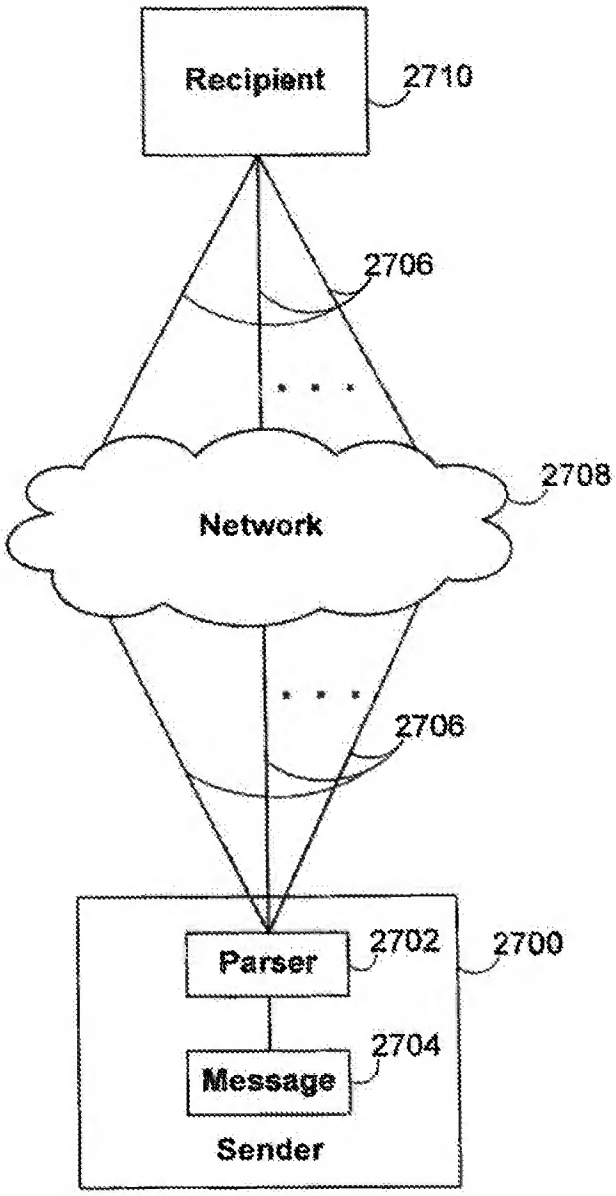

Recipient
2710
2706
2708
Network
2706
Parser
2702
2700
2704
Message
Sender

Figure 28

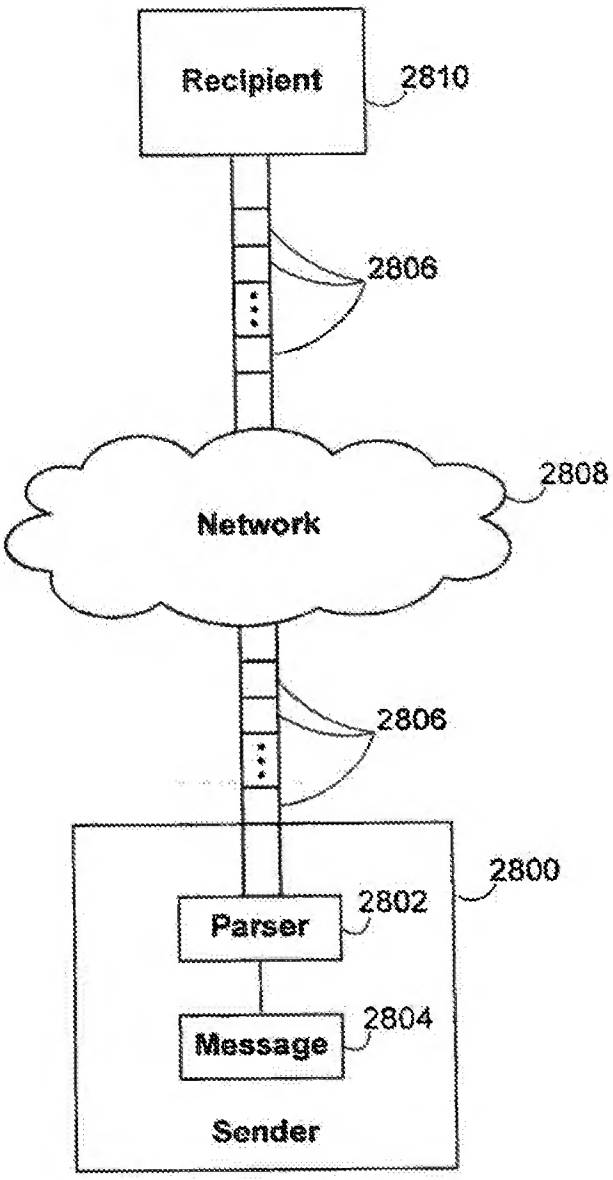

Recipient
2810
2806
Network
2808
2806
2800
Parser
2802
Message
2804
Sender

Figure 29

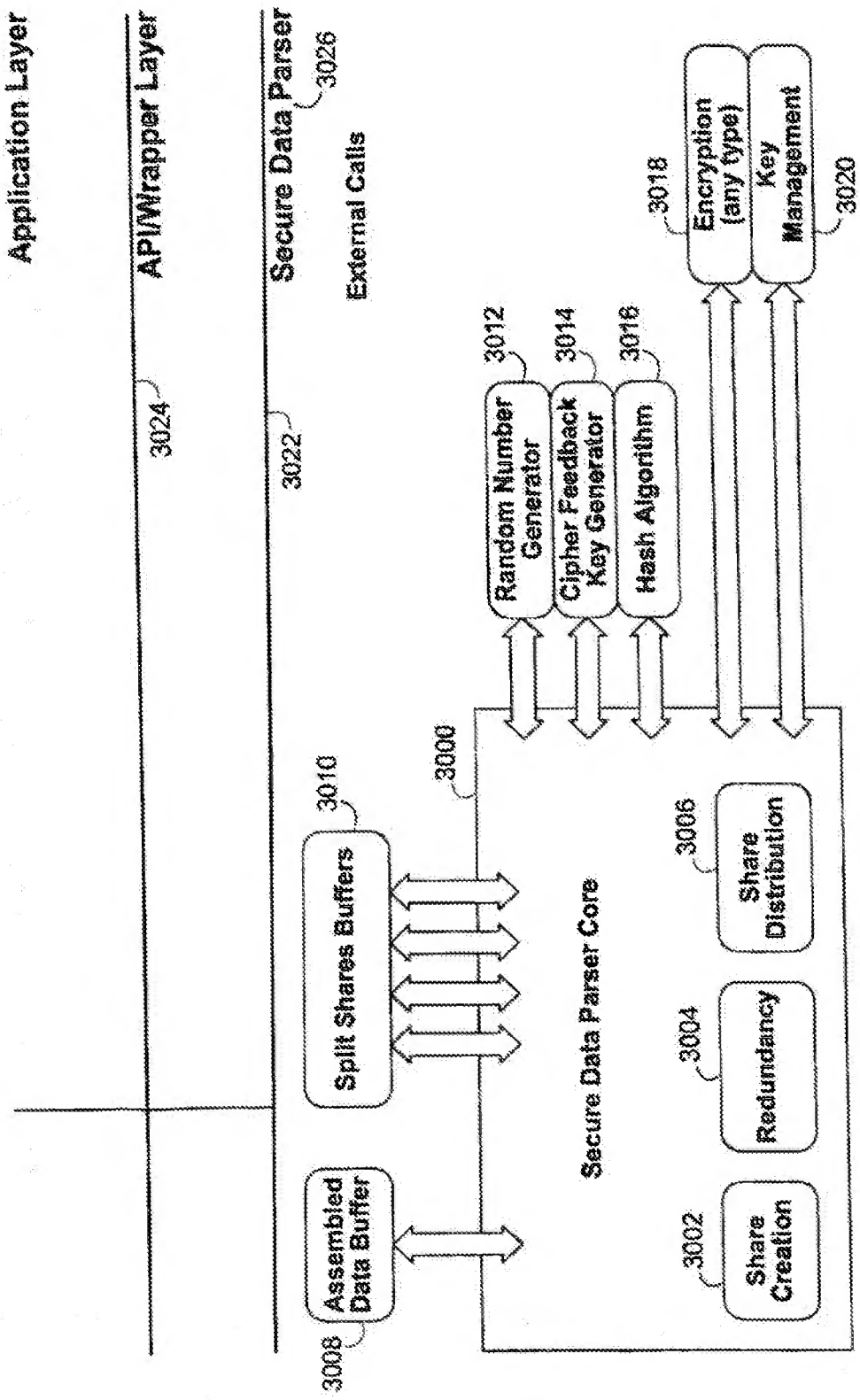

Application Layer
API/Wrapper Layer
Secure Data Parser
3026
External Calls
3024
3022
3012
Random Number Generator
3014
Cipher Feedback Key Generator
3016
Hash Algorithm
3018
Encryption (any type)
Key Management
3020
3000
3010
Split Shares Buffers
Secure Data Parser Core
3006
Share Distribution
3004
Redundancy
3002
Share Creation
3008
Assembled Data Buffer

Figure 30

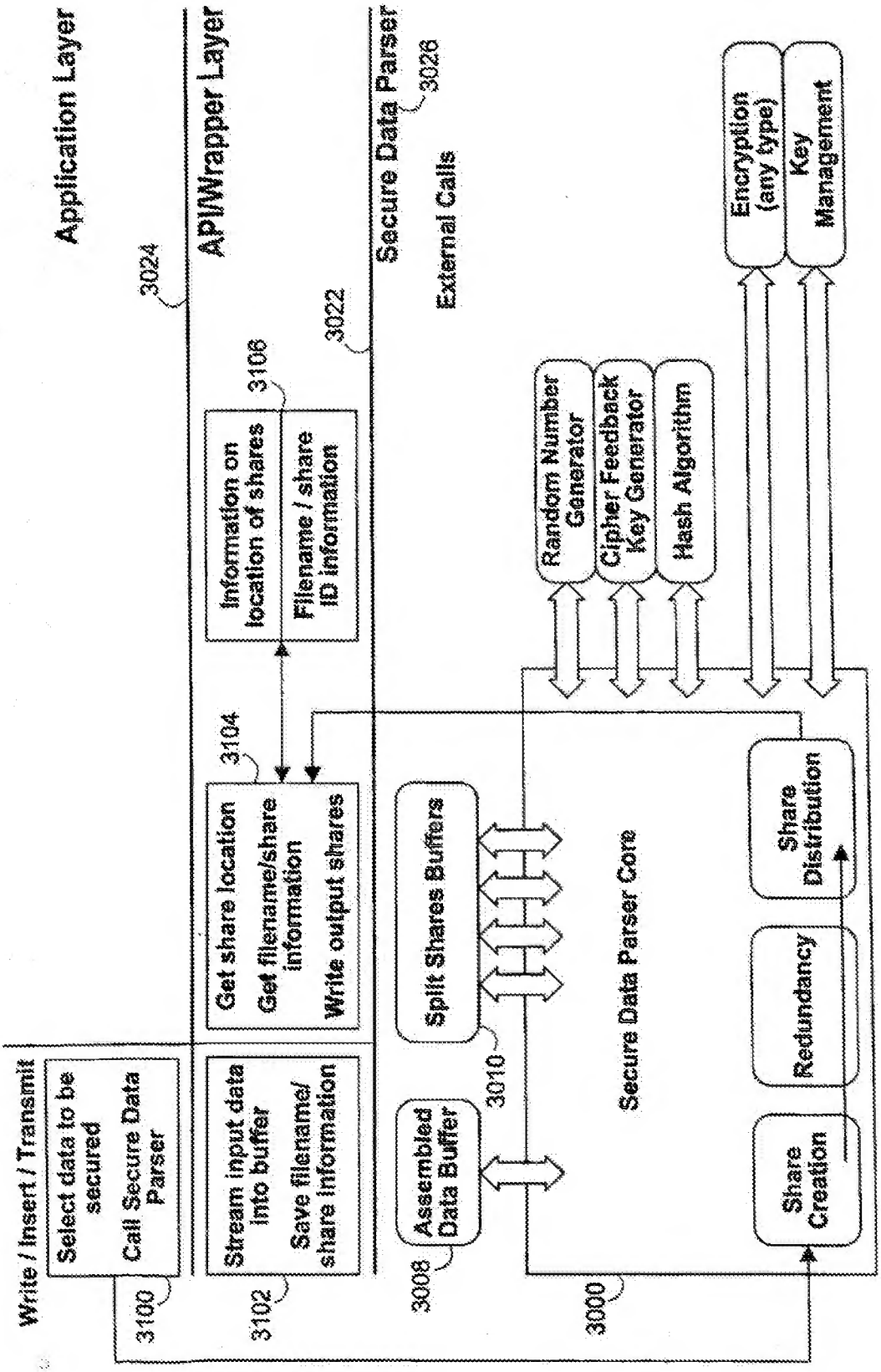

Application Layer
API/Wrapper Layer
Secure Data Parser
3026
External Calls
3024
3022
3106
3104
3100
3102
3008
3010
3000
Write / Insert / Transmit
Select data to be secured
Call Secure Data Parser
Stream input data into buffer
Save filename/ share information
Get share location
Get filename/share information
Write output shares
Information on location of shares
Filename / share ID information
Assembled Data Buffer
Split Shares Buffers
Secure Data Parser Core
Share Creation
Redundancy
Share Distribution
Random Number Generator
Cipher Feedback Key Generator
Hash Algorithm
Encryption (any type)
Key Management

Figure 31

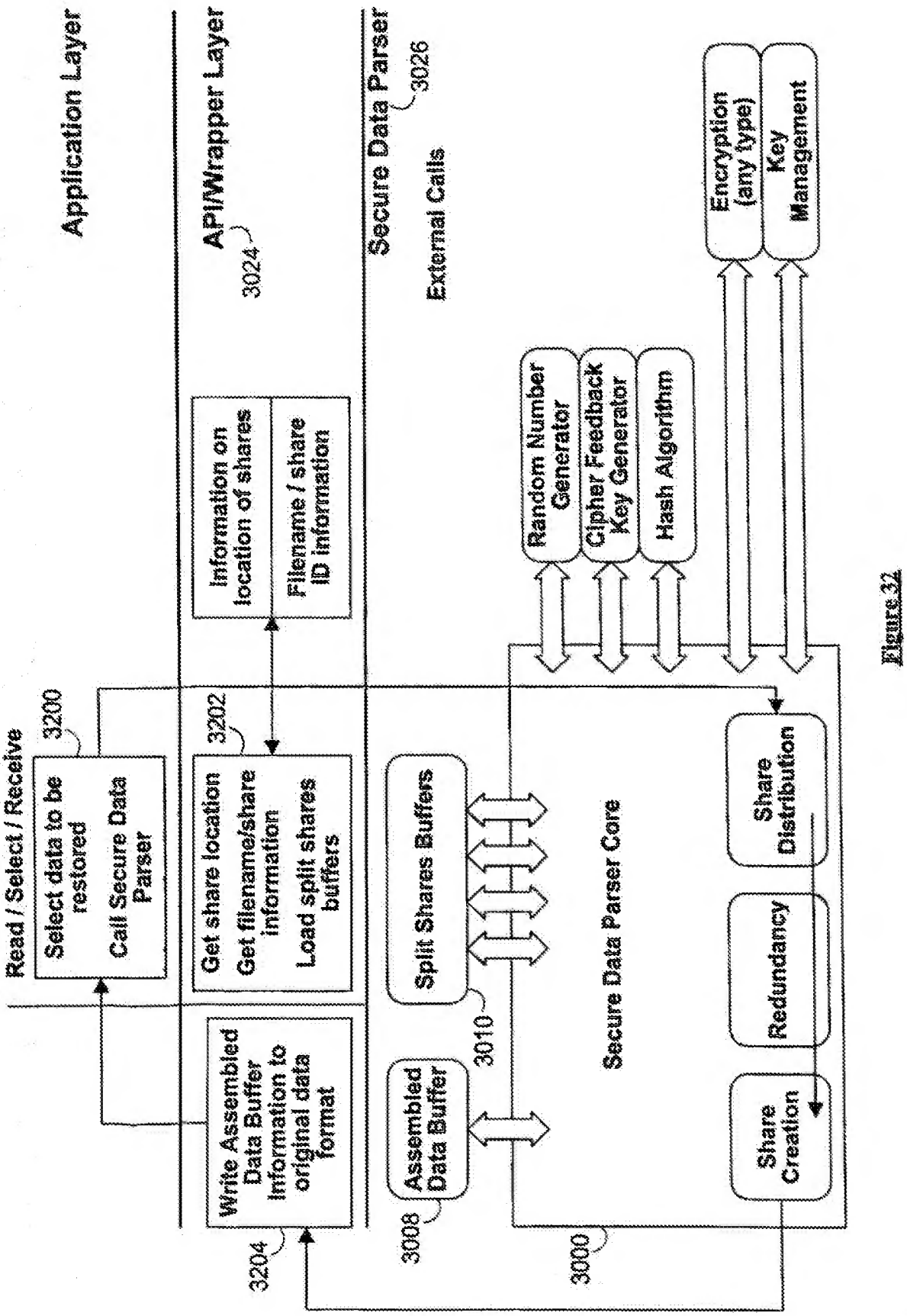

Application Layer
API/Wrapper Layer
3024
Secure Data Parser
3026
External Calls
Read / Select / Receive
3200
Select data to be restored
Call Secure Data Parser
3202
Get share location
Get filename/share information
Load split shares buffers
Information on location of shares
Filename / share ID information
Write Assembled Data Buffer Information to original data format
3204
Assembled Data Buffer
3008
Split Shares Buffers
3010
Secure Data Parser Core
3000
Share Creation
Redundancy
Share Distribution
Random Number Generator
Cipher Feedback Key Generator
Hash Algorithm
Encryption (any type)
Key Management

Figure 32

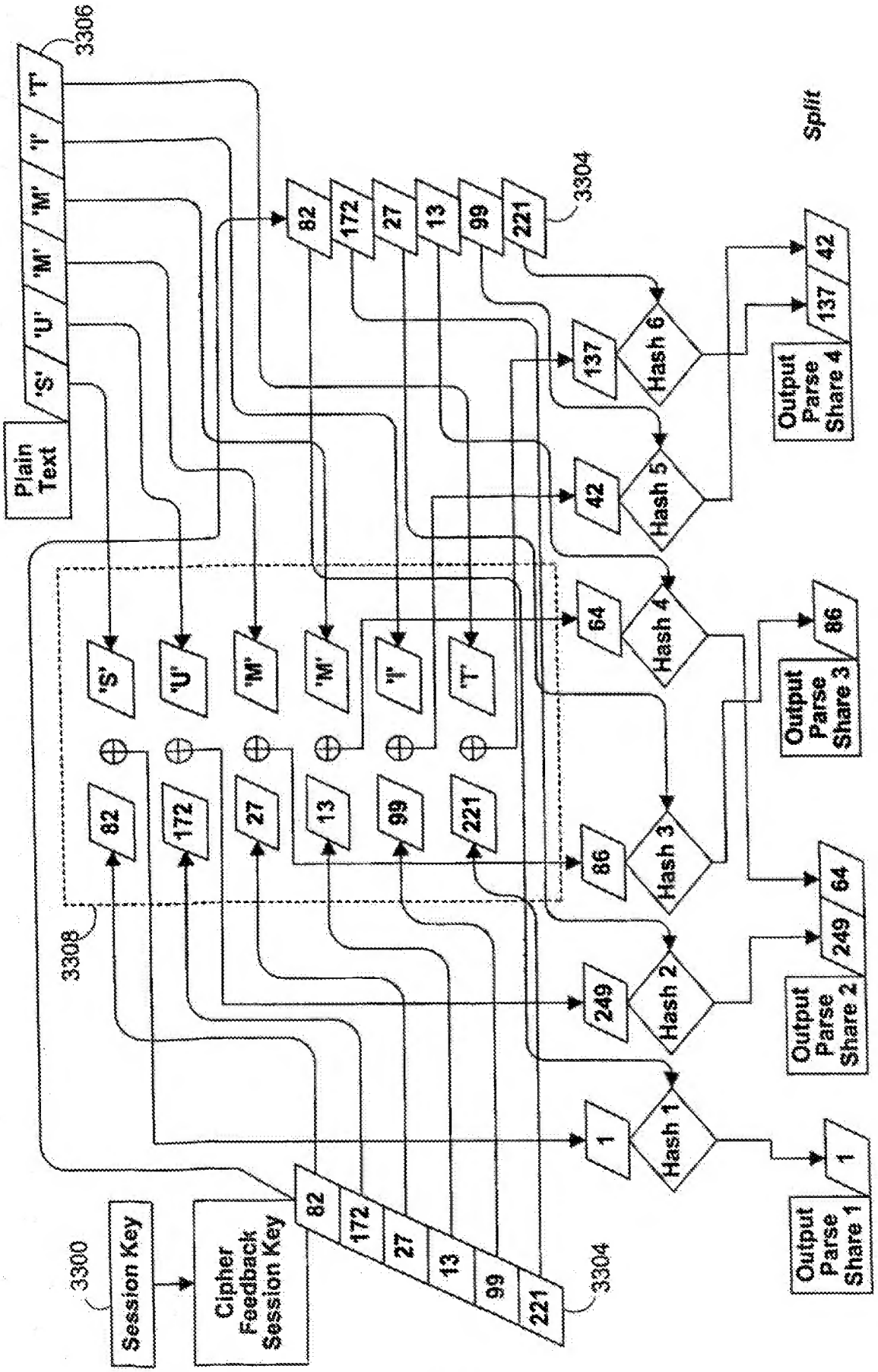

Split
Plain Text
'S' 'U' 'M' 'M' 'I' 'T'
3306
3304
82 172 27 13 99 221
3308
Session Key
3300
Cipher Feedback Session Key
Hash 1
Hash 2
Hash 3
Hash 4
Hash 5
Hash 6
Output Parse Share 1
Output Parse Share 2
Output Parse Share 3
Output Parse Share 4

Figure 33

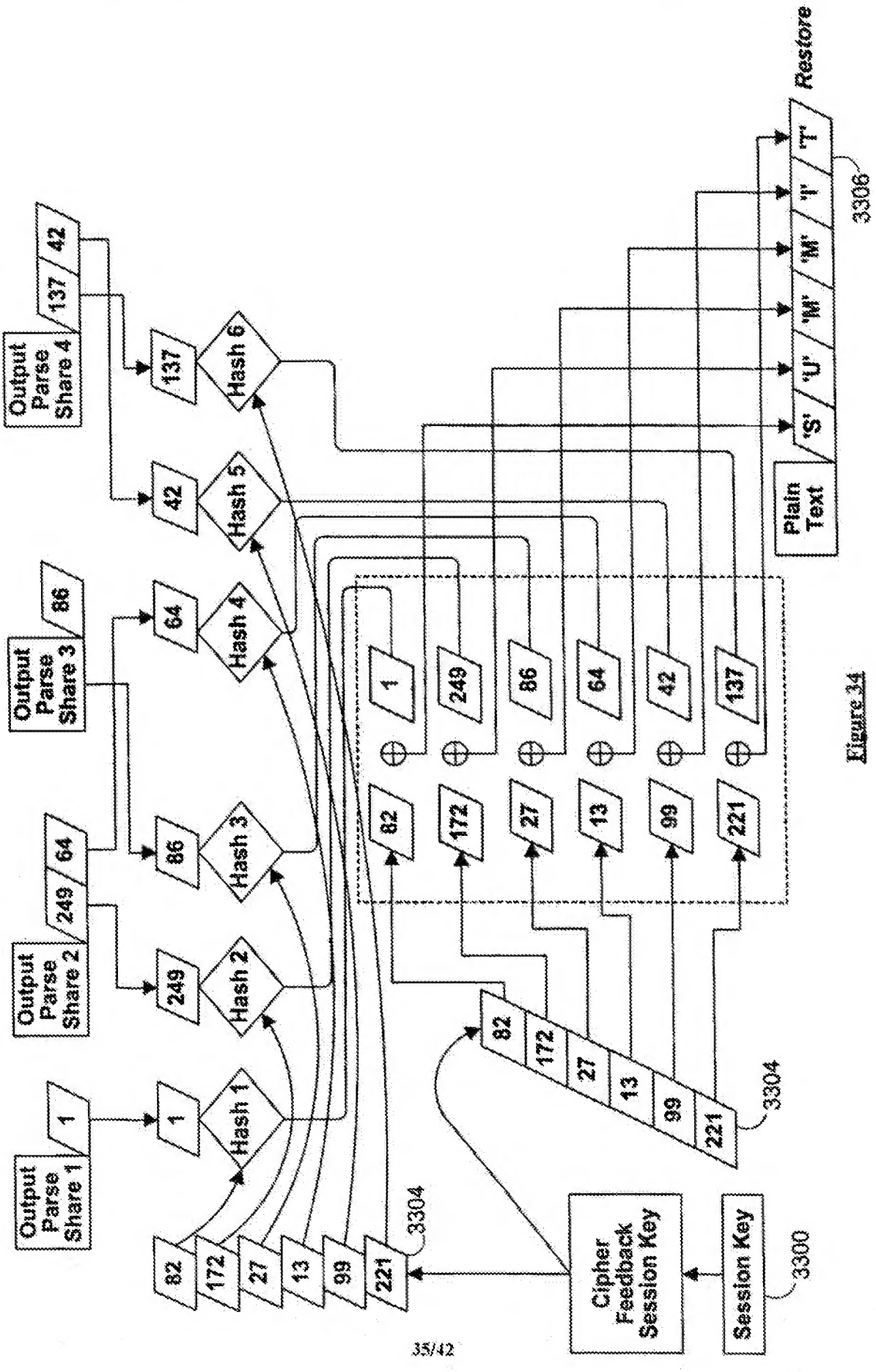

Output Parse Share 1
Output Parse Share 2
Output Parse Share 3
Output Parse Share 4
Hash 1
Hash 2
Hash 3
Hash 4
Hash 5
Hash 6
Plain Text
Restore
'S' 'U' 'M' 'M' 'I' 'T'
3306
3304
3300
Cipher Feedback Session Key
Session Key

Figure 34

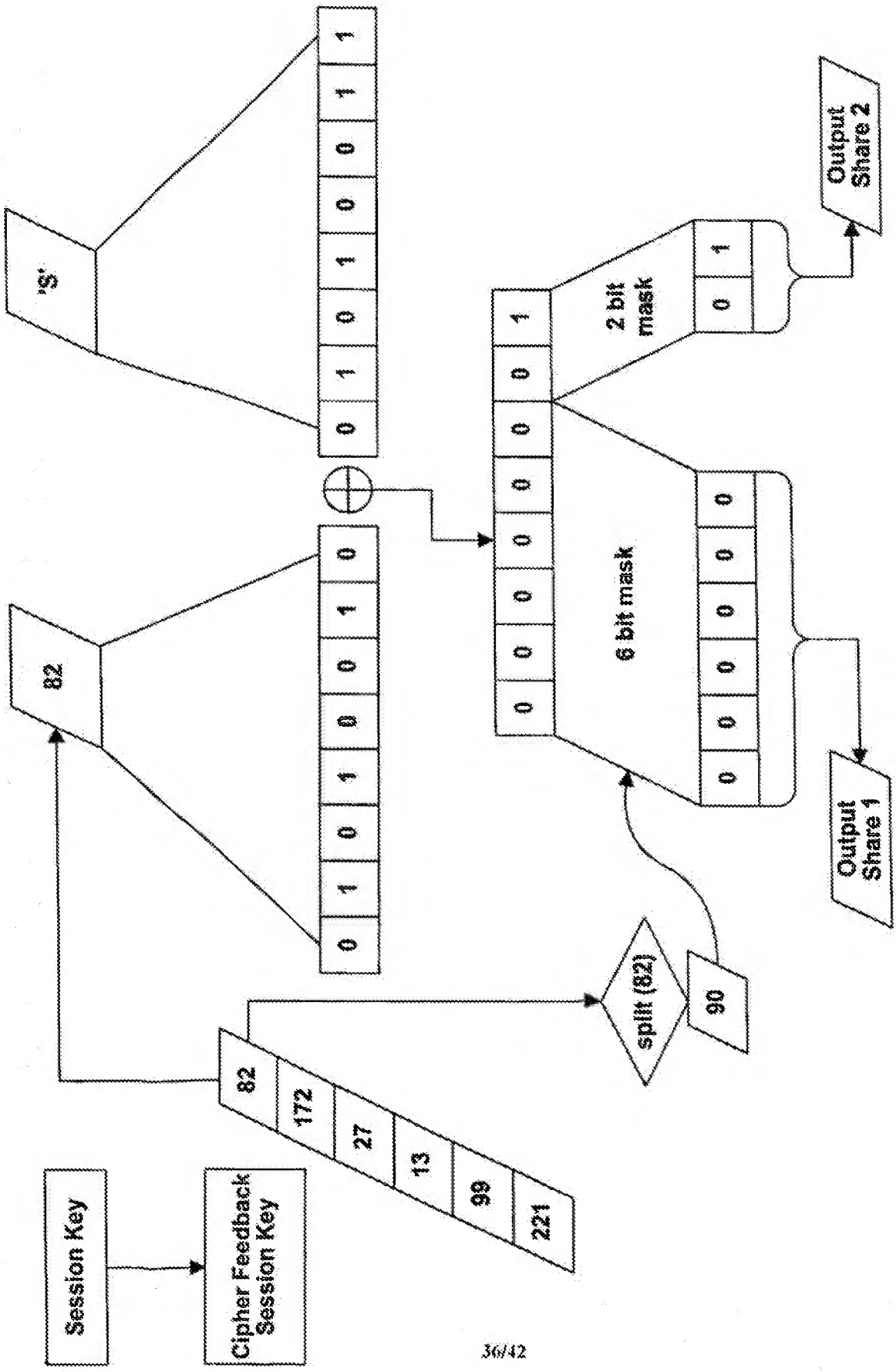

Session Key
Cipher Feedback Session Key
82
172
27
13
99
221
split (82)
90
82
'S'
0 1 0 1 0 0 0 1
0 1 0 1 0 0 1 0
0 0 0 0 0 0 0 1
6 bit mask
2 bit mask
0 0 0 0 0 0
0 1
Output Share 1
Output Share 2

Figure 35

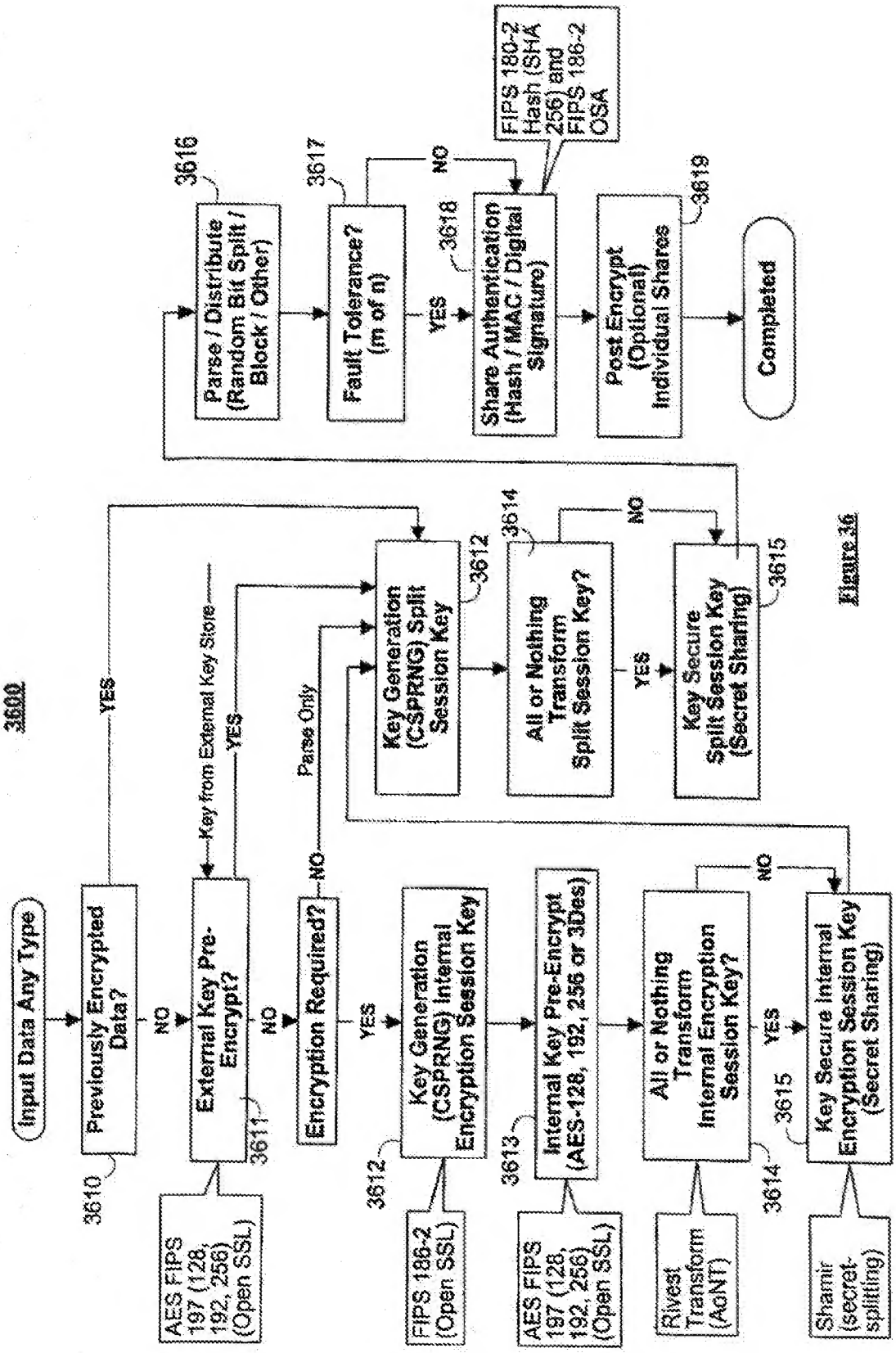

3600
Input Data Any Type
Previously Encrypted Data?
YES
NO
3610
External Key Pre-Encrypt?
Key from External Key Store
YES
NO
3611
AES FIPS 197 (128, 192, 256) (Open SSL)
Encryption Required?
Parse Only
NO
YES
Key Generation (CSPRNG) Split Session Key
3612
Key Generation (CSPRNG) Internal Encryption Session Key
3612
FIPS 186-2 (Open SSL)
All or Nothing Transform Split Session Key?
3614
YES
NO
Internal Key Pre-Encrypt (AES-128, 192, 256 or 3Des)
3613
AES FIPS 197 (128, 192, 256) (Open SSL)
Key Secure Split Session Key (Secret Sharing)
3615
All or Nothing Transform Internal Encryption Session Key?
3614
YES
NO
Rivest Transform (AoNT)
Key Secure Internal Encryption Session Key (Secret Sharing)
3615
Shamir (secret-splitting)
Parse / Distribute (Random Bit Split / Block / Other)
3616
Fault Tolerance? (m of n)
3617
YES
NO
3618
FIPS 180-2 Hash (SHA 256) and FIPS 186-2 OSA
Share Authentication (Hash / MAC / Digital Signature)
Post Encrypt (Optional) Individual Shares
3619
Completed

Figure 36

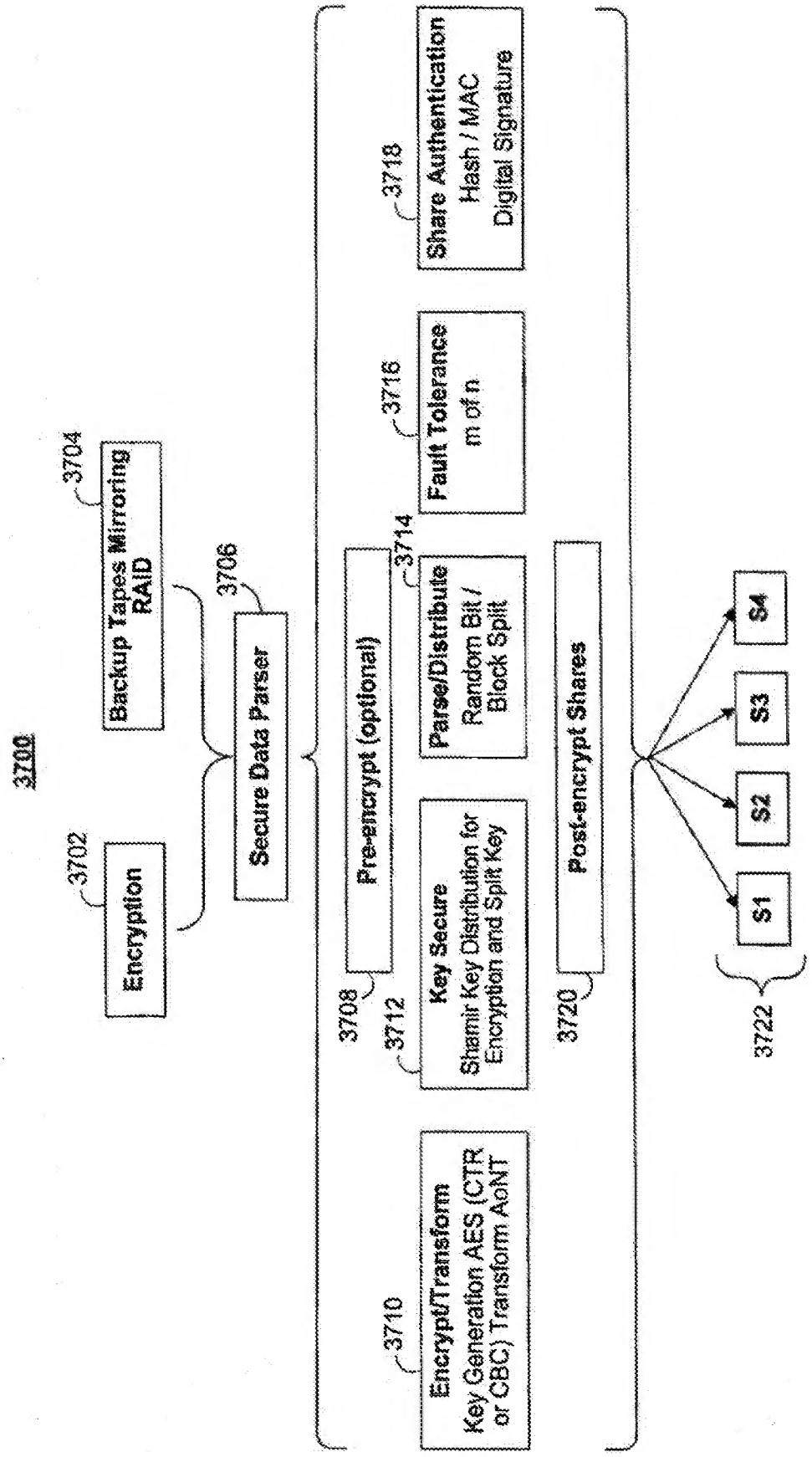

3700
3702
Encryption
3704
Backup Tapes Mirroring RAID
3706
Secure Data Parser
3708
Pre-encrypt (optional)
3710
Encrypt/Transform
Key Generation AES (CTR or CBC) Transform AoNT
3712
Key Secure
Shamir Key Distribution for Encryption and Split Key
3714
Parse/Distribute
Random Bit / Block Split
3716
Fault Tolerance
m of n
3718
Share Authentication
Hash / MAC
Digital Signature
3720
Post-encrypt Shares
3722
S1
S2
S3
S4

Figure 37

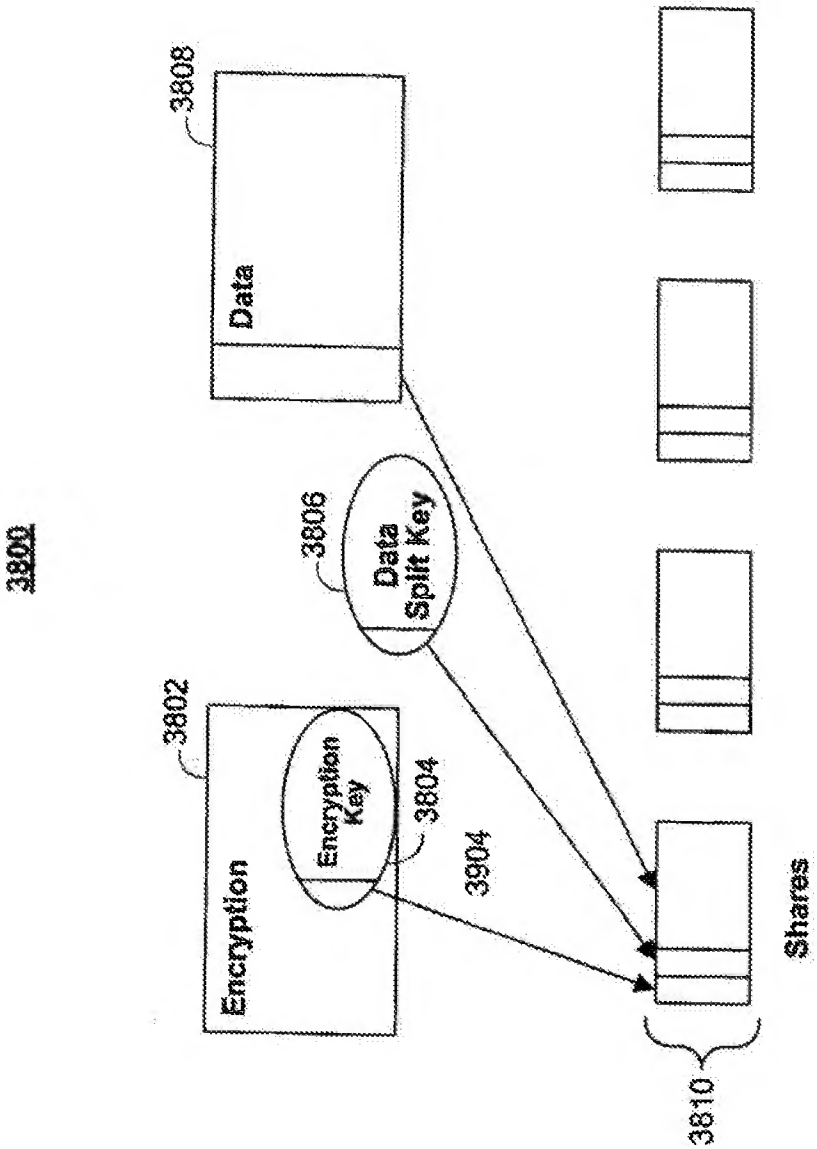

3800
3808
Data
3806
Data Split Key
3802
Encryption
Encryption Key
3804
3904
Shares
3810

Figure 38

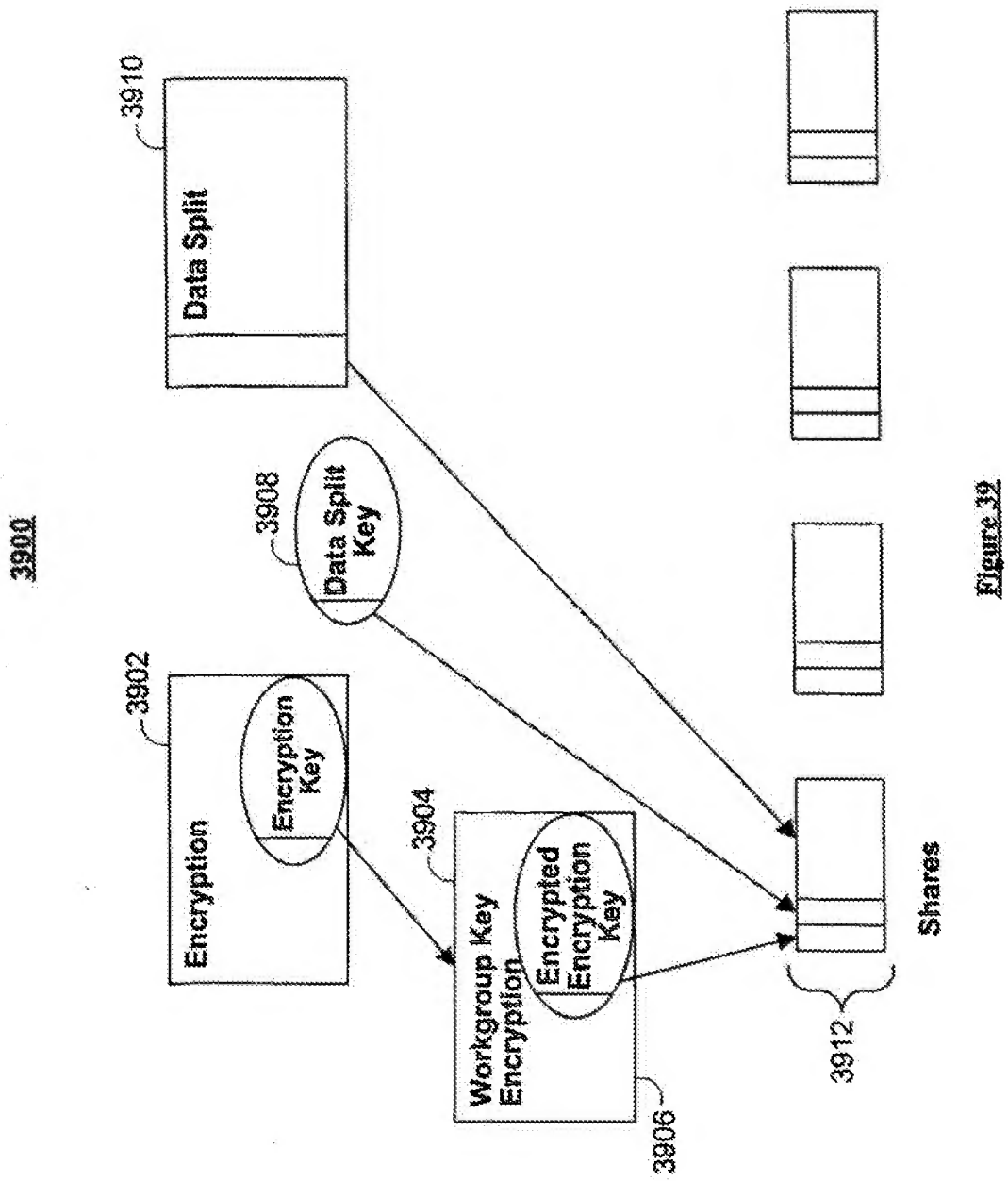

3900
3910
Data Split
3908
Data Split Key
3902
Encryption
Encryption Key
3904
Workgroup Key Encryption
Encrypted Encryption Key
3906
3912
Shares

Figure 39

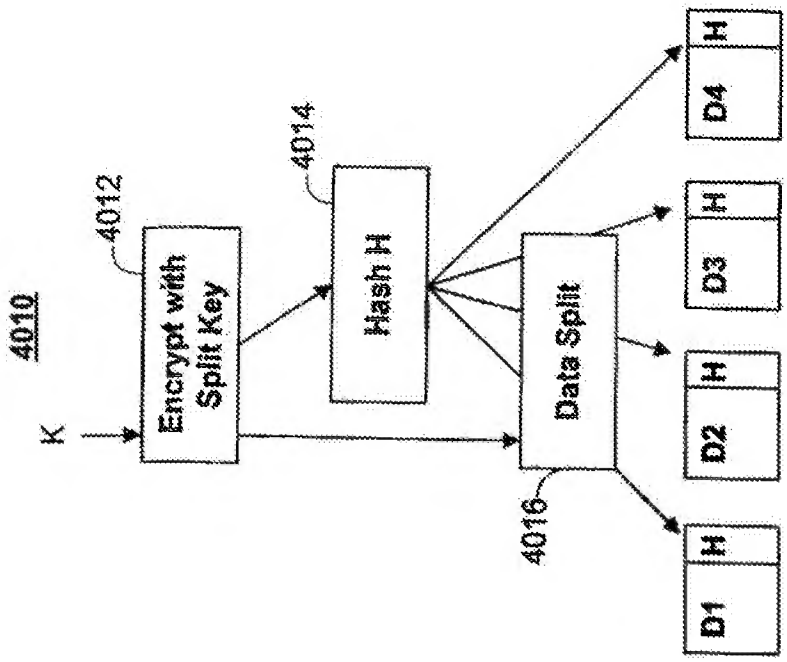


Figure 40B

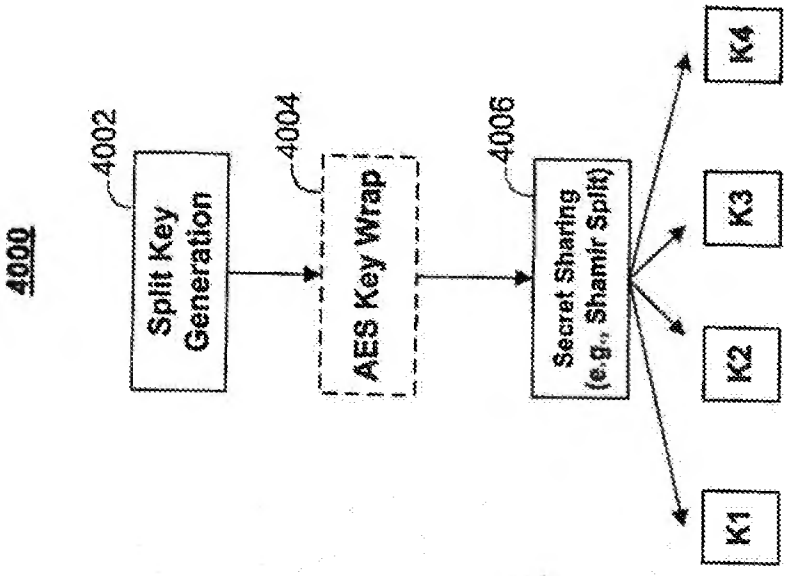


Figure 40A

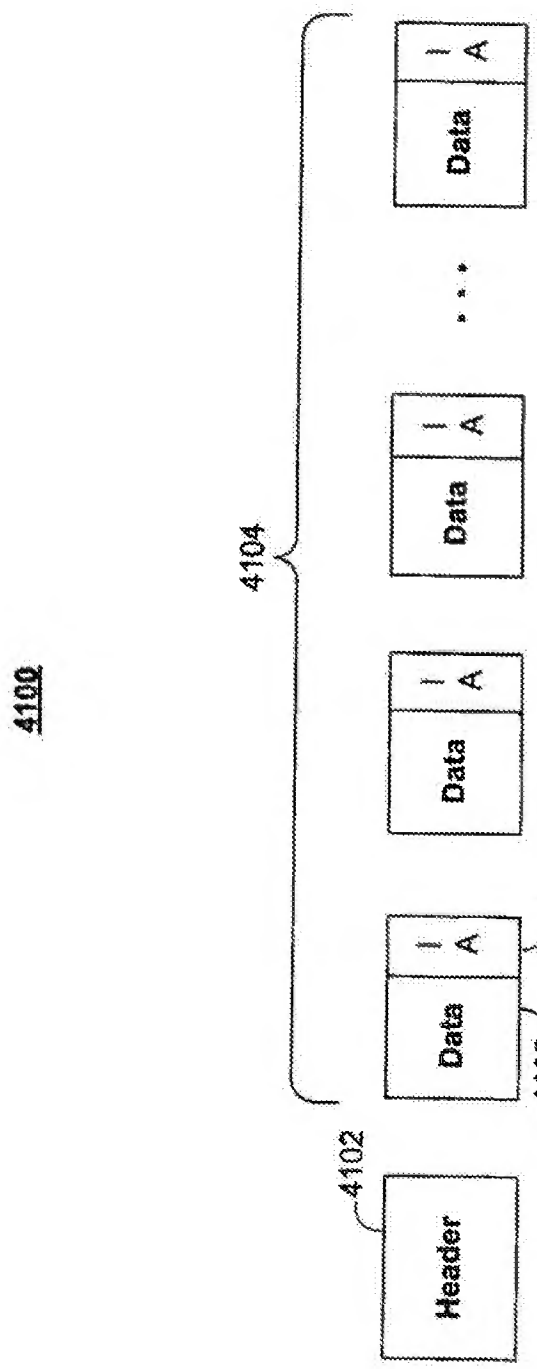

4100
4104
Header
4102
Data
I
A
4106
4108
Data
I
A
Data
I
A
Data
I
A

Figure 41

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
8 May 2008 (08.05.2008)

PCT

(10) International Publication Number
WO 2008/054406 A3

(51) International Patent Classification:
H04K 1/00 (2006.01)

(21) International Application Number:
PCT/US2006/045086

(22) International Filing Date:
20 November 2006 (20.11.2006)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/738,231 18 November 2005 (18.11.2005) US

(71) Applicants and
(72) Inventors: **ORSINI, Rick L.** [US/US]; 2100 Kings Forest Lane, Flower Mound, TX 75028 (US). **O'HARE, Mark S.** [US/US]; 8 Kennedy Court, Coto De Caza, CA 92679 (US). **DAVENPORT, Roger** [US/US]; 3208 Cr 4119, Campbell, TX 75422 (US). **WINICK, Steven** [US/US]; 3 Expressway Plaza, Roslyn Heights, NY 11577 (US).

(74) Agents: **INGERMAN, Jeffrey H.** et al.; Fish & Neave IP Group, Ropes & Gray LLP, 1251 Avenue of the Americas, New York, NY 10020 (US).

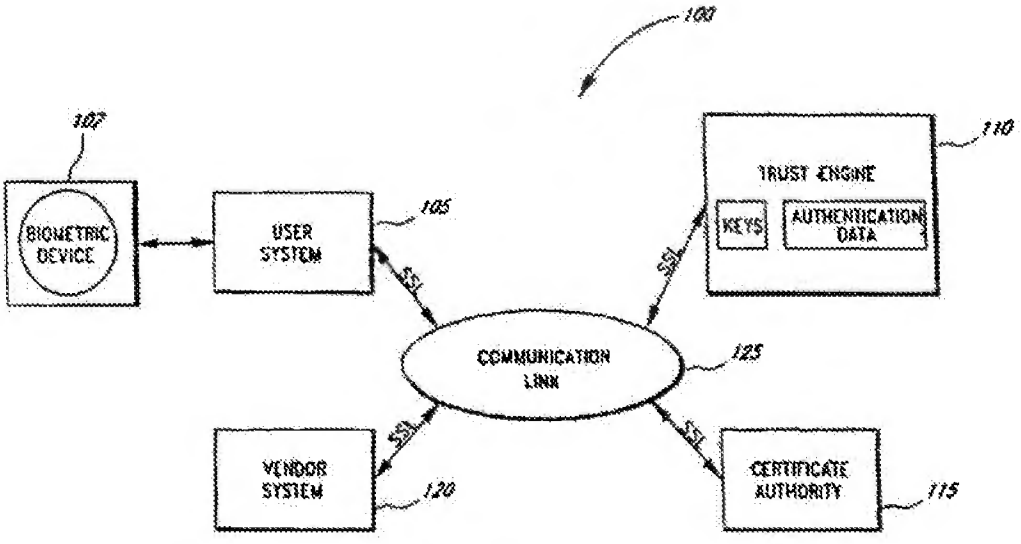
(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LV, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, SV, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, LV, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:
— *with international search report*
— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments*

(88) Date of publication of the international search report:
25 September 2008

(54) Title: SECURE DATA PARSER METHOD AND SYSTEM



(57) Abstract: A secure data parser is provided that may be integrated into any suitable system for securely storing and communicating data. The secure data parser parses data and then splits the data into multiple portions that are stored or communicated distinctly. Encryption of the original data, the portions of data, or both may be employed for additional security. The secure data parser may be used to protect data in motion by splitting original data into portions of data, that may be communicated using multiple communications paths.

WO 2008/054406 A3

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US 08/45066

A. CLASSIFICATION OF SUBJECT MATTER
IPC(8) - H04K 1/00 (2008.04)
USPC - 380/28
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
USPC: 380/28

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
USPC: 380/33,247,268,270,286, 380/33 | 713/156,176,186 (view text search terms below)

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
pubWEST(PGPB,USPT,EPAB,JPAB; PLUR=YES); DialogPRO(Engineering); Google Scholar; Text search terms: Secure data unauthorized access or use, Cryptographic system, Integrity, Remote location, Security user-dependent, Mobile user, Parsing, Splitting or separating, Data stream, Data block, Hash OR MAC signature

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 2002/0010679 A1 (FELSHER) 24 January 2002 (24.01.2002) entire document, especially Abstract and para [0120], [0123], [0128], [0177], [0184], [0220], [0226], [0255], [0266]-[0267], [0287]-[0288], [0294], [0325] and [0345]	1-15
Y	US 2004/0122883 A1 (SIDMAN) 24 June 2004 (24.06.2004) entire document, especially Abstract, FIGS.5-6, 13 and 22 and para [0020], [0082], [0086], [0092], [0101], [0110], [0132], [0139] and [0172]	1-15
A	MYERS et al. 'A secure, publisher-centric Web caching infrastructure'. In: INFOCOM 2001 - Proceedings. IEEE Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies [online], vol. 3, p. 1235-1243. Published 04/22/2001. [retrieved on 2008-07-08]. Retrieved from the Internet: <URL: http://people.ischool.berkeley.edu/~chuang/pubs/gemini.pdf >	1-15

☐ Further documents are listed in the continuation of Box C. ☐

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
06 July 2008 (06.07.2008)	17 JUL 2008
Name and mailing address of the ISA/US Mail Stop PCT, Attn: ISA/US, Commissioner for Patents P.O. Box 1450, Alexandria, Virginia 22313-1450 Facsimile No. 571-273-3201	Authorized officer: Lee W. Young PCT Helpdesk: 571-272-4300 PCT OSP: 571-272-7774

Form PCT/ISA/210 (second sheet) (April 2007)

PTO/SB/81 (01-06)
Approved for use through 11/30/2011. OMB 0651-0035
U.S. Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

POWER OF ATTORNEY OR REVOCATION OF POWER OF ATTORNEY WITH A NEW POWER OF ATTORNEY AND CHANGE OF CORRESPONDENCE ADDRESS

Application Number	11/258,839
Filing Date	October 25, 2005
First Named Inventor	Rick L. Orsini
Title	SECURE DATA PARSER METHOD AND SYSTEM
Art Unit	2432
Examiner Name	Samson B. Lemma
Attorney Docket No.	SFC-2

I hereby revoke all previous powers of attorney given in the above-identified application.

[ ] A Power of Attorney is submitted herewith.

OR

[x] I hereby appoint Practitioner(s) associated with the following Customer Number as my/our attorney(s) or agent(s) to prosecute the application identified above, and to transact all business in the United States Patent and Trademark Office connected therewith:

1473

OR

[ ] I hereby appoint Practitioner(s) named below as my/our attorney(s) or agent(s) to prosecute the application identified above, and to transact all business in the United States Patent and Trademark Office connected therewith:

Practitioner(s) Name	Registration Number	Practitioner(s) Name	Registration Number

Please recognize or change the correspondence address for the above-identified application to:

[x] The address associated with the above-mentioned Customer Number:

OR

[ ] The address associated with Customer Number:

OR

[ ] Firm or Individual Name

Address					
City		State		Zip	
Country		Telephone		Email	

I am the:

[ ] Applicant/Inventor

OR

[x] Assignee of record of the entire interest. See 37 CFR 3.71.
Statement under 37 CFR 3.73(b) (Form PTO/SB/96) submitted herewith or filed on _____

SIGNATURE of Applicant or Assignee of Record

Signature		Date	6-29-11
Name	Mark S. O'Hare	Telephone	(949) 533-9901
Title and Company	President & CEO, Security First Corp.		

NOTE: Signatures of all the inventors or assignees of record of the entire interest or their representative(s) are required. Submit multiple forms if more than one signature is required, see below*.

[x] *Total of 1 forms are submitted.

2708271A1

PTO/SB/26 (07-09)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

TERMINAL DISCLAIMER TO OBVIATE A DOUBLE PATENTING REJECTION OVER A "PRIOR" PATENT	Docket Number (Optional) SFC-2

In re Application of: Orsini et al.

Application No.: 11/258,839

Filed: October 25, 2005

For: SECURE DATA PARSER METHOD AND SYSTEM

The owner*, Security First Corp., of 100 percent interest in the instant application hereby disclaims, except as provided below, the terminal part of the statutory term of any patent granted on the instant application which would extend beyond the expiration date of the full statutory term of **prior patent** No. 7,391,865 as the term of said prior patent is defined in 35 U.S.C. 154 and 173, and as the term of said **prior patent** is presently shortened by any terminal disclaimer. The owner hereby agrees that any patent so granted on the instant application shall be enforceable only for and during such period that it and the **prior patent** are commonly owned. This agreement runs with any patent granted on the instant application and is binding upon the grantee, its successors or assigns.

In making the above disclaimer, the owner does not disclaim the terminal part of the term of any patent granted on the instant application that would extend to the expiration date of the full statutory term as defined in 35 U.S.C. 154 and 173 of the **prior patent**, "as the term of said **prior patent** is presently shortened by any terminal disclaimer," in the event that said **prior patent** later:

expires for failure to pay a maintenance fee;
is held unenforceable;
is found invalid by a court of competent jurisdiction;
is statutorily disclaimed in whole or terminally disclaimed under 37 CFR 1.321;
has all claims canceled by a reexamination certificate;
is reissued; or
is in any manner terminated prior to the expiration of its full statutory term as presently shortened by any terminal disclaimer.

Check either box 1 or 2 below, if appropriate.

1. ☐ For submissions on behalf of a business/organization (e.g., corporation, partnership, university, government agency, etc.), the undersigned is empowered to act on behalf of the business/organization.

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

2. ☒ The undersigned is an attorney or agent of record. Reg. No. 64,813

/Laura Zager/ — Signature

June 30, 2011 — Date

Laura Zager — Typed or printed name

(617) 951-7069 — Telephone Number

☒ Terminal disclaimer fee under 37 CFR 1.20(d) is included.

WARNING: Information on this form may become public. Credit card information should not be included on this form. Provide credit card information and authorization on PTO-2038.

*Statement under 37 CFR 3.73(b) is required if terminal disclaimer is signed by the assignee (owner). Form PTO/SB/96 may be used for making this certification. See MPEP § 324.

27091688_1

Electronic Acknowledgement Receipt

EFS ID:	10422225
Application Number:	11258839
International Application Number:	
Confirmation Number:	2420
Title of Invention:	Secure data parser method and system
First Named Inventor/Applicant Name:	Rick L. Orsini
Customer Number:	01473
Filer:	Laura A. Zager/Joanne Ryan
Filer Authorized By:	Laura A. Zager
Attorney Docket Number:	SFC-2
Receipt Date:	30-JUN-2011
Filing Date:	25-OCT-2005
Time Stamp:	15:34:13
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	yes
Payment Type	Deposit Account
Payment was successfully received in RAM	$875
RAM confirmation Number	2281
Deposit Account	061075
Authorized User	

The Director of the USPTO is hereby authorized to charge indicated fees and credit any overpayment as follows:

Charge any Additional Fees required under 37 C.F.R. Section 1.16 (National application filing, search, and examination fees)

Charge any Additional Fees required under 37 C.F.R. Section 1.17 (Patent application and reexamination processing fees)

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Foreign Reference	AU2004248616B2.pdf	5991582 344ddfa0af6d45bc819902112fe50f42e259330c	no	147
Warnings:					
Information:					
2	Foreign Reference	WO2008054406.pdf	20874488 b22d425d9933db7a869593afdd9add66c01c2081	no	236
Warnings:					
Information:					
3	Non Patent Literature	Damgard_Noninteractive_and_Reusable.pdf	1229546 20867d7a8a27f5bfb3384748f34405308b630b97	no	12
Warnings:					
Information:					
4	Non Patent Literature	Krawczyk-SecretSharingMadeShort.pdf	6179755 5a077c6568e3742d41da82c989cada0cfec3933e	no	11
Warnings:					
Information:					
5	Non Patent Literature	Mitchell_CJ_Making_Serial_Number_based.pdf	5832739 5fa7e9f0e2cacb847e4a7f3dafa942374ff3d997	no	5
Warnings:					
Information:					
6	Extension of Time	SFC-2_Extension.pdf	30974 13b7548070ce8ac5b5bf658b2f49dbb454ccb66e	no	1
Warnings:					
Information:					
7		SFC-2_Response.pdf	46672 d40aee688b6738042c71faffd62a87fa434536e8	yes	17

Multipart Description/PDF files in .zip description

Document Description	Start	End
Amendment/Req. Reconsideration-After Non-Final Reject	1	1
Claims	2	15
Applicant Arguments/Remarks Made in an Amendment	16	17

Warnings:					
Information:					
8	Terminal Disclaimer Filed	SFC-2_TD12148365.pdf	22677 63e84c535a53231f5dd8762a4f53d4e3d9c79c6e	no	1
Warnings:					
Information:					
9	Terminal Disclaimer Filed	SFC-2_TD7391865.pdf	21740 edbf895741067f13c94626ed74c32716bb46321b	no	1
Warnings:					
Information:					
10	Power of Attorney	SFC-2_POA.pdf	1952030 d532227b67ba11ba24522a1694323a37ab1c2ddf	no	1
Warnings:					
Information:					
11	Assignee showing of ownership per 37 CFR 3.73(b).	SFC-2_Statement_Under_373b.pdf	1718219 ae6f46534d15746a7fec725bcecf58b99e4ce5dd	no	1
Warnings:					
Information:					
12	Transmittal Letter	SFC-2_Supp_IDS.pdf	16227 904752b4163855dd43b263066b375915a31f56e1	no	2
Warnings:					
Information:					
13	Information Disclosure Statement (IDS) Form (SB08)	SFC-2_SB08.pdf	53038 f6ac8b012b6cf1e4db25867544c1c03720d03ea6	no	7
Warnings:					
Information:					
This is not an USPTO supplied IDS fillable form					
14	Fee Worksheet (SB06)	fee-info.pdf	33342 498401e059da9ed35afaf849788533014403 6db9	no	2
Warnings:					
Information:					
		Total Files Size (in bytes):	44003029		

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111

If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.

National Stage of an International Application under 35 U.S.C. 371

If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office

If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

PTO/SB/06 (07-06)
Approved for use through 1/31/2007. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

PATENT APPLICATION FEE DETERMINATION RECORD

Substitute for Form PTO-875

Application or Docket Number	Filing Date	☐ To be Mailed
11/258,839	10/25/2005	

APPLICATION AS FILED – PART I

FOR	(Column 1) NUMBER FILED	(Column 2) NUMBER EXTRA	SMALL ENTITY ☒ RATE ($)	SMALL ENTITY ☒ FEE ($)	OR	OTHER THAN SMALL ENTITY RATE ($)	OTHER THAN SMALL ENTITY FEE ($)
☐ BASIC FEE (37 CFR 1.16(a), (b), or (c))	N/A	N/A	N/A			N/A	
☐ SEARCH FEE (37 CFR 1.16(k), (i), or (m))	N/A	N/A	N/A			N/A	
☐ EXAMINATION FEE (37 CFR 1.16(o), (p), or (q))	N/A	N/A	N/A			N/A	
TOTAL CLAIMS (37 CFR 1.16(i))	minus 20 =	*	X $ =		OR	X $ =	
INDEPENDENT CLAIMS (37 CFR 1.16(h))	minus 3 =	*	X $ =			X $ =	
☐ APPLICATION SIZE FEE (37 CFR 1.16(s))	If the specification and drawings exceed 100 sheets of paper, the application size fee due is $250 ($125 for small entity) for each additional 50 sheets or fraction thereof. See 35 U.S.C. 41(a)(1)(G) and 37 CFR 1.16(s).						
☐ MULTIPLE DEPENDENT CLAIM PRESENT (37 CFR 1.16(j))							
* If the difference in column 1 is less than zero, enter "0" in column 2.			TOTAL			TOTAL	

APPLICATION AS AMENDED – PART II

AMENDMENT 06/30/2011	(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA	SMALL ENTITY RATE ($)	SMALL ENTITY ADDITIONAL FEE ($)	OR	OTHER THAN SMALL ENTITY RATE ($)	OTHER THAN SMALL ENTITY ADDITIONAL FEE ($)
Total (37 CFR 1.16(i))	* 66	Minus	** 85	= 0	X $26 =	0	OR	X $ =	
Independent (37 CFR 1.16(h))	* 11	Minus	***21	= 0	X $110 =	0	OR	X $ =	
☐ Application Size Fee (37 CFR 1.16(s))									
☐ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))							OR		
					TOTAL ADD'L FEE	0	OR	TOTAL ADD'L FEE	

AMENDMENT	(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA	RATE ($)	ADDITIONAL FEE ($)	OR	RATE ($)	ADDITIONAL FEE ($)
Total (37 CFR 1.16(i))	*	Minus	**	=	X $ =		OR	X $ =	
Independent (37 CFR 1.16(h))	*	Minus	***	=	X $ =		OR	X $ =	
☐ Application Size Fee (37 CFR 1.16(s))									
☐ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))							OR		
					TOTAL ADD'L FEE		OR	TOTAL ADD'L FEE	

* If the entry in column 1 is less than the entry in column 2, write "0" in column 3.

** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, enter "20".

*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, enter "3".

The "Highest Number Previously Paid For" (Total or Independent) is the highest number found in the appropriate box in column 1.

Legal Instrument Examiner:
/RUTH GRADEN/

This collection of information is required by 37 CFR 1.16. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 12 minutes to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

If you need assistance in completing the form, call 1-800-PTO-9199 and select option 2.

PATENTS
SFC-2

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

PATENT APPLICATION

Applicants	:	Rick L. Orsini et al.
Application No.	:	11/258,839 Confirmation No. 2420
Filed	:	October 25, 2005
For	:	SECURE DATA PARSER METHOD AND SYSTEM
Art Unit	:	2432
Examiner	:	Samson B. Lemma

Boston, Massachusetts
June 30, 2011

Commissioner for Patents
P.O. Box 1450
Alexandria, Virginia 22313-1450

THIRD SUPPLEMENTAL INFORMATION DISCLOSURE STATEMENT

Sir:

Pursuant to 37 C.F.R. §§ 1.56 and 1.97, Applicants hereby make the documents listed on the accompanying Form PTO/SB/08A of record in the above-identified patent application. Pursuant to 37 C.F.R. § 1.98(a)(2)(ii) no copies of the "U.S. Patent Documents" and "U.S. Patent Publications" are provided. Copies of the "Foreign" and "Non-Patent Literature" documents are submitted herewith.

Applicants respectfully request that these documents be printed on any patent which may issue on this application. Applicants request that a copy of Form PTO/SB/08A, as

considered and initialed by the Examiner, be returned with the next communication.

This Information Disclosure Statement is being submitted after the mailing date of the first Office Action, but before the mailing date of a final action under 37 C.F.R. § 1.113, a notice of allowance under 37 C.F.R. § 1.311, or an action that otherwise closes prosecution in the application. The Director is hereby authorized to charge the $180.00 fee set forth in 37 C.F.R. § 1.17(p) to Deposit Account No. 06-1075.

Consideration of the foregoing in relation to this patent application is respectfully requested.

Applicants reserve the right to establish the patentability of the claimed invention over any of the information provided herewith, and/or to prove that this information may not be prior art, and/or to prove that this information may not be enabling for the teachings purportedly offered.

Respectfully submitted,

/Laura Zager/

Laura Zager
Registration No. 52,367
Agent for Applicants
Customer No. 1473

INFORMATION DISCLOSURE STATEMENT BY APPLICANT

(**Not for submission under 37 CFR 1.99)**

Application Number	11258839
Filing Date	2005-10-25
First Named Inventor	Rick L. Orsini
Art Unit	2432
Examiner Name	Samson B. Lemma
Attorney Docket Number	SFC-2

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

☐ That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

☐ See attached certification statement.

☒ Fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☐ None

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Laura Zager/	Date (YYYY-MM-DD)	2011-06-30
Name/Print	Laura Zager	Registration Number	64,813

This collection of information is required by 37 CFR 1.97 and 1.98. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 1 hour to complete, including gathering, preparing and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these record s.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Australian Government

IP Australia

LETTERS PATENT

STANDARD PATENT

2004248616

I, Fatima Beattie, the Commissioner of Patents, grant a Standard Patent with the following particulars:

Name and Address of Patentee(s):

Security First Corporation
22362 Gilberto, Suite 130, Rancho Santa Margarita, CA, 92688, United States of America

Name of Actual Inventor(s):

Orsini, Rick; Davenport, Roger; Vanzandt, John and O'Hare, Mark.

Title of Invention:

Secure data parser method and system

Term of Letters Patent:

Twenty years from 10 June 2004

Priority Details :

Number	*Date*	*Filed with*
10/458,928	11 June 2003	US

COMMONWEALTH OF AUSTRALIA
AUSTRALIA
PATENT OFFICE

Dated this 23rd day of July 2009

Fatima Beattie
Commissioner of Patents

PATENTS ACT 1990

(12) STANDARD PATENT (11) Application No. **AU 2004248616 B2**
(19) AUSTRALIAN PATENT OFFICE

(54) Title
Secure data parser method and system

(51) International Patent Classification(s)
H04L 9/00 (2006.01)

(21) Application No: **2004248616** (22) Date of Filing: **2004.06.10**

(87) WIPO No: **WO04/111791**

(30) Priority Data

(31) Number	(32) Date	(33) Country
10/458,928	**2003.06.11**	**US**

(43) Publication Date: **2004.12.23**
(44) Accepted Journal Date: **2009.04.09**

(71) Applicant(s)
Security First Corporation

(72) Inventor(s)
Orsini, Rick;Davenport, Roger;Vanzandt, John;O'Hare, Mark

(74) Agent / Attorney
Cullen & Co, Level 26 239 George Street, Brisbane, QLD, 4000

(56) Related Art
US5,910,987

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
23 December 2004 (23.12.2004)

PCT

(10) International Publication Number
WO 2004/111791 A3

(51) International Patent Classification[7]: H04L 9/00

(21) International Application Number:
PCT/US2004/018426

(22) International Filing Date: 10 June 2004 (10.06.2004)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
10/458,928 11 June 2003 (11.06.2003) US

(71) Applicant *(for all designated States except US)*: SECURITY FIRST CORPORATION [US/US]; 22362 Gilberto, Suite 130, Rancho Santa Margarita, CA 92688 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **ORSINI, Rick** [US/US]; 2100 Kings Forest Lane, Flower Mound, TX 75028 (US). **VANZANDT, John** [US/US]; 27099 Pacific Terrace Drive, Mission Viejo, CA 92692 (US). **O'HARE, Mark** [US/US]; 45 Oak Knoll, Coto de Caza, CA 92679 (US). **DAVENPORT, Roger** [US/US]; 14423 Heartside Place, Farmers Branch, TX 75234 (US).

(74) Agents: **SCHWARTZ, Jeff** et al.; Preston Gates Ellis & Rouvelas Meeds LLP, 1735 New York Avenue NW, Suite 500, Washington, DC 20006 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NA, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.

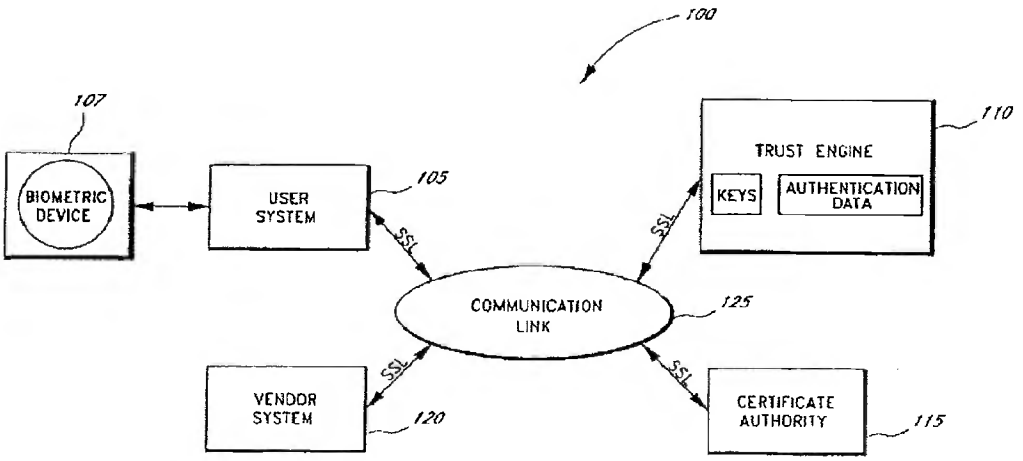
(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IT, LU, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:
— *with international search report*

(88) Date of publication of the international search report:
28 April 2005

[Continued on next page]

(54) Title: SECURE DATA PARSER METHOD AND SYSTEM



(57) Abstract: The present invention provides a method and system for securing sensitive data from unauthorized access or use. The method and system of the present invention is useful in a wide variety of settings, including commercial settings generally available to the public which may be extremely large or small with respect to the number of users. The method and system of the present invention is also useful in a more private setting, such as with a corporation or governmental agency, as well as between corporation, governmental agencies or any other entity.

WO 2004/111791 A3

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

2004248616 25 Feb 2009

SECURE DATA PARSER METHOD AND SYSTEM

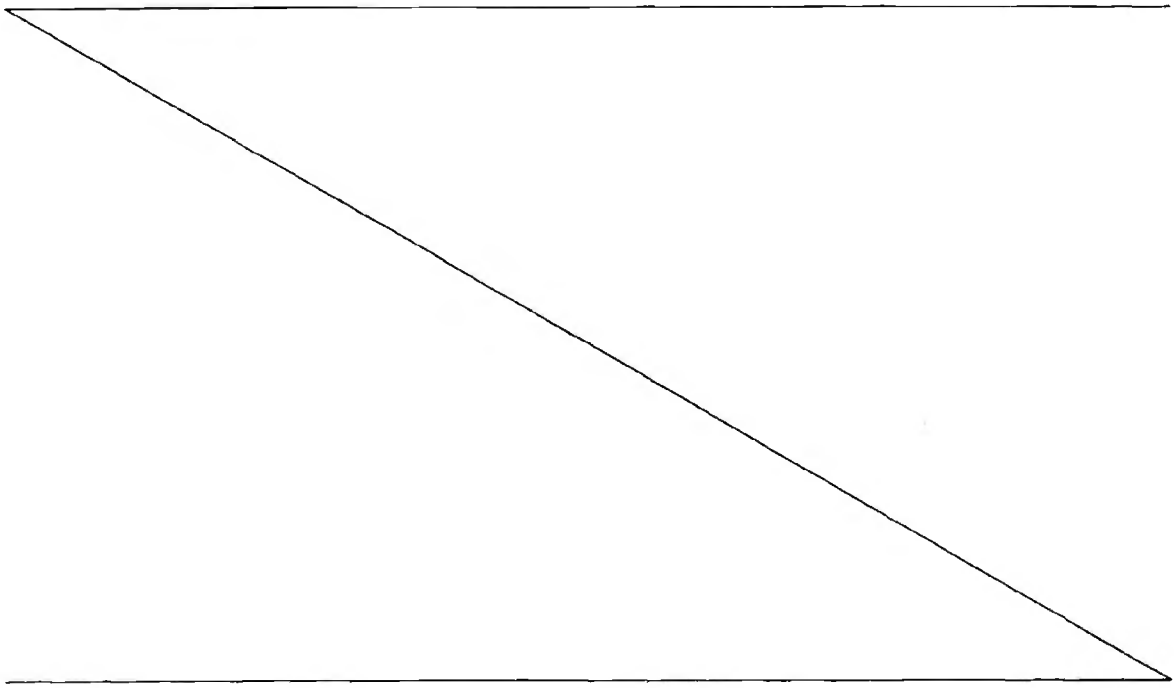
Field of the Invention

The present invention relates in general to a system for securing data from unauthorized access or use.

Description of the Related Art

In today's society, individuals and businesses conduct an ever-increasing amount of activities on and over computer systems. These computer systems, including proprietary and non-proprietary computer networks, are often storing, archiving, and transmitting all types of sensitive information. Thus, an ever-increasing need exists for ensuring data stored and transmitted over these systems cannot be read or otherwise compromised.

One common solution for securing computer systems is to provide login and password functionality. However, password management has proven to be quite costly with a large percentage of help desk calls relating to password issues. Moreover, passwords provide little security in that they are generally stored in a file susceptible to inappropriate access, through, for example, brute-force attacks.



Another solution for securing computer systems is to provide cryptographic infrastructures. Cryptography, in general, refers to protecting data by transforming, or encrypting, it into an unreadable format. Only those who possess the key(s) to the encryption can decrypt the data into a useable format. Cryptography is used to identify users, e.g., authentication, to allow access privileges, e.g., authorization, to create digital certificates and signatures, and the like. One popular cryptography system is a public key system that uses two keys, a public key known to everyone and a private key known only to the individual or business owner thereof. Generally, the data encrypted with one key is decrypted with the other and neither key is recreatable from the other.

Unfortunately, even the foregoing typical public-key cryptographic systems are still highly reliant on the user for security. For example, cryptographic systems issue the private key to the user, for example, through the user's browser. Unsophisticated users then generally store the private key on a hard drive accessible to others through an open computer system, such as, for example, the Internet. On the other hand, users may choose poor names for files containing their private key, such as, for example, "key." The result of the foregoing and other acts is to allow the key or keys to be susceptible to compromise.

In addition to the foregoing compromises, a user may save his or her private key on a computer system configured with an archiving or backup system, potentially resulting in copies of the private key traveling through multiple computer storage devices or other systems. This security breach is often referred to as "key migration." Similar to key migration, many applications provide access to a user's private key through, at most, simple login and password access. As mentioned in the foregoing, login and password access often does not provide adequate security.

One solution for increasing the security of the foregoing cryptographic systems is to include biometrics as part of the authentication or authorization. Biometrics generally include measurable physical characteristics, such as, for example, finger prints or speech that can be checked by an automated system, such as, for example, pattern matching or recognition of finger print patterns or speech patterns. In such systems, a user's biometric and/or keys may be stored on mobile computing devices,

such as, for example, a smartcard, laptop, personal digital assistant, or mobile phone, thereby allowing the biometric or keys to be usable in a mobile environment.

The foregoing mobile biometric cryptographic system still suffers from a variety of drawbacks. For example, the mobile user may lose or break the smartcard or portable computing device, thereby having his or her access to potentially important data entirely cut-off. Alternatively, a malicious person may steal the mobile user's smartcard or portable computing device and use it to effectively steal the mobile user's digital credentials. On the other hand, the portable-computing device may be connected to an open system, such as the Internet, and, like passwords, the file where the biometric is stored may be susceptible to compromise through user inattentiveness to security or malicious intruders.

Summary of the Invention

Based on the foregoing, a need exists to provide a cryptographic system whose security is user-independent while still supporting mobile users.

Accordingly, one aspect of the present invention is to provide a method for securing virtually any type of data from unauthorized access or use. The method comprises one or more steps of parsing, splitting or separating the data to be secured into two or more parts or portions. The method also comprises encrypting the data to be secured. Encryption of the data may be performed prior to or after the first parsing, splitting or separating of the data. In addition, the encrypting step may be repeated for one or more portions of the data. Similarly, the parsing, splitting or separating steps may be repeated for one or more portions of the data. The method also optionally comprises storing the parsed, split or separated data that has been encrypted in one location or in multiple locations. This method also optionally comprises reconstituting or re-assembling the secured data into its original form for authorized access or use. This method may be incorporated into the operations of any computer, server, engine or the like, that is capable of executing the desired steps of the method.

Another aspect of the present invention provides a system for securing virtually any type of data from unauthorized access or use. This system comprises a data splitting module, a cryptographic handling module, and, optionally, a data assembly

module. The system may, in one embodiment, further comprise one or more data storage facilities where secure data may be stored.

Accordingly, one aspect of the invention is to provide a secure server, or trust engine, having server-centric keys, or in other words, storing cryptographic keys and user authentication data on a server. According to this embodiment, a user accesses the trust engine in order to perform authentication and cryptographic functions, such as, but not limited to, for example, authentication, authorization, digital signing and generation, storage, and retrieval of certificates, encryption, notary-like and power-of-attorney-like actions, and the like.

Another aspect of the invention is to provide a reliable, or trusted, authentication process. Moreover, subsequent to a trustworthy positive authentication, a wide number of differing actions may be taken, from providing cryptographic technology, to system or device authorization and access, to permitting use or control of one or a wide number of electronic devices.

Another aspect of the invention is to provide cryptographic keys and authentication data in an environment where they are not lost, stolen, or compromised, thereby advantageously avoiding a need to continually reissue and manage new keys and authentication data. According to another aspect of the invention, the trust engine allows a user to use one key pair for multiple activities, vendors, and/or authentication requests. According to yet another aspect of the invention, the trust engine performs at least one step of cryptographic processing, such as, but not limited to, encrypting, authenticating, or signing, on the server side, thereby allowing clients or users to possess only minimal computing resources.

According to yet another aspect of the invention, the trust engine includes one or multiple depositories for storing portions of each cryptographic key and authentication data. The portions are created through a data splitting process that prohibits reconstruction without a predetermined portion from more than one location in one depository or from multiple depositories. According to another embodiment, the multiple depositories may be geographically remote such that a rogue employee or otherwise compromised system at one depository will not provide access to a user's key or authentication data.

According to yet another embodiment, the authentication process advantageously allows the trust engine to process multiple authentication activities in parallel. According to yet another embodiment, the trust engine may advantageously track failed access attempts and thereby limit the number of times malicious intruders may attempt to subvert the system.

According to yet another embodiment, the trust engine may include multiple instantiations where each trust engine may predict and share processing loads with the others. According to yet another embodiment, the trust engine may include a redundancy module for polling a plurality of authentication results to ensure that more than one system authenticates the user.

Therefore, one aspect of the invention includes a secure cryptographic system, which may be remotely accessible, for storing data of any type, including, but not limited to, a plurality of private cryptographic keys to be associated with a plurality of users. The cryptographic system associates each of the plurality of users with one or more different keys from the plurality of private cryptographic keys and performs cryptographic functions for each user using the associated one or more different keys without releasing the plurality of private cryptographic keys to the users. The cryptographic system comprises a depository system having at least one server which stores the data to be secured, such as a plurality of private cryptographic keys and a plurality of enrollment authentication data. Each enrollment authentication data identifies one of multiple users and each of the multiple users is associated with one or more different keys from the plurality of private cryptographic keys. The cryptographic system also may comprise an authentication engine which compares authentication data received by one of the multiple users to enrollment authentication data corresponding to the one of multiple users and received from the depository system, thereby producing an authentication result. The cryptographic system also may comprise a cryptographic engine which, when the authentication result indicates proper identification of the one of the multiple users, performs cryptographic functions on behalf of the one of the multiple users using the associated one or more different keys received from the depository system. The cryptographic system also may comprise a transaction engine

connected to route data from the multiple users to the depository server system, the authentication engine, and the cryptographic engine.

Another aspect of the invention includes a secure cryptographic system that is optionally remotely accessible. The cryptographic system comprises a depository system having at least one server which stores at least one private key and any other data, such as, but not limited to, a plurality of enrollment authentication data, wherein each enrollment authentication data identifies one of possibly multiple users. The cryptographic system may also optionally comprise an authentication engine which compares authentication data received by users to enrollment authentication data corresponding to the user and received from the depository system, thereby producing an authentication result. The cryptographic system also comprises a cryptographic engine which, when the authentication result indicates proper identification of the user, performs cryptographic functions on behalf of the user using at least said private key, which may be received from the depository system. The cryptographic system may also optionally comprise a transaction engine connected to route data from the users to other engines or systems such as, but not limited to, the depository server system, the authentication engine, and the cryptographic engine.

Another aspect of the invention includes a method of facilitating cryptographic functions. The method comprises associating a user from multiple users with one or more keys from a plurality of private cryptographic keys stored on a secure location, such as a secure server. The method also comprises receiving authentication data from the user, and comparing the authentication data to authentication data corresponding to the user, thereby verifying the identity of the user. The method also comprises utilizing the one or more keys to perform cryptographic functions without releasing the one or more keys to the user.

Another aspect of the invention includes an authentication system for uniquely identifying a user through secure storage of the user's enrollment authentication data. The authentication system comprises one or more data storage facilities, wherein each data storage facility includes a computer accessible storage medium which stores at least one of portions of enrollment authentication data. The authentication system also comprises an authentication engine which communicates with the data storage facility

or facilities. The authentication engine comprises a data splitting module which operates on the enrollment authentication data to create portions, a data assembling module which processes the portions from at least one of the data storage facilities to assemble the enrollment authentication data, and a data comparator module which receives current authentication data from a user and compares the current authentication data with the assembled enrollment authentication data to determine whether the user has been uniquely identified.

Another aspect of the invention includes a cryptographic system. The cryptographic system comprises one or more data storage facilities, wherein each data storage facility includes a computer accessible storage medium which stores at least one portion of one ore more cryptographic keys. The cryptographic system also comprises a cryptographic engine which communicates with the data storage facilities. The cryptographic engine also comprises a data splitting module which operate on the cryptographic keys to create portions, a data assembling module which processes the portions from at least one of the data storage facilities to assemble the cryptographic keys, and a cryptographic handling module which receives the assembled cryptographic keys and performs cryptographic functions therewith.

Another aspect of the invention includes a method of storing any type of data, including, but not limited to, authentication data in geographically remote secure data storage facilities thereby protecting the data against composition of any individual data storage facility. The method comprises receiving data at a trust engine, combining at the trust engine the data with a first substantially random value to form a first combined value, and combining the data with a second substantially random value to form a second combined value. The method comprises creating a first pairing of the first substantially random value with the second combined value, creating a second pairing of the first substantially random value with the second substantially random value, and storing the first pairing in a first secure data storage facility. The method comprises storing the second pairing in a second secure data storage facility remote from the first secure data storage facility.

Another aspect of the invention includes a method of storing any type of data, including, but not limited to, authentication data comprising receiving data, combining

the data with a first set of bits to form a second set of bits, and combining the data with a third set of bits to form a fourth set of bits. The method also comprises creating a first pairing of the first set of bits with the third set of bits. The method also comprises creating a second pairing of the first set of bits with the fourth set of bits, and storing one of the first and second pairings in a first computer accessible storage medium. The method also comprises storing the other of the first and second pairings in a second computer accessible storage medium.

Another aspect of the invention includes a method of storing cryptographic data in geographically remote secure data storage facilities thereby protecting the cryptographic data against comprise of any individual data storage facility. The method comprises receiving cryptographic data at a trust engine, combining at the trust engine the cryptographic data with a first substantially random value to form a first combined value, and combining the cryptographic data with a second substantially random value to form a second combined value. The method also comprises creating a first pairing of the first substantially random value with the second combined value, creating a second pairing of the first substantially random value with the second substantially random value, and storing the first pairing in a first secure data storage facility. The method also comprises storing the second pairing in a secure second data storage facility remote from the first secure data storage facility.

Another aspect of the invention includes a method of storing cryptographic data comprising receiving authentication data and combining the cryptographic data with a first set of bits to form a second set of bits. The method also comprises combining the cryptographic data with a third set of bits to form a fourth set of bits, creating a first pairing of the first set of bits with the third set of bits, and creating a second pairing of the first set of bits with the fourth set of bits. The method also comprises storing one of the first and second pairings in a first computer accessible storage medium, and storing the other of the first and second pairings in a second computer accessible storage medium.

Another aspect of the invention includes a method of handling sensitive data of any type or form in a cryptographic system, wherein the sensitive data exists in a useable form only during actions by authorized users, employing the sensitive data.

The method also comprises receiving in a software module, substantially randomized or encrypted sensitive data from a first computer accessible storage medium, and receiving in the software module, substantially randomized or encrypted data which may or may not be sensitive data, from one or more other computer accessible storage medium. The method also comprises processing the substantially randomized pre-encrypted sensitive data and the substantially randomized or encrypted data which may or may not be sensitive data, in the software module to assemble the sensitive data and employing the sensitive data in a software engine to perform an action. The action includes, but is not limited to, one of authenticating a user and performing a cryptographic function.

Another aspect of the invention includes a secure authenticating system. The secure authentication system comprises a plurality of authentication engines. Each authentication engine receives enrolment authentication data designed to uniquely identify a user to a degree of certainty. Each authentication engine receives current authentication data to compare to the enrolment authentication data, and each authentication engine determines an authentication results. The secure authentication system also comprises a redundancy system which receives the authentication result of at least two of the authentication engines and determines whether the user has been uniquely identified.

Definitions of the specific embodiments of the invention as claimed herein follow.

According to a first embodiment of the present invention there is provided a method for securing data. The method comprises the steps of:

a) encrypting a data set to provide an encrypted data set;

b) generating two or more portions of data from the encrypted data set. The generating comprises: splitting the data set into a number of data units, generating random or pseudo-random numbers, associating the random or pseudo-random numbers with at least two shares, associating the random or pseudo-random numbers with the data units, determining into which of the at least two shares to store each data unit according to the association of the random or pseudo-random numbers with the at least two shares and with the data units, and storing, using electronic storage, the data units and data indicative of at least a portion of the random or pseudo-random numbers in the at least two shares according to the determination.

The two or more portions of data each contain a randomized or pseudo-randomized distribution of the encrypted data set; and

c) encrypting one or more of the portions of data from step b), whereby the data set is restorable from at least two of the two or more portions of data from step b). Restoring the data set comprises: decrypting the one or more portions of data from step c), reconstituting the encrypted data set by recombining data from the at least two of the two or more portions of data that was randomly or pseudo-randomly distributed in step b), and decrypting the encrypted data set into the data set.

According to a second embodiment of the present invention there is provided a method for securing data. The method comprises the steps of:

a) generating two or more portions of data from a data set. The generating comprises: splitting the data set into a number of data units, generating random or pseudo-random numbers, associating the random or pseudo-random numbers with at least two shares, associating the random or pseudo-random numbers with the data units, determining into which of the at least two shares to store each data unit according to the association of the random or pseudo-random numbers with the at least two shares and with the data units, and storing, using electronic storage, the data units and data indicative of at least a portion of the random or pseudo-random numbers in the at least two shares according to the determination. The two or more portions of data each contain a random or pseudo-random distribution of the data set; and

b) encrypting one or more of the portions of data of step a), whereby the data set is restorable from at least two of the two or more portions of data. Restoring the data set comprises: decrypting the one or more portions of data from step b), and reconstituting the data set by recombining data from the at least two of the two or more portions of data that was randomly or pseudo-randomly distributed in step a).

According to a third embodiment of the present invention there is provided a method for securing data. The method comprises the steps of:

a) generating an encryption master key and encrypting a data set using the encryption master key;

b) generating two or more portions of data from the encrypted data set and the encryption master key according to one separating pattern. Generating the two or more

portions comprises: splitting the data set into a number of data units, generating random or pseudo-random numbers, associating the random or pseudo-random numbers with at least two shares, associating the random or pseudo-random numbers with the data units, determining into which of the at least two shares to store each data unit according to the association of the random or pseudo-random numbers with the at least two shares and with the data units, storing, using electronic storage, the data units and data indicative of at least a portion of the random or pseudo-random numbers in the at least two shares according to the determination, and appending an encryption master key portion to the at least two shares, wherein the two or more portions comprise a random or pseudo-random distribution of data from the encrypted data set; and

c) generating one or more encryption keys for the portions of data from step b) and encrypting said portions of data using said one or more encryption keys, whereby the data set is restorable from at least two portions of the two or more portions of data. Restoring the data set comprises: decrypting the encrypted portions of data, reconstituting the encrypted data set by recombining data from the at least two portions of the two or more portions of data that was randomly or pseudo-randomly distributed in step b), and decrypting the encrypted data set into the data set.

According to a fourth embodiment of the present invention there is provided a method for securing data. The method comprises the steps of:

a) generating an encryption master key and encrypting a data set using the encryption master key;

b) generating two or more portions of data from the encrypted data set and the encryption master key according to one separating pattern. Generating the two or more portions comprises: splitting the data set into a number of data units, generating random or pseudo-random numbers, associating the random or pseudo-random numbers with at least two shares, associating the random or pseudo-random numbers with the data units, determining into which of the at least two shares to store each data unit according to the association of the random or pseudo-random numbers with the at least two shares and with the data units, storing, using electronic storage, the data units and data indicative of at least a portion of the random or pseudo-random numbers in the at least two shares according to the determination, and storing

encryption master key portions in the at least two shares, wherein the two or more portions comprise a random or pseudo-random distribution of data from the encrypted data set; and

c) generating one or more encryption keys for the encrypted data set portions of step b) and encrypting said portions of data using said encryption key, whereby the data set is restorable from at least two portions of the two or more portions of data. Restoring the data set comprises: decrypting the encrypted portions from step c), reconstituting the encrypted data set by recombining data from the at least two portions of the two or more portions of data that was randomly or pseudo-randomly distributed in step b), and decrypting the encrypted data set into the data set.

According to a fifth embodiment of the present invention there is provided a method for securing data. The method comprises the steps of:

a) encrypting a data set to provide an encrypted data set;

b) generating two or more portions of data from the encrypted data set according to the contents of a unique key value. The generating comprises: splitting the data set into a number of data units, generating the unique key values, associating the unique key values with at least two shares, associating the unique key values with the data units, determining into which of the at least two shares to store each data unit according to the association of the unique key values with the at least two shares and with the data units, and storing, using electronic storage, the data units and data indicative of at least a portion of the unique key values in the at least two shares according to the determination. The encrypted data set is randomly or pseudo-randomly distributed among the two or more portions of data; and

c) encrypting one or more of the portions of data from step b), whereby the data set is restorable from at least a subset of the portions of data. Restoring the data set comprises:

decrypting the one or more portions of data from step c), reconstituting the encrypted data set by recombining data from the at least two of the two or more portions of data that was randomly or pseudo-randomly distributed in step b), and decrypting the encrypted data set into the data set.

According to a sixth embodiment of the present invention there is provided a method for securing data. The method comprises the steps of:

a) splitting a data set into N number of data units;

b) selecting X number of shares for data unit storage;

c) generating N number of random or pseudo-random numbers that correspond to the X number of shares;

d) assigning the random or pseudo-random numbers to the data units; and

e) storing, using electronic storage, the data units and data indicative of at least a portion of the random or pseudo-random numbers in the shares that correspond to the random or pseudo-random numbers, whereby the data set is restorable from at least a subset of the X number of shares. Restoring the data set comprises reconstituting the data set by recombining the data units from the at least a subset of the X number of shares according to the substantially random numbers.

According to a seventh embodiment of the present invention there is provided a method for securing a data set. The method comprises the step of generating at least two portions of data from the data set. The generating comprises: splitting the data set into a number of data units, generating random or pseudo-random numbers, associating the random or pseudo-random numbers with at least two shares, associating the random or pseudo-random numbers with the data units, determining into which of the at least two shares to store each data unit according to the association of the random or pseudo-random numbers with the at least two shares and with the data units, and storing, using electronic storage, the data units and data indicative of at least a portion of the random or pseudo-random numbers in the at least two shares according to the determination. Each of the at least two portions of data respectively contains a random or pseudo-random distribution of a respective subset of the data set, whereby the data set is restorable from at least two portions of the at least two portions of data by recombining data from the at least two portions of the at least two portions of data that was substantially randomly distributed.

According to an eighth embodiment of the present invention there is provided a method for securing a data set. The method comprises the steps of:

a) randomly or pseudo-randomly selecting a first group of data units from the data set;

b) randomly or pseudo-randomly selecting a second group of data units from the data set. The randomly or pseudo-randomly selecting the first group of data units and the second group of data units comprises: generating random or pseudo-random numbers, associating the

random or pseudo-random numbers with at least two shares, associating the random or pseudo-random numbers with data units in the first group of data units and the second group of data units, and determining into which of the at least two shares to store each data unit according to the association of the random or pseudo-random numbers with the at least two shares and with the data units. Each of the first group of the data units and the second group of the data units contains less than all of the data units in the data set; and

c) storing the first group of data units and the second group of data units separately in the at least two shares. The data set is restorable from at least a portion of the first group of data units and at least a portion of the second group of data units.

Brief Description of the Drawings

The present invention is described in more detail below in connection with the attached drawings, which are meant to illustrate and not to limit the invention, and in which:

FIGURE 1 illustrates a block diagram of a cryptographic system, according to aspect of an embodiment of the invention;

FIGURE 2 illustrates a block diagram of the trust engine of FIGURE 1, according to aspect of an embodiment of the invention;

FIGURE 3 illustrates a block diagram of the transaction engine of FIGURE 2, according to aspect of an embodiment of the invention;

FIGURE 4 illustrates a block diagram of the depository of FIGURE 2, according to aspects of an embodiment of the invention;

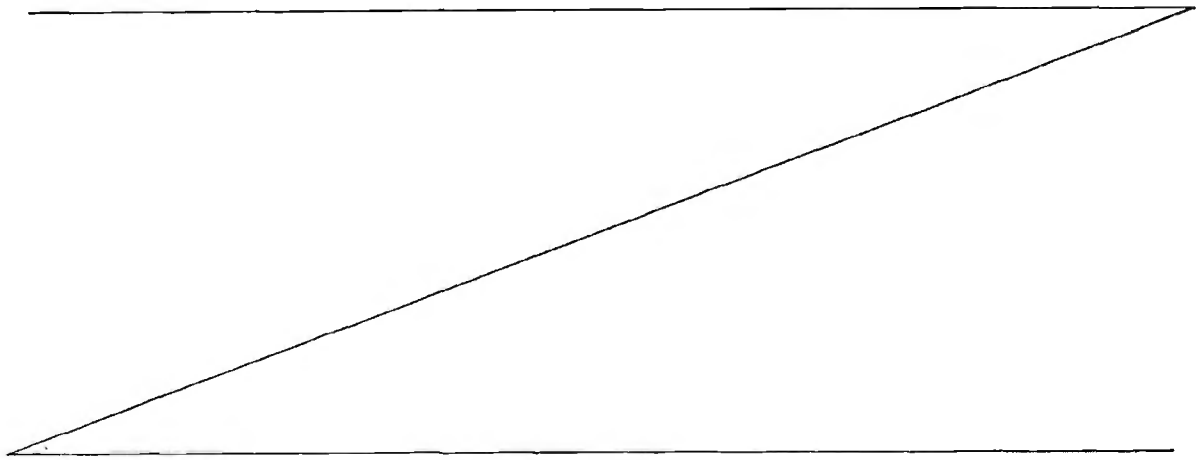


FIGURE 5 illustrates a block diagram of the authentication engine of FIGURE 2, according to aspects of an embodiment of the invention;

FIGURE 6 illustrates a block diagram of the cryptographic engine of FIGURE 2, according to aspects of an embodiment of the invention;

FIGURE 7 illustrates a block diagram of a depository system, according to aspects of another embodiment of the invention;

FIGURE 8 illustrates a flow chart of a data splitting process according to aspects of an embodiment of the invention;

FIGURE 9, Panel A illustrates a data flow of an enrollment process according to aspects of an embodiment of the invention;

FIGURE 9, Panel B illustrates a flow chart of an interoperability process according to aspects of an embodiment of the invention;

FIGURE 10 illustrates a data flow of an authentication process according to aspects of an embodiment of the invention;

FIGURE 11 illustrates a data flow of a signing process according to aspects of an embodiment of the invention;

FIGURE 12 illustrates a data flow and an encryption/decryption process according to aspects and yet another embodiment of the invention;

FIGURE 13 illustrates a simplified block diagram of a trust engine system according to aspects of another embodiment of the invention;

FIGURE 14 illustrates a simplified block diagram of a trust engine system according to aspects of another embodiment of the invention;

FIGURE 15 illustrates a block diagram of the redundancy module of FIGURE 14, according to aspects of an embodiment of the invention;

FIGURE 16 illustrates a process for evaluating authentications according to one aspect of the invention;

FIGURE 17 illustrates a process for assigning a value to an authentication according to one aspect as shown in FIGURE 16 of the invention;

FIGURE 18 illustrates a process for performing trust arbitrage in an aspect of the invention as shown in FIGURE 17; and

EX1010, Page 2304

FIGURE 19 illustrates a sample transaction between a user and a vendor according to aspects of an embodiment of the invention where an initial web based contact leads to a sales contract signed by both parties.

FIGURE 20 illustrates a sample user system with a cryptographic service provider module which provides security functions to a user system.

FIGURE 21 illustrates a process for parsing, splitting or separating data with encryption and storage of the encryption master key with the data.

FIGURE 22 illustrates a process for parsing, splitting or separating data with encryption and storing the encryption master key separately from the data.

FIGURE 23 illustrates the intermediary key process for parsing, splitting or separating data with encryption and storage of the encryption master key with the data.

FIGURE 24 illustrates the intermediary key process for parsing, splitting or separating data with encryption and storing the encryption master key separately from the data.

FIGURE 25 illustrates utilization of the cryptographic methods and systems of the present invention with a small working group.

Detailed Description of the Preferred Embodiment

One aspect of the present invention is to provide a cryptographic system where one or more secure servers, or a trust engine, stores cryptographic keys and user authentication data. Users access the functionality of conventional cryptographic systems through network access to the trust engine, however, the trust engine does not release actual keys and other authentication data and therefore, the keys and data remain secure. This server-centric storage of keys and authentication data provides for user-independent security, portability, availability, and straightforwardness.

Because users can be confident in, or trust, the cryptographic system to perform user and document authentication and other cryptographic functions, a wide variety of functionality may be incorporated into the system. For example, the trust engine provider can ensure against agreement repudiation by, for example, authenticating the agreement participants, digitally signing the agreement on behalf of or for the participants, and storing a record of the agreement digitally signed by each participant.

In addition, the cryptographic system may monitor agreements and determine to apply varying degrees of authentication, based on, for example, price, user, vendor, geographic location, place of use, or the like.

To facilitate a complete understanding of the invention, the remainder of the detailed description describes the invention with reference to the figures, wherein like elements are referenced with like numerals throughout.

FIGURE 1 illustrates a block diagram of a cryptographic system 100, according to aspects of an embodiment of the invention. As shown in FIGURE 1, the cryptographic system 100 includes a user system 105, a trust engine 110, a certificate authority 115, and a vendor system 120, communicating through a communication link 125.

According to one embodiment of the invention, the user system 105 comprises a conventional general-purpose computer having one or more microprocessors, such as, for example, an Intel-based processor. Moreover, the user system 105 includes an appropriate operating system, such as, for example, an operating system capable of including graphics or windows, such as Windows, Unix, Linux, or the like. As shown in FIGURE 1, the user system 105 may include a biometric device 107. The biometric device 107 may advantageously capture a user's biometric and transfer the captured biometric to the trust engine 110. According to one embodiment of the invention, the biometric device may advantageously comprise a device having attributes and features similar to those disclosed in U.S. Patent Application No. 08/926,277, filed on September 5, 1997, entitled *"RELIEF OBJECT IMAGE GENERATOR,"* U.S. Patent Application No. 09/558,634, filed on April 26, 2000, entitled *"IMAGING DEVICE FOR A RELIEF OBJECT AND SYSTEM AND METHOD OF USING THE IMAGE DEVICE,"* U.S. Patent Application No. 09/435,011, filed on November 5, 1999, entitled *"RELIEF OBJECT SENSOR ADAPTOR,"* and U.S. Patent Application No. 09/477,943, filed on January 5, 2000, entitled *"PLANAR OPTICAL IMAGE SENSOR AND SYSTEM FOR GENERATING AN ELECTRONIC IMAGE OF A RELIEF OBJECT FOR FINGERPRINT READING,"* all of which are owned by the instant assignee, and all of which are hereby incorporated by reference herein.

In addition, the user system 105 may connect to the communication link 125 through a conventional service provider, such as, for example, a dial up, digital subscriber line (DSL), cable modem, fiber connection, or the like. According to another embodiment, the user system 105 connects the communication link 125 through network connectivity such as, for example, a local or wide area network. According to one embodiment, the operating system includes a TCP/IP stack that handles all incoming and outgoing message traffic passed over the communication link 125.

Although the user system 105 is disclosed with reference to the foregoing embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize from the disclosure herein, a wide number of alternatives embodiments of the user system 105, including almost any computing device capable of sending or receiving information from another computer system. For example, the user system 105 may include, but is not limited to, a computer workstation, an interactive television, an interactive kiosk, a personal mobile computing device, such as a digital assistant, mobile phone, laptop, or the like, a wireless communications device, a smartcard, an embedded computing device, or the like, which can interact with the communication link 125. In such alternative systems, the operating systems will likely differ and be adapted for the particular device. However, according to one embodiment, the operating systems advantageously continue to provide the appropriate communications protocols needed to establish communication with the communication link 125.

FIGURE 1 illustrates the trust engine 110. According to one embodiment, the trust engine 110 comprises one or more secure servers for accessing and storing sensitive information, which may be any type or form of data, such as, but not limited to text, audio, video, user authentication data and public and private cryptographic keys. According to one embodiment, the authentication data includes data designed to uniquely identify a user of the cryptographic system 100. For example, the authentication data may include a user identification number, one or more biometrics, and a series of questions and answers generated by the trust engine 110 or the user, but answered initially by the user at enrollment. The foregoing questions may include demographic data, such as place of birth, address, anniversary, or the like, personal

data, such as mother's maiden name, favorite ice cream, or the like, or other data designed to uniquely identify the user. The trust engine 110 compares a user's authentication data associated with a current transaction, to the authentication data provided at an earlier time, such as, for example, during enrollment. The trust engine 110 may advantageously require the user to produce the authentication data at the time of each transaction, or, the trust engine 110 may advantageously allow the user to periodically produce authentication data, such as at the beginning of a string of transactions or the logging onto a particular vendor website.

According to the embodiment where the user produces biometric data, the user provides a physical characteristic, such as, but not limited to, facial scan, hand scan, ear scan, iris scan, retinal scan, vascular pattern, DNA, a fingerprint, writing or speech, to the biometric device 107. The biometric device advantageously produces an electronic pattern, or biometric, of the physical characteristic. The electronic pattern is transferred through the user system 105 to the trust engine 110 for either enrollment or authentication purposes.

Once the user produces the appropriate authentication data and the trust engine 110 determines a positive match between that authentication data (current authentication data) and the authentication data provided at the time of enrollment (enrollment authentication data), the trust engine 110 provides the user with complete cryptographic functionality. For example, the properly authenticated user may advantageously employ the trust engine 110 to perform hashing, digitally signing, encrypting and decrypting (often together referred to only as encrypting), creating or distributing digital certificates, and the like. However, the private cryptographic keys used in the cryptographic functions will not be available outside the trust engine 110, thereby ensuring the integrity of the cryptographic keys.

According to one embodiment, the trust engine 110 generates and stores cryptographic keys. According to another embodiment, at least one cryptographic key is associated with each user. Moreover, when the cryptographic keys include public-key technology, each private key associated with a user is generated within, and not released from, the trust engine 110. Thus, so long as the user has access to the trust engine 110, the user may perform cryptographic functions using his or her private or

public key. Such remote access advantageously allows users to remain completely mobile and access cryptographic functionality through practically any Internet connection, such as cellular and satellite phones, kiosks, laptops, hotel rooms and the like.

[0068] According to another embodiment, the trust engine 110 performs the cryptographic functionality using a key pair generated for the trust engine 110. According to this embodiment, the trust engine 110 first authenticates the user, and after the user has properly produced authentication data matching the enrollment authentication data, the trust engine 110 uses its own cryptographic key pair to perform cryptographic functions on behalf of the authenticated user.

A skilled artisan will recognize from the disclosure herein that the cryptographic keys may advantageously include some or all of symmetric keys, public keys, and private keys. In addition, a skilled artisan will recognize from the disclosure herein that the foregoing keys may be implemented with a wide number of algorithms available from commercial technologies, such as, for example, RSA, ELGAMAL, or the like.

FIGURE 1 also illustrates the certificate authority 115. According to one embodiment, the certificate authority 115 may advantageously comprise a trusted third-party organization or company that issues digital certificates, such as, for example, VeriSign, Baltimore, Entrust, or the like. The trust engine 110 may advantageously transmit requests for digital certificates, through one or more conventional digital certificate protocols, such as, for example, PKCS10, to the certificate authority 115. In response, the certificate authority 115 will issue a digital certificate in one or more of a number of differing protocols, such as, for example, PKCS7. According to one embodiment of the invention, the trust engine 110 requests digital certificates from several or all of the prominent certificate authorities 115 such that the trust engine 110 has access to a digital certificate corresponding to the certificate standard of any requesting party.

According to another embodiment, the trust engine 110 internally performs certificate issuances. In this embodiment, the trust engine 110 may access a certificate system for generating certificates and/or may internally generate certificates when they are requested, such as, for example, at the time of key generation or in the certificate

standard requested at the time of the request. The trust engine 110 will be disclosed in greater detail below.

FIGURE 1 also illustrates the vendor system 120. According to one embodiment, the vendor system 120 advantageously comprises a Web server. Typical Web servers generally serve content over the Internet using one of several internet markup languages or document format standards, such as the Hyper-Text Markup Language (HTML) or the Extensible Markup Language (XML). The Web server accepts requests from browsers like Netscape and Internet Explorer and then returns the appropriate electronic documents. A number of server or client-side technologies can be used to increase the power of the Web server beyond its ability to deliver standard electronic documents. For example, these technologies include Common Gateway Interface (CGI) scripts, Secure Sockets Layer (SSL) security, and Active Server Pages (ASPs). The vendor system 120 may advantageously provide electronic content relating to commercial, personal, educational, or other transactions.

Although the vendor system 120 is disclosed with reference to the foregoing embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize from the disclosure herein that the vendor system 120 may advantageously comprise any of the devices described with reference to the user system 105 or combination thereof.

FIGURE 1 also illustrates the communication link 125 connecting the user system 105, the trust engine 110, the certificate authority 115, and the vendor system 120. According to one embodiment, the communication link 125 preferably comprises the Internet. The Internet, as used throughout this disclosure is a global network of computers. The structure of the Internet, which is well known to those of ordinary skill in the art, includes a network backbone with networks branching from the backbone. These branches, in turn, have networks branching from them, and so on. Routers move information packets between network levels, and then from network to network, until the packet reaches the neighborhood of its destination. From the destination, the destination network's host directs the information packet to the appropriate terminal, or node. In one advantageous embodiment, the Internet routing hubs comprise domain name system (DNS) servers using Transmission Control Protocol/Internet Protocol

(TCP/IP) as is well known in the art. The routing hubs connect to one or more other routing hubs via high-speed communication links.

One popular part of the Internet is the World Wide Web. The World Wide Web contains different computers, which store documents capable of displaying graphical and textual information. The computers that provide information on the World Wide Web are typically called "websites." A website is defined by an Internet address that has an associated electronic page. The electronic page can be identified by a Uniform Resource Locator (URL). Generally, an electronic page is a document that organizes the presentation of text, graphical images, audio, video, and so forth.

Although the communication link 125 is disclosed in terms of its preferred embodiment, one of ordinary skill in the art will recognize from the disclosure herein that the communication link 125 may include a wide range of interactive communications links. For example, the communication link 125 may include interactive television networks, telephone networks, wireless data transmission systems, two-way cable systems, customized private or public computer networks, interactive kiosk networks, automatic teller machine networks, direct links, satellite or cellular networks, and the like.

FIGURE 2 illustrates a block diagram of the trust engine 110 of FIGURE 1 according to aspects of an embodiment of the invention. As shown in FIGURE 2, the trust engine 110 includes a transaction engine 205, a depository 210, an authentication engine 215, and a cryptographic engine 220. According to one embodiment of the invention, the trust engine 110 also includes mass storage 225. As further shown in FIGURE 2, the transaction engine 205 communicates with the depository 210, the authentication engine 215, and the cryptographic engine 220, along with the mass storage 225. In addition, the depository 210 communicates with the authentication engine 215, the cryptographic engine 220, and the mass storage 225. Moreover, the authentication engine 215 communicates with the cryptographic engine 220. According to one embodiment of the invention, some or all of the foregoing communications may advantageously comprise the transmission of XML documents to IP addresses that correspond to the receiving device. As mentioned in the foregoing, XML documents advantageously allow designers to create their own customized document tags, enabling

the definition, transmission, validation, and interpretation of data between applications and between organizations. Moreover, some or all of the foregoing communications may include conventional SSL technologies.

According to one embodiment, the transaction engine 205 comprises a data routing device, such as a conventional Web server available from Netscape, Microsoft, Apache, or the like. For example, the Web server may advantageously receive incoming data from the communication link 125. According to one embodiment of the invention, the incoming data is addressed to a front-end security system for the trust engine 110. For example, the front-end security system may advantageously include a firewall, an intrusion detection system searching for known attack profiles, and/or a virus scanner. After clearing the front-end security system, the data is received by the transaction engine 205 and routed to one of the depository 210, the authentication engine 215, the cryptographic engine 220, and the mass storage 225. In addition, the transaction engine 205 monitors incoming data from the authentication engine 215 and cryptographic engine 220, and routes the data to particular systems through the communication link 125. For example, the transaction engine 205 may advantageously route data to the user system 105, the certificate authority 115, or the vendor system 120.

According to one embodiment, the data is routed using conventional HTTP routing techniques, such as, for example, employing URLs or Uniform Resource Indicators (URIs). URIs are similar to URLs, however, URIs typically indicate the source of files or actions, such as, for example, executables, scripts, and the like. Therefore, according to the one embodiment, the user system 105, the certificate authority 115, the vendor system 120, and the components of the trust engine 210, advantageously include sufficient data within communication URLs or URIs for the transaction engine 205 to properly route data throughout the cryptographic system.

Although the data routing is disclosed with reference to its preferred embodiment, a skilled artisan will recognize a wide number of possible data routing solutions or strategies. For example, XML or other data packets may advantageously be unpacked and recognized by their format, content, or the like, such that the transaction engine 205 may properly route data throughout the trust engine 110.

Moreover, a skilled artisan will recognize that the data routing may advantageously be adapted to the data transfer protocols conforming to particular network systems, such as, for example, when the communication link 125 comprises a local network.

According to yet another embodiment of the invention, the transaction engine 205 includes conventional SSL encryption technologies, such that the foregoing systems may authenticate themselves, and vise-versa, with transaction engine 205, during particular communications. As will be used throughout this disclosure, the term "½ SSL" refers to communications where a server but not necessarily the client, is SSL authenticated, and the term "FULL SSL" refers to communications where the client and the server are SSL authenticated. When the instant disclosure uses the term "SSL", the communication may comprise ½ or FULL SSL.

As the transaction engine 205 routes data to the various components of the cryptographic system 100, the transaction engine 205 may advantageously create an audit trail. According to one embodiment, the audit trail includes a record of at least the type and format of data routed by the transaction engine 205 throughout the cryptographic system 100. Such audit data may advantageously be stored in the mass storage 225.

FIGURE 2 also illustrates the depository 210. According to one embodiment, the depository 210 comprises one or more data storage facilities, such as, for example, a directory server, a database server, or the like. As shown in FIGURE 2, the depository 210 stores cryptographic keys and enrollment authentication data. The cryptographic keys may advantageously correspond to the trust engine 110 or to users of the cryptographic system 100, such as the user or vendor. The enrollment authentication data may advantageously include data designed to uniquely identify a user, such as, user ID, passwords, answers to questions, biometric data, or the like. This enrollment authentication data may advantageously be acquired at enrollment of a user or another alternative later time. For example, the trust engine 110 may include periodic or other renewal or reissue of enrollment authentication data.

According to one embodiment, the communication from the transaction engine 205 to and from the authentication engine 215 and the cryptographic engine 220 comprises secure communication, such as, for example conventional SSL technology.

In addition, as mentioned in the foregoing, the data of the communications to and from the depository 210 may be transferred using URLs, URIs, HTTP or XML documents, with any of the foregoing advantageously having data requests and formats embedded therein.

As mentioned above, the depository 210 may advantageously comprises a plurality of secure data storage facilities. In such an embodiment, the secure data storage facilities may be configured such that a compromise of the security in one individual data storage facility will not compromise the cryptographic keys or the authentication data stored therein. For example, according to this embodiment, the cryptographic keys and the authentication data are mathematically operated on so as to statistically and substantially randomize the data stored in each data storage facility. According to one embodiment, the randomization of the data of an individual data storage facility renders that data undecipherable. Thus, compromise of an individual data storage facility produces only a randomized undecipherable number and does not compromise the security of any cryptographic keys or the authentication data as a whole.

FIGURE 2 also illustrates the trust engine 110 including the authentication engine 215. According to one embodiment, the authentication engine 215 comprises a data comparator configured to compare data from the transaction engine 205 with data from the depository 210. For example, during authentication, a user supplies current authentication data to the trust engine 110 such that the transaction engine 205 receives the current authentication data. As mentioned in the foregoing, the transaction engine 205 recognizes the data requests, preferably in the URL or URI, and routes the authentication data to the authentication engine 215. Moreover, upon request, the depository 210 forwards enrollment authentication data corresponding to the user to the authentication engine 215. Thus, the authentication engine 215 has both the current authentication data and the enrollment authentication data for comparison.

According to one embodiment, the communications to the authentication engine comprise secure communications, such as, for example, SSL technology. Additionally, security can be provided within the trust engine 110 components, such as, for example, super-encryption using public key technologies. For example, according to one

embodiment, the user encrypts the current authentication data with the public key of the authentication engine 215. In addition, the depository 210 also encrypts the enrollment authentication data with the public key of the authentication engine 215. In this way, only the authentication engine's private key can be used to decrypt the transmissions.

As shown in FIGURE 2, the trust engine 110 also includes the cryptographic engine 220. According to one embodiment, the cryptographic engine comprises a cryptographic handling module, configured to advantageously provide conventional cryptographic functions, such as, for example, public-key infrastructure (PKI) functionality. For example, the cryptographic engine 220 may advantageously issue public and private keys for users of the cryptographic system 100. In this manner, the cryptographic keys are generated at the cryptographic engine 220 and forwarded to the depository 210 such that at least the private cryptographic keys are not available outside of the trust engine 110. According to another embodiment, the cryptographic engine 220 randomizes and splits at least the private cryptographic key data, thereby storing only the randomized split data. Similar to the splitting of the enrollment authentication data, the splitting process ensures the stored keys are not available outside the cryptographic engine 220. According to another embodiment, the functions of the cryptographic engine can be combined with and performed by the authentication engine 215.

According to one embodiment, communications to and from the cryptographic engine include secure communications, such as SSL technology. In addition, XML documents may advantageously be employed to transfer data and/or make cryptographic function requests.

FIGURE 2 also illustrates the trust engine 110 having the mass storage 225. As mentioned in the foregoing, the transaction engine 205 keeps data corresponding to an audit trail and stores such data in the mass storage 225. Similarly, according to one embodiment of the invention, the depository 210 keeps data corresponding to an audit trail and stores such data in the mass storage device 225. The depository audit trail data is similar to that of the transaction engine 205 in that the audit trail data comprises a record of the requests received by the depository 210 and the response thereof. In

addition, the mass storage 225 may be used to store digital certificates having the public key of a user contained therein.

Although the trust engine 110 is disclosed with reference to its preferred and alternative embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize in the disclosure herein, a wide number of alternatives for the trust engine 110. For example, the trust engine 110, may advantageously perform only authentication, or alternatively, only some or all of the cryptographic functions, such as data encryption and decryption. According to such embodiments, one of the authentication engine 215 and the cryptographic engine 220 may advantageously be removed, thereby creating a more straightforward design for the trust engine 110. In addition, the cryptographic engine 220 may also communicate with a certificate authority such that the certificate authority is embodied within the trust engine 110. According to yet another embodiment, the trust engine 110 may advantageously perform authentication and one or more cryptographic functions, such as, for example, digital signing.

FIGURE 3 illustrates a block diagram of the transaction engine 205 of FIGURE 2, according to aspects of an embodiment of the invention. According to this embodiment, the transaction engine 205 comprises an operating system 305 having a handling thread and a listening thread. The operating system 305 may advantageously be similar to those found in conventional high volume servers, such as, for example, Web servers available from Apache. The listening thread monitors the incoming communication from one of the communication link 125, the authentication engine 215, and the cryptographic engine 220 for incoming data flow. The handling thread recognizes particular data structures of the incoming data flow, such as, for example, the foregoing data structures, thereby routing the incoming data to one of the communication link 125, the depository 210, the authentication engine 215, the cryptographic engine 220, or the mass storage 225. As shown in FIGURE 3, the incoming and outgoing data may advantageously be secured through, for example, SSL technology.

FIGURE 4 illustrates a block diagram of the depository 210 of FIGURE 2 according to aspects of an embodiment of the invention. According to this

embodiment, the depository 210 comprises one or more lightweight directory access protocol (LDAP) servers. LDAP directory servers are available from a wide variety of manufacturers such as Netscape, ISO, and others. FIGURE 4 also shows that the directory server preferably stores data 405 corresponding to the cryptographic keys and data 410 corresponding to the enrollment authentication data. According to one embodiment, the depository 210 comprises a single logical memory structure indexing authentication data and cryptographic key data to a unique user ID. The single logical memory structure preferably includes mechanisms to ensure a high degree of trust, or security, in the data stored therein. For example, the physical location of the depository 210 may advantageously include a wide number of conventional security measures, such as limited employee access, modern surveillance systems, and the like. In addition to, or in lieu of, the physical securities, the computer system or server may advantageously include software solutions to protect the stored data. For example, the depository 210 may advantageously create and store data 415 corresponding to an audit trail of actions taken. In addition, the incoming and outgoing communications may advantageously be encrypted with public key encryption coupled with conventional SSL technologies.

According to another embodiment, the depository 210 may comprise distinct and physically separated data storage facilities, as disclosed further with reference to FIGURE 7.

FIGURE 5 illustrates a block diagram of the authentication engine 215 of FIGURE 2 according to aspects of an embodiment of the invention. Similar to the transaction engine 205 of FIGURE 3, the authentication engine 215 comprises an operating system 505 having at least a listening and a handling thread of a modified version of a conventional Web server, such as, for example, Web servers available from Apache. As shown in FIGURE 5, the authentication engine 215 includes access to at least one private key 510. The private key 510 may advantageously be used for example, to decrypt data from the transaction engine 205 or the depository 210, which was encrypted with a corresponding public key of the authentication engine 215.

FIGURE 5 also illustrates the authentication engine 215 comprising a comparator 515, a data splitting module 520, and a data assembling module 525.

According to the preferred embodiment of the invention, the comparator 515 includes technology capable of comparing potentially complex patterns related to the foregoing biometric authentication data. The technology may include hardware, software, or combined solutions for pattern comparisons, such as, for example, those representing finger print patterns or voice patterns. In addition, according to one embodiment, the comparator 515 of the authentication engine 215 may advantageously compare conventional hashes of documents in order to render a comparison result. According to one embodiment of the invention, the comparator 515 includes the application of heuristics 530 to the comparison. The heuristics 530 may advantageously address circumstances surrounding an authentication attempt, such as, for example, the time of day, IP address or subnet mask, purchasing profile, email address, processor serial number or ID, or the like.

Moreover, the nature of biometric data comparisons may result in varying degrees of confidence being produced from the matching of current biometric authentication data to enrollment data. For example, unlike a traditional password which may only return a positive or negative match, a fingerprint may be determined to be a partial match, e.g. a 90% match, a 75% match, or a 10% match, rather than simply being correct or incorrect. Other biometric identifiers such as voice print analysis or face recognition may share this property of probabilistic authentication, rather than absolute authentication.

When working with such probabilistic authentication or in other cases where an authentication is considered less than absolutely reliable, it is desirable to apply the heuristics 530 to determine whether the level of confidence in the authentication provided is sufficiently high to authenticate the transaction which is being made.

It will sometimes be the case that the transaction at issue is a relatively low value transaction where it is acceptable to be authenticated to a lower level of confidence. This could include a transaction which has a low dollar value associated with it (e.g., a $10 purchase) or a transaction with low risk (e.g., admission to a members-only web site).

Conversely, for authenticating other transactions, it may be desirable to require a high degree of confidence in the authentication before allowing the transaction to

proceed. Such transactions may include transactions of large dollar value (e.g., signing a multi-million dollar supply contract) or transaction with a high risk if an improper authentication occurs (e.g., remotely logging onto a government computer).

The use of the heuristics 530 in combination with confidence levels and transactions values may be used as will be described below to allow the comparator to provide a dynamic context-sensitive authentication system.

According to another embodiment of the invention, the comparator 515 may advantageously track authentication attempts for a particular transaction. For example, when a transaction fails, the trust engine 110 may request the user to re-enter his or her current authentication data. The comparator 515 of the authentication engine 215 may advantageously employ an attempt limiter 535 to limit the number of authentication attempts, thereby prohibiting brute-force attempts to impersonate a user's authentication data. According to one embodiment, the attempt limiter 535 comprises a software module monitoring transactions for repeating authentication attempts and, for example, limiting the authentication attempts for a given transaction to three. Thus, the attempt limiter 535 will limit an automated attempt to impersonate an individual's authentication data to, for example, simply three "guesses." Upon three failures, the attempt limiter 535 may advantageously deny additional authentication attempts. Such denial may advantageously be implemented through, for example, the comparator 515 returning a negative result regardless of the current authentication data being transmitted. On the other hand, the transaction engine 205 may advantageously block any additional authentication attempts pertaining to a transaction in which three attempts have previously failed.

The authentication engine 215 also includes the data splitting module 520 and the data assembling module 525. The data splitting module 520 advantageously comprises a software, hardware, or combination module having the ability to mathematically operate on various data so as to substantially randomize and split the data into portions. According to one embodiment, original data is not recreatable from an individual portion. The data assembling module 525 advantageously comprises a software, hardware, or combination module configured to mathematically operate on the foregoing substantially randomized portions, such that the combination thereof

provides the original deciphered data. According to one embodiment, the authentication engine 215 employs the data splitting module 520 to randomize and split enrollment authentication data into portions, and employs the data assembling module 525 to reassemble the portions into usable enrollment authentication data.

FIGURE 6 illustrates a block diagram of the cryptographic engine 220 of the trust engine 200 of FIGURE 2 according to aspects of one embodiment of the invention. Similar to the transaction engine 205 of FIGURE 3, the cryptographic engine 220 comprises an operating system 605 having at least a listening and a handling thread of a modified version of a conventional Web server, such as, for example, Web servers available from Apache. As shown in FIGURE 6, the cryptographic engine 220 comprises a data splitting module 610 and a data assembling module 620 that function similar to those of FIGURE 5. However, according to one embodiment, the data splitting module 610 and the data assembling module 620 process cryptographic key data, as opposed to the foregoing enrollment authentication data. Although, a skilled artisan will recognize from the disclosure herein that the data splitting module 910 and the data splitting module 620 may be combined with those of the authentication engine 215.

The cryptographic engine 220 also comprises a cryptographic handling module 625 configured to perform one, some or all of a wide number of cryptographic functions. According to one embodiment, the cryptographic handling module 625 may comprise software modules or programs, hardware, or both. According to another embodiment, the cryptographic handling module 625 may perform data comparisons, data parsing, data splitting, data separating, data hashing, data encryption or decryption, digital signature verification or creation, digital certificate generation, storage, or requests, cryptographic key generation, or the like. Moreover, a skilled artisan will recognize from the disclosure herein that the cryptographic handling module 825 may advantageously comprises a public-key infrastructure, such as Pretty Good Privacy (PGP), an RSA-based public-key system, or a wide number of alternative key management systems. In addition, the cryptographic handling module 625 may perform public-key encryption, symmetric-key encryption, or both. In addition to the foregoing, the cryptographic handling module 625 may include one or more computer programs or

modules, hardware, or both, for implementing seamless, transparent, interoperability functions.

A skilled artisan will also recognize from the disclosure herein that the cryptographic functionality may include a wide number or variety of functions generally relating to cryptographic key management systems.

FIGURE 7 illustrates a simplified block diagram of a depository system 700 according to aspects of an embodiment of the invention. As shown in FIGURE 7, the depository system 700 advantageously comprises multiple data storage facilities, for example, data storage facilities D1, D2, D3, and D4. However, it is readily understood by those of ordinary skill in the art that the depository system may have only one data storage facility. According to one embodiment of the invention, each of the data storage facilities D1 through D4 may advantageously comprise some or all of the elements disclosed with reference to the depository 210 of FIGURE 4. Similar to the depository 210, the data storage facilities D1 through D4 communicate with the transaction engine 205, the authentication engine 215, and the cryptographic engine 220, preferably through conventional SSL. Communication links transferring, for example, XML documents. Communications from the transaction engine 205 may advantageously include requests for data, wherein the request is advantageously broadcast to the IP address of each data storage facility D1 through D4. On the other hand, the transaction engine 205 may broadcast requests to particular data storage facilities based on a wide number of criteria, such as, for example, response time, server loads, maintenance schedules, or the like.

In response to requests for data from the transaction engine 205, the depository system 700 advantageously forwards stored data to the authentication engine 215 and the cryptographic engine 220. The respective data assembling modules receive the forwarded data and assemble the data into useable formats. On the other hand, communications from the authentication engine 215 and the cryptographic engine 220 to the data storage facilities D1 through D4 may include the transmission of sensitive data to be stored. For example, according to one embodiment, the authentication engine 215 and the cryptographic engine 220 may advantageously employ their respective data splitting modules to divide sensitive data into undecipherable portions, and then

transmit one or more undecipherable portions of the sensitive data to a particular data storage facility.

According to one embodiment, each data storage facility, D1 through D4, comprises a separate and independent storage system, such as, for example, a directory server. According to another embodiment of the invention, the depository system 700 comprises multiple geographically separated independent data storage systems. By distributing the sensitive data into distinct and independent storage facilities D1 through D4, some or all of which may be advantageously geographically separated, the depository system 700 provides redundancy along with additional security measures. For example, according to one embodiment, only data from two of the multiple data storage facilities, D1 through D4, are needed to decipher and reassemble the sensitive data. Thus, as many as two of the four data storage facilities D1 through D4 may be inoperative due to maintenance, system failure, power failure, or the like, without affecting the functionality of the trust engine 110. In addition, because, according to one embodiment, the data stored in each data storage facility is randomized and undecipherable, compromise of any individual data storage facility does not necessarily compromise the sensitive data. Moreover, in the embodiment having geographical separation of the data storage facilities, a compromise of multiple geographically remote facilities becomes increasingly difficult. In fact, even a rogue employee will be greatly challenged to subvert the needed multiple independent geographically remote data storage facilities.

Although the depository system 700 is disclosed with reference to its preferred and alternative embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize from the disclosure herein, a wide number of alternatives for the depository system 700. For example, the depository system 700 may comprise one, two or more data storage facilities. In addition, sensitive data may be mathematically operated such that portions from two or more data storage facilities are needed to reassemble and decipher the sensitive data.

As mentioned in the foregoing, the authentication engine 215 and the cryptographic engine 220 each include a data splitting module 520 and 610, respectively, for splitting any type or form of sensitive data, such as, for example, text,

audio, video, the authentication data and the cryptographic key data. FIGURE 8 illustrates a flowchart of a data splitting process 800 performed by the data splitting module according to aspects of an embodiment of the invention. As shown in FIGURE 8, the data splitting process 800 begins at step 805 when sensitive data "S" is received by the data splitting module of the authentication engine 215 or the cryptographic engine 220. Preferably, in step 810, the data splitting module then generates a substantially random number, value, or string or set of bits, "A." For example, the random number A may be generated in a wide number of varying conventional techniques available to one of ordinary skill in the art, for producing high quality random numbers suitable for use in cryptographic applications. In addition, according to one embodiment, the random number A comprises a bit length which may be any suitable length, such as shorter, longer or equal to the bit length of the sensitive data, S.

In addition, in step 820 the data splitting process 800 generates another statistically random number "C." According to the preferred embodiment, the generation of the statistically random numbers A and C may advantageously be done in parallel. The data splitting module then combines the numbers A and C with the sensitive data S such that new numbers "B" and "D" are generated. For example, number B may comprise the binary combination of A XOR S and number D may comprise the binary combination of C XOR S. The XOR function, or the "exclusive-or" function, is well known to those of ordinary skill in the art. The foregoing combinations preferably occur in steps 825 and 830, respectively, and, according to one embodiment, the foregoing combinations also occur in parallel. The data splitting process 800 then proceeds to step 835 where the random numbers A and C and the numbers B and D are paired such that none of the pairings contain sufficient data, by themselves, to reorganize and decipher the original sensitive data S. For example, the numbers may be paired as follows: AC, AD, BC, and BD. According to one embodiment, each of the foregoing pairings is distributed to one of the depositories D1 through D4 of FIGURE 7. According to another embodiment, each of the foregoing pairings is randomly distributed to one of the depositories D1 through D4. For example, during a first data splitting process 800, the pairing AC may be sent to depository D2, through, for example, a random selection of D2's IP address. Then,

during a second data splitting process 800, the pairing AC may be sent to depository D4, through, for example, a random selection of D4's IP address. In addition, the pairings may all be stored on one depository, and may be stored in separate locations on said depository.

Based on the foregoing, the data splitting process 800 advantageously places portions of the sensitive data in each of the four data storage facilities D1 through D4, such that no single data storage facility D1 through D4 includes sufficient encrypted data to recreate the original sensitive data S. As mentioned in the foregoing, such randomization of the data into individually unusable encrypted portions increases security and provides for maintained trust in the data even if one of the data storage facilities, D1 through D4, is compromised.

Although the data splitting process 800 is disclosed with reference to its preferred embodiment, the invention is not intended to be limited thereby. Rather a skilled artisan will recognize from the disclosure herein, a wide number of alternatives for the data splitting process 800. For example, the data splitting process may advantageously split the data into two numbers, for example, random number A and number B and, randomly distribute A and B through two data storage facilities. Moreover, the data splitting process 800 may advantageously split the data among a wide number of data storage facilities through generation of additional random numbers. The data may be split into any desired, selected, predetermined, or randomly assigned size unit, including but not limited to, a bit, bits, bytes, kilobytes, megabytes or larger, or any combination or sequence of sizes. In addition, varying the sizes of the data units resulting from the splitting process may render the data more difficult to restore to a useable form, thereby increasing security of sensitive data. It is readily apparent to those of ordinary skill in the art that the split data unit sizes may be a wide variety of data unit sizes or patterns of sizes or combinations of sizes. For example, the data unit sizes may be selected or predetermined to be all of the same size, a fixed set of different sizes, a combination of sizes, or randomly generates sizes. Similarly, the data units may be distributed into one or more shares according to a fixed or predetermined data unit size, a pattern or combination of data unit sizes, or a randomly generated data unit size or sizes per share.

As mentioned in the foregoing, in order to recreate the sensitive data S, the data portions need to be derandomized and reorganized. This process may advantageously occur in the data assembling modules, 525 and 620, of the authentication engine 215 and the cryptographic engine 220, respectively. The data assembling module, for example, data assembly module 525, receives data portions from the data storage facilities D1 through D4, and reassembles the data into useable form. For example, according to one embodiment where the data splitting module 520 employed the data splitting process 800 of FIGURE 8, the data assembling module 525 uses data portions from at least two of the data storage facilities D1 through D4 to recreate the sensitive data S. For example, the pairings of AC, AD, BC, and BD, were distributed such that any two provide one of A and B, or, C and D. Noting that S = A XOR B or S = C XOR D indicates that when the data assembling module receives one of A and B, or, C and D, the data assembling module 525 can advantageously reassemble the sensitive data S. Thus, the data assembling module 525 may assemble the sensitive data S, when, for example, it receives data portions from at least the first two of the data storage facilities D1 through D4 to respond to an assemble request by the trust engine 110.

Based on the above data splitting and assembling processes, the sensitive data S exists in usable format only in a limited area of the trust engine 110. For example, when the sensitive data S includes enrollment authentication data, usable, nonrandomized enrollment authentication data is available only in the authentication engine 215. Likewise, when the sensitive data S includes private cryptographic key data, usable, nonrandomized private cryptographic key data is available only in the cryptographic engine 220.

Although the data splitting and assembling processes are disclosed with reference to their preferred embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize from the disclosure herein, a wide number of alternatives for splitting and reassembling the sensitive data S. For example, public-key encryption may be used to further secure the data at the data storage facilities D1 through D4. In addition, it is readily apparent to those of ordinary skill in the art that the data splitting module described herein is also a separate and distinct embodiment of the present invention that may be incorporated into, combined with or

otherwise made part of any pre-existing computer systems, software suites, database, or combinations thereof, or other embodiments of the present invention, such as the trust engine, authentication engine, and transaction engine disclosed and described herein.

FIGURE 9A illustrates a data flow of an enrollment process 900 according to aspects of an embodiment of the invention. As shown in FIGURE 9A, the enrollment process 900 begins at step 905 when a user desires to enroll with the trust engine 110 of the cryptographic system 100. According to this embodiment, the user system 105 advantageously includes a client-side applet, such as a Java-based, that queries the user to enter enrollment data, such as demographic data and enrollment authentication data. According to one embodiment, the enrollment authentication data includes user ID, password(s), biometric(s), or the like. According to one embodiment, during the querying process, the client-side applet preferably communicates with the trust engine 110 to ensure that a chosen user ID is unique. When the user ID is nonunique, the trust engine 110 may advantageously suggest a unique user ID. The client-side applet gathers the enrollment data and transmits the enrollment data, for example, through and XML document, to the trust engine 110, and in particular, to the transaction engine 205. According to one embodiment, the transmission is encoded with the public key of the authentication engine 215.

According to one embodiment, the user performs a single enrollment during step 905 of the enrollment process 900. For example, the user enrolls himself or herself as a particular person, such as Joe User. When Joe User desires to enroll as Joe User, CEO of Mega Corp., then according to this embodiment, Joe User enrolls a second time, receives a second unique user ID and the trust engine 110 does not associate the two identities. According to another embodiment of the invention, the enrollment process 900 provides for multiple user identities for a single user ID. Thus, in the above example, the trust engine 110 will advantageously associate the two identities of Joe User. As will be understood by a skilled artisan from the disclosure herein, a user may have many identities, for example, Joe User the head of household, Joe User the member of the Charitable Foundations, and the like. Even though the user may have multiple identities, according to this embodiment, the trust engine 110 preferably stores

only one set of enrollment data. Moreover, users may advantageously add, edit/update, or delete identities as they are needed.

Although the enrollment process 900 is disclosed with reference to its preferred embodiment, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize from the disclosure herein, a wide number of alternatives for gathering of enrollment data, and in particular, enrollment authentication data. For example, the applet may be common object model (COM) based applet or the like.

On the other hand, the enrollment process may include graded enrollment. For example, at a lowest level of enrollment, the user may enroll over the communication link 125 without producing documentation as to his or her identity. According to an increased level of enrollment, the user enrolls using a trusted third party, such as a digital notary. For example, and the user may appear in person to the trusted third party, produce credentials such as a birth certificate, driver's license, military ID, or the like, and the trusted third party may advantageously include, for example, their digital signature in enrollment submission. The trusted third party may include an actual notary, a government agency, such as the Post Office or Department of Motor Vehicles, a human resources person in a large company enrolling an employee, or the like. A skilled artisan will understand from the disclosure herein that a wide number of varying levels of enrollment may occur during the enrollment process 900.

After receiving the enrollment authentication data, at step 915, the transaction engine 205, using conventional FULL SSL technology forwards the enrollment authentication data to the authentication engine 215. In step 920, the authentication engine 215 decrypts the enrollment authentication data using the private key of the authentication engine 215. In addition, the authentication engine 215 employs the data splitting module to mathematically operate on the enrollment authentication data so as to split the data into at least two independently undecipherable, randomized, numbers. As mentioned in the foregoing, at least two numbers may comprise a statistically random number and a binary X0Red number. In step 925, the authentication engine 215 forwards each portion of the randomized numbers to one of the data storage facilities D1 through D4. As mentioned in the foregoing, the authentication engine 215

may also advantageously randomize which portions are transferred to which depositories.

Often during the enrollment process 900, the user will also desire to have a digital certificate issued such that he or she may receive encrypted documents from others outside the cryptographic system 100. As mentioned in the foregoing, the certificate authority 115 generally issues digital certificates according to one or more of several conventional standards. Generally, the digital certificate includes a public key of the user or system, which is known to everyone.

Whether the user requests a digital certificate at enrollment, or at another time, the request is transferred through the trust engine 110 to the authentication engine 215. According to one embodiment, the request includes an XML document having, for example, the proper name of the user. According to step 935, the authentication engine 215 transfers the request to the cryptographic engine 220 instructing the cryptographic engine 220 to generate a cryptographic key or key pair.

Upon request, at step 935, the cryptographic engine 220 generates at least one cryptographic key. According to one embodiment, the cryptographic handling module 625 generates a key pair, where one key is used as a private key, and one is used as a public key. The cryptographic engine 220 stores the private key and, according to one embodiment, a copy of the public key. In step 945, the cryptographic engine 220 transmits a request for a digital certificate to the transaction engine 205. According to one embodiment, the request advantageously includes a standardized request, such as PKCS10, embedded in, for example, an XML document. The request for a digital certificate may advantageously correspond to one or more certificate authorities and the one or more standard formats the certificate authorities require.

In step 950 the transaction engine 205 forwards this request to the certificate authority 115, who, in step 955, returns a digital certificate. The return digital certificate may advantageously be in a standardized format, such as PKCS7, or in a proprietary format of one or more of the certificate authorities 115. In step 960, the digital certificate is received by the transaction engine 205, and a copy is forwarded to the user and a copy is stored with the trust engine 110. The trust engine 110 stores a copy of the certificate such that the trust engine 110 will not need to rely on the

EX1010, Page 2328

availability of the certificate authority 115. For example, when the user desires to send a digital certificate, or a third party requests the user's digital certificate, the request for the digital certificate is typically sent to the certificate authority 115. However, if the certificate authority 115 is conducting maintenance or has been victim of a failure or security compromise, the digital certificate may not be available.

At any time after issuing the cryptographic keys, the cryptographic engine 220 may advantageously employ the data splitting process 800 described above such that the cryptographic keys are split into independently undecipherable randomized numbers. Similar to the authentication data, at step 965 the cryptographic engine 220 transfers the randomized numbers to the data storage facilities D1 through D4.

A skilled artisan will recognize from the disclosure herein that the user may request a digital certificate anytime after enrollment. Moreover, the communications between systems may advantageously include FULL SSL or public-key encryption technologies. Moreover, the enrollment process may issue multiple digital certificates from multiple certificate authorities, including one or more proprietary certificate authorities internal or external to the trust engine 110.

As disclosed in steps 935 through 960, one embodiment of the invention includes the request for a certificate that is eventually stored on the trust engine 110. Because, according to one embodiment, the cryptographic handling module 625 issues the keys used by the trust engine 110, each certificate corresponds to a private key. Therefore, the trust engine 110 may advantageously provide for interoperability through monitoring the certificates owned by, or associated with, a user. For example, when the cryptographic engine 220 receives a request for a cryptographic function, the cryptographic handling module 625 may investigate the certificates owned by the requesting user to determine whether the user owns a private key matching the attributes of the request. When such a certificate exists, the cryptographic handling module 625 may use the certificate or the public or private keys associated therewith, to perform the requested function. When such a certificate does not exist, the cryptographic handling module 625 may advantageously and transparently perform a number of actions to attempt to remedy the lack of an appropriate key. For example, FIGURE 9B illustrates a flowchart of an interoperability process 970, which according

to aspects of an embodiment of the invention, discloses the foregoing steps to ensure the cryptographic handling module 625 performs cryptographic functions using appropriate keys.

As shown in FIGURE 9B, the interoperability process 970 begins with step 972 where the cryptographic handling module 925 determines the type of certificate desired. According to one embodiment of the invention, the type of certificate may advantageously be specified in the request for cryptographic functions, or other data provided by the requestor. According to another embodiment, the certificate type may be ascertained by the data format of the request. For example, the cryptographic handling module 925 may advantageously recognize the request corresponds to a particular type.

According to one embodiment, the certificate type may include one or more algorithm standards, for example, RSA, ELGAMAL, or the like. In addition, the certificate type may include one or more key types, such as symmetric keys, public keys, strong encryption keys such as 256 bit keys, less secure keys, or the like. Moreover, the certificate type may include upgrades or replacements of one or more of the foregoing algorithm standards or keys, one or more message or data formats, one or more data encapsulation or encoding schemes, such as Base 32 or Base 64. The certificate type may also include compatibility with one or more third-party cryptographic applications or interfaces, one or more communication protocols, or one or more certificate standards or protocols. A skilled artisan will recognize from the disclosure herein that other differences may exist in certificate types, and translations to and from those differences may be implemented as disclosed herein.

Once the cryptographic handling module 625 determines the certificate type, the interoperability process 970 proceeds to step 974, and determines whether the user owns a certificate matching the type determined in step 974. When the user owns a matching certificate, for example, the trust engine 110 has access to the matching certificate through, for example, prior storage thereof, the cryptographic handling module 825 knows that a matching private key is also stored within the trust engine 110. For example, the matching private key may be stored within the depository 210 or depository system 700. The cryptographic handling module 625 may advantageously

request the matching private key be assembled from, for example, the depository 210, and then in step 976, use the matching private key to perform cryptographic actions or functions. For example, as mentioned in the foregoing, the cryptographic handling module 625 may advantageously perform hashing, hash comparisons, data encryption or decryption, digital signature verification or creation, or the like.

When the user does not own a matching certificate, the interoperability process 970 proceeds to step 978 where the cryptographic handling module 625 determines whether the users owns a cross-certified certificate. According to one embodiment, cross-certification between certificate authorities occurs when a first certificate authority determines to trust certificates from a second certificate authority. In other words, the first certificate authority determines that certificates from the second certificate authority meets certain quality standards, and therefore, may be "certified" as equivalent to the first certificate authority's own certificates. Cross-certification becomes more complex when the certificate authorities issue, for example, certificates having levels of trust. For example, the first certificate authority may provide three levels of trust for a particular certificate, usually based on the degree of reliability in the enrollment process, while the second certificate authority may provide seven levels of trust. Cross-certification may advantageously track which levels and which certificates from the second certificate authority may be substituted for which levels and which certificates from the first. When the foregoing cross-certification is done officially and publicly between two certification authorities, the mapping of certificates and levels to one another is often called "chaining."

According to another embodiment of the invention, the cryptographic handling module 625 may advantageously develop cross-certifications outside those agreed upon by the certificate authorities. For example, the cryptographic handling module 625 may access a first certificate authority's certificate practice statement (CPS), or other published policy statement, and using, for example, the authentication tokens required by particular trust levels, match the first certificate authority's certificates to those of another certificate authority.

When, in step 978, the cryptographic handling module 625 determines that the users owns a cross-certified certificate, the interoperability process 970 proceeds to step

976, and performs the cryptographic action or function using the cross-certified public key, private key, or both. Alternatively, when the cryptographic handling module 625 determines that the users does not own a cross-certified certificate, the interoperability process 970 proceeds to step 980, where the cryptographic handling module 625 selects a certificate authority that issues the requested certificate type, or a certificate cross-certified thereto. In step 982, the cryptographic handling module 625 determines whether the user enrollment authentication data, discussed in the foregoing, meets the authentication requirements of the chosen certificate authority. For example, if the user enrolled over a network by, for example, answering demographic and other questions, the authentication data provided may establish a lower level of trust than a user providing biometric data and appearing before a third-party, such as, for example, a notary. According to one embodiment, the foregoing authentication requirements may advantageously be provided in the chosen authentication authority's CPS.

When the user has provided the trust engine 110 with enrollment authentication data meeting the requirements of chosen certificate authority, the interoperability process 970 proceeds to step 984, where the cryptographic handling module 825 acquires the certificate from the chosen certificate authority. According to one embodiment, the cryptographic handling module 625 acquires the certificate by following steps 945 through 960 of the enrollment process 900. For example, the cryptographic handling module 625 may advantageously employ one or more public keys from one or more of the key pairs already available to the cryptographic engine 220, to request the certificate from the certificate authority. According to another embodiment, the cryptographic handling module 625 may advantageously generate one or more new key pairs, and use the public keys corresponding thereto, to request the certificate from the certificate authority.

According to another embodiment, the trust engine 110 may advantageously include one or more certificate issuing modules capable of issuing one or more certificate types. According to this embodiment, the certificate issuing module may provide the foregoing certificate. When the cryptographic handling module 625 acquires the certificate, the interoperability process 970 proceeds to step 976, and

performs the cryptographic action or function using the public key, private key, or both corresponding to the acquired certificate.

When the user, in step 982, has not provided the trust engine 110 with enrollment authentication data meeting the requirements of chosen certificate authority, the cryptographic handling module 625 determines, in step 986 whether there are other certificate authorities that have different authentication requirements. For example, the cryptographic handling module 625 may look for certificate authorities having lower authentication requirements, but still issue the chosen certificates, or cross-certifications thereof.

When the foregoing certificate authority having lower requirements exists, the interoperability process 970 proceeds to step 980 and chooses that certificate authority. Alternatively, when no such certificate authority exists, in step 988, the trust engine 110 may request additional authentication tokens from the user. For example, the trust engine 110 may request new enrollment authentication data comprising, for example, biometric data. Also, the trust engine 110 may request the user appear before a trusted third party and provide appropriate authenticating credentials, such as, for example, appearing before a notary with a drivers license, social security card, bank card, birth certificate, military ID, or the like. When the trust engine 110 receives updated authentication data, the interoperability process 970 proceeds to step 984 and acquires the foregoing chosen certificate.

Through the foregoing interoperability process 970, the cryptographic handling module 625 advantageously provides seamless, transparent, translations and conversions between differing cryptographic systems. A skilled artisan will recognize from the disclosure herein, a wide number of advantages and implementations of the foregoing interoperable system. For example, the foregoing step 986 of the interoperability process 970 may advantageously include aspects of trust arbitrage, discussed in further detail below, where the certificate authority may under special circumstances accept lower levels of cross-certification. In addition, the interoperability process 970 may include ensuring interoperability between and employment of standard certificate revocations, such as employing certificate revocation lists (CRL), online certificate status protocols (OCSP), or the like.

EX1010, Page 2333

FIGURE 10 illustrates a data flow of an authentication process 1000 according to aspects of an embodiment of the invention. According to one embodiment, the authentication process 1000 includes gathering current authentication data from a user and comparing that to the enrollment authentication data of the user. For example, the authentication process 1000 begins at step 1005 where a user desires to perform a transaction with, for example, a vendor. Such transactions may include, for example, selecting a purchase option, requesting access to a restricted area or device of the vendor system 120, or the like. At step 1010, a vendor provides the user with a transaction ID and an authentication request. The transaction ID may advantageously include a 192 bit quantity having a 32 bit timestamp concatenated with a 128 bit random quantity, or a "nonce," concatenated with a 32 bit vendor specific constant. Such a transaction ID uniquely identifies the transaction such that copycat transactions can be refused by the trust engine 110.

The authentication request may advantageously include what level of authentication is needed for a particular transaction. For example, the vendor may specify a particular level of confidence that is required for the transaction at issue. If authentication cannot be made to this level of confidence, as will be discussed below, the transaction will not occur without either further authentication by the user to raise the level of confidence, or a change in the terms of the authentication between the vendor and the server. These issues are discussed more completely below.

[0143] According to one embodiment, the transaction ID and the authentication request may be advantageously generated by a vendor-side applet or other software program. In addition, the transmission of the transaction ID and authentication data may include one or more XML documents encrypted using conventional SSL technology, such as, for example, ½ SSL, or, in other words vendor-side authenticated SSL.

After the user system 105 receives the transaction ID and authentication request, the user system 105 gathers the current authentication data, potentially including current biometric information, from the user. The user system 105, at step 1015, encrypts at least the current authentication data "B" and the transaction ID, with the public key of the authentication engine 215, and transfers that data to the trust engine 110. The

transmission preferably comprises XML documents encrypted with at least conventional ½ SSL technology. In step 1020, the transaction engine 205 receives the transmission, preferably recognizes the data format or request in the URL or URI, and forwards the transmission to the authentication engine 215.

During steps 1015 and 1020, the vendor system 120, at step 1025, forwards the transaction ID and the authentication request to the trust engine 110, using the preferred FULL SSL technology. This communication may also include a vendor ID, although vendor identification may also be communicated through a non-random portion of the transaction ID. At steps 1030 and 1035, the transaction engine 205 receives the communication, creates a record in the audit trail, and generates a request for the user's enrollment authentication data to be assembled from the data storage facilities D1 through D4. At step 1040, the depository system 700 transfers the portions of the enrollment authentication data corresponding to the user to the authentication engine 215. At step 1045, the authentication engine 215 decrypts the transmission using its private key and compares the enrollment authentication data to the current authentication data provided by the user.

The comparison of step 1045 may advantageously apply heuristical context sensitive authentication, as referred to in the forgoing, and discussed in further detail below. For example, if the biometric information received does not match perfectly, a lower confidence match results. In particular embodiments, the level of confidence of the authentication is balanced against the nature of the transaction and the desires of both the user and the vendor. Again, this is discussed in greater detail below.

At step 1050, the authentication engine 215 fills in the authentication request with the result of the comparison of step 1045. According to one embodiment of the invention, the authentication request is filled with a YES/NO or TRUE/FALSE result of the authentication process 1000. In step 1055 the filled-in authentication request is returned to the vendor for the vendor to act upon, for example, allowing the user to complete the transaction that initiated the authentication request. According to one embodiment, a confirmation message is passed to the user.

Based on the foregoing, the authentication process 1000 advantageously keeps sensitive data secure and produces results configured to maintain the integrity of the

sensitive data. For example, the sensitive data is assembled only inside the authentication engine 215. For example, the enrollment authentication data is undecipherable until it is assembled in the authentication engine 215 by the data assembling module, and the current authentication data is undecipherable until it is unwrapped by the conventional SSL technology and the private key of the authentication engine 215. Moreover, the authentication result transmitted to the vendor does not include the sensitive data, and the user may not even know whether he or she produced valid authentication data.

Although the authentication process 1000 is disclosed with reference to its preferred and alternative embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize from the disclosure herein, a wide number of alternatives for the authentication process 1000. For example, the vendor may advantageously be replaced by almost any requesting application, even those residing with the user system 105. For example, a client application, such as Microsoft Word, may use an application program interface (API) or a cryptographic API (CAPI) to request authentication before unlocking a document. Alternatively, a mail server, a network, a cellular phone, a personal or mobile computing device, a workstation, or the like, may all make authentication requests that can be filled by the authentication process 1000. In fact, after providing the foregoing trusted authentication process 1000, the requesting application or device may provide access to or use of a wide number of electronic or computer devices or systems.

Moreover, the authentication process 1000 may employ a wide number of alternative procedures in the event of authentication failure. For example, authentication failure may maintain the same transaction ID and request that the user reenter his or her current authentication data. As mentioned in the foregoing, use of the same transaction ID allows the comparator of the authentication engine 215 to monitor and limit the number of authentication attempts for a particular transaction, thereby creating a more secure cryptographic system 100.

In addition, the authentication process 1000 may be advantageously be employed to develop elegant single sign-on solutions, such as, unlocking a sensitive data vault. For example, successful or positive authentication may provide the

authenticated user the ability to automatically access any number of passwords for an almost limitless number of systems and applications. For example, authentication of a user may provide the user access to password, login, financial credentials, or the like, associated with multiple online vendors, a local area network, various personal computing devices, Internet service providers, auction providers, investment brokerages, or the like. By employing a sensitive data vault, users may choose truly large and random passwords because they no longer need to remember them through association. Rather, the authentication process 1000 provides access thereto. For example, a user may choose a random alphanumeric string that is twenty plus digits in length rather than something associated with a memorable data, name, etc.

According to one embodiment, a sensitive data vault associated with a given user may advantageously be stored in the data storage facilities of the depository 210, or split and stored in the depository system 700. According to this embodiment, after positive user authentication, the trust engine 110 serves the requested sensitive data, such as, for example, to the appropriate password to the requesting application. According to another embodiment, the trust engine 110 may include a separate system for storing the sensitive data vault. For example, the trust engine 110 may include a stand-alone software engine implementing the data vault functionality and figuratively residing "behind" the foregoing front-end security system of the trust engine 110. According to this embodiment, the software engine serves the requested sensitive data after the software engine receives a signal indicating positive user authentication from the trust engine 110.

In yet another embodiment, the data vault may be implemented by a third-party system. Similar to the software engine embodiment, the third-party system may advantageously serve the requested sensitive data after the third-party system receives a signal indicating positive user authentication from the trust engine 110. According to yet another embodiment, the data vault may be implemented on the user system 105. A user-side software engine may advantageously serve the foregoing data after receiving a signal indicating positive user authentication from the trust engine 110.

Although the foregoing data vaults are disclosed with reference to alternative embodiments, a skilled artisan will recognize from the disclosure herein, a wide number

of additional implementations thereof. For example, a particular data vault may include aspects from some or all of the foregoing embodiments. In addition, any of the foregoing data vaults may employ one or more authentication requests at varying times. For example, any of the data vaults may require authentication every one or more transactions, periodically, every one or more sessions, every access to one or more Webpages or Websites, at one or more other specified intervals, or the like.

FIGURE 11 illustrates a data flow of a signing process 1100 according to aspects of an embodiment of the invention. As shown in FIGURE 11, the signing process 1100 includes steps similar to those of the authentication process 1000 described in the foregoing with reference to FIGURE 10. According to one embodiment of the invention, the signing process 1100 first authenticates the user and then performs one or more of several digital signing functions as will be discussed in further detail below. According to another embodiment, the signing process 1100 may advantageously store data related thereto, such as hashes of messages or documents, or the like. This data may advantageously be used in an audit or any other event, such as for example, when a participating party attempts to repudiate a transaction.

As shown in FIGURE 11, during the authentication steps, the user and vendor may advantageously agree on a message, such as, for example, a contract. During signing, the signing process 1100 advantageously ensures that the contract signed by the user is identical to the contract supplied by the vendor. Therefore, according to one embodiment, during authentication, the vendor and the user include a hash of their respective copies of the message or contract, in the data transmitted to the authentication engine 215. By employing only a hash of a message or contract, the trust engine 110 may advantageously store a significantly reduced amount of data, providing for a more efficient and cost effective cryptographic system. In addition, the stored hash may be advantageously compared to a hash of a document in question to determine whether the document in question matches one signed by any of the parties. The ability to determine whether the document is identical to one relating to a transaction provides for additional evidence that can be used against a claim for repudiation by a party to a transaction.

In step 1103, the authentication engine 215 assembles the enrollment authentication data and compares it to the current authentication data provided by the user. When the comparator of the authentication engine 215 indicates that the enrollment authentication data matches the current authentication data, the comparator of the authentication engine 215 also compares the hash of the message supplied by the vendor to the hash of the message supplied by the user. Thus, the authentication engine 215 advantageously ensures that the message agreed to by the user is identical to that agreed to by the vendor.

In step 1105, the authentication engine 215 transmits a digital signature request to the cryptographic engine 220. According to one embodiment of the invention, the request includes a hash of the message or contract. However, a skilled artisan will recognize from the disclosure herein that the cryptographic engine 220 may encrypt virtually any type of data, including, but not limited to, video, audio, biometrics, images or text to form the desired digital signature. Returning to step 1105, the digital signature request preferably comprises an XML document communicated through conventional SSL technologies.

In step 1110, the authentication engine 215 transmits a request to each of the data storage facilities D1 through D4, such that each of the data storage facilities D1 through D4 transmit their respective portion of the cryptographic key or keys corresponding to a signing party. According to another embodiment, the cryptographic engine 220 employs some or all of the steps of the interoperability process 970 discussed in the foregoing, such that the cryptographic engine 220 first determines the appropriate key or keys to request from the depository 210 or the depository system 700 for the signing party, and takes actions to provide appropriate matching keys. According to still another embodiment, the authentication engine 215 or the cryptographic engine 220 may advantageously request one or more of the keys associated with the signing party and stored in the depository 210 or depository system 700.

According to one embodiment, the signing party includes one or both the user and the vendor. In such case, the authentication engine 215 advantageously requests the cryptographic keys corresponding to the user and/or the vendor. According to another

embodiment, the signing party includes the trust engine 110. In this embodiment, the trust engine 110 is certifying that the authentication process 1000 properly authenticated the user, vendor, or both. Therefore, the authentication engine 215 requests the cryptographic key of the trust engine 110, such as, for example, the key belonging to the cryptographic engine 220, to perform the digital signature. According to another embodiment, the trust engine 110 performs a digital notary-like function. In this embodiment, the signing party includes the user, vendor, or both, along with the trust engine 110. Thus, the trust engine 110 provides the digital signature of the user and/or vendor, and then indicates with its own digital signature that the user and/or vendor were properly authenticated. In this embodiment, the authentication engine 215 may advantageously request assembly of the cryptographic keys corresponding to the user, the vendor, or both. According to another embodiment, the authentication engine 215 may advantageously request assembly of the cryptographic keys corresponding to the trust engine 110.

According to another embodiment, the trust engine 110 performs power of attorney-like functions. For example, the trust engine 110 may digitally sign the message on behalf of a third party. In such case, the authentication engine 215 requests the cryptographic keys associated with the third party. According to this embodiment, the signing process 1100 may advantageously include authentication of the third party, before allowing power of attorney-like functions. In addition, the authentication process 1000 may include a check for third party constraints, such as, for example, business logic or the like dictating when and in what circumstances a particular third-party's signature may be used.

Based on the foregoing, in step 1110, the authentication engine requested the cryptographic keys from the data storage facilities D1 through D4 corresponding to the signing party. In step 1115, the data storage facilities D1 through D4 transmit their respective portions of the cryptographic key corresponding to the signing party to the cryptographic engine 220. According to one embodiment, the foregoing transmissions include SSL technologies. According to another embodiment, the foregoing transmissions may advantageously be super-encrypted with the public key of the cryptographic engine 220.

In step 1120, the cryptographic engine 220 assembles the foregoing cryptographic keys of the signing party and encrypts the message therewith, thereby forming the digital signature(s). In step 1125 of the signing process 1100, the cryptographic engine 220 transmits the digital signature(s) to the authentication engine 215. In step 1130, the authentication engine 215 transmits the filled-in authentication request along with a copy of the hashed message and the digital signature(s) to the transaction engine 205. In step 1135, the transaction engine 205 transmits a receipt comprising the transaction ID, an indication of whether the authentication was successful, and the digital signature(s), to the vendor. According to one embodiment, the foregoing transmission may advantageously include the digital signature of the trust engine 110. For example, the trust engine 110 may encrypt the hash of the receipt with its private key, thereby forming a digital signature to be attached to the transmission to the vendor.

According to one embodiment, the transaction engine 205 also transmits a confirmation message to the user. Although the signing process 1100 is disclosed with reference to its preferred and alternative embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize from the disclosure herein, a wide number of alternatives for the signing process 1100. For example, the vendor may be replaced with a user application, such as an email application. For example, the user may wish to digitally sign a particular email with his or her digital signature. In such an embodiment, the transmission throughout the signing process 1100 may advantageously include only one copy of a hash of the message. Moreover, a skilled artisan will recognize from the disclosure herein that a wide number of client applications may request digital signatures. For example, the client applications may comprise word processors, spreadsheets, emails, voicemail, access to restricted system areas, or the like.

In addition, a skilled artisan will recognize from the disclosure herein that steps 1105 through 1120 of the signing process 1100 may advantageously employ some or all of the steps of the interoperability process 970 of FIGURE 9B, thereby providing interoperability between differing cryptographic systems that may, for example, need to process the digital signature under differing signature types.

FIGURE 12 illustrates a data flow of an encryption/decryption process 1200 according to aspects of an embodiment of the invention. As shown in FIGURE 12, the decryption process 1200 begins by authenticating the user using the authentication process 1000. According to one embodiment, the authentication process 1000 includes in the authentication request, a synchronous session key. For example, in conventional PKI technologies, it is understood by skilled artisans that encrypting or decrypting data using public and private keys is mathematically intensive and may require significant system resources. However, in symmetric key cryptographic systems, or systems where the sender and receiver of a message share a single common key that is used to encrypt and decrypt a message, the mathematical operations are significantly simpler and faster. Thus, in the conventional PKI technologies, the sender of a message will generate synchronous session key, and encrypt the message using the simpler, faster symmetric key system. Then, the sender will encrypt the session key with the public key of the receiver. The encrypted session key will be attached to the synchronously encrypted message and both data are sent to the receiver. The receiver uses his or her private key to decrypt the session key, and then uses the session key to decrypt the message. Based on the foregoing, the simpler and faster symmetric key system is used for the majority of the encryption/decryption processing. Thus, in the decryption process 1200, the decryption advantageously assumes that a synchronous key has been encrypted with the public key of the user. Thus, as mentioned in the foregoing, the encrypted session key is included in the authentication request.

Returning to the decryption process 1200, after the user has been authenticated in step 1205, the authentication engine 215 forwards the encrypted session key to the cryptographic engine 220. In step 1210, the authentication engine 215 forwards a request to each of the data storage facilities, D1 through D4, requesting the cryptographic key data of the user. In step 1215, each data storage facility, D1 through D4, transmits their respective portion of the cryptographic key to the cryptographic engine 220. According to one embodiment, the foregoing transmission is encrypted with the public key of the cryptographic engine 220.

In step 1220 of the decryption process 1200, the cryptographic engine 220 assembles the cryptographic key and decrypts the session key therewith. In step 1225,

the cryptographic engine forwards the session key to the authentication engine 215. In step 1227, the authentication engine 215 fills in the authentication request including the decrypted session key, and transmits the filled-in authentication request to the transaction engine 205. In step 1230, the transaction engine 205 forwards the authentication request along with the session key to the requesting application or vendor. Then, according to one embodiment, the requesting application or vendor uses the session key to decrypt the encrypted message.

Although the decryption process 1200 is disclosed with reference to its preferred and alternative embodiments, a skilled artisan will recognize from the disclosure herein, a wide number of alternatives for the decryption process 1200. For example, the decryption process 1200 may forego synchronous key encryption and rely on full public-key technology. In such an embodiment, the requesting application may transmit the entire message to the cryptographic engine 220, or, may employ some type of compression or reversible hash in order to transmit the message to the cryptographic engine 220. A skilled artisan will also recognize from the disclosure herein that the foregoing communications may advantageously include XML documents wrapped in SSL technology.

The encryption/decryption process 1200 also provides for encryption of documents or other data. Thus, in step 1235, a requesting application or vendor may advantageously transmit to the transaction engine 205 of the trust engine 110, a request for the public key of the user. The requesting application or vendor makes this request because the requesting application or vendor uses the public key of the user, for example, to encrypt the session key that will be used to encrypt the document or message. As mentioned in the enrollment process 900, the transaction engine 205 stores a copy of the digital certificate of the user, for example, in the mass storage 225. Thus, in step 1240 of the encryption process 1200, the transaction engine 205 requests the digital certificate of the user from the mass storage 225. In step 1245, the mass storage 225 transmits the digital certificate corresponding to the user, to the transaction engine 205. In step 1250, the transaction engine 205 transmits the digital certificate to the requesting application or vendor. According to one embodiment, the encryption portion of the encryption process 1200 does not include the authentication of a user.

This is because the requesting vendor needs only the public key of the user, and is not requesting any sensitive data.

A skilled artisan will recognize from the disclosure herein that if a particular user does not have a digital certificate, the trust engine 110 may employ some or all of the enrollment process 900 in order to generate a digital certificate for that particular user. Then, the trust engine 110 may initiate the encryption/decryption process 1200 and thereby provide the appropriate digital certificate. In addition, a skilled artisan will recognize from the disclosure herein that steps 1220 and 1235 through 1250 of the encryption/decryption process 1200 may advantageously employ some or all of the steps of the interoperability process of FIGURE 9B, thereby providing interoperability between differing cryptographic systems that may, for example, need to process the encryption.

FIGURE 13 illustrates a simplified block diagram of a trust engine system 1300 according to aspects of yet another embodiment of the invention. As shown in FIGURE 13, the trust engine system 1300 comprises a plurality of distinct trust engines 1305, 1310, 1315, and 1320, respectively. To facilitate a more complete understanding of the invention, FIGURE 13 illustrates each trust engine, 1305, 1310, 1315, and 1320 as having a transaction engine, a depository, and an authentication engine. However, a skilled artisan will recognize that each transaction engine may advantageously comprise some, a combination, or all of the elements and communication channels disclosed with reference to FIGURES 1-8. For example, one embodiment may advantageously include trust engines having one or more transaction engines, depositories, and cryptographic servers or any combinations thereof.

According to one embodiment of the invention, each of the trust engines 1305, 1310, 1315 and 1320 are geographically separated, such that, for example, the trust engine 1305 may reside in a first location, the trust engine 1310 may reside in a second location, the trust engine 1315 may reside in a third location, and the trust engine 1320 may reside in a fourth location. The foregoing geographic separation advantageously decreases system response time while increasing the security of the overall trust engine system 1300.

EX1010, Page 2344

For example, when a user logs onto the cryptographic system 100, the user may be nearest the first location and may desire to be authenticated. As described with reference to FIGURE 10, to be authenticated, the user provides current authentication data, such as a biometric or the like, and the current authentication data is compared to that user's enrollment authentication data. Therefore, according to one example, the user advantageously provides current authentication data to the geographically nearest trust engine 1305. The transaction engine 1321 of the trust engine 1305 then forwards the current authentication data to the authentication engine 1322 also residing at the first location. According to another embodiment, the transaction engine 1321 forwards the current authentication data to one or more of the authentication engines of the trust engines 1310, 1315, or 1320.

The transaction engine 1321 also requests the assembly of the enrollment authentication data from the depositories of, for example, each of the trust engines, 1305 through 1320. According to this embodiment, each depository provides its portion of the enrollment authentication data to the authentication engine 1322 of the trust engine 1305. The authentication engine 1322 then employs the encrypted data portions from, for example, the first two depositories to respond, and assembles the enrollment authentication data into deciphered form. The authentication engine 1322 compares the enrollment authentication data with the current authentication data and returns an authentication result to the transaction engine 1321 of the trust engine 1305.

Based on the above, the trust engine system 1300 employs the nearest one of a plurality of geographically separated trust engines, 1305 through 1320, to perform the authentication process. According to one embodiment of the invention, the routing of information to the nearest transaction engine may advantageously be performed at client-side applets executing on one or more of the user system 105, vendor system 120, or certificate authority 115. According to an alternative embodiment, a more sophisticated decision process may be employed to select from the trust engines 1305 through 1320. For example, the decision may be based on the availability, operability, speed of connections, load, performance, geographic proximity, or a combination thereof, of a given trust engine.

In this way, the trust engine system 1300 lowers its response time while maintaining the security advantages associated with geographically remote data storage facilities, such as those discussed with reference to FIGURE 7 where each data storage facility stores randomized portions of sensitive data. For example, a security compromise at, for example, the depository 1325 of the trust engine 1315 does not necessarily compromise the sensitive data of the trust engine system 1300. This is because the depository 1325 contains only non-decipherable randomized data that, without more, is entirely useless.

According to another embodiment, the trust engine system 1300 may advantageously include multiple cryptographic engines arranged similar to the authentication engines. The cryptographic engines may advantageously perform cryptographic functions such as those disclosed with reference to FIGURES 1-8. According to yet another embodiment, the trust engine system 1300 may advantageously replace the multiple authentication engines with multiple cryptographic engines, thereby performing cryptographic functions such as those disclosed with reference to FIGURES 1-8. According to yet another embodiment of the invention, the trust engine system 1300 may replace each multiple authentication engine with an engine having some or all of the functionality of the authentication engines, cryptographic engines, or both, as disclosed in the foregoing,

Although the trust engine system 1300 is disclosed with reference to its preferred and alternative embodiments, a skilled artisan will recognize that the trust engine system 1300 may comprise portions of trust engines 1305 through 1320. For example, the trust engine system 1300 may include one or more transaction engines, one or more depositories, one or more authentication engines, or one or more cryptographic engines or combinations thereof.

FIGURE 14 illustrates a simplified block diagram of a trust engine System 1400 according to aspects of yet another embodiment of the invention. As shown in FIGURE 14, the trust engine system 1400 includes multiple trust engines 1405, 1410, 1415 and 1420. According to one embodiment, each of the trust engines 1405, 1410, 1415 and 1420, comprise some or all of the elements of trust engine 110 disclosed with reference to FIGURES 1-8. According to this embodiment, when the client side applets of the

user system 105, the vendor system 120, or the certificate authority 115, communicate with the trust engine system 1400, those communications are sent to the IP address of each of the trust engines 1405 through 1420. Further, each transaction engine of each of the trust engines, 1405, 1410, 1415, and 1420, behaves similar to the transaction engine 1321 of the trust engine 1305 disclosed with reference to FIGURE 13. For example, during an authentication process, each transaction engine of each of the trust engines 1405, 1410, 1415, and 1420 transmits the current authentication data to their respective authentication engines and transmits a request to assemble the randomized data stored in each of the depositories of each of the trust engines 1405 through 1420. FIGURE 14 does not illustrate all of these communications; as such illustration would become overly complex. Continuing with the authentication process, each of the depositories then communicates its portion of the randomized data to each of the authentication engines of the each of the trust engines 1405 through 1420. Each of the authentication engines of the each of the trust engines employs its comparator to determine whether the current authentication data matches the enrollment authentication data provided by the depositories of each of the trust engines 1405 through 1420. According to this embodiment, the result of the comparison by each of the authentication engines is then transmitted to a redundancy module of the other three trust engines. For example, the result of the authentication engine from the trust engine 1405 is transmitted to the redundancy modules of the trust engines 1410, 1415, and 1420. Thus, the redundancy module of the trust engine 1405 likewise receives the result of the authentication engines from the trust engines 1410, 1415, and 1420.

FIGURE 15 illustrates a block diagram of the redundancy module of FIGURE 14. The redundancy module comprises a comparator configured to receive the authentication result from three authentication engines and transmit that result to the transaction engine of the fourth trust engine. The comparator compares the authentication result form the three authentication engines, and if two of the results agree, the comparator concludes that the authentication result should match that of the two agreeing authentication engines. This result is then transmitted back to the transaction engine corresponding to the trust engine not associated with the three authentication engines.

Based on the foregoing, the redundancy module determines an authentication result from data received from authentication engines that are preferably geographically remote from the trust engine of that the redundancy module. By providing such redundancy functionality, the trust engine system 1400 ensures that a compromise of the authentication engine of one of the trust engines 1405 through 1420, is insufficient to compromise the authentication result of the redundancy module of that particular trust engine. A skilled artisan will recognize that redundancy module functionality of the trust engine system 1400 may also be applied to the cryptographic engine of each of the trust engines 1405 through 1420. However, such cryptographic engine communication was not shown in FIGURE 14 to avoid complexity. Moreover, a skilled artisan will recognize a wide number of alternative authentication result conflict resolution algorithms for the comparator of FIGURE 15 are suitable for use in the present invention.

According to yet another embodiment of the invention, the trust engine system 1400 may advantageously employ the redundancy module during cryptographic comparison steps. For example, some or all of the foregoing redundancy module disclosure with reference to FIGURES 14 and 15 may advantageously be implemented during a hash comparison of documents provided by one or more parties during a particular transaction.

Although the foregoing invention has been described in terms of certain preferred and alternative embodiments, other embodiments will be apparent to those of ordinary skill in the art from the disclosure herein. For example, the trust engine 110 may issue short-term certificates, where the private cryptographic key is released to the user for a predetermined period of time. For example, current certificate standards include a validity field that can be set to expire after a predetermined amount of time. Thus, the trust engine 110 may release a private key to a user where the private key would be valid for, for example, 24 hours. According to such an embodiment, the trust engine 110 may advantageously issue a new cryptographic key pair to be associated with a particular user and then release the private key of the new cryptographic key pair. Then, once the private cryptographic key is released, the trust engine 110 immediately

expires any internal valid use of such private key, as it is no longer securable by the trust engine 110.

In addition, a skilled artisan will recognize that the cryptographic system 100 or the trust engine 110 may include the ability to recognize any type of devices, such as, but not limited to, a laptop, a cell phone, a network, a biometric device or the like. According to one embodiment, such recognition may come from data supplied in the request for a particular service, such as, a request for authentication leading to access or use, a request for cryptographic functionality, or the like. According to one embodiment, the foregoing request may include a unique device identifier, such as, for example, a processor ID. Alternatively, the request may include data in a particular recognizable data format. For example, mobile and satellite phones often do not include the processing power for full X509.v3 heavy encryption certificates, and therefore do not request them. According to this embodiment, the trust engine 110 may recognize the type of data format presented, and respond only in kind.

In an additional aspect of the system described above, context sensitive authentication can be provided using various techniques as will be described below. Context sensitive authentication, for example as shown in FIGURE 16, provides the possibility of evaluating not only the actual data which is sent by the user when attempting to authenticate himself, but also the circumstances surrounding the generation and delivery of that data. Such techniques may also support transaction specific trust arbitrage between the user and trust engine 110 or between the vendor and trust engine 110, as will be described below.

As discussed above, authentication is the process of proving that a user is who he says he is. Generally, authentication requires demonstrating some fact to an authentication authority. The trust engine 110 of the present invention represents the authority to which a user must authenticate himself. The user must demonstrate to the trust engine 110 that he is who he says he is by either: knowing something that only the user should know (knowledge-based authentication), having something that only the user should have (token-based authentication), or by being something that only the user should be (biometric-based authentication).

Examples of knowledge-based authentication include without limitation a password, PIN number, or lock combination. Examples of token-based authentication include without limitation a house key, a physical credit card, a driver's license, or a particular phone number. Examples of biometric-based authentication include without limitation a fingerprint, handwriting analysis, facial scan, hand scan, ear scan, iris scan, vascular pattern, DNA, a voice analysis, or a retinal scan.

Each type of authentication has particular advantages and disadvantages, and each provides a different level of security. For example, it is generally harder to create a false fingerprint that matches someone else's than it is to overhear someone's password and repeat it. Each type of authentication also requires a different type of data to be known to the authenticating authority in order to verify someone using that form of authentication.

As used herein, "authentication" will refer broadly to the overall process of verifying someone's identity to be who he says he is. An "authentication technique" will refer to a particular type of authentication based upon a particular piece of knowledge, physical token, or biometric reading. "Authentication data" refers to information which is sent to or otherwise demonstrated to an authentication authority in order to establish identity. "Enrollment data" will refer to the data which is initially submitted to an authentication authority in order to establish a baseline for comparison with authentication data. An "authentication instance" will refer to the data associated with an attempt to authenticate by an authentication technique.

The internal protocols and communications involved in the process of authenticating a user is described with reference to FIGURE 10 above. The part of this process within which the context sensitive authentication takes place occurs within the comparison step shown as step 1045 of FIGURE 10. This step takes place within the authentication engine 215 and involves assembling the enrollment data 410 retrieved from the depository 210 and comparing the authentication data provided by the user to it. One particular embodiment of this process is shown in FIGURE 16 and described below.

The current authentication data provided by the user and the enrollment data retrieved from the depository 210 are received by the authentication engine 215 in step

1600 of FIGURE 16. Both of these sets of data may contain data which is related to separate techniques of authentication. The authentication engine 215 separates the authentication data associated with each individual authentication instance in step 1605. This is necessary so that the authentication data is compared with the appropriate subset of the enrollment data for the user (e.g. fingerprint authentication data should be compared with fingerprint enrollment data, rather than password enrollment data).

Generally, authenticating a user involves one or more individual authentication instances, depending on which authentication techniques are available to the user. These methods are limited by the enrollment data which were provided by the user during his enrollment process (if the user did not provide a retinal scan when enrolling, he will not be able to authenticate himself using a retinal scan), as well as the means which may be currently available to the user (e.g. if the user does not have a fingerprint reader at his current location, fingerprint authentication will not be practical). In some cases, a single authentication instance may be sufficient to authenticate a user; however, in certain circumstances a combination of multiple authentication instances may be used in order to more confidently authenticate a user for a particular transaction.

Each authentication instance consists of data related to a particular authentication technique (e.g. fingerprint, password, smart card, etc.) and the circumstances which surround the capture and delivery of the data for that particular technique. For example, a particular instance of attempting to authenticate via password will generate not only the data related to the password itself, but also circumstantial data, known as "metadata", related to that password attempt. This circumstantial data includes information such as: the time at which the particular authentication instance took place, the network address from which the authentication information was delivered, as well as any other information as is known to those of skill in the art which may be determined about the origin of the authentication data (the type of connection, the processor serial number, etc.).

In many cases, only a small amount of circumstantial metadata will be available. For example, if the user is located on a network which uses proxies or network address translation or another technique which masks the address of the originating computer, only the address of the proxy or router may be determined. Similarly, in many cases

information such as the processor serial number will not be available because of either limitations of the hardware or operating system being used, disabling of such features by the operator of the system, or other limitations of the connection between the user's system and the trust engine 110.

As shown in FIGURE 16, once the individual authentication instances represented within the authentication data are extracted and separated in step 1605, the authentication engine 215 evaluates each instance for its reliability in indicating that the user is who he claims to be. The reliability for a single authentication instance will generally be determined based on several factors. These may be grouped as factors relating to the reliability associated with the authentication technique, which are evaluated in step 1610, and factors relating to the reliability of the particular authentication data provided, which are evaluated in step 1815. The first group includes without limitation the inherent reliability of the authentication technique being used, and the reliability of the enrollment data being used with that method. The second group includes without limitation the degree of match between the enrollment data and the data provided with the authentication instance, and the metadata associated with that authentication instance. Each of these factors may vary independently of the others.

The inherent reliability of an authentication technique is based on how hard it is for an imposter to provide someone else's correct data, as well as the overall error rates for the authentication technique. For passwords and knowledge based authentication methods, this reliability is often fairly low because there is nothing that prevents someone from revealing their password to another person and for that second person to use that password. Even a more complex knowledge based system may have only moderate reliability since knowledge may be transferred from person to person fairly easily. Token based authentication, such as having a proper smart card or using a particular terminal to perform the authentication, is similarly of low reliability used by itself, since there is no guarantee that the right person is in possession of the proper token.

However, biometric techniques are more inherently reliable because it is generally difficult to provide someone else with the ability to use your fingerprints in a convenient manner, even intentionally. Because subverting biometric authentication

techniques is more difficult, the inherent reliability of biometric methods is generally higher than that of purely knowledge or token based authentication techniques. However, even biometric techniques may have some occasions in which a false acceptance or false rejection is generated. These occurrences may be reflected by differing reliabilities for different implementations of the same biometric technique. For example, a fingerprint matching system provided by one company may provide a higher reliability than one provided by a different company because one uses higher quality optics or a better scanning resolution or some other improvement which reduces the occurrence of false acceptances or false rejections.

Note that this reliability may be expressed in different manners. The reliability is desirably expressed in some metric which can be used by the heuristics 530 and algorithms of the authentication engine 215 to calculate the confidence level of each authentication. One preferred mode of expressing these reliabilities is as a percentage or fraction. For instance, fingerprints might be assigned an inherent reliability of 97%, while passwords might only be assigned an inherent reliability of 50%. Those of skill in the art will recognize that these particular values are merely exemplary and may vary between specific implementations.

The second factor for which reliability must be assessed is the reliability of the enrollment. This is part of the "graded enrollment" process referred to above. This reliability factor reflects the reliability of the identification provided during the initial enrollment process. For instance, if the individual initially enrolls in a manner where they physically produce evidence of their identity to a notary or other public official, and enrollment data is recorded at that time and notarized, the data will be more reliable than data which is provided over a network during enrollment and only vouched for by a digital signature or other information which is not truly tied to the individual.

Other enrollment techniques with varying levels of reliability include without limitation: enrollment at a physical office of the trust engine 110 operator; enrollment at a user's place of employment; enrollment at a post office or passport office; enrollment through an affiliated or trusted party to the trust engine 110 operator; anonymous or pseudonymous enrollment in which the enrolled identity is not yet identified with a particular real individual, as well as such other means as are known in the art.

These factors reflect the trust between the trust engine 110 and the source of identification provided during the enrollment process. For instance, if enrollment is performed in association with an employer during the initial process of providing evidence of identity, this information may be considered extremely reliable for purposes within the company, but may be trusted to a lesser degree by a government agency, or by a competitor. Therefore, trust engines operated by each of these other organizations may assign different levels of reliability to this enrollment.

Similarly, additional data which is submitted across a network, but which is authenticated by other trusted data provided during a previous enrollment with the same trust engine 110 may be considered as reliable as the original enrollment data was, even though the latter data were submitted across an open network. In such circumstances, a subsequent notarization will effectively increase the level of reliability associated with the original enrollment data. In this way for example, an anonymous or pseudonymous enrollment may then be raised to a full enrollment by demonstrating to some enrollment official the identity of the individual matching the enrolled data.

The reliability factors discussed above are generally values which may be determined in advance of any particular authentication instance. This is because they are based upon the enrollment and the technique, rather than the actual authentication. In one embodiment, the step of generating reliability based upon these factors involves looking up previously determined values for this particular authentication technique and the enrollment data of the user. In a further aspect of an advantageous embodiment of the present invention, such reliabilities may be included with the enrollment data itself. In this way, these factors are automatically delivered to the authentication engine 215 along with the enrollment data sent from the depository 210.

While these factors may generally be determined in advance of any individual authentication instance, they still have an effect on each authentication instance which uses that particular technique of authentication for that user. Furthermore, although the values may change over time (e.g. if the user re-enrolls in a more reliable fashion), they are not dependent on the authentication data itself. By contrast, the reliability factors associated with a single specific instance's data may vary on each occasion. These

factors, as discussed below, must be evaluated for each new authentication in order to generate reliability scores in step 1815.

The reliability of the authentication data reflects the match between the data provided by the user in a particular authentication instance and the data provided during the authentication enrollment. This is the fundamental question of whether the authentication data matches the enrollment data for the individual the user is claiming to be. Normally, when the data do not match, the user is considered to not be successfully authenticated, and the authentication fails. The manner in which this is evaluated may change depending on the authentication technique used. The comparison of such data is performed by the comparator 515 function of the authentication engine 215 as shown in FIGURE 5.

For instance, matches of passwords are generally evaluated in a binary fashion. In other words, a password is either a perfect match, or a failed match. It is usually not desirable to accept as even a partial match a password which is close to the correct password if it is not exactly correct. Therefore, when evaluating a password authentication, the reliability of the authentication returned by the comparator 515 is typically either 100% (correct) or 0% (wrong), with no possibility of intermediate values.

Similar rules to those for passwords are generally applied to token based authentication methods, such as smart cards. This is because having a smart card which has a similar identifier or which is similar to the correct one, is still just as wrong as having any other incorrect token. Therefore tokens tend also to be binary authenticators: a user either has the right token, or he doesn't.

However, certain types of authentication data, such as questionnaires and biometrics, are generally not binary authenticators. For example, a fingerprint may match a reference fingerprint to varying degrees. To some extent, this may be due to variations in the quality of the data captured either during the initial enrollment or in subsequent authentications. (A fingerprint may be smudged or a person may have a still healing scar or burn on a particular finger.) In other instances the data may match less than perfectly because the information itself is somewhat variable and based upon pattern matching. (A voice analysis may seem close but not quite right because of

background noise, or the acoustics of the environment in which the voice is recorded, or because the person has a cold.) Finally, in situations where large amounts of data are being compared, it may simply be the case that much of the data matches well, but some doesn't. (A ten-question questionnaire may have resulted in eight correct answers to personal questions, but two incorrect answers.) For any of these reasons, the match between the enrollment data and the data for a particular authentication instance may be desirably assigned a partial match value by the comparator 515. In this way, the fingerprint might be said to be a 85% match, the voice print a 65% match, and the questionnaire an 80% match, for example.

This measure (degree of match) produced by the comparator 515 is the factor representing the basic issue of whether an authentication is correct or not. However, as discussed above, this is only one of the factors which may be used in determining the reliability of a given authentication instance. Note also that even though a match to some partial degree may be determined, that ultimately, it may be desirable to provide a binary result based upon a partial match. In an alternate mode of operation, it is also possible to treat partial matches as binary, i.e. either perfect (100%) or failed (0%) matches, based upon whether or not the degree of match passes a particular threshold level of match. Such a process may be used to provide a simple pass/fail level of matching for systems which would otherwise produce partial matches.

Another factor to be considered in evaluating the reliability of a given authentication instance concerns the circumstances under which the authentication data for this particular instance are provided. As discussed above, the circumstances refer to the metadata associated with a particular authentication instance. This may include without limitation such information as: the network address of the authenticator, to the extent that it can be determined; the time of the authentication; the mode of transmission of the authentication data (phone line, cellular, network, etc.); and the serial number of the system of the authenticator.

These factors can be used to produce a profile of the type of authentication that is normally requested by the user. Then, this information can be used to assess reliability in at least two manners. One manner is to consider whether the user is requesting authentication in a manner which is consistent with the normal profile of

authentication by this user. If the user normally makes authentication requests from one network address during business days (when she is at work) and from a different network address during evenings or weekends (when she is at home), an authentication which occurs from the home address during the business day is less reliable because it is outside the normal authentication profile. Similarly, if the user normally authenticates using a fingerprint biometric and in the evenings, an authentication which originates during the day using only a password is less reliable.

An additional way in which the circumstantial metadata can be used to evaluate the reliability of an instance of authentication is to determine how much corroboration the circumstance provides that the authenticator is the individual he claims to be. For instance, if the authentication comes from a system with a serial number known to be associated with the user, this is a good circumstantial indicator that the user is who they claim to be. Conversely, if the authentication is coming from a network address which is known to be in Los Angeles when the user is known to reside in London, this is an indication that this authentication is less reliable based on its circumstances.

It is also possible that a cookie or other electronic data may be placed upon the system being used by a user when they interact with a vendor system or with the trust engine 110. This data is written to the storage of the system of the user and may contain an identification which may be read by a Web browser or other software on the user system. If this data is allowed to reside on the user system between sessions (a "persistent cookie"), it may be sent with the authentication data as further evidence of the past use of this system during authentication of a particular user. In effect, the metadata of a given instance, particularly a persistent cookie, may form a sort of token based authenticator itself.

Once the appropriate reliability factors based on the technique and data of the authentication instance are generated as described above in steps 1610 and 1615 respectively, they are used to produce an overall reliability for the authentication instance provided in step 1620. One means of doing this is simply to express each reliability as a percentage and then to multiply them together.

For example, suppose the authentication data is being sent in from a network address known to be the user's home computer completely in accordance with the

user's past authentication profile (100%), and the technique being used is fingerprint identification (97%), and the initial finger print data was roistered through the user's employer with the trust engine 110 (90%), and the match between the authentication data and the original fingerprint template in the enrollment data is very good (99%). The overall reliability of this authentication instance could then be calculated as the product of these reliabilities: 100% * 97% * 90% * 99% - 86.4% reliability.

This calculated reliability represents the reliability of one single instance of authentication. The overall reliability of a single authentication instance may also be calculated using techniques which treat the different reliability factors differently, for example by using formulas where different weights are assigned to each reliability factor. Furthermore, those of skill in the art will recognize that the actual values used may represent values other than percentages and may use non-arithmetic systems. One embodiment may include a module used by an authentication requestor to set the weights for each factor and the algorithms used in establishing the overall reliability of the authentication instance.

The authentication engine 215 may use the above techniques and variations thereof to determine the reliability of a single authentication instance, indicated as step 1620. However, it may be useful in many authentication situations for multiple authentication instances to be provided at the same time. For example, while attempting to authenticate himself using the system of the present invention, a user may provide a user identification, fingerprint authentication data, a smart card, and a password. In such a case, three independent authentication instances are being provided to the trust engine 110 for evaluation. Proceeding to step 1625, if the authentication engine 215 determines that the data provided by the user includes more than one authentication instance, then each instance in turn will be selected as shown in step 1630 and evaluated as described above in steps 1610, 1615 and 1620.

Note that many of the reliability factors discussed may vary from one of these instances to another. For instance, the inherent reliability of these techniques is likely to be different, as well as the degree of match provided between the authentication data and the enrollment data. Furthermore, the user may have provided enrollment data at different times and under different circumstances for each of these techniques,

providing different enrollment reliabilities for each of these instances as well. Finally, even though the circumstances under which the data for each of these instances is being submitted is the same, the use of such techniques may each fit the profile of the user differently, and so may be assigned different circumstantial reliabilities. (For example, the user may normally use their password and fingerprint, but not their smart card.)

As a result, the final reliability for each of these authentication instances may be different from One another. However, by using multiple instances together, the overall confidence level for the authentication will tend to increase.

Once the authentication engine has performed steps 1610 through 1620 for all of the authentication instances provided in the authentication data, the reliability of each instance is used in step 1635 to evaluate the overall authentication confidence level. This process of combining the individual authentication instance reliabilities into the authentication confidence level may be modeled by various methods relating the individual reliabilities produced, and may also address the particular interaction between some of these authentication techniques. (For example, multiple knowledge-based systems such as passwords may produce less confidence than a single password and even a fairly weak biometric, such as a basic voice analysis.)

One means in which the authentication engine 215 may combine the reliabilities of multiple concurrent authentication instances to generate a final confidence level is to multiply the unreliability of each instance to arrive at a total unreliability. The unreliability is generally the complementary percentage of the reliability. For example, a technique which is 84% reliable is 16% unreliable. The three authentication instances described above (fingerprint, smart card, password)which produce reliabilities of 86%, 75%, and 72% would have corresponding unreliabilities of (100- 86)%, (100- 75)% and (100- 72)%, or 14%, 25%, and 28%, respectively. By multiplying these unreliabilities, we get a cumulative unreliability of 14% * 25% * 28% - .98% unreliability, which corresponds to a reliability of 99.02%.

In an additional mode of operation, additional factors and heuristics 530 may be applied within the authentication engine 215 to account for the interdependence of various authentication techniques. For example, if someone has unauthorized access to a particular home computer, they probably have access to the phone line at that address

as well. Therefore, authenticating based on an originating phone number as well as upon the serial number of the authenticating system does not add much to the overall confidence in the authentication. However, knowledge based authentication is largely independent of token based authentication (i.e. if someone steals your cellular phone or keys, they are no more likely to know your PIN or password than if they hadn't).

Furthermore, different vendors or other authentication requestors may wish to weigh different aspects of the authentication differently. This may include the use of separate weighing factors or algorithms used in calculating the reliability of individual instances as well as the use of different means to evaluate authentication events with multiple instances.

For instance, vendors for certain types of transactions, for instance corporate email systems, may desire to authenticate primarily based upon heuristics and other circumstantial data by default. Therefore, they may apply high weights to factors related to the metadata and other profile related information associated with the circumstances surrounding authentication events. This arrangement could be used to ease the burden on users during normal operating hours, by not requiring more from the user than that he be logged on to the correct machine during business hours. However, another vendor may weigh authentications coming from a particular technique most heavily, for instance fingerprint matching, because of a policy decision that such a technique is most suited to authentication for the particular vendor's purposes.

Such varying weights may be defined by the authentication requestor in generating the authentication request and sent to the trust engine 110 with the authentication request in one mode of operation. Such options could also be set as preferences during an initial enrollment process for the authentication requestor and stored within the authentication engine in another mode of operation.

[0227] Once the authentication engine 215 produces an authentication confidence level for the authentication data provided, this confidence level is used to complete the authentication request in step 1640, and this information is forwarded from the authentication engine 215 to the transaction engine 205 for inclusion in a message to the authentication requestor.

The process described above is merely exemplary, and those of skill in the art will recognize that the steps need not be performed in the order shown or that only certain of the steps are desired to be performed, or that a variety of combinations of steps may be desired. Furthermore, certain steps, such as the evaluation of the reliability of each authentication instance provided, may be carried out in parallel with one another if circumstances permit.

In a further aspect of this invention, a method is provided to accommodate conditions when the authentication confidence level produced by the process described above fails to meet the required trust level of the vendor or other party requiring the authentication. In circumstances such as these where a gap exists between the level of confidence provided and the level of trust desired, the operator of the trust engine 110 is in a position to provide opportunities for one or both parties to provide alternate data or requirements in order to close this trust gap. This process will be referred to as "trust arbitrage" herein.

Trust arbitrage may take place within a framework of cryptographic authentication as described above with reference to FIGURES 10 and 11. As shown therein, a vendor or other party will request authentication of a particular user in association with a particular transaction. In one circumstance, the vendor simply requests an authentication, either positive or negative, and after receiving appropriate data from the user, the trust engine 110 will provide such a binary authentication. In circumstances such as these, the degree of confidence required in order to secure a positive authentication is determined based upon preferences set within the trust engine 110.

However, it is also possible that the vendor may request a particular level of trust in order to complete a particular transaction. This required level may be included with the authentication request (e.g. authenticate this user to 98% confidence) or may be determined by the trust engine 110 based on other factors associated with the transaction (i.e. authenticate this user as appropriate for this transaction). One such factor might be the economic value of the transaction. For transactions which have greater economic value, a higher degree of trust may be required. Similarly, for transactions with high degrees of risk a high degree of trust may be required.

Conversely, for transactions which are either of low risk or of low value, lower trust levels may be required by the vendor or other authentication requestor.

The process of trust arbitrage occurs between the steps of the trust engine 110 receiving the authentication data in step 1050 of FIGURE 10 and the return of an authentication result to the vendor in step 1055 of FIGURE 10. Between these steps, the process which leads to the evaluation of trust levels and the potential trust arbitrage occurs as shown in FIGURE 17. In circumstances where simple binary authentication is performed, the process shown in FIGURE 17 reduces to having the transaction engine 205 directly compare the authentication data provided with the enrollment data for the identified user as discussed above with reference to FIGURE 10, flagging any difference as a negative authentication.

As shown in FIGURE 17, the first step after receiving the data in step 1050 is for the transaction engine 205 to determine the trust level which is required for a positive authentication for this particular transaction in step 1710. This step may be performed by one of several different methods. The required trust level may be specified to the trust engine 110 by the authentication requestor at the time when the authentication request is made. The authentication requestor may also set a preference in advance which is stored within the depository 210 or other storage which is accessible by the transaction engine 205. This preference may then be read and used each time an authentication request is made by this authentication requestor. The preference may also be associated with a particular user as a security measure such that a particular level of trust is always required in order to authenticate that user, the user preference being stored in the depository 210 or other storage media accessible by the transaction engine 205. The required level may also be derived by the transaction engine 205 or authentication engine 215 based upon information provided in the authentication request, such as the value and risk level of the transaction to be authenticated.

In one mode of operation, a policy management module or other software which is used when generating the authentication request is used to specify the required degree of trust for the authentication of the transaction. This may be used to provide a series of rules to follow when assigning the required level of trust based upon the policies which

are specified within the policy management module. One advantageous mode of operation is for such a module to be incorporated with the web server of a vendor in order to appropriately determine required level of trust for transactions initiated with the vendor's web server. In this way, transaction requests from users may be assigned a required trust level in accordance with the policies of the vendor and such information may be forwarded to the trust engine 110 along with the authentication request.

This required trust level correlates with the degree of certainty that the vendor wants to have that the individual authenticating is in fact who he identifies himself as. For example, if the transaction is one where the vendor wants a fair degree of certainty because goods are changing hands, the vendor may require a trust level of 85%. For situation where the vendor is merely authenticating the user to allow him to view members only content or exercise privileges on a chat room, the downside risk may be small enough that the vendor requires only a 60% trust level. However, to enter into a production contract with a value of tens of thousands of dollars, the vendor may require a trust level of 99% or more.

This required trust level represents a metric to which the user must authenticate himself in order to complete the transaction. If the required trust level is 85% for example, the user must provide authentication to the trust engine 110 sufficient for the trust engine 110 to say with 85% confidence that the user is who they say they are. It is the balance between this required trust level and the authentication confidence level which produces either a positive authentication (to the satisfaction of the vendor) or a possibility of trust arbitrage.

As shown in FIGURE 17, after the transaction engine 205 receives the required trust level, it compares in step 1720 the required trust level to the authentication confidence level which the authentication engine 215 calculated for the current authentication (as discussed with reference to FIGURE 16). If the authentication confidence level is higher than the required trust level for the transaction in step 1730, then the process moves to step 1740 where a positive authentication for this transaction is produced by the transaction engine 205. A message to this effect will then be inserted into the authentication results and returned to the vendor by the transaction engine 205 as shown in step 1055 (see FIGURE 10).

However, if the authentication confidence level does not fulfill the required trust level in step 1730, then a confidence gap exists for the current authentication, and trust arbitrage is conducted in step 1750. Trust arbitrage is described more completely with reference to FIGURE 18 below. This process as described below takes place within the transaction engine 205 of the trust engine 110. Because no authentication or other cryptographic operations are needed to execute trust arbitrage (other than those required for the SSL communication between the transaction engine 205 and other components), the process may be performed outside the authentication engine 215. However, as will be discussed below, any reevaluation of authentication data or other cryptographic or authentication events will require the transaction engine 205 to resubmit the appropriate data to the authentication engine 215. Those of skill in the art will recognize that the trust arbitrage process could alternately be structured to take place partially or entirely within the authentication engine 215 itself.

As mentioned above, trust arbitrage is a process where the trust engine 110 mediates a negotiation between the vendor and user in an attempt to secure a positive authentication where appropriate. As shown in step 1805, the transaction engine 205 first determines whether or not the current situation is appropriate for trust arbitrage. This may be determined based upon the circumstances of the authentication, e.g. whether this authentication has already been through multiple cycles of arbitrage, as well as upon the preferences of either the vendor or user, as will be discussed further below.

In such circumstances where arbitrage is not possible, the process proceeds to step 1810 where the transaction engine 205 generates a negative authentication and then inserts it into the authentication results which are sent to the vendor in step 1055 (see FIGURE 10). One limit which may be advantageously used to prevent authentications from pending indefinitely is to set a time-out period from the initial authentication request. In this way, any transaction which is not positively authenticated within the time limit is denied further arbitrage and negatively authenticated. Those of skill in the art will recognize that such a time limit may vary depending upon the circumstances of the transaction and the desires of the user and vendor. Limitations may also be placed

upon the number of attempts that may be made at providing a successful authentication. Such limitations may be handled by an attempt limiter 535 as shown in FIGURE 5.

If arbitrage is not prohibited in step 1805, the transaction engine 205 will then engage in negotiation with one or both of the transacting parties. The transaction engine 205 may send a message to the user requesting some form of additional authentication in order to boost the authentication confidence level produced as shown in step 1820. In the simplest form, this may simply indicates that authentication was insufficient. A request to produce one or more additional authentication instances to improve the overall confidence level of the authentication may also be sent.

If the user provides some additional authentication instances in step 1825, then the transaction engine 205 adds these authentication instances to the authentication data for the transaction and forwards it to the authentication engine 215 as shown in step 1015 (see FIGURE 10), and the authentication is reevaluated based upon both the pre-existing authentication instances for this transaction and the newly provided authentication instances.

An additional type of authentication may be a request from the trust engine 110 to make some form of person-to-person contact between the trust engine 110 operator (or a trusted associate) and the user, for example, by phone call. This phone call or other non-computer authentication can be used to provide personal contact with the individual and also to conduct some form of questionnaire based authentication. This also may give the opportunity to verify an originating telephone number and potentially a voice analysis of the user when he calls in. Even if no additional authentication data can be provided, the additional context associated with the user's phone number may improve the reliability of the authentication context. Any revised data or circumstances based upon this phone call are fed into the trust engine 110 for use in consideration of the authentication request.

Additionally, in step 1820 the trust engine 110 may provide an opportunity for the user to purchase insurance, effectively buying a more confident authentication. The operator of the trust engine 110 may, at times, only want to make such an option available if the confidence level of the authentication is above a certain threshold to begin with. In effect, this user side insurance is a way for the trust engine 110 to vouch

for the user when the authentication meets the normal required trust level of the trust engine 110 for authentication, but does not meet the required trust level of the vendor for this transaction. In this way, the user may still successfully authenticate to a very high level as may be required by the vendor, even though he only has authentication instances which produce confidence sufficient for the trust engine 110.

This function of the trust engine 110 allows the trust engine 110 to vouch for someone who is authenticated to the satisfaction of the trust engine 110, but not of the vendor. This is analogous to the function performed by a notary in adding his signature to a document in order to indicate to someone reading the document at a later time that the person whose signature appears on the document is in fact the person who signed it. The signature of the notary testifies to the act of signing by the user. In the same way, the trust engine is providing an indication that the person transacting is who they say they are.

However, because the trust engine 110 is artificially boosting the level of confidence provided by the user, there is a greater risk to the trust engine 110 operator, since the user is not actually meeting the required trust level of the vendor. The cost of the insurance is designed to offset the risk of a false positive authentication to the trust engine 110 (who may be effectively notarizing the authentications of the user). The user pays the trust engine 110 operator to take the risk of authenticating to a higher level of confidence than has actually been provided.

Because such an insurance system allows someone to effectively buy a higher confidence rating from the trust engine 110, both vendors and users may wish to prevent the use of user side insurance in certain transactions. Vendors may wish to limit positive authentications to circumstances where they know that actual authentication data supports the degree of confidence which they require and so may indicate to the trust engine 110 that user side insurance is not to be allowed. Similarly, to protect his online identity, a user may wish to prevent the use of user side insurance on his account, or may wish to limit its use to situations where the authentication confidence level without the insurance is higher than a certain limit. This may be used as a security measure to prevent someone from overhearing a password or stealing a smart card and using them to falsely authenticate to a low level of confidence, and then

purchasing insurance to produce a very high level of (false) confidence. These factors may be evaluated in determining whether user side insurance is allowed.

If user purchases insurance in step 1840, then the authentication confidence level is adjusted based upon the insurance purchased in step 1845, and the authentication confidence level and required trust level are again compared in step 1730 (see FIGURE 17). The process continues from there, and may lead to either a positive authentication in step 1740 (see FIGURE 17), or back into the trust arbitrage process in step 1750 for either further arbitrage (if allowed) or a negative authentication in step 1810 if further arbitrage is prohibited.

In addition to sending a message to the user in step 1820, the transaction engine 205 may also send a message to the vendor in step 1830 which indicates that a pending authentication is currently below the required trust level. The message may also offer various options on how to proceed to the vendor. One of these Options is to simply inform the vendor of what the current authentication confidence level is and ask if the vendor wishes to maintain their current unfulfilled required trust level. This may be beneficial because in some cases, the vendor may have independent means for authenticating the transaction or may have been using a default set of requirements which generally result in a higher required level being initially specified than is actually needed for the particular transaction at hand.

For instance, it may be standard practice that all incoming purchase order transactions with the vendor are expected to meet a 98% trust level. However, if an order was recently discussed by phone between the vendor and a long-standing customer, and immediately thereafter the transaction is authenticated, but only to a 93% confidence level, the vendor may wish to simply lower the acceptance threshold for this transaction, because the phone call effectively provides additional authentication to the vendor. In certain circumstances, the vendor may be willing to lower their required trust level, but not all the way to the level of the current authentication confidence. For instance, the vendor in the above example might consider that the phone call prior to the order might merit a 4% reduction in the degree of trust needed; however, this is still greater than the 93% confidence produced by the user.

If the vendor does adjust their required trust level in step 1835, then the authentication confidence level produced by the authentication and the required trust level are compared in step 1730 (see FIGURE 17). If the confidence level now exceeds the required trust level, a positive authentication may be generated in the transaction engine 205 in step 1740 (see FIGURE 17). If not, further arbitrage may be attempted as discussed above if it is permitted.

In addition to requesting an adjustment to the required trust level, the transaction engine 205 may also offer vendor side insurance to the vendor requesting the authentication. This insurance serves a similar purpose to that described above for the user side insurance. Here, however, rather than the cost corresponding to the risk being taken by the trust engine 110 in authenticating above the actual authentication confidence level produced, the cost of the insurance corresponds to the risk being taken by the vendor in accepting a lower trust level in the authentication.

Instead of just lowering their actual required trust level, the vendor has the option of purchasing insurance to protect itself from the additional risk associated with a lower level of trust in the authentication of the user. As described above, it may be advantageous for the vendor to only consider purchasing such insurance to cover the trust gap in conditions where the existing authentication is already above a certain threshold.

The availability of such vendor side insurance allows the vendor the option to either: lower his trust requirement directly at no additional cost to himself, bearing the risk of a false authentication himself (based on the lower trust level required); or, buying insurance for the trust gap between the authentication confidence level and his requirement, with the trust engine 110 operator bearing the risk of the lower confidence level which has been provided. By purchasing the insurance, the vendor effectively keeps his high trust level requirement; because the risk of a false authentication is shifted to the trust engine 110 operator.

If the vendor purchases insurance in step 1840, the authentication confidence level and required trust levels are compared in step 1730 (see FIGURE 17), and the process continues as described above.

Note that it is also possible that both the user and the vendor respond to messages from the trust engine 110. Those of skill in the art will recognize that there are multiple ways in which such situations can be handled. One advantageous mode of handling the possibility of multiple responses is simply to treat the responses in a first-come, first-served manner. For example, if the vendor responds with a lowered required trust level and immediately thereafter the user also purchases insurance to raise his authentication level, the authentication is first reevaluated based upon the lowered trust requirement from the vendor. If the authentication is now positive, the user's insurance purchase is ignored. In another advantageous mode of operation, the user might only be charged for the level of insurance required to meet the new, lowered trust requirement of the vendor (if a trust gap remained even with the lowered vendor trust requirement).

If no response from either party is received during the trust arbitrage process at step 1850 within the time limit set for the authentication, the arbitrage is reevaluated in step 1805. This effectively begins the arbitrage process again. If the time limit was final or other circumstances prevent further arbitrage in step 1805, a negative authentication is generated by the transaction engine 205 in step 1810 and returned to the vendor in step 1055 (see FIGURE 10). If not, new messages may be sent to the user and vendor, and the process may be repeated as desired.

Note that for certain types of transactions, for instance, digitally signing documents which are not part of a transaction, there may not necessarily be a vendor or other third party; therefore the transaction is primarily between the user and the trust engine 110. In circumstances such as these, the trust engine 110 will have its own required trust level which must be satisfied in order to generate a positive authentication. However, in such circumstances, it will often not be desirable for the trust engine 110 to offer insurance to the user in order for him to raise the confidence of his own signature.

The process described above and shown in FIGURES 16-18 may be carried out using various communications modes as described above with reference to the trust engine 110. For instance, the messages may be web-based and sent using SSL connections between the trust engine 110 and applets downloaded in real time to

browsers running on the user or vendor systems. In an alternate mode of operation, certain dedicated applications may be in use by the user and vendor which facilitate such arbitrage and insurance transactions. In another alternate mode of operation, secure email operations may be used to mediate the arbitrage described above, thereby allowing deferred evaluations and batch processing of authentications. Those of skill in the art will recognize that different communications modes may be used as are appropriate for the circumstances and authentication requirements of the vendor.

The following description with reference to FIGURE 19 describes a sample transaction which integrates the various aspects of the present invention as described above. This example illustrates the overall process between a user and a vendor as mediates by the trust engine 110. Although the various steps and components as described in detail above may be used to carry out the following transaction, the process illustrated focuses on the interaction between the trust engine 110, user and vendor.

The transaction begins when the user, while viewing web pages online, fills out an order form on the web site of the vendor in step 1900. The user wishes to submit this order form to the vendor, signed with his digital signature. In order to do this, the user submits the order form with his request for a signature to the trust engine 110 in step 1905. The user will also provide authentication data which will be used as described above to authenticate his identity.

In step 1910 the authentication data is compared to the enrollment data by the trust engine 110 as discussed above, and if a positive authentication is produced, the hash of the order form, signed with the private key of the user, is forwarded to the vendor along with the order form itself.

The vendor receives the signed form in step 1915, and then the vendor will generate an invoice or other contract related to the purchase to be made in step 1920. This contract is sent back to the user with a request for a signature in step 1925. The vendor also sends an authentication request for this contract transaction to the trust engine 110 in step 1930 including a hash of the contract which will be signed by both parties. To allow the contract to be digitally signed by both parties, the vendor also includes authentication data for itself so that the vendor's signature upon the contract can later be verified if necessary.

As discussed above, the trust engine 110 then verifies the authentication data provided by the vendor to confirm the vendor's identity, and if the data produces a positive authentication in step 1935, continues with step 1955 when the data is received from the user. If the vendor's authentication data does not match the enrollment data of the vendor to the desired degree, a message is returned to the vendor requesting further authentication. Trust arbitrage may be performed here if necessary, as described above, in order for the vendor to successfully authenticate itself to the trust engine 110.

When the user receives the contract in step 1940, he reviews it, generates authentication data to sign it if it is acceptable in step 1945, and then sends a hash of the contract and his authentication data to the trust engine 110 in step 1950. The trust engine 110 verifies the authentication data in step 1955 and if the authentication is good, proceeds to process the contract as described below. As discussed above with reference to FIGURES 17 and 18, trust arbitrage may be performed as appropriate to close any trust gap which exists between the authentication confidence level and the required authentication level for the transaction.

The trust engine 110 signs the hash of the contract with the user's private key, and sends this signed hash to the vendor in step 1960, signing the complete message on its own behalf, i.e., including a hash of the complete message (including the user's signature) encrypted with the private key 510 of the trust engine 110. This message is received by the vendor in step 1965. The message represents a signed contract (hash of contract encrypted using user's private key) and a receipt from the trust engine 110 (the hash of the message including the signed contract, encrypted using the trust engine 110's private key).

The trust engine 110 similarly prepares a hash of the contract with the vendor's private key in step 1970, and forwards this to the user, signed by the trust engine 110. In this way, the user also receives a copy of the contract, signed by the vendor, as well as a receipt, signed by the trust engine 110, for delivery of the signed contract in step 1975.

In addition to the foregoing, an additional aspect of the invention provides a cryptographic Service Provider Module (SPM) which may be available to a client side application as a means to access functions provided by the trust engine 110 described

above. One advantageous way to provide such a service is for the cryptographic SPM is to mediate communications between a third party Application Programming Interface (API) and a trust engine 110 which is accessible via a network or other remote connection. A sample cryptographic SPM is described below with reference to FIGURE 20.

For example, on a typical system, a number of API's are available to programmers. Each API provides a set of function calls which may be made by an application 2000 running upon the system. Examples of API's which provide programming interfaces suitable for cryptographic functions, authentication functions, and other security function include the Cryptographic API (CAPI) 2010 provided by Microsoft with its Windows operating systems, and the Common Data Security Architecture (CDSA), sponsored by IBM, Intel and other members of the Open Group. CAPI will be used as an exemplary security API in the discussion that follows. However, the cryptographic SPM described could be used with CDSA or other security API's as are known in the art.

This API is used by a user system 105 or vendor system 120 when a call is made for a cryptographic function. Included among these functions may be requests associated with performing various cryptographic operations, such as encrypting a document with a particular key, signing a document, requesting a digital certificate, verifying a signature upon a signed document, and such other cryptographic functions as are described herein or known to those of skill in the art.

Such cryptographic functions are normally performed locally to the system upon which CAPI 2010 is located. This is because generally the functions called require the use of either resources of the local user system 105, such as a fingerprint reader, or software functions which are programmed using libraries which are executed on the local machine. Access to these local resources is normally provided by one or more Service Provider Modules (SPM's) 2015, 2020 as referred to above which provide resources with which the cryptographic functions are carried out. Such SPM's may include software libraries 2015 to perform encrypting or decrypting operations, or drivers and applications 2020 which are capable of accessing specialized hardware 2025, such as biometric scanning devices. In much the way that CAPI 2010 provides

functions which may be used by an application 2000 of the system 105, the SPM's 2015, 2020 provide CAPI with access to the lower level functions and resources associated with the available services upon the system.

In accordance with the invention, it is possible to provide a cryptographic SPM 2030 which is capable of accessing the cryptographic functions provided by the trust engine 110 and making these functions available to an application 2000 through CAPI 2010. Unlike embodiments where CAPI 2010 is only able to access resources which are locally available through SPM's 2015, 2020, a cryptographic SPM 2030 as described herein would be able to submit requests for cryptographic operations to a remotely-located, network-accessible trust engine 110 in order to perform the operations desired.

For instance, if an application 2000 has a need for a cryptographic operation, such as signing a document, the application 2000 makes a function call to the appropriate CAPI 2010 function. CAPI 2010 in turn will execute this function, making use of the resources which are made available to it by the SPM's 2015, 2020 and the cryptographic SPM 2030. In the case of a digital signature function, the cryptographic SPM 2030 will generate an appropriate request which will be sent to the trust engine 110 across the communication link 125.

The operations which occur between the cryptographic SPM 2030 and the trust engine 110 are the same operations that would be possible between any other system and the trust engine 110. However, these functions are effectively made available to a user system 105 through CAPI 2010 such that they appear to be locally available upon the user system 105 itself. However, unlike ordinary SPM's 2015, 2020, the functions are being carried out on the remote trust engine 110 and the results relayed to the cryptographic SPM 2030 in response to appropriate requests across the communication link 125.

This cryptographic SPM 2030 makes a number of operations available to the user system 105 or a vendor system 120 which might not otherwise be available. These functions include without limitation: encryption and decryption of documents; issuance of digital certificates; digital signing of documents; verification of digital signatures; and such other operations as will be apparent to those of skill in the art.

In a separate embodiment, the present invention comprises a complete system for performing the data securing methods of the present invention on any data set. The computer system of this embodiment comprises a data splitting module that comprises the functionality shown in FIGURE 8 and described herein. In one embodiment of the present invention, the data splitting module comprises a parser program or software suite which comprises data splitting, encryption and decryption, reconstitution or reassembly functionality. This embodiment may further comprise a data storage facility or multiple data storage facilities, as well. The data splitting module, or parser, comprises a cross-platform software module suite which integrates within an electronic infrastructure, or as an add-on to any application which requires the ultimate security of its data elements. This parsing process operates on any type of data set, and on any and all file types, or in a database on any row, column or cell of data in that database.

The parsing process of the present invention may, in one embodiment, be designed in a modular tiered fashion, and any encryption process is suitable for use in the process of the present invention. The modular tiers of the parsing process of the present invention may include, but are not limited to, 1) cryptographic split, dispersed and securely stored in multiple locations; 2) encrypt, cryptographically split, dispersed and securely stored in multiple locations; 3) encrypt, cryptographically split, encrypt each share, then dispersed and securely stored in multiple locations; and 4) encrypt, cryptographically split, encrypt each share with a different type of encryption than was used in the first step, then dispersed and securely stored in multiple locations.

The process comprises, in one embodiment, splitting of the data according to the contents of a generated random number, or key and performing the same cryptographic splitting of the key used in the encryption of splitting of the data to be secured into two or more portions, or shares, of parsed data, and in one embodiment, preferably into four or more portions of parsed data, encrypting all of the portions, then scattering and storing these portions back into the database, or relocating them to any named device, fixed or removable, depending on the requestor's need for privacy and security. Alternatively, in another embodiment, encryption may occur prior to the splitting of the data set by the splitting module or parser. The original data processed as described in this embodiment is encrypted and obfuscated and is secured. The dispersion of the

encrypted elements, if desired, can be virtually anywhere, including, but not limited to, a single server or data storage device, or among separate data storage facilities or devices. Encryption key management in one embodiment may be included within the software suite, or in another embodiment may be integrated into an existing infrastructure or any other desired location.

A cryptographic split (cryptosplit) partitions the data into N number of shares. The partitioning can be on any size unit of data, including an individual bit, bits, bytes, kilobytes, megabytes, or larger units, as well as any pattern or combination of data unit sizes whether predetermined or randomly generated. The units can also be of different sized, based on either a random or predetermined set of values. This means the data can be viewed as a sequence of these units. In this manner the size of the data units themselves may render the data more secure, for example by using one or more predetermined or randomly generated pattern, sequence or combination of data unit sizes. The units are then distributed (either randomly or by a predetermined set of values) into the N shares. This distribution could also involve a shuffling of the order of the units in the shares. It is readily apparent to those of ordinary skill in the art that the distribution of the data units into the shares may be performed according to a wide variety of possible selections, including but not limited to size-fixed, predetermined sizes, or one or more combination, pattern or sequence of data unit sizes that are predetermined or randomly generated.

One example of this cryptographic split process, or cryptosplit, would be to consider the data to be 23 bytes in size, with the data unit size chosen to be one byte, and with the number of shares selected to be 4. Each byte would be distributed into one of the 4 shares. Assuming a random distribution, a key would be obtained to create a sequence of 23 random numbers (r1, r2, r3 through r23), each with a value between 1 and 4 corresponding to the four shares. Each of the units of data (in this example 23 individual bytes of data) is associated with one of the 23 random numbers corresponding to one of the four shares. The distribution of the bytes of data into the four shares would occur by placing the first byte of the data into share number r1, byte two into share r2, byte three into share r3, through the 23rd byte of data into share r23. It is readily apparent to those of ordinary skill in the art that a wide variety of other

possible steps or combination or sequence of steps, including the size of the data units, may be used in the cryptosplit process of the present invention, and the above example is a non-limiting description of one process for cryptosplitting data. To recreate the original data, the reverse operation would be performed.

In another embodiment of the cryptosplit process of the present invention, an option for the cryptosplitting process is to provide sufficient redundancy in the shares such that only a subset of the shares are needed to reassemble or restore the data to its original or useable form. As a non-limiting example, the cryptosplit may be done as a "3 of 4" cryptosplit such that only three of the four shares are necessary to reassemble or restore the data to its original or useable form. This is also referred to as a "M of N cryptosplit" wherein N is the total number of shares, and M is at least one less than N. It is readily apparent to those of ordinary skill in the art that there are many possibilities for creating this redundancy in the cryptosplitting process of the present invention.

In one embodiment of the cryptosplitting process of the present invention, each unit of data is stored in two shares, the primary share and the backup share. Using the "3 of 4" cryptosplitting process described above, any one share can be missing, and this is sufficient to reassemble or restore the original data with no missing data units since only three of the total four shares are required. As described herein, a random number is generated that corresponds to one of the shares. The random number is associated with a data unit, and stored in the corresponding share, based on a key. One key is used, in this embodiment, to generate the primary and backup share random number. As described herein for the cryptosplitting process of the present invention, a set of random numbers (also referred to as primary share numbers) from 0 to 3 are generated equal to the number of data units. Then another set of random numbers is generated (also referred to as backup share numbers) from 1 to 3 equal to the number of data units. Each unit of data is then associated with a primary share number and a backup share number. Alternatively, a set of random numbers may be generated that is fewer than the number of data units, and repeating the random number set, but this may reduce the security of the sensitive data. The primary share number is used to determine into which share the data unit is stored. The backup share number is combined with the primary share number to create a third share number between 0 and 3, and this number

is used to determine into which share the data unit is stored. In this example, the equation to determine the third share number is:

(primary share number + backup share number) MOD 4 = third share number.

In the embodiment described above where the primary share number is between 0 and 3, and the backup share number is between 1 and 3 ensures that the third share number is different from the primary share number. This results in the data unit being stored in two different shares. It is readily apparent to those of ordinary skill in the art that there are many ways of performing redundant cryptosplitting and non-redundant cryptosplitting in addition to the embodiments disclosed herein. For example, the data units in each share could be shuffled utilizing a different algorithm. This data unit shuffling may be performed as the original data is split into the data units, or after the data units are placed into the shares, or after the share is full, for example.

The various cryptosplitting processes and data shuffling processes described herein, and all other embodiments of the cryptosplitting and data shuffling methods of the present invention may be performed on data units of any size, including but not limited to, as small as an individual bit, bits, bytes, kilobytes, megabytes or larger.

[0285] An example of one embodiment of source code that would perform the cryptosplitting process described herein is:

[0286] DATA [1:24] – array of bytes with the data to be split

[0287] SHARES[0:3; 1:24] – 2-dimensionalarray with each row representing one of the shares

[0288] RANDOM[1:24] – array random numbers in the range of 0..3

[0289] S1 = 1;

[0290] S2 = 1;

[0291] S3 = 1;

[0292] S4 = 1;

[0293] For J = 1 to 24 do

Begin

IF RANDOM[J[==0 then

Begin

```
                SHARES[1,S1] = DATA [J];
                S1 = S1 + 1;
                End
        ELSE IF RANDOM[J[ ==1 then
                Begin
                SHARES[2,S2] = DATA [J];
                S2 = S2 + 1;
                END
        ELSE IF RANDOM[J[ ==2 then
                Begin
                Shares[3,S3] = data [J];
                S3 = S3 + 1;
                End
        Else  begin
                Shares[4,S4] = data [J];
                S4 = S4 + 1;
                End;
    END;
```

An example of one embodiment of source code that would perform the cryptosplitting RAID process described herein is:

Generate two sets of numbers, PrimaryShare is 0 to 3, BackupShare is 1 to 3. Then put each data unit into share[primaryshare[1]] and share[(primaryshare[1]+backupshare[1]) mod 4, with the same process as in cryptosplitting described above. This method will be scalable to any size N, where only N-1 shares are necessary to restore the data.

The retrieval, recombining, reassembly or reconstituting of the encrypted data elements may utilize any number of authentication techniques, including, but not limited to, biometrics, such as fingerprint recognition, facial scan, hand scan, iris scan, retinal scan, ear scan, vascular pattern recognition or DNA analysis. The data splitting or parser modules of the present invention may be integrated into a wide variety of infrastructure products or applications as desired.

Traditional encryption technologies known in the art rely on one or more key used to encrypt the data and render it unusable without the key. The data, however, remains whole and intact and subject to attack. The parser software suite of the present invention, in one embodiment, addresses this problem by performing a cryptographic split or parsing of the encrypted file into two or more portions or shares, and in another embodiment, preferably four or more shares, adding another layer of encryption to each share of the data, then storing the shares in different physical and/or logical locations. When one or more data shares are physically removed from the system, either by using a removable device, such as a data storage device, or by placing the share under another party's control, any possibility of compromise of secured data is effectively removed.

An example of one embodiment of the parser software suite of the present invention and an example of how it may be utilized is shown in FIGURE 21 and described below. However, it is readily apparent to those of ordinary skill in the art that the parser software suite of the present invention may be utilized in a wide variety of ways in addition to the non-limiting example below. As a deployment option, and in one embodiment, the parser may be implemented with external session key management or secure internal storage of session keys. Upon implementation, a Parser Master Key will be generated which will be used for securing the application and for encryption purposes. It should be also noted that the incorporation of the Parser Master key in the resulting secured data allows for a flexibility of sharing of secured data by individuals within a workgroup, enterprise or extended audience.

As shown in Figure 21, this embodiment of the present invention shows the steps of the process performed by the parser software suite on data to store the session master key with the parsed data:

1. Generating a session master key and encrypt the data using RS1 stream cipher.
2. Separating the resulting encrypted data into four shares or portions of parsed data according to the pattern of the session master key.
3. In this embodiment of the method, the session master key will be stored along with the secured data shares in a data depository.

Separating the session master key according to the pattern of the Parser Master Key and append the key data to the encrypted parsed data.

4. The resulting four shares of data will contain encrypted portions of the original data and portions of the session master key. Generate a stream cipher key for each of the four data shares.

5. Encrypting each share, then store the encryption keys in different locations from the encrypted data portions or shares: Share 1 gets Key 4, Share 2 gets Key 1, Share 3 gets Key 2, Share 4 gets Key 3.

To restore the original data format, the steps are reversed.

It is readily apparent to those of ordinary skill in the art that certain steps of the methods described herein may be performed in different order, or repeated multiple times, as desired. It is also readily apparent to those skilled in the art that the portions of the data may be handled differently from one another. For example, multiple parsing steps may be performed on only one portion of the parsed data. Each portion of parsed data may be uniquely secured in any desirable way provided only that the data may be reassembled, reconstituted, reformed, decrypted or restored to its original or other usable form.

As shown in FIGURE 22 and described herein, another embodiment of the present invention comprises the steps of the process performed by the parser software suite on data to store the session master key data in one or more separate key management table:

1. Generating a session master key and encrypt the data using RS1 stream cipher.

2. Separating the resulting encrypted data into four shares or portions of parsed data according to the pattern of the session master key.

3. In this embodiment of the method of the present invention, the session master key will be stored in a separate key management table in a data depository. Generating a unique transaction ID for this transaction. Storing the transaction ID and session master key in a separate key management table. Separating the transaction ID according to the

pattern of the Parser Master Key and append the data to the encrypted parsed or separated data.

4. The resulting four shares of data will contain encrypted portions of the original data and portions of the transaction ID.
5. Generating a stream cipher key for each of the four data shares.
6. Encrypting each share, then store the encryption keys in different locations from the encrypted data portions or shares: Share 1 gets Key 4, Share 2 gets Key 1, Share 3 gets Key 2, Share 4 gets Key 3.

To restore the original data format, the steps are reversed.

It is readily apparent to those of ordinary skill in the art that certain steps of the method described herein may be performed in different order, or repeated multiple times, as desired. It is also readily apparent to those skilled in the art that the portions of the data may be handled differently from one another. For example, multiple separating or parsing steps may be performed on only one portion of the parsed data. Each portion of parsed data may be uniquely secured in any desirable way provided only that the data may be reassembled, reconstituted, reformed, decrypted or restored to its original or other usable form.

As shown in Figure 23, this embodiment of the present invention shows the steps of the process performed by the parser software suite on data to store the session master key with the parsed data:

1. Accessing the parser master key associated with the authenticated user
2. Generating a unique Session Master key
3. Derive an Intermediary Key from an exclusive OR function of the Parser Master Key and Session Master key
4. Optional encryption of the data using an existing or new encryption algorithm keyed with the Intermediary Key.
5. Separating the resulting optionally encrypted data into four shares or portions of parsed data according to the pattern of the Intermediary key.
6. In this embodiment of the method, the session master key will be stored along with the secured data shares in a data depository.

Separating the session master key according to the pattern of the Parser Master Key and append the key data to the optionally encrypted parsed data shares.

7. The resulting multiple shares of data will contain optionally encrypted portions of the original data and portions of the session master key.
8. Optionally generate an encryption key for each of the four data shares.
9. Optionally encrypting each share with an existing or new encryption algorithm, then store the encryption keys in different locations from the encrypted data portions or shares: for example, Share 1 gets Key 4, Share 2 gets Key 1, Share 3 gets Key 2, Share 4 gets Key 3.

To restore the original data format, the steps are reversed.

It is readily apparent to those of ordinary skill in the art that certain steps of the methods described herein may be performed in different order, or repeated multiple times, as desired. It is also readily apparent to those skilled in the art that the portions of the data may be handled differently from one another. For example, multiple parsing steps may be performed on only one portion of the parsed data. Each portion of parsed data may be uniquely secured in any desirable way provided only that the data may be reassembled, reconstituted, reformed, decrypted or restored to its original or other usable form.

As shown in FIGURE 24 and described herein, another embodiment of the present invention comprises the steps of the process performed by the parser software suite on data to store the session master key data in one or more separate key management table:

1. Accessing the Parser Master Key associated with the authenticated user
2. Generating a unique Session Master Key
3. Derive an Intermediary Key from an exclusive OR function of the Parser Master Key and Session Master key
4. Optionally encrypt the data using an existing or new encryption algorithm keyed with the Intermediary Key.

5. Separating the resulting optionally encrypted data into four shares or portions of parsed data according to the pattern of the Intermediary Key.

6. In this embodiment of the method of the present invention, the session master key will be stored in a separate key management table in a data depository. Generating a unique transaction ID for this transaction. Storing the transaction ID and session master key in a separate key management table or passing the Session Master Key and transaction ID back to the calling program for external management. Separating the transaction ID according to the pattern of the Parser Master Key and append the data to the optionally encrypted parsed or separated data.

7. The resulting four shares of data will contain optionally encrypted portions of the original data and portions of the transaction ID.

8. Optionally generate an encryption key for each of the four data shares.

9. Optionally encrypting each share, then store the encryption keys in different locations from the encrypted data portions or shares. For example: Share 1 gets Key 4, Share 2 gets Key 1, Share 3 gets Key 2, Share 4 gets Key 3.

To restore the original data format, the steps are reversed.

It is readily apparent to those of ordinary skill in the art that certain steps of the method described herein may be performed in different order, or repeated multiple times, as desired. It is also readily apparent to those skilled in the art that the portions of the data may be handled differently from one another. For example, multiple separating or parsing steps may be performed on only one portion of the parsed data. Each portion of parsed data may be uniquely secured in any desirable way provided only that the data may be reassembled, reconstituted, reformed, decrypted or restored to its original or other usable form.

A wide variety of encryption methodologies are suitable for use in the methods of the present invention, as is readily apparent to those skilled in the art. The One Time

Pad algorithm, is often considered one of the most secure encryption methods, and is suitable for use in the method of the present invention. Using the One Time Pad algorithm requires that a key be generated which is as long as the data to be secured. The use of this method may be less desirable in certain circumstances such as those resulting in the generation and management of very long keys because of the size of the data set to be secured. In the One-Time Pad (OTP) algorithm, the simple exclusive-or function, XOR, is used. For two binary streams x and y of the same length, x XOR y means the bitwise exclusive-or of x and y.

At the bit level is generated:

0 XOR 0 = 0

0 XOR 1 = 1

1 XOR 0 = 1

1 XOR 1 = 0

An example of this process is described herein for an n-byte secret, s, (or data set) to be split. The process will generate an n-byte random value, a, and then set:

b = a XOR s.

Note that one can derive "s" via the equation:

s = a XOR b.

The values a and b are referred to as shares or portions and are placed in separate depositories. Once the secret s is split into two or more shares, it is discarded in a secure manner.

The parser software suite of the present invention may utilize this function, performing multiple XOR functions incorporating multiple distinct secret key values: K1, K2, K3, Kn, K5. At the beginning of the operation, the data to be secured is passed through the first encryption operation, secure data = data XOR secret key 5:

S = D XOR K5

In order to securely store the resulting encrypted data in, for example, four shares, S1, S2, S3, Sn, the data is parsed into "n" segments, or shares, according to the value of K5. This operation results in "n" pseudorandom shares of the original encrypted data. Subsequent XOR functions may then be performed on each share with

the remaining secret key values, for example: Secure data segment 1 = encrypted data share 1 XOR secret key 1:

SD1 = S1 XOR K1

SD2 = S2 XOR K2

SD3 = S3 XOR K3

SDn = Sn XOR Kn.

In one embodiment, it may not be desired to have any one depository contain enough information to decrypt the information held there, so the key required to decrypt the share is stored in a different data depository:

Depository 1: SD1, Kn

Depository 2: SD2, K1

Depository 3: SD3, K2

Depository n: SDn, K3.

Additionally, appended to each share may be the information required to retrieve the original session encryption key, K5. Therefore, in the key management example described herein, the original session master key is referenced by a transaction ID split into "n" shares according to the contents of the installation dependant Parser Master Key (TID1, TID2, TID3, TIDn):

Depository 1: SD1, Kn, TID1

Depository 2: SD2, K1, TID2

Depository 3: SD3, K2, TID3

Depository n: SDn, K3, TIDn.

In the incorporated session key example described herein, the session master key is split into "n" shares according to the contents of the installation dependant Parser Master Key (SK1, SK2, SK3, SKn):

Depository 1: SD1, Kn, SK1

Depository 2: SD2, K1, SK2

Depository 3: SD3, K2, SK3

Depository n: SDn, K3, SKn.

Unless all four shares are retrieved, the data cannot be reassembled according to this example. Even if all four shares are captured, there is no possibility of

reassembling or restoring the original information without access to the session master key and the Parser Master Key.

This example has described an embodiment of the method of the present invention, and also describes, in another embodiment, the algorithm used to place shares into depositories so that shares from all depositories can be combined to form the secret authentication material. The computations needed are very simple and fast. However, with the One Time Pad (OTP) algorithm there may be circumstances that cause it to be less desirable, such as a large data set to be secured, because the key size is the same size as the data to be stored. Therefore, there would be a need to store and transmit about twice the amount of the original data which may be less desirable under certain circumstances.

Stream Cipher RS1

The stream cipher RS1 splitting technique is very similar to the OTP splitting technique described herein. Instead of an n-byte random value, an n' = min(n, 16)-byte random value is generated and used to key the RS1 Stream Cipher algorithm. The advantage of the RS1 Stream Cipher algorithm is that a pseudorandom key is generated from a much smaller seed number. The speed of execution of the RS1 Stream Cipher encryption is also rated at approximately 10 times the speed of the well known in the art Triple DES encryption without compromising security. The RS1 Stream Cipher algorithm is well known in the art, and may be used to generate the keys used in the XOR function. The RS1 Stream Cipher algorithm is interoperable with other commercially available stream cipher algorithms, such as the RC4™ stream cipher algorithm of RSA Security, Inc and is suitable for use in the methods of the present invention.

Using the key notation above, K1 thru K5 are now an n' byte random values and we set:

SD1 = S1 XOR E(K1)

SD2 = S2 XOR E(K2)

SD3 = S3 XOR E(K3)

$$SDn = Sn \text{ XOR } E(Kn)$$

where E(K1) thru E(Kn) are the first n' bytes of output from the RS1 Stream Cipher algorithm keyed by K1 thru Kn. The shares are now placed into data depositories as described herein.

In this stream cipher RS1 algorithm, the required computations needed are nearly as simple and fast as the OTP algorithm. The benefit in this example using the RS1 Stream Cipher is that the system needs to store and transmit on average only about 16 bytes more than the size of the original data to be secured per share. When the size of the original data is more than 16 bytes, this RS1 algorithm is more efficient than the OTP algorithm because it is simply shorter. It is readily apparent to those of ordinary skill in the art that a wide variety of encryption methods or algorithms are suitable for use in the present invention, including, but not limited to RS1, OTP, RC4™, Triple DES and AES.

There are major advantages provided by the data security methods and computer systems of the present invention over traditional encryption methods. One advantage is the security gained from moving shares of the data to different locations on one or more data depositories or storage devices, that may be in different logical, physical or geographical locations. When the shares of data are split physically and under the control of different personnel, for example, the possibility of compromising the data is greatly reduced.

[0357] Another advantage provided by the methods and system of the present invention is the combination of the steps of the method of the present invention for securing data to provide a comprehensive process of maintaining security of sensitive data. The data is encrypted with a secure key and split into one or more shares, and in one embodiment, four shares, according to the secure key. The secure key is stored safely with a reference pointer which is secured into four shares according to a secure key. The data shares are then encrypted individually and the keys are stored safely with different encrypted shares. When combined, the entire process for securing data according to the methods disclosed herein becomes a comprehensive package for data security.

The data secured according to the methods of the present invention is readily retrievable and restored, reconstituted, reassembled, decrypted, or otherwise returned into its original or other suitable form for use. In order to restore the original data, the following items may be utilized:

1. All shares or portions of the data set.
2. Knowledge of and ability to reproduce the process flow of the method used to secure the data.
3. Access to the session master key.
4. Access to the Parser Master Key.

Therefore, it may be desirable to plan a secure installation wherein at least one of the above elements may be physically separated from the remaining components of the system (under the control of a different system administrator for example).

Protection against a rogue application invoking the data securing methods application may be enforced by use of the Parser Master Key. A mutual authentication handshake between the Secure Parser™ and the application may be required in this embodiment of the present invention prior to any action taken.

The security of the system dictates that there be no "backdoor" method for recreation of the original data. For installations where data recovery issues may arise, the Secure Parser™ can be enhanced to provide a mirror of the four shares and session master key depository. Hardware options such as RAID (redundant array of inexpensive disks, used to spread information over several disks) and software options such as replication can assist as well in the data recovery planning.

Key Management

In one embodiment of the present invention, the data securing method uses three sets of keys for an encryption operation. Each set of keys may have individual key storage, retrieval, security and recovery options, based on the installation. The keys that may be used, include, but are not limited to:

1. The Parser Master Key

EX1010, Page 2388

This key is an individual key associated with the installation of the data parser. It is installed on the server on which the parser has been deployed. There are a variety of options suitable for securing this key including, but not limited to, a smart card, separate hardware key store, standard key stores, custom key stores or within a secured database table, for example.

2. The Session Master Key

A Session Master Key may be generated each time data is secured. The Session Master Key is used to encrypt the data prior to the parsing operation. It may also be incorporated (if the Session Master Key is not integrated into the parsed data) as a means of parsing the encrypted data. The Session Master Key may be secured in a variety of manners, including, but not limited to, a standard key store, custom key store, separate database table, or secured within the encrypted shares, for example.

3. The Share Encryption Keys

For each share or portions of a data set that is created, an individual Share Encryption Key may be generated to further encrypt the shares. The Share Encryption Keys may be stored in different shares than the share that was encrypted.

It is readily apparent to those of ordinary skill in the art that the data securing methods and computer system of the present invention are widely applicable to any type of data in any setting or environment. In addition to commercial applications conducted over the Internet or between customers and vendors, the data securing methods and computer systems of the present invention are highly applicable to non-commercial or private settings or environments. Any data set that is desired to be kept secure from any unauthorized user may be secured using the methods and systems described herein. For example, access to a particular database within a company or organization may be advantageously restricted to only selected users by employing the methods and systems of the present invention for securing data. Another example is the generation, modification or access to documents wherein it is desired to restrict access or prevent unauthorized or accidental access or disclosure outside a group of selected individuals, computers or workstations. These and other examples of the ways in which the methods and systems of data securing of the present invention are applicable to any non-commercial or commercial environment or setting for any setting, including, but

not limited to any organization, government agency or corporation.

In another embodiment of the present invention, the data securing method uses three sets of keys for an encryption operation. Each set of keys may have individual key storage, retrieval, security and recovery options, based on the installation. The keys that may be used, include, but are not limited to:

1. The Parser Master Key

This key is an individual key associated with the installation of the data parser. It is installed on the server on which the parser has been deployed. There are a variety of options suitable for securing this key including, but not limited to, a smart card, separate hardware key store, standard key stores, custom key stores or within a secured database table, for example.

2. The Session Master Key

A Session Master Key may be generated each time data is secured. The Session Master Key is used in conjunction with the Parser Master key to derive the Intermediary Key. The Session Master Key may be secured in a variety of manners, including, but not limited to, a standard key store, custom key store, separate database table, or secured within the encrypted shares, for example.

3. The Intermediary Key

An Intermediary Key may be generated each time data is secured. The Intermediary Key is used to encrypt the data prior to the parsing operation. It may also be incorporated as a means of parsing the encrypted data.

4. The Share Encryption Keys

For each share or portions of a data set that is created, an individual Share Encryption Key may be generated to further encrypt the shares. The Share Encryption Keys may be stored in different shares than the share that was encrypted.

It is readily apparent to those of ordinary skill in the art that the data securing methods and computer system of the present invention are widely applicable to any type of data in any setting or environment. In addition to commercial applications conducted over the Internet or between customers and vendors, the data securing methods and computer systems of the present invention are highly applicable to non-commercial or private settings or environments. Any data set that is desired to be kept secure from any

unauthorized user may be secured using the methods and systems described herein. For example, access to a particular database within a company or organization may be advantageously restricted to only selected users by employing the methods and systems of the present invention for securing data. Another example is the generation, modification or access to documents wherein it is desired to restrict access or prevent unauthorized or accidental access or disclosure outside a group of selected individuals, computers or workstations. These and other examples of the ways in which the methods and systems of data securing of the present invention are applicable to any non-commercial or commercial environment or setting for any setting, including, but not limited to any organization, government agency or corporation.

Workgroup, Project, Individual PC/Laptop or Cross Platform Data Security

The data securing methods and computer systems of the present invention are also useful in securing data by workgroup, project, individual PC/Laptop and any other platform that is in use in, for example, businesses, offices, government agencies, or any setting in which sensitive data is created, handled or stored. The present invention provides methods and computer systems to secure data that is known to be sought after by organizations, such as the U.S. Government, for implementation across the entire government organization or between governments at a state or federal level.

The data securing methods and computer systems of the present invention provide the ability to not only parse flat files but also data fields, sets and or table of any type. Additionally, all forms of data are capable of being secured under this process, including, but not limited to, text, video, images, biometrics and voice data. Scalability, speed and data throughput of the methods of securing data of the present invention are only limited to the hardware the user has at their disposal.

In one embodiment of the present invention, the data securing methods are utilized as described below in a workgroup environment. In one embodiment, as shown in FIGURE 23 and described below, the Workgroup Scale data securing method of the present invention uses the private key management functionality of the TrustEngine to store the user/group relationships and the associated private keys (Parser Group Master Keys) necessary for a group of users to share secure data. The method of the present

invention has the capability to secure data for an enterprise, workgroup, or individual user, depending on how the Parser Master Key was deployed.

In one embodiment, additional key management and user/group management programs may be provided, enabling wide scale workgroup implementation with a single point of administration and key management. Key generation, management and revocation are handled by the single maintenance program, which all become especially important as the number of users increase. In another embodiment, key management may also be set up across one or several different system administrators, which may not allow any one person or group to control data as needed. This allows for the management of secured data to be obtained by roles, responsibilities, membership, rights, etc., as defined by an organization, and the access to secured data can be limited to just those who are permitted or required to have access only to the portion they are working on, while others, such as managers or executives, may have access to all of the secured data. This embodiment allows for the sharing of secured data among different groups within a company or organization while at the same time only allowing certain selected individuals, such as those with the authorized and predetermined roles and responsibilities, to observe the data as a whole. In addition, this embodiment of the methods and systems of the present invention also allows for the sharing of data among, for example, separate companies, or separate departments or divisions of companies, or any separate organization departments, groups, agencies, or offices, or the like, of any government or organization or any kind, where some sharing is required, but not any one party may be permitted to have access to all the data. Particularly apparent examples of the need and utility for such a method and system of the present invention are to allow sharing, but maintain security, in between government areas, agencies and offices, and between different divisions, departments or offices of a large company, or any other organization, for example.

An example of the applicability of the methods of the present invention on a smaller scale is as follows. A Parser Master key is used as a serialization or branding of the Parser to an organization. As the scale of use of the Parser Master key is reduced from the whole enterprise to a smaller workgroup, the data securing methods described herein are used to share files within groups of users.

In the example shown in FIGURE 25 and described below, there are six users defined along with their title or role within the organization. The side bar represents five possible groups that the users can belong to according to their role. The arrow represents membership by the user in one or more of the groups.

When configuring the SecureParser for use in this example, the system administrator accesses the user and group information from the operating system by a maintenance program. This maintenance program generates and assigns Parser Group Master Keys to users based on their membership in groups.

In this example, there are three members in the Senior Staff group. For this group, the actions would be:

1. Access Parser Group Master Key for the Senior Staff group (generate a key if not available);
2. Generate a digital certificate associating CEO with the Senior Staff group;
3. Generate a digital certificate associating CFO with the Senior Staff group;
4. Generate a digital certificate associating Vice President, Marketing with the Senior Staff group.

The same set of actions would be done for each group, and each member within each group. When the maintenance program is complete, the Parser Group Master Key becomes a shared credential for each member of the group. Revocation of the assigned digital certificate may be done automatically when a user is removed from a group through the maintenance program without affecting the remaining members of the group.

Once the shared credentials have been defined, the Parser process remains the same. When a file, document or data element is to be secured, the user is prompted for the target group to be used when securing the data. The resulting secured data is only accessible by other members of the target group. This functionality of the methods and systems of the present invention may be used with any other computer system or software platform, any may be, for example, integrated into existing application programs or used standalone for file security.

It is readily apparent to those of ordinary skill in the art that any one or combination of encryption algorithms are suitable for use in the methods and systems of the present invention. For example, the encryption steps may, in one embodiment, be repeated to produce a multi-layered encryption scheme. In addition, a different encryption algorithm, or combination of encryption algorithms, may be used in repeat encryption steps such that different encryption algorithms are applied to the different layers of the multi-layered encryption scheme. As such, the encryption scheme itself may become a component of the methods of the present invention for securing sensitive data from unauthorized use or access.

Additionally, other combinations, admissions, substitutions and modifications will be apparent to the skilled artisan in view of the disclosure herein. Accordingly, the present invention is not intended to be limited by the reaction of the preferred embodiments but is to be defined by a reference to the appended claims.

The term "comprise" and variants of the term such as "comprises" or "comprising" are used herein to denote the inclusion of a stated integer or stated integers but not to exclude any other integer or any other integers, unless in the context or usage an exclusive interpretation of the term is required.

Any reference to publications cited in this specification is not an admission that the disclosures constitute common general knowledge in Australia.

2004248616 25 Feb 2009

THE CLAIMS DEFINING THE INVENTION ARE AS FOLLOWS:

1. A method for securing data, the method comprising the steps of:

a) encrypting a data set to provide an encrypted data set;

b) generating two or more portions of data from the encrypted data set, wherein the generating comprises:

splitting the data set into a number of data units,

generating random or pseudo-random numbers,

associating the random or pseudo-random numbers with at least two shares,

associating the random or pseudo-random numbers with the data units,

determining into which of the at least two shares to store each data unit according to the association of the random or pseudo-random numbers with the at least two shares and with the data units, and

storing, using electronic storage, the data units and data indicative of at least a portion of the random or pseudo-random numbers in the at least two shares according to the determination;

wherein the two or more portions of data each contain a randomized or pseudo-randomized distribution of the encrypted data set; and

c) encrypting one or more of the portions of data from step b), whereby the data set is restorable from at least two of the two or more portions of data from step b), wherein restoring the data set comprises:

decrypting the one or more portions of data from step c),

reconstituting the encrypted data set by recombining data from the at least two of the two or more portions of data that was randomly or pseudo-randomly distributed in step b), and

decrypting the encrypted data set into the data set.

2. The method of claim 1, wherein the generating of step b) generates at least four portions of data.

3. The method of any one of the preceding claims, wherein step b) and step c) are repeated one or more times, and optionally, wherein the encrypting of step c) is performed using an encryption algorithm that is different from the encryption algorithm in step a).

4. The method of any one of the preceding claims, wherein the at least two data shares are on different locations of the same data depository.

5. The method of any one of claims 1 to 3, wherein the at least two data shares are on different data depositories.

6. The method of any one of claims 1 to 3, wherein the at least two data shares are on different data depositories in different geographic locations.

7. The method of any one of the preceding claims, wherein the encryption of step c) provides an encryption key, and wherein the encryption key is stored together with the data encrypted using said encryption key in step c).

8. The method of any one of claims 1 to 6, wherein the encryption of step c) provides an encryption key, and wherein the encryption key is stored separately from the data encrypted using said encryption key in step c).

9. The method of any one of the preceding claims, wherein the data set of step a) comprises data selected from the group consisting of encryption key data, text, video, audio, images, biometrics, and digital data.

10. A method for securing data, the method comprising the steps of:

a) generating two or more portions of data from a data set, wherein the generating comprises:

splitting the data set into a number of data units,

generating random or pseudo-random numbers,

associating the random or pseudo-random numbers with at least two shares,

associating the random or pseudo-random numbers with the data units,

determining into which of the at least two shares to store each data unit according to the association of the random or pseudo-random numbers with the at least two shares and with the data units, and

storing, using electronic storage, the data units and data indicative of at least a portion of the random or pseudo-random numbers in the at least two shares according to the determination;

wherein the two or more portions of data each contain a random or pseudo-random distribution of the data set; and

b) encrypting one or more of the portions of data of step a), whereby the data set is restorable from at least two of the two or more portions of data, wherein restoring the data set comprises:

decrypting the one or more portions of data from step b), and

reconstituting the data set by recombining data from the at least two of the two or more portions of data that was randomly or pseudo-randomly distributed in step a).

11. The method of claim 10, wherein the generating of step a) generates at least four portions of data.

12. The method of claim 10 or 11, wherein step a) and step b) are repeated one or more times, and, optionally, wherein the encryption of step b) is repeated using a different encryption algorithm.

13. The method of any one of claims 10 to 12, wherein the at least two data shares are on different locations of the same data depository.

14. The method of any one of claims 10 to 12, wherein the at least two data shares are on different data depositories.

15. The method of any one of claims 10 to 12, wherein the at least two data shares are on different data depositories in different geographic locations.

16. The method of any one of claims 10 to 15, wherein the encryption of step b) provides an encryption key, and wherein the encryption key is stored together with the data encrypted using said encryption key in step b).

17. The method of any one of claims 10 to 15, wherein the encryption of step b) provides an encryption key, and wherein the encryption key is stored separately from the data encrypted using said encryption key in step b).

18. The method of any one of claims 10 to 17, wherein the data set of step a) comprises data selected from a group consisting of encryption key data, text, video, audio, images, biometrics, and digital data.

19. The method of any one of claims 10 to 18, wherein the encryption of step b) is performed using an encryption algorithm selected from a group consisting of RS1, RC4™, and OTP.

20. A method for securing data, the method comprising the steps of:

a) generating an encryption master key and encrypting a data set using the encryption master key;

b) generating two or more portions of data from the encrypted data set and the encryption master key according to one separating pattern, wherein generating the two or more portions comprises:

splitting the data set into a number of data units,

generating random or pseudo-random numbers,

associating the random or pseudo-random numbers with at least two shares,

associating the random or pseudo-random numbers with the data units,

determining into which of the at least two shares to store each data unit according to the association of the random or pseudo-random numbers with the at least two shares and with the data units,

storing, using electronic storage, the data units and data indicative of at least a portion of the random or pseudo-random numbers in the at least two shares according to the determination, and

appending an encryption master key portion to the at least two shares, wherein the two or more portions comprise a random or pseudo-random distribution of data from the encrypted data set; and

c) generating one or more encryption keys for the portions of data from step b) and encrypting said portions of data using said one or more encryption keys, whereby the data set is restorable from at least two portions of the two or more portions of data, wherein restoring the data set comprises:

decrypting the encrypted portions of data,

reconstituting the encrypted data set by recombining data from the at least two portions of the two or more portions of data that was randomly or pseudo-randomly distributed in step b), and

decrypting the encrypted data set into the data set.

21. The method of claim 20, wherein the storing of encrypted data portions is on two or more different locations of one data depository.

22. The method of claim 20 or 21, wherein the storing of encrypted data portions is on two or more data depositories.

23. The method of any one of claims 20 to 21, wherein the storing of encrypted data portions is on two or more different locations of one data depository.

24. The method of any one of claims 20 to 23, wherein the storing of the encryption keys is on two or more different data depositories.

25. The method of any one of claims 20 to 24, wherein the encryption keys generated in step c) is stored with the encrypted data of step c) on different locations on one or more data depository.

26. The method of any one of claims 20 to 25, wherein the encryption key generated in step c) is stored on a different data depository from the encrypted data of step c) that was encrypted using said encryption key.

27. The method of any one of claims 20 to 26, wherein the encrypted data of step b) is separated into four or more portions.

28. The method of any one of claims 20 to 27, wherein the encryption master key of step b) is separated into four or more portions.

29. The method of any one of claims 20 to 28, wherein step b) and step c) are repeated one or more times, and optionally, wherein the encrypting of step c) is performed using an encryption algorithm that is different from the encryption algorithm used in step a).

30. A method for securing data, the method comprising the steps of:

a) generating an encryption master key and encrypting a data set using the encryption master key;

b) generating two or more portions of data from the encrypted data set and the encryption master key according to one separating pattern, wherein generating the two or more portions comprises:

splitting the data set into a number of data units,

generating random or pseudo-random numbers,

associating the random or pseudo-random numbers with at least two shares,

associating the random or pseudo-random numbers with the data units,

determining into which of the at least two shares to store each data unit according to the association of the random or pseudo-random numbers with the at least two shares and with the data units,

storing, using electronic storage, the data units and data indicative of at least a portion of the random or pseudo-random numbers in the at least two shares according to the determination, and

storing encryption master key portions in the at least two shares, wherein the two or more portions comprise a random or pseudo-random distribution of data from the encrypted data set; and

c) generating one or more encryption keys for the encrypted data set portions of step b) and encrypting said portions of data using said encryption key, whereby the data set is restorable from at least two portions of the two or more portions of data, wherein restoring the data set comprises:

decrypting the encrypted portions from step c),

reconstituting the encrypted data set by recombining data from the at least two portions of the two or more portions of data that was randomly or pseudo-randomly distributed in step b), and

decrypting the encrypted data set into the data set.

31. The method of claim 30, wherein the storing of encrypted data portions is on two or more different locations of one data depository.

32. The method of claim 30 or 31, wherein the storing of encrypted data portions is on two or more data depositories.

33. The method of claim 30 or 31, wherein the storing of the encryption keys is on two or more different locations of one data depository.

34. The method of claim 30 or 31, wherein the storing of the encryption keys is on two or more different data depositories.

35. The method of any one of claims 30 to 34, wherein the encryption keys generated in step c) and used to encrypt a data set in step c) is stored with the encrypted data set that was encrypted using the encryption key on one or more data depositories.

36. The method of any one of claims 30 to 34, wherein the encryption key generated in step c) and used to encrypt a data set in step c) is stored in a different location on one or more data depositories from the encrypted data set that was encrypted using the encryption key.

37. The method of any one of claims 30 to 36, wherein the encrypted data of step b) is separated into four or more portions.

38. The method of any one of claims 30 to 37, wherein the encryption master key of step b) is separated into four or more portions.

39. The method of any one of claims 30 to 38, wherein step b) and step c) are repeated one or more times, and optionally, wherein the encrypting of step c) is performed using an encryption algorithm that is different from the encryption algorithm used in step a).

40. A method for securing data, the method comprising the steps of:

a) encrypting a data set to provide an encrypted data set;

b) generating two or more portions of data from the encrypted data set according to the contents of a unique key value, wherein the generating comprises:

splitting the data set into a number of data units,

generating the unique key values,

associating the unique key values with at least two shares,

associating the unique key values with the data units,

determining into which of the at least two shares to store each data unit according to the association of the unique key values with the at least two shares and with the data units, and

storing, using electronic storage, the data units and data indicative of at least a portion of the unique key values in the at least two shares according to the determination;

wherein the encrypted data set is randomly or pseudo-randomly distributed among the two or more portions of data; and

c) encrypting one or more of the portions of data from step b), whereby the data set is restorable from at least a subset of the portions of data, wherein restoring the data set comprises:

decrypting the one or more portions of data from step c),

reconstituting the encrypted data set by recombining data from the at least two of the two or more portions of data that was randomly or pseudo-randomly distributed in step b), and

decrypting the encrypted data set into the data set.

41. The method of claim 40, wherein the generating of step b) generates four or more portions of data according to the contents of a unique key value.

42. The method of claim 40 or 41, wherein step b) and step c) are repeated one or more times, and optionally, wherein the encrypting of step c) is performed using an encryption algorithm that is different from the encryption algorithm in step a).

43. The method of any one of claims 40 to 42, wherein the at least two data shares are on different locations of the same data depository.

44. The method of any one of claims 40 to 42, wherein the at least two data shares are on different data depositories.

45. The method of any one of claims 40 to 42, wherein the at least two data shares are on different data depositories in different geographic locations.

46. The method of any one of claims 40 to 45, wherein the encryption of step c) provides an encryption key, and wherein the encryption key is stored together with the data encrypted using said encryption key in step c).

47. The method of any one of claims 40 to 45, wherein the encryption of step c) provides an encryption key, and wherein the encryption key is stored separately from the data encrypted using said encryption key in step c).

48. The method of any one of claims 40 to 47, wherein the data set of step a) comprises data selected from the group consisting of encryption key data, text, video, audio, images, biometrics, and digital data.

49. The method of any one of claims 40 to 48 wherein said portions of data comprise one or more bits of data.

50. A method for securing data, the method comprising the steps of:

a) splitting a data set into N number of data units;

b) selecting X number of shares for data unit storage;

c) generating N number of random or pseudo-random numbers that correspond to the X number of shares;

d) assigning the random or pseudo-random numbers to the data units; and

e) storing, using electronic storage, the data units and data indicative of at least a portion of the random or pseudo-random numbers in the shares that correspond to the random or pseudo-random numbers, whereby the data set is restorable from at least a subset of the X number of shares, wherein restoring the data set comprises reconstituting the data set by recombining the data units from the at least a subset of the X number of shares according to the substantially random numbers.

51. The method of claim 50, wherein said data units comprise at least one bit.

52. A method for securing a data set, the method comprising the steps of:
generating at least two portions of data from the data set, wherein the generating comprises:
splitting the data set into a number of data units,
generating random or pseudo-random numbers,
associating the random or pseudo-random numbers with at least two shares,
associating the random or pseudo-random numbers with the data units,
determining into which of the at least two shares to store each data unit according to the association of the random or pseudo-random numbers with the at least two shares and with the data units, and
storing, using electronic storage, the data units and data indicative of at least a portion of the random or pseudo-random numbers in the at least two shares according to the determination;
wherein each of the at least two portions of data respectively contains a random or pseudo-random distribution of a respective subset of the data set, whereby the data set is restorable from at least two portions of the at least two portions of data by recombining data from the at least two portions of the at least two portions of data that was substantially randomly distributed.

53. The method of claim 52 wherein generating the at least two portions of data comprises generating the at least two portions of data from the data set, wherein the at least two portions of data each contain a substantially randomized distribution of bits of data from the data set.

54. The method of claim 52 wherein generating the at least two portions of data comprises generating the at least two portions of data from the data set, wherein the at least two portions of data each contain a substantially randomized distribution of bytes of data from the data set.

55. The method of claim 52 wherein generating the at least two portions of data comprises generating the at least two portions of data from the data set, wherein the at least two portions of data each contain a substantially randomized distribution of blocks of data from the data set.

56. The method of claim 52 wherein generating the at least two portions of data comprises generating the at least two portions of data from the data set using redundancy.

57. A method for securing a data set, the method comprising the steps of:

(a) randomly or pseudo-randomly selecting a first group of data units from the data set;

(b) randomly or pseudo-randomly selecting a second group of data units from the data set, wherein the randomly or pseudo-randomly selecting the first group of data units and the second group of data units comprises:

generating random or pseudo-random numbers,

associating the random or pseudo-random numbers with at least two shares,

associating the random or pseudo-random numbers with data units in the first group of data units and the second group of data units, and

determining into which of the at least two shares to store each data unit according to the association of the random or pseudo-random numbers with the at least two shares and with the data units;

wherein each of the first group of the data units and the second group of the data units contains less than all of the data units in the data set; and

(c) storing the first group of data units and the second group of data units separately in the at least two shares, wherein the data set is restorable from at least a portion of the first group of data units and at least a portion of the second group of data units.

Figure 1

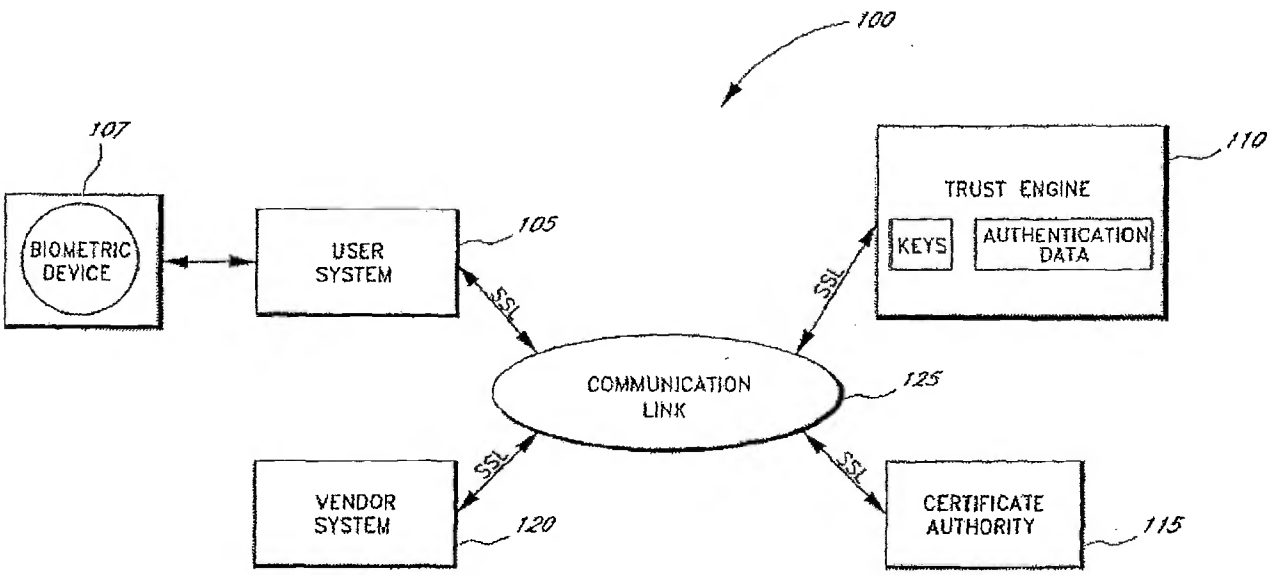

100
107
BIOMETRIC DEVICE
USER SYSTEM
105
SSL
TRUST ENGINE
KEYS
AUTHENTICATION DATA
110
SSL
COMMUNICATION LINK
125
SSL
SSL
VENDOR SYSTEM
120
CERTIFICATE AUTHORITY
115

Figure 2

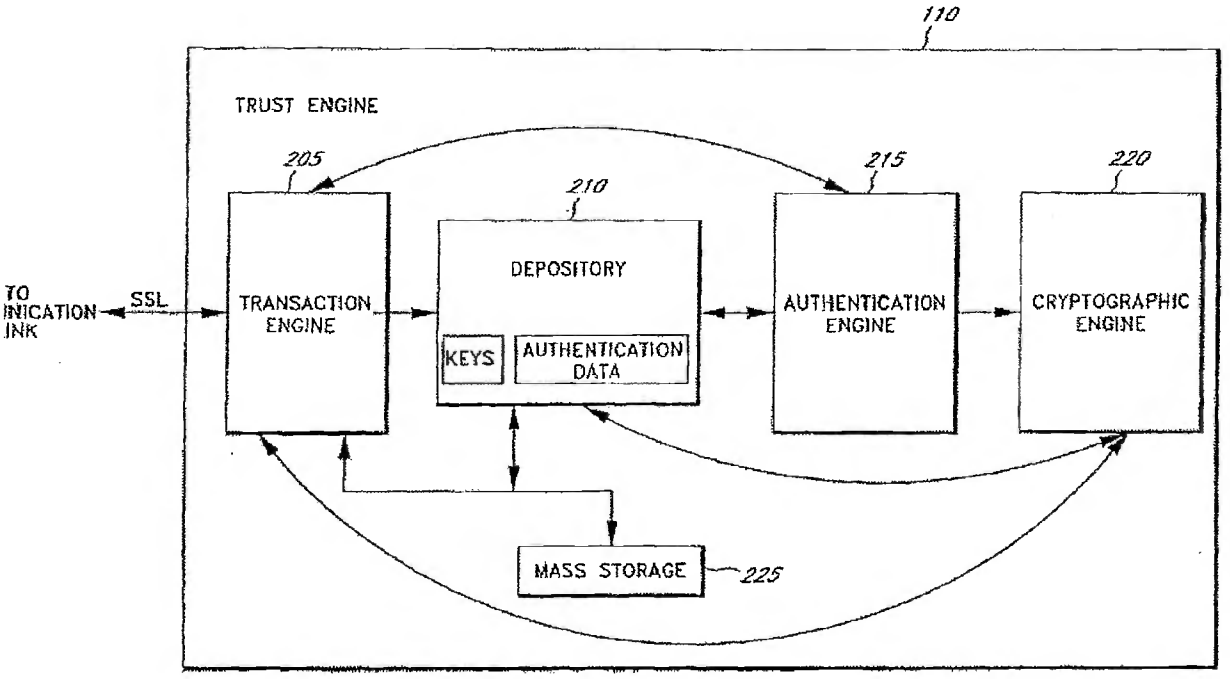

110
TRUST ENGINE
205
210
215
220
TO
INICATION
INK
SSL
TRANSACTION ENGINE
DEPOSITORY
KEYS
AUTHENTICATION DATA
AUTHENTICATION ENGINE
CRYPTOGRAPHIC ENGINE
MASS STORAGE
225

Figure 3

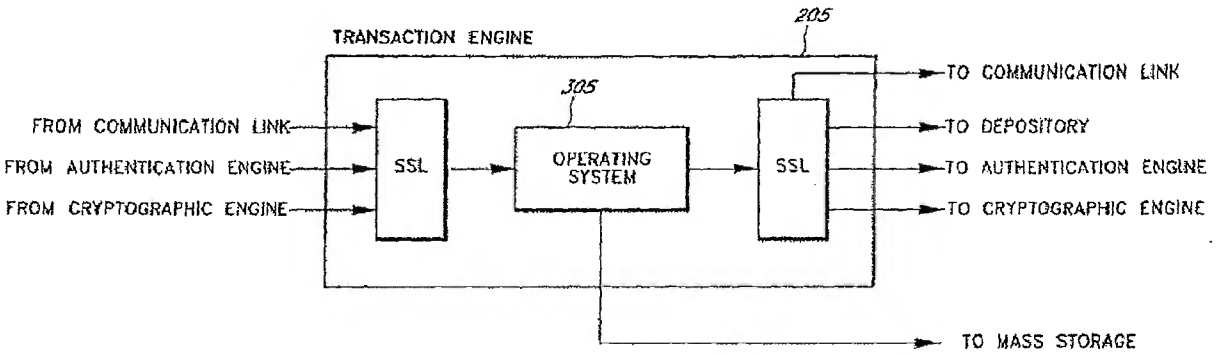

205
TRANSACTION ENGINE
305
FROM COMMUNICATION LINK
FROM AUTHENTICATION ENGINE
FROM CRYPTOGRAPHIC ENGINE
SSL
OPERATING SYSTEM
SSL
TO COMMUNICATION LINK
TO DEPOSITORY
TO AUTHENTICATION ENGINE
TO CRYPTOGRAPHIC ENGINE
TO MASS STORAGE

Figure 4

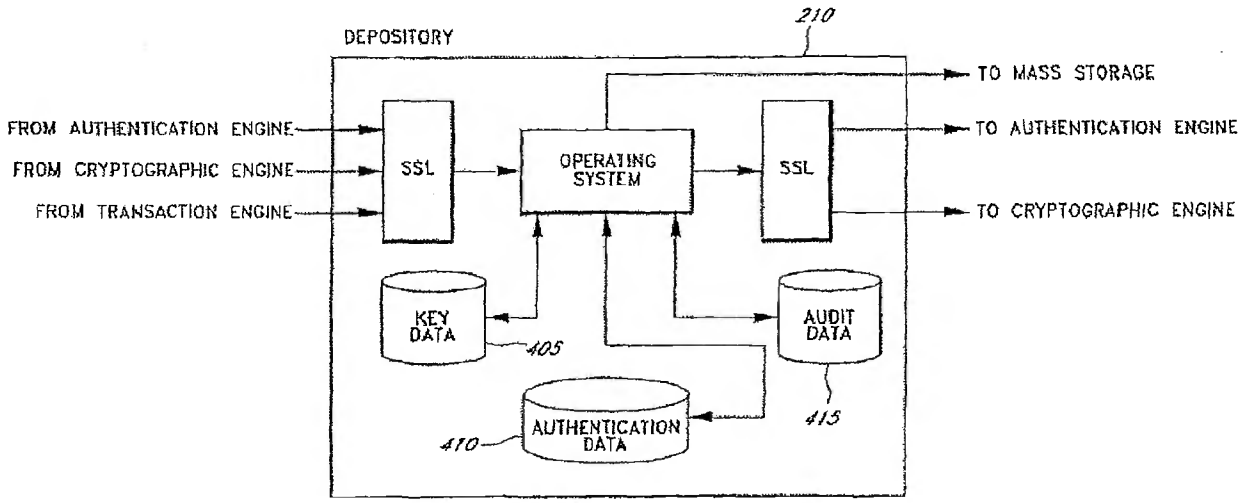

DEPOSITORY
210
FROM AUTHENTICATION ENGINE
FROM CRYPTOGRAPHIC ENGINE
FROM TRANSACTION ENGINE
SSL
OPERATING SYSTEM
SSL
TO MASS STORAGE
TO AUTHENTICATION ENGINE
TO CRYPTOGRAPHIC ENGINE
KEY DATA
405
AUDIT DATA
415
AUTHENTICATION DATA
410

Figure 5

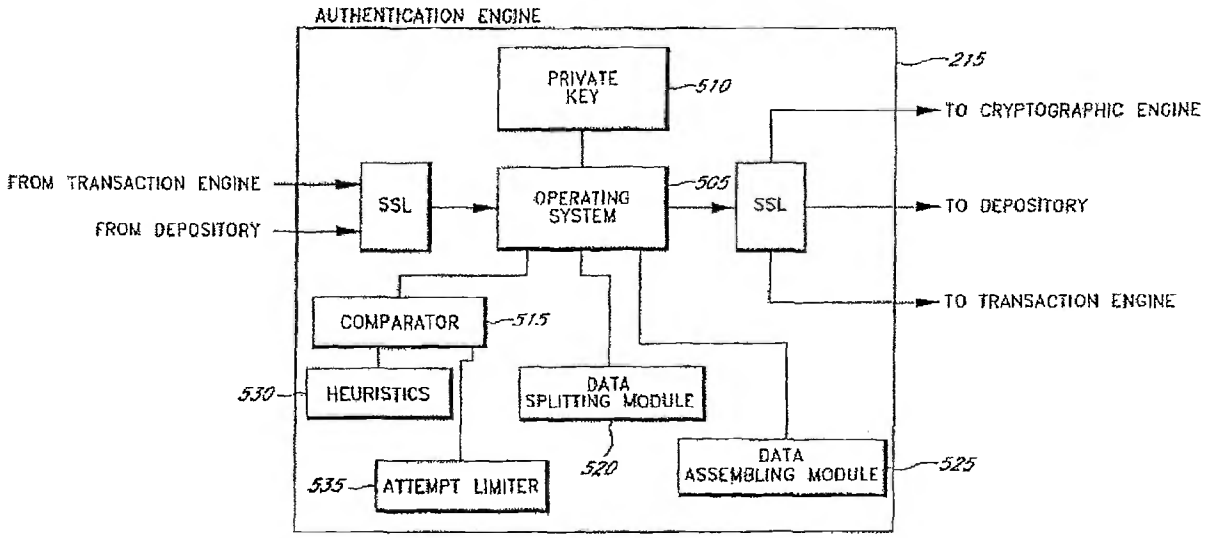

AUTHENTICATION ENGINE
215
PRIVATE KEY
510
TO CRYPTOGRAPHIC ENGINE
FROM TRANSACTION ENGINE
FROM DEPOSITORY
SSL
OPERATING SYSTEM
505
SSL
TO DEPOSITORY
TO TRANSACTION ENGINE
COMPARATOR
515
530
HEURISTICS
DATA SPLITTING MODULE
520
DATA ASSEMBLING MODULE
525
535
ATTEMPT LIMITER

Figure 6

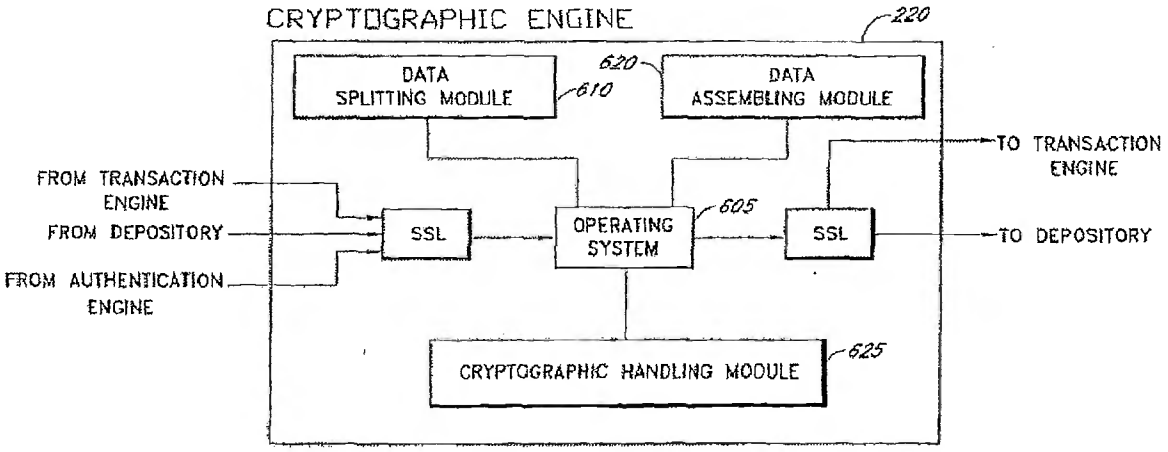

CRYPTOGRAPHIC ENGINE
220
DATA SPLITTING MODULE
610
620
DATA ASSEMBLING MODULE
FROM TRANSACTION ENGINE
FROM DEPOSITORY
FROM AUTHENTICATION ENGINE
SSL
OPERATING SYSTEM
605
SSL
TO TRANSACTION ENGINE
TO DEPOSITORY
CRYPTOGRAPHIC HANDLING MODULE
625

Figure 7

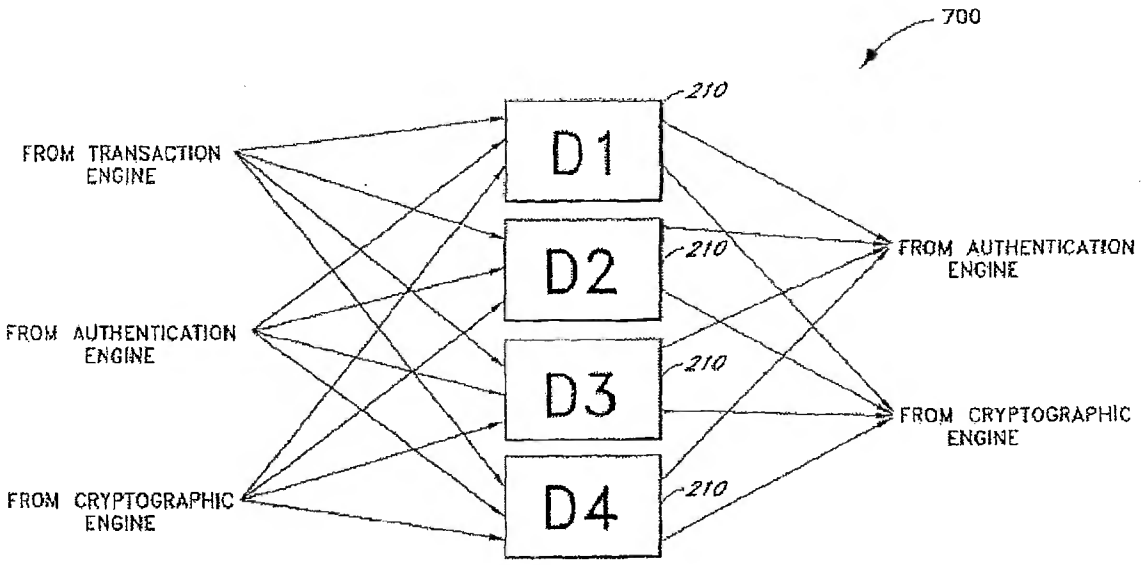

700
210
D1
D2
D3
D4
210
210
210
FROM TRANSACTION ENGINE
FROM AUTHENTICATION ENGINE
FROM CRYPTOGRAPHIC ENGINE
FROM AUTHENTICATION ENGINE
FROM CRYPTOGRAPHIC ENGINE

Figure 8

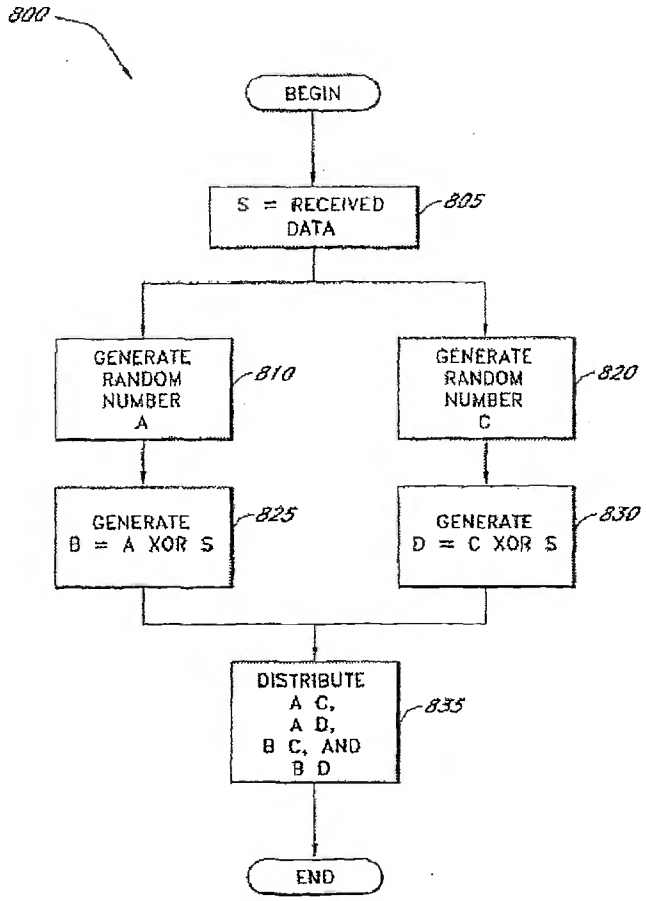

800
BEGIN
S = RECEIVED DATA
805
GENERATE RANDOM NUMBER A
810
GENERATE RANDOM NUMBER C
820
GENERATE B = A XOR S
825
GENERATE D = C XOR S
830
DISTRIBUTE A C, A D, B C, AND B D
835
END

Figure 9, Panel A



ENROLLMENT DATA FLOW			
SEND	RECEIVE	SSL	ACTION
905 USER	TRANSACTION ENGINE(TE)	1/2	TRANSMIT ENROLLMENT AUTHENTICATION DATA (B) AND THE USER ID (UID) ENCRYPTED WITH THE PUBLIC KEY OF THE AUTHENTICATION ENGINE (AE) AS (PUB_AE(UID,B))
915 TE	AE	FULL	FORWARD TRANSMISSION
920			AE DECRYPTS AND SPLITS FORWARDED DATA
925 AE	THE Xth DEPOSITORY (DX)	FULL	STORE RESPECTIVE PORTION OF DATA
WHEN DIGITAL CERTIFICATE REQUESTED			
930 AE	CRYPTOGRAPHIC ENGINE (CE)	FULL	REQUEST KEY GENERATION
935			CE GENERATES AND SPLITS KEY
945 CE	TE	FULL	TRANSMIT REQUEST FOR DIGITAL CERTIFICATE
950 TE	CERTIFICATION AUTHORITY (CA)	1/2	TRANSMIT REQUEST
955 CA	TE	1/2	TRANSMIT DIGITAL CERTIFICATE
960 TE	USER	1/2	TRANSMIT DIGITAL CERTIFICATE
TE	MS	FULL	STORE DIGITAL CERTIFICATE
965 CE	DX	FULL	STORE RESPECTIVE PORTION OF KEY

Figure 9, Panel B

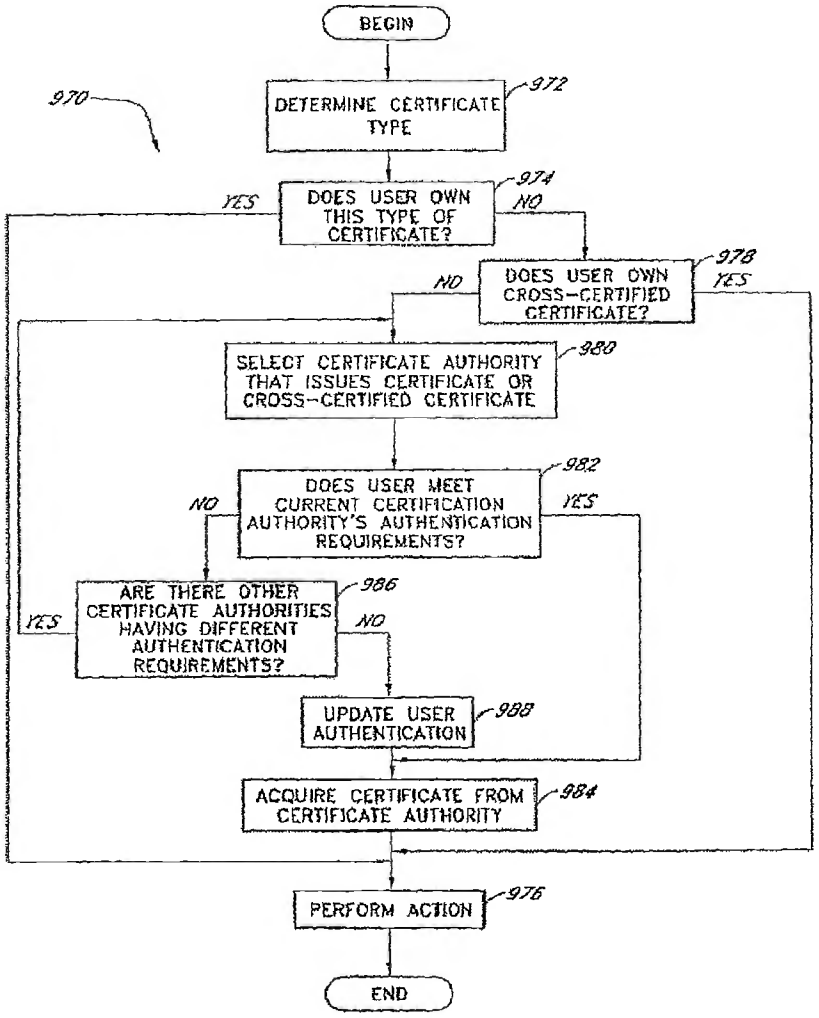
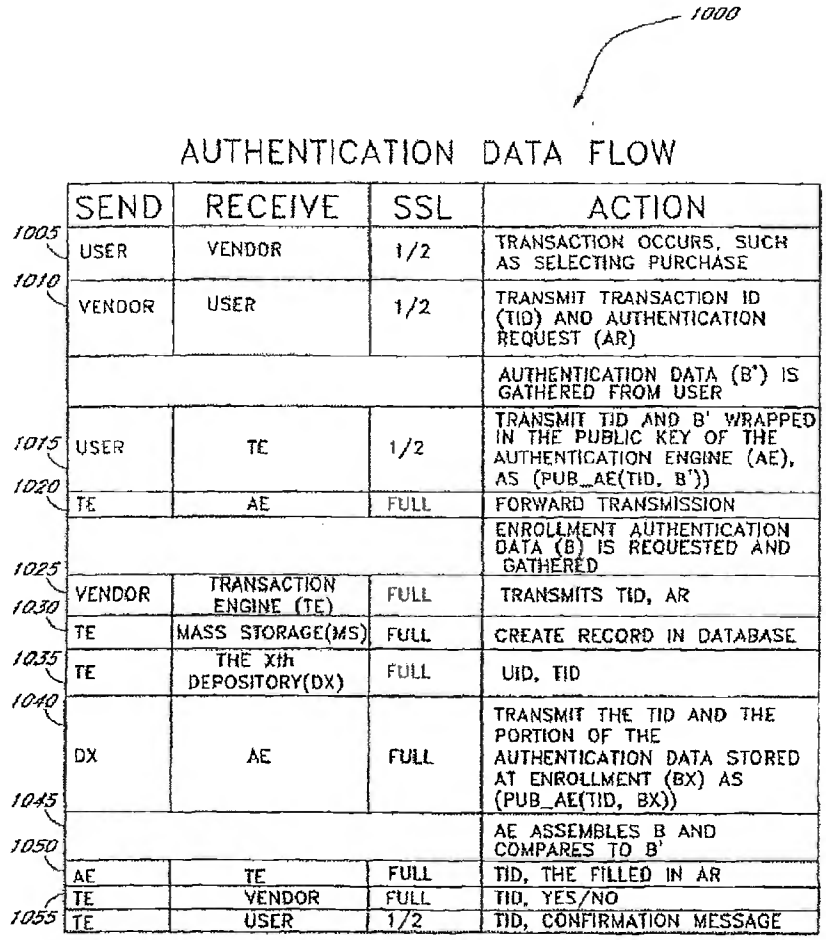


Figure 10



SEND	RECEIVE	SSL	ACTION
USER	VENDOR	1/2	TRANSACTION OCCURS, SUCH AS SELECTING PURCHASE
VENDOR	USER	1/2	TRANSMIT TRANSACTION ID (TID) AND AUTHENTICATION REQUEST (AR)
			AUTHENTICATION DATA (B') IS GATHERED FROM USER
USER	TE	1/2	TRANSMIT TID AND B' WRAPPED IN THE PUBLIC KEY OF THE AUTHENTICATION ENGINE (AE), AS (PUB_AE(TID, B'))
TE	AE	FULL	FORWARD TRANSMISSION
			ENROLLMENT AUTHENTICATION DATA (B) IS REQUESTED AND GATHERED
VENDOR	TRANSACTION ENGINE (TE)	FULL	TRANSMITS TID, AR
TE	MASS STORAGE(MS)	FULL	CREATE RECORD IN DATABASE
TE	THE Xth DEPOSITORY(DX)	FULL	UID, TID
DX	AE	FULL	TRANSMIT THE TID AND THE PORTION OF THE AUTHENTICATION DATA STORED AT ENROLLMENT (BX) AS (PUB_AE(TID, BX))
			AE ASSEMBLES B AND COMPARES TO B'
AE	TE	FULL	TID, THE FILLED IN AR
TE	VENDOR	FULL	TID, YES/NO
TE	USER	1/2	TID, CONFIRMATION MESSAGE

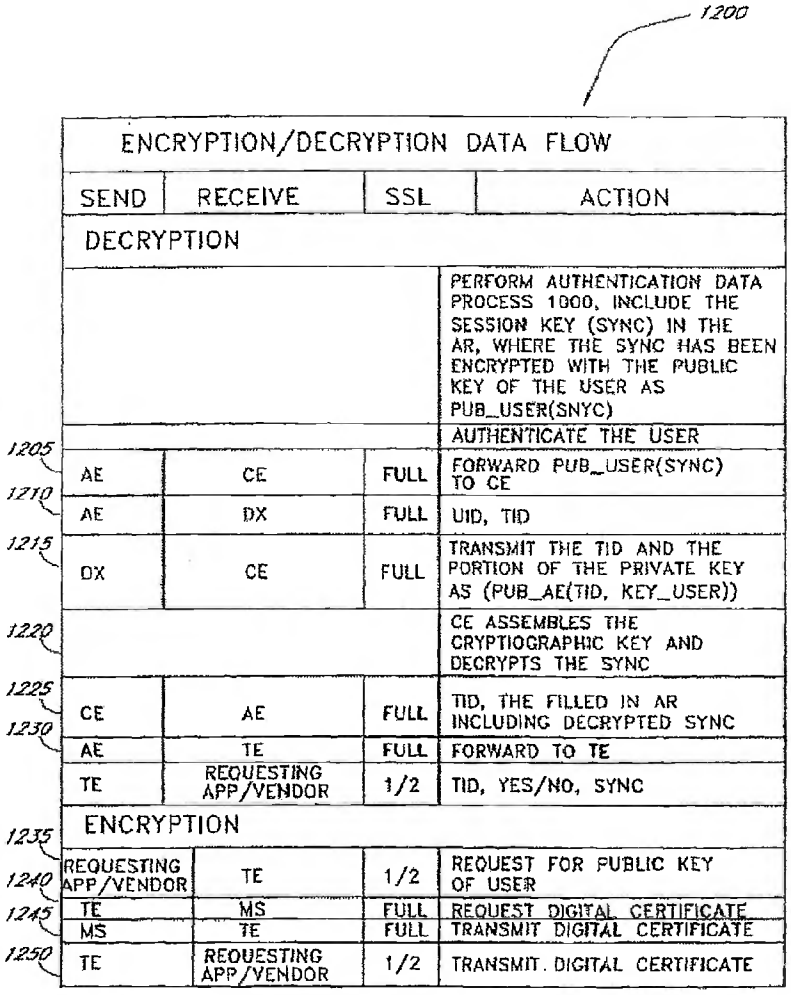
Figure 11

1100

SIGNING DATA FLOW			
SEND	RECEIVE	SSL	ACTION
USER	VENDOR	1/2	TRANSACTION OCCURS, SUCH AS AGREEING ON A DEAL
VENDOR	USER	1/2	TRANSMIT TRANSACTION IDENTIFICATION NUMBER (TID), AUTHENTICATION REQUEST (AR), AND AGREEMENT OR MESSAGE (M)
			CURRENT AUTHENTICATION DATA (B') AND A HASH OF THE MESSAGE RECEIVED BY THE USER (h(M')) IS GATHERED FROM USER
USER	TE	1/2	TRANSMIT TID, B', AR, AND h(M') WRAPPED IN THE PUBLIC KEY OF THE AUTHENTICATION ENGINE (AE) AS (PUB_AE(TID, B', h(M'))
TE	AE	FULL	FORWARD TRANSMISSION
			GATHER ENROLLMENT AUTHENTICATION DATA
VENDOR	TRANSACTION ENGINE (TE)	FULL	TRANSMITS UID, TID, AR, AND A HASH OF THE MESSAGE (h(M)).
TE	MASS STORAGE (MS)	FULL	CREATE RECORD IN DATABASE
TE	THE Xth DEPOSITORY(DX)	FULL	UID, TID
DX	AE	FULL	TRANSMIT THE TID AND THE PORTION OF THE AUTHENTICATION DATA STORED AT ENROLLMENT (BX).AS (PUB_AE(TID, BX))
			THE ORIGINAL VENDOR MESSAGE IS TRANSMITTED TO THE AE
TE	AE	FULL	TRANSMIT h(M)
			AE ASSEMBLES B, COMPARES TO B' AND COMPARES h(M) TO h(M')
AE	CRYPTOGRAPHIC ENGINE (CE)	FULL	REQUEST FOR DIGITAL SIGNATURE AND A MESSAGE TO BE SIGNED, FOR EXAMPLE, THE HASHED MESSAGE
AE	DX	FULL	TID, SIGNING UID
DX	CE	FULL	TRANSMIT THE PORTION OF THE CRYPTOGRAPHIC KEY CORRESPONDING TO THE SIGNING PARTY
			CE ASSEMBLES KEY AND SIGNS
CE	AE	FULL	TRANSMIT THE DIGITAL SIGNATURE (S) OF SIGNING PARTY
AE	TE	FULL	TID, THE FILLED IN AR, h(M), AND S
TE	VENDOR	FULL	TID, A RECEIPT=(TID, YES/NO, AND S), AND THE DIGITAL SIGNATURE OF THE TRUST ENGINE, FOR EXAMPLE, A HASH OF THE RECEIPT ENCRYPTED WITH THE TRUST ENGINE'S PRIVATE KEY (Priv_TE(h(RECEIPT)))
TE	USER	1/2	TID, CONFIRMATION MESSAGE

1103 1105 1110 1115 1120 1125 1130 1135 1140

Figure 12



ENCRYPTION/DECRYPTION DATA FLOW			
SEND	RECEIVE	SSL	ACTION
DECRYPTION			
			PERFORM AUTHENTICATION DATA PROCESS 1000, INCLUDE THE SESSION KEY (SYNC) IN THE AR, WHERE THE SYNC HAS BEEN ENCRYPTED WITH THE PUBLIC KEY OF THE USER AS PUB_USER(SNYC)
			AUTHENTICATE THE USER
AE	CE	FULL	FORWARD PUB_USER(SYNC) TO CE
AE	DX	FULL	UID, TID
DX	CE	FULL	TRANSMIT THE TID AND THE PORTION OF THE PRIVATE KEY AS (PUB_AE(TID, KEY_USER))
			CE ASSEMBLES THE CRYPTIOGRAPHIC KEY AND DECRYPTS THE SYNC
CE	AE	FULL	TID, THE FILLED IN AR INCLUDING DECRYPTED SYNC
AE	TE	FULL	FORWARD TO TE
TE	REQUESTING APP/VENDOR	1/2	TID, YES/NO, SYNC
ENCRYPTION			
REQUESTING APP/VENDOR	TE	1/2	REQUEST FOR PUBLIC KEY OF USER
TE	MS	FULL	REQUEST DIGITAL CERTIFICATE
MS	TE	FULL	TRANSMIT DIGITAL CERTIFICATE
TE	REQUESTING APP/VENDOR	1/2	TRANSMIT. DIGITAL CERTIFICATE

Figure 13

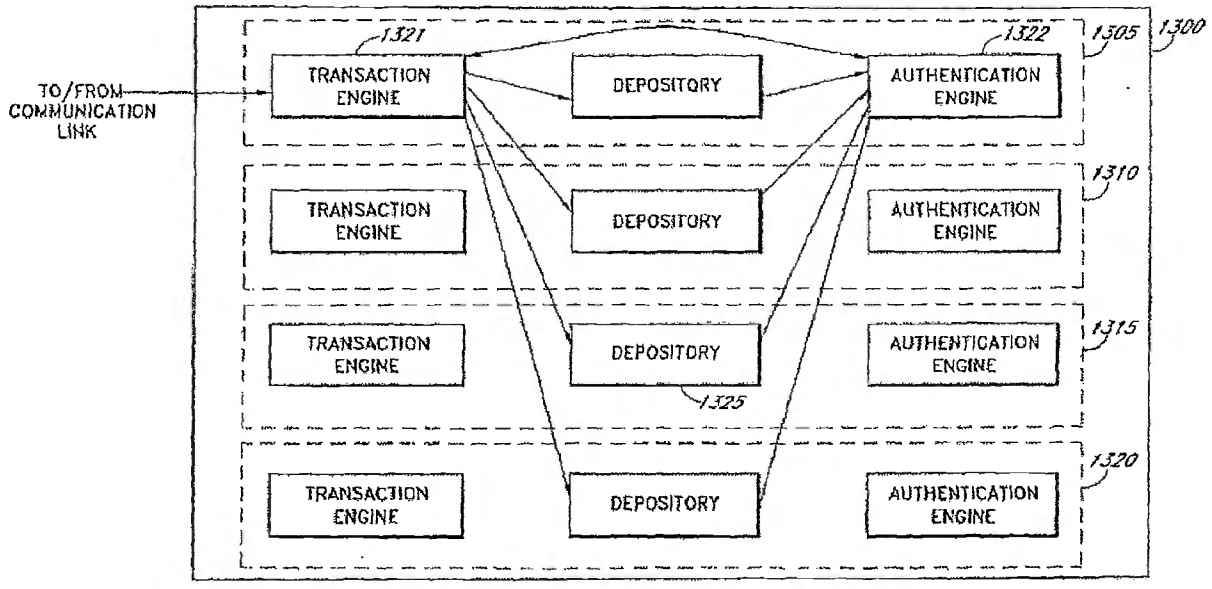

TO/FROM COMMUNICATION LINK
1321
TRANSACTION ENGINE
DEPOSITORY
1322
AUTHENTICATION ENGINE
1305
1300
1310
TRANSACTION ENGINE
DEPOSITORY
AUTHENTICATION ENGINE
1315
TRANSACTION ENGINE
DEPOSITORY
1325
AUTHENTICATION ENGINE
1320
TRANSACTION ENGINE
DEPOSITORY
AUTHENTICATION ENGINE

Figure 14

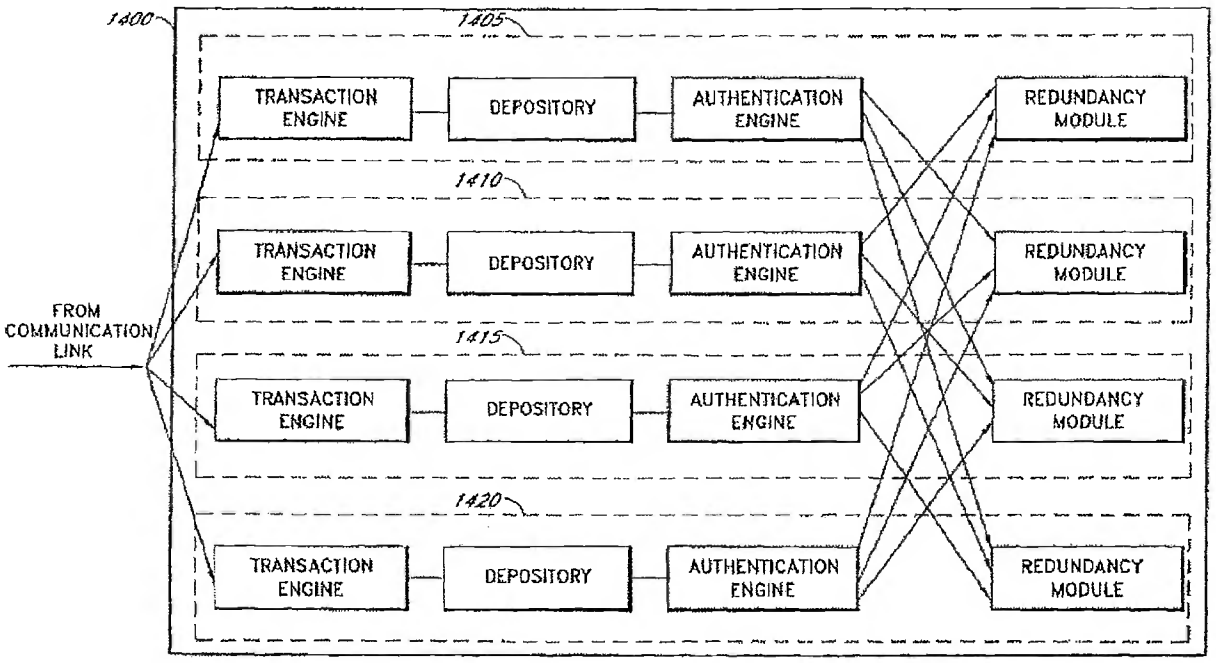

1400
1405
TRANSACTION ENGINE
DEPOSITORY
AUTHENTICATION ENGINE
REDUNDANCY MODULE
1410
TRANSACTION ENGINE
DEPOSITORY
AUTHENTICATION ENGINE
REDUNDANCY MODULE
FROM COMMUNICATION LINK
1415
TRANSACTION ENGINE
DEPOSITORY
AUTHENTICATION ENGINE
REDUNDANCY MODULE
1420
TRANSACTION ENGINE
DEPOSITORY
AUTHENTICATION ENGINE
REDUNDANCY MODULE

Figure 15

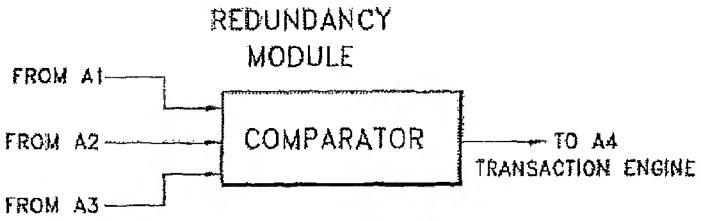

REDUNDANCY
MODULE
FROM A1
FROM A2
FROM A3
COMPARATOR
TO A4
TRANSACTION ENGINE

Figure 16

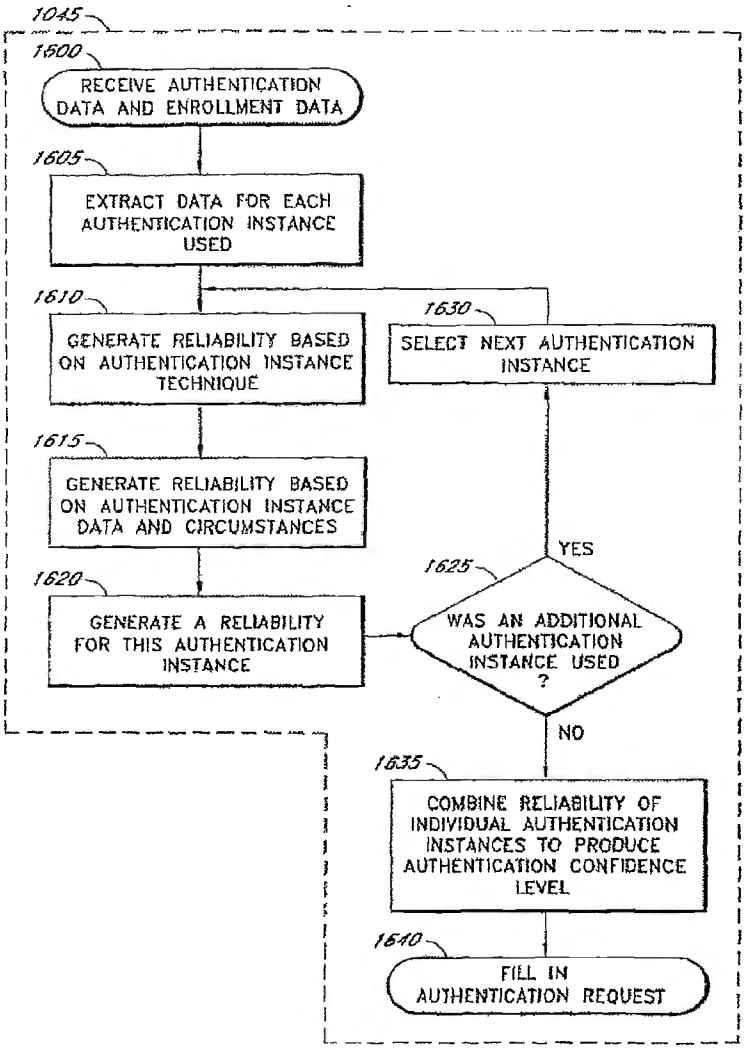

1045
1600
RECEIVE AUTHENTICATION DATA AND ENROLLMENT DATA
1605
EXTRACT DATA FOR EACH AUTHENTICATION INSTANCE USED
1610
GENERATE RELIABILITY BASED ON AUTHENTICATION INSTANCE TECHNIQUE
1615
GENERATE RELIABILITY BASED ON AUTHENTICATION INSTANCE DATA AND CIRCUMSTANCES
1620
GENERATE A RELIABILITY FOR THIS AUTHENTICATION INSTANCE
1625
WAS AN ADDITIONAL AUTHENTICATION INSTANCE USED ?
YES
NO
1630
SELECT NEXT AUTHENTICATION INSTANCE
1635
COMBINE RELIABILITY OF INDIVIDUAL AUTHENTICATION INSTANCES TO PRODUCE AUTHENTICATION CONFIDENCE LEVEL
1640
FILL IN AUTHENTICATION REQUEST

Figure 17

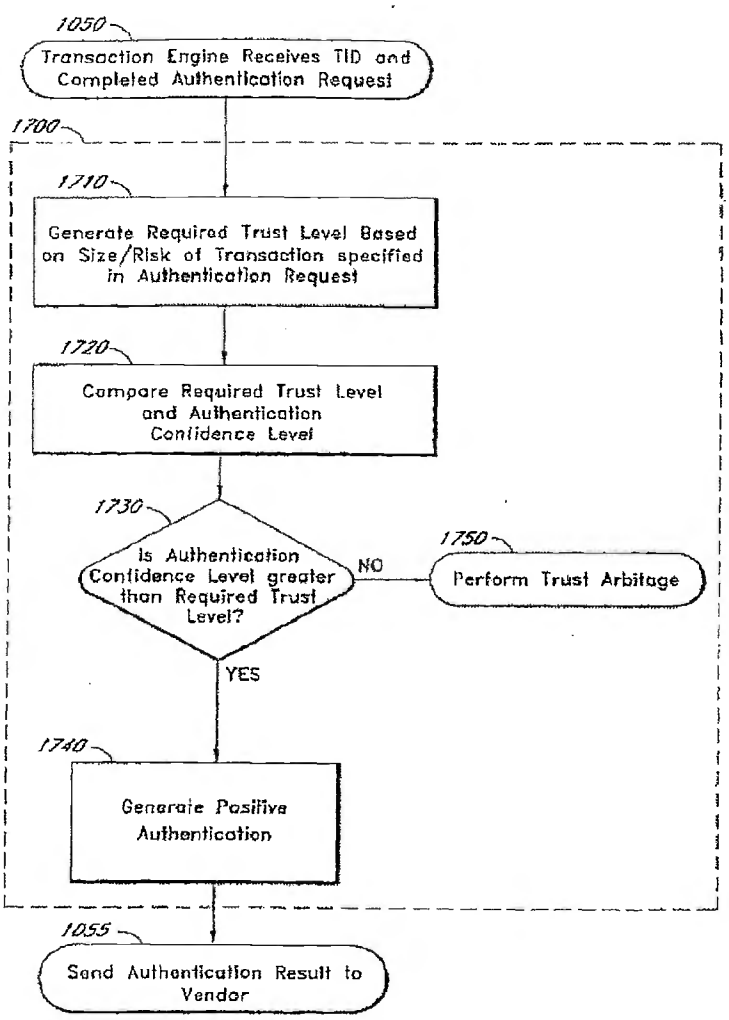

1050
Transaction Engine Receives TID and Completed Authentication Request
1700
1710
Generate Required Trust Level Based on Size/Risk of Transaction specified in Authentication Request
1720
Compare Required Trust Level and Authentication Confidence Level
1730
Is Authentication Confidence Level greater than Required Trust Level?
NO
1750
Perform Trust Arbitage
YES
1740
Generate Positive Authentication
1055
Send Authentication Result to Vendor

Figure 18

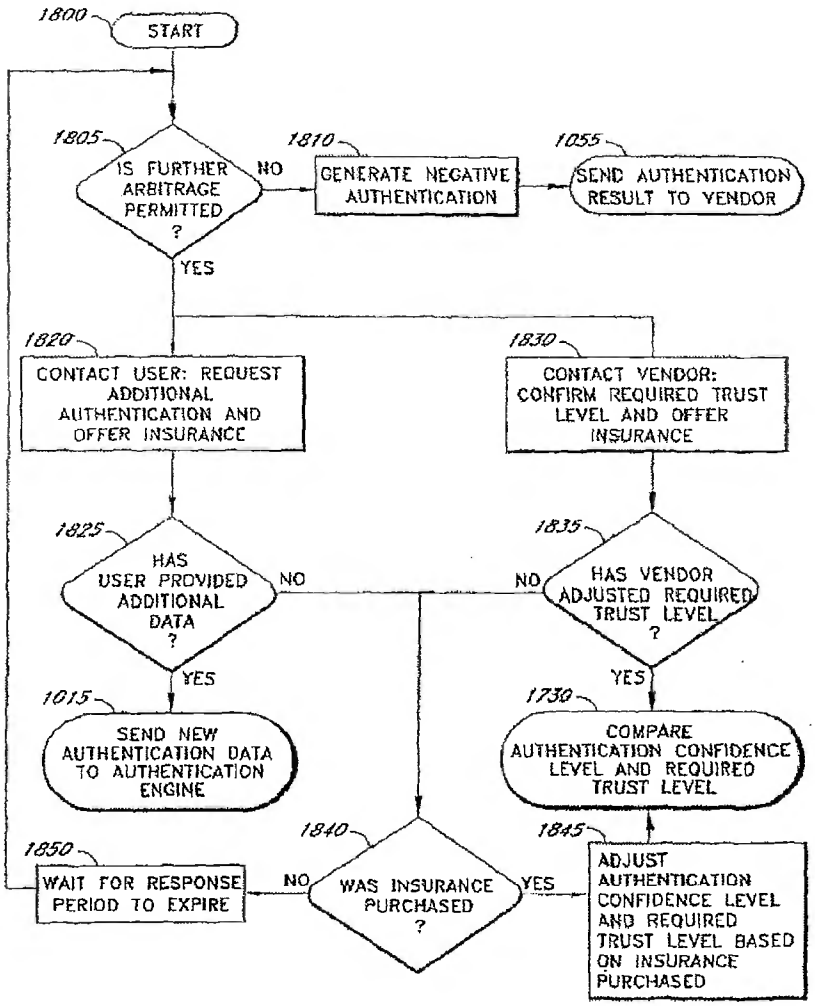

1800
START
1805
IS FURTHER ARBITRAGE PERMITTED ?
NO
1810
GENERATE NEGATIVE AUTHENTICATION
1055
SEND AUTHENTICATION RESULT TO VENDOR
YES
1820
CONTACT USER: REQUEST ADDITIONAL AUTHENTICATION AND OFFER INSURANCE
1830
CONTACT VENDOR: CONFIRM REQUIRED TRUST LEVEL AND OFFER INSURANCE
1825
HAS USER PROVIDED ADDITIONAL DATA ?
NO
1835
HAS VENDOR ADJUSTED REQUIRED TRUST LEVEL ?
NO
YES
YES
1015
SEND NEW AUTHENTICATION DATA TO AUTHENTICATION ENGINE
1730
COMPARE AUTHENTICATION CONFIDENCE LEVEL AND REQUIRED TRUST LEVEL
1840
WAS INSURANCE PURCHASED ?
1845
ADJUST AUTHENTICATION CONFIDENCE LEVEL AND REQUIRED TRUST LEVEL BASED ON INSURANCE PURCHASED
1850
WAIT FOR RESPONSE PERIOD TO EXPIRE
NO
YES

Figure 19

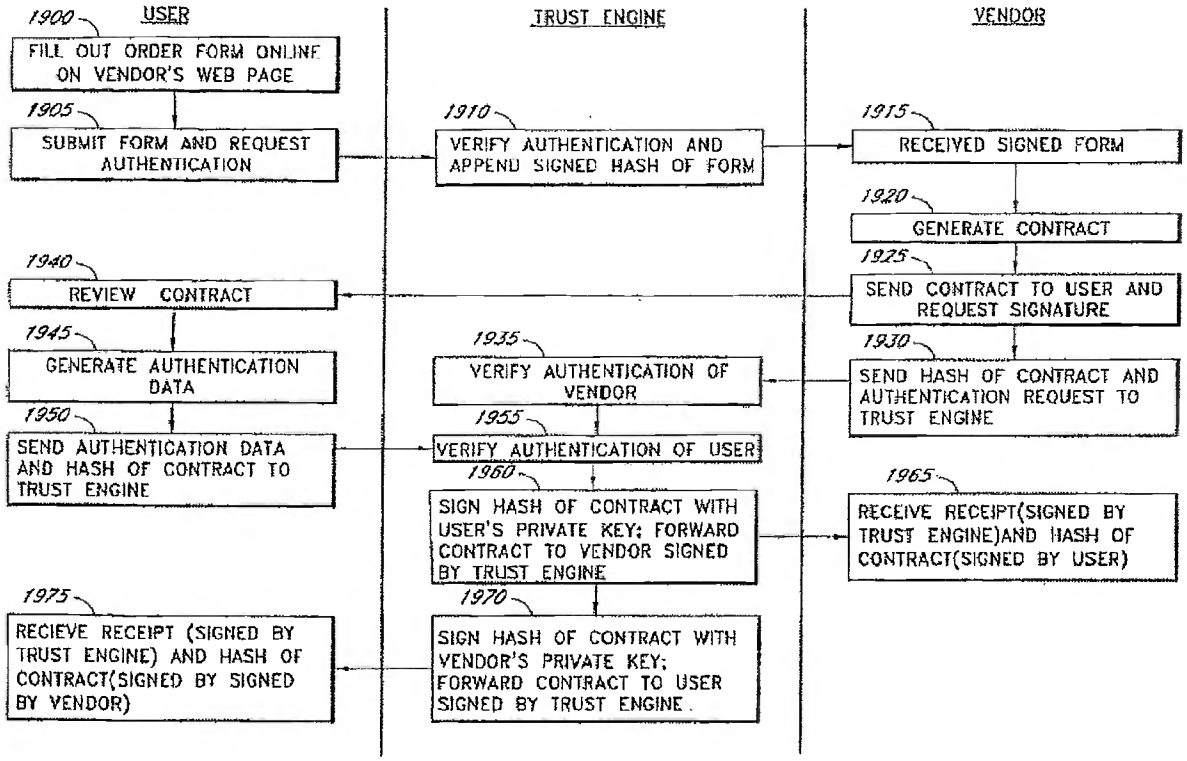

USER
TRUST ENGINE
VENDOR
1900
FILL OUT ORDER FORM ONLINE ON VENDOR'S WEB PAGE
1905
SUBMIT FORM AND REQUEST AUTHENTICATION
1910
VERIFY AUTHENTICATION AND APPEND SIGNED HASH OF FORM
1915
RECEIVED SIGNED FORM
1920
GENERATE CONTRACT
1925
SEND CONTRACT TO USER AND REQUEST SIGNATURE
1930
SEND HASH OF CONTRACT AND AUTHENTICATION REQUEST TO TRUST ENGINE
1935
VERIFY AUTHENTICATION OF VENDOR
1940
REVIEW CONTRACT
1945
GENERATE AUTHENTICATION DATA
1950
SEND AUTHENTICATION DATA AND HASH OF CONTRACT TO TRUST ENGINE
1955
VERIFY AUTHENTICATION OF USER
1960
SIGN HASH OF CONTRACT WITH USER'S PRIVATE KEY; FORWARD CONTRACT TO VENDOR SIGNED BY TRUST ENGINE
1965
RECEIVE RECEIPT(SIGNED BY TRUST ENGINE)AND HASH OF CONTRACT(SIGNED BY USER)
1970
SIGN HASH OF CONTRACT WITH VENDOR'S PRIVATE KEY; FORWARD CONTRACT TO USER SIGNED BY TRUST ENGINE.
1975
RECIEVE RECEIPT (SIGNED BY TRUST ENGINE) AND HASH OF CONTRACT(SIGNED BY SIGNED BY VENDOR)

Figure 20

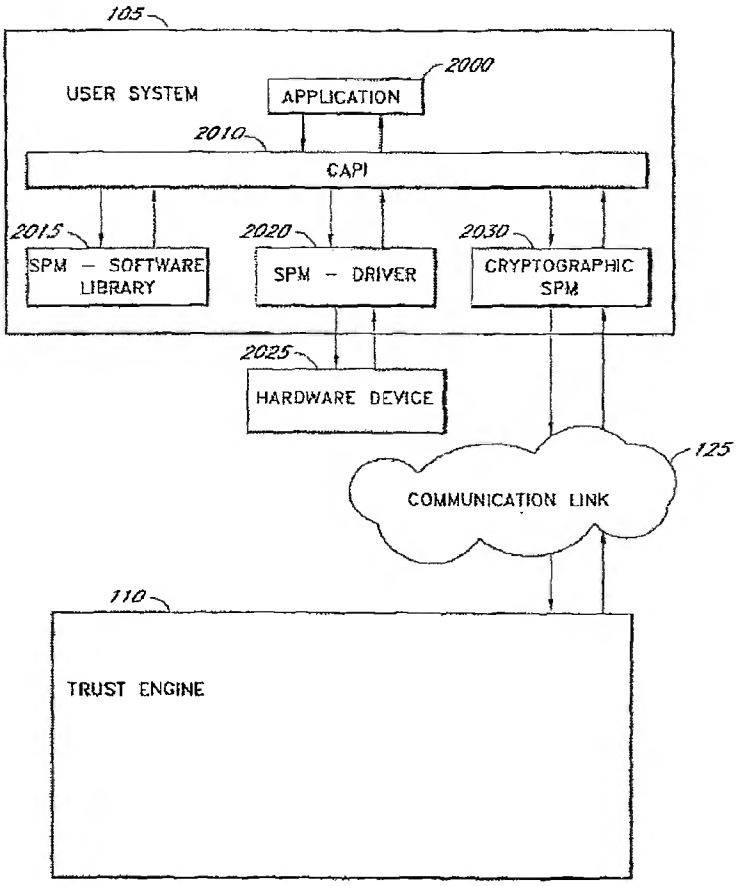

105
USER SYSTEM
APPLICATION
2000
2010
CAPI
2015
SPM – SOFTWARE LIBRARY
2020
SPM – DRIVER
2030
CRYPTOGRAPHIC SPM
2025
HARDWARE DEVICE
125
COMMUNICATION LINK
110
TRUST ENGINE

Figure 21

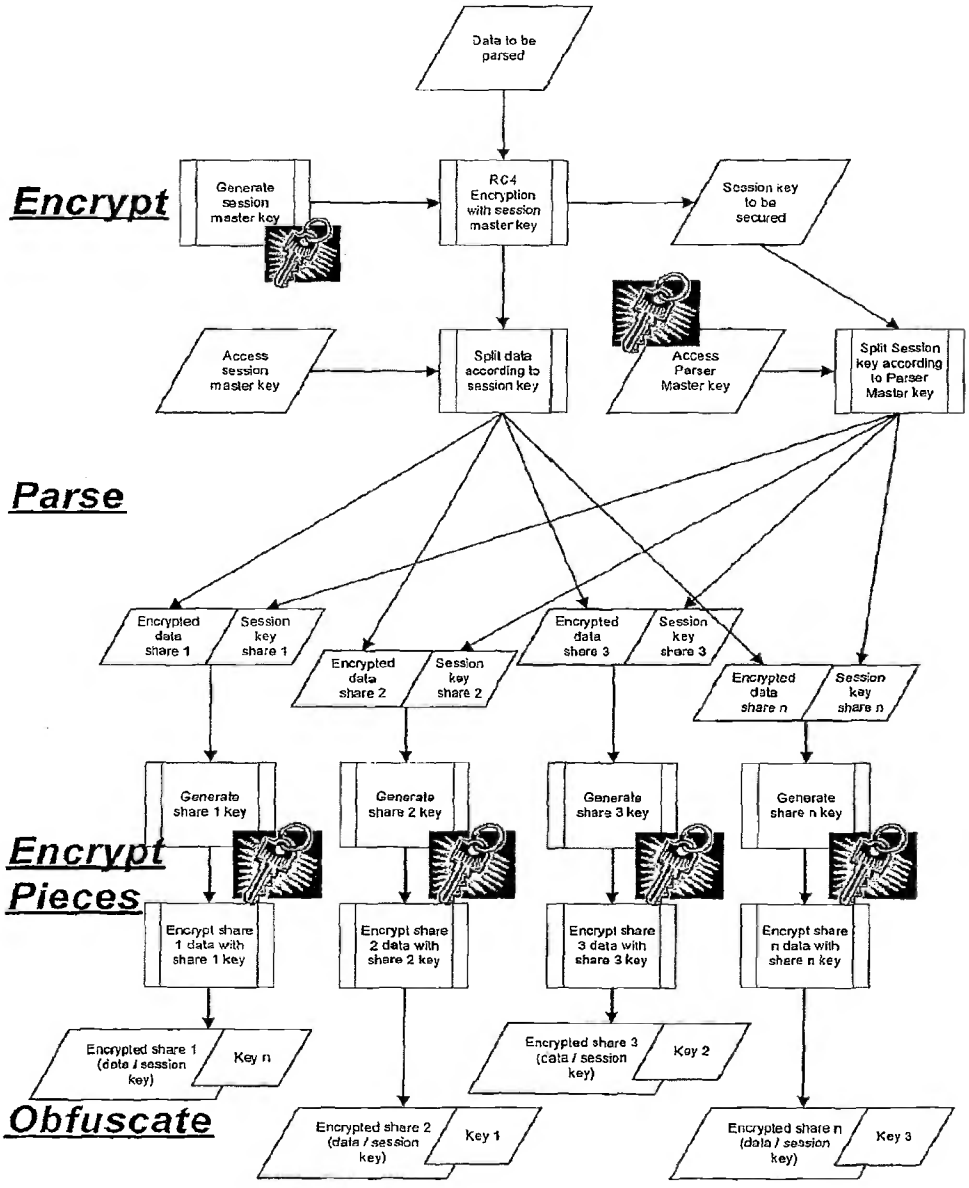

Data to be parsed
Encrypt
Generate session master key
RC4 Encryption with session master key
Session key to be secured
Access session master key
Split data according to session key
Access Parser Master key
Split Session key according to Parser Master key
Parse
Encrypted data share 1
Session key share 1
Encrypted data share 2
Session key share 2
Encrypted data share 3
Session key share 3
Encrypted data share n
Session key share n
Generate share 1 key
Generate share 2 key
Generate share 3 key
Generate share n key
Encrypt Pieces
Encrypt share 1 data with share 1 key
Encrypt share 2 data with share 2 key
Encrypt share 3 data with share 3 key
Encrypt share n data with share n key
Encrypted share 1 (data / session key)
Key n
Encrypted share 3 (data / session key)
Key 2
Obfuscate
Encrypted share 2 (data / session key)
Key 1
Encrypted share n (data / session key)
Key 3

Figure 22

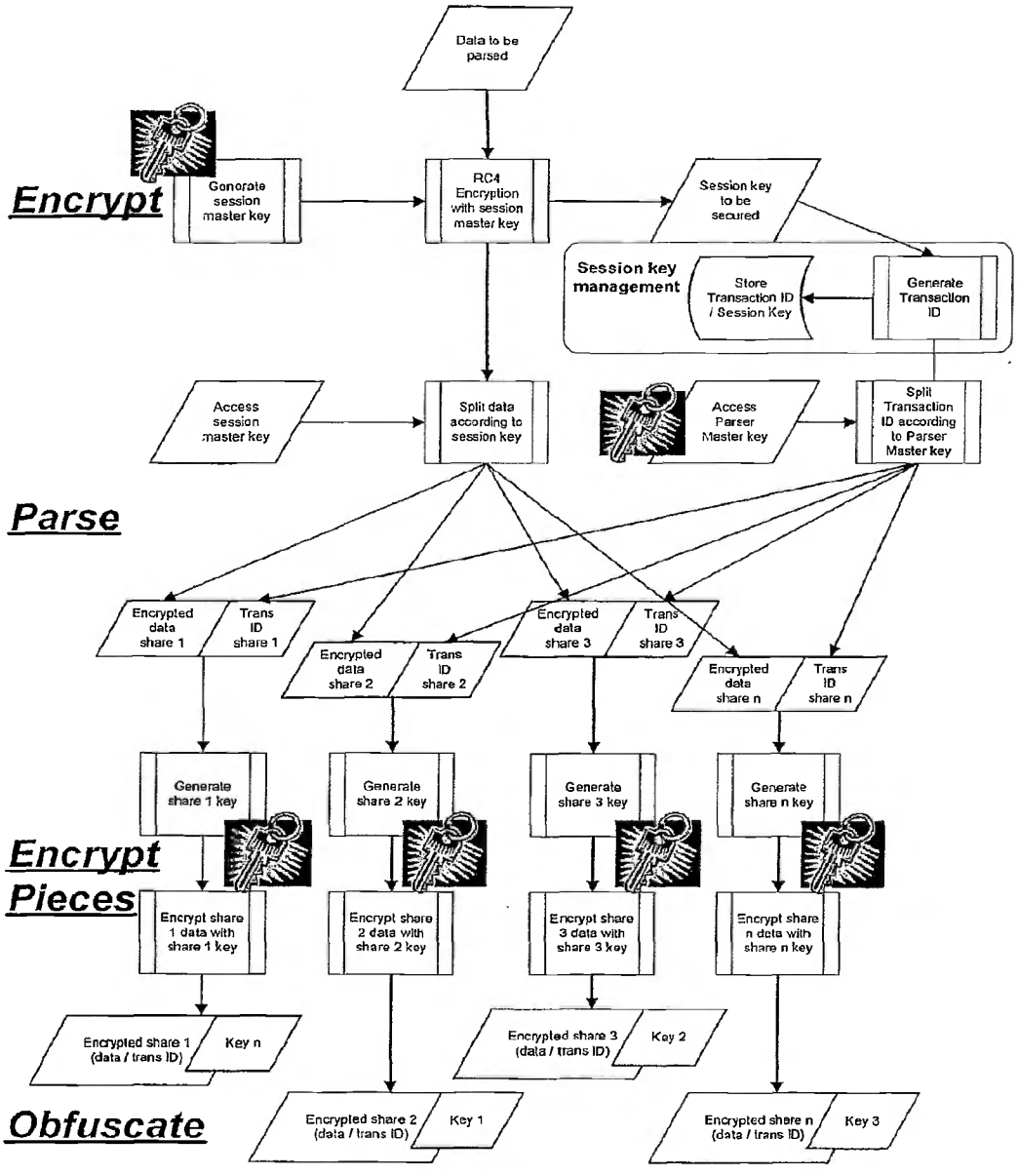

Data to be parsed
Encrypt
Generate session master key
RC4 Encryption with session master key
Session key to be secured
Session key management
Store Transaction ID / Session Key
Generate Transaction ID
Access session master key
Split data according to session key
Access Parser Master key
Split Transaction ID according to Parser Master key
Parse
Encrypted data share 1
Trans ID share 1
Encrypted data share 2
Trans ID share 2
Encrypted data share 3
Trans ID share 3
Encrypted data share n
Trans ID share n
Generate share 1 key
Generate share 2 key
Generate share 3 key
Generate share n key
Encrypt Pieces
Encrypt share 1 data with share 1 key
Encrypt share 2 data with share 2 key
Encrypt share 3 data with share 3 key
Encrypt share n data with share n key
Encrypted share 1 (data / trans ID)
Key n
Encrypted share 3 (data / trans ID)
Key 2
Obfuscate
Encrypted share 2 (data / trans ID)
Key 1
Encrypted share n (data / trans ID)
Key 3

Figure 23

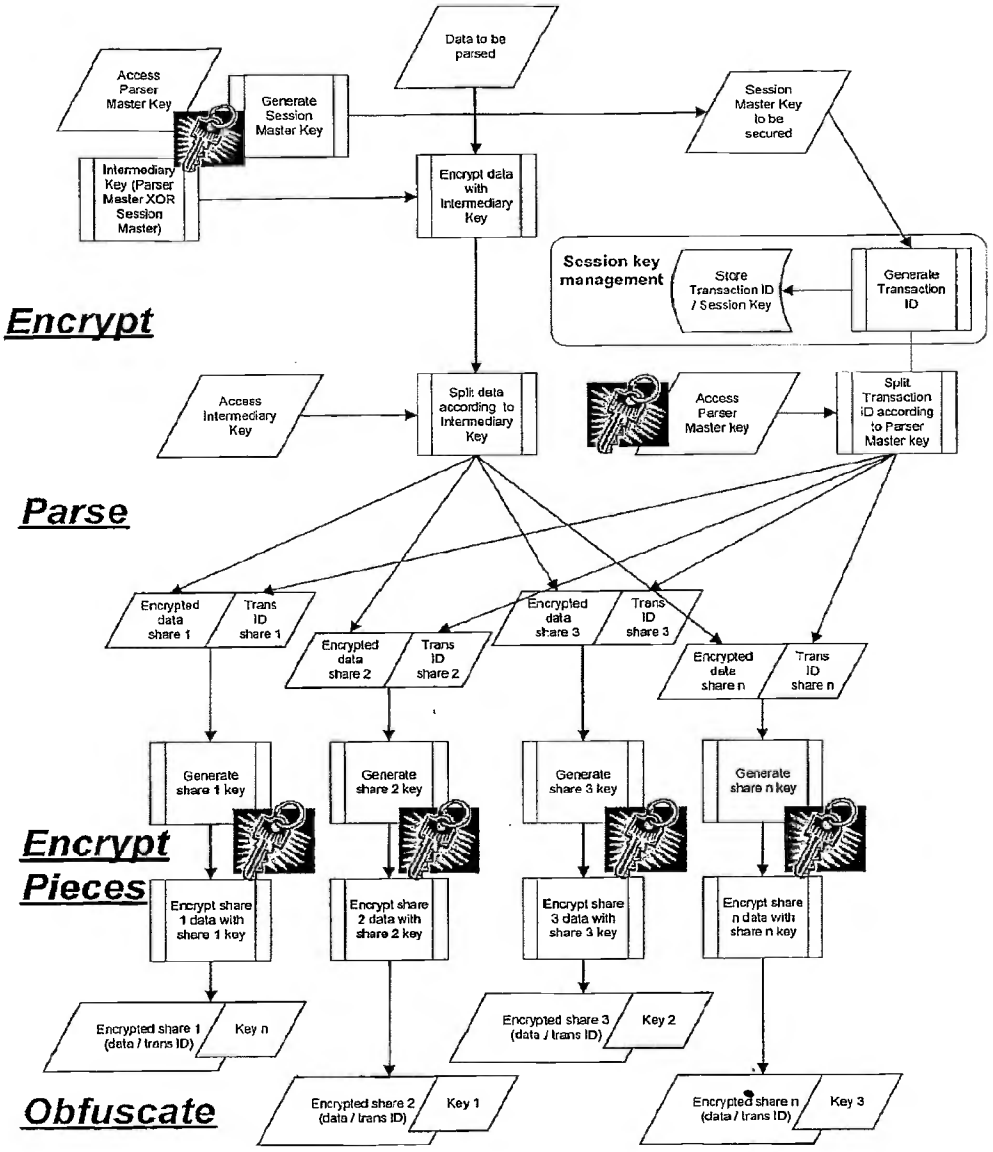

Data to be parsed
Access Parser Master Key
Generate Session Master Key
Session Master Key to be secured
Intermediary Key (Parser Master XOR Session Master)
Encrypt data with Intermediary Key
Session key management
Store Transaction ID / Session Key
Generate Transaction ID
Encrypt
Access Intermediary Key
Split data according to Intermediary Key
Access Parser Master key
Split Transaction ID according to Parser Master key
Parse
Encrypted data share 1
Trans ID share 1
Encrypted data share 2
Trans ID share 2
Encrypted data share 3
Trans ID share 3
Encrypted data share n
Trans ID share n
Generate share 1 key
Generate share 2 key
Generate share 3 key
Generate share n key
Encrypt Pieces
Encrypt share 1 data with share 1 key
Encrypt share 2 data with share 2 key
Encrypt share 3 data with share 3 key
Encrypt share n data with share n key
Encrypted share 1 (data / trans ID)
Key n
Encrypted share 3 (data / trans ID)
Key 2
Obfuscate
Encrypted share 2 (data / trans ID)
Key 1
Encrypted share n (data / trans ID)
Key 3

Figure 24

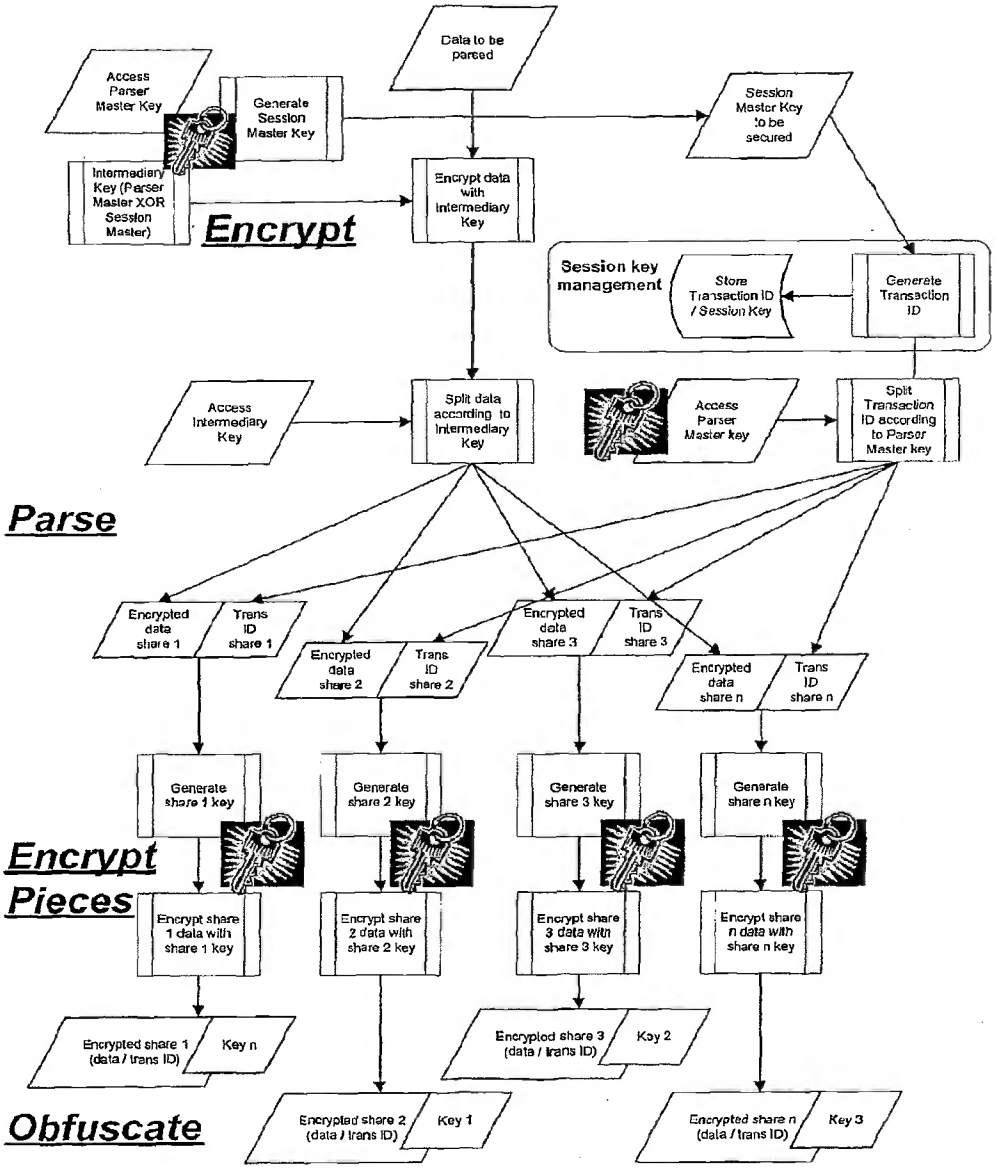
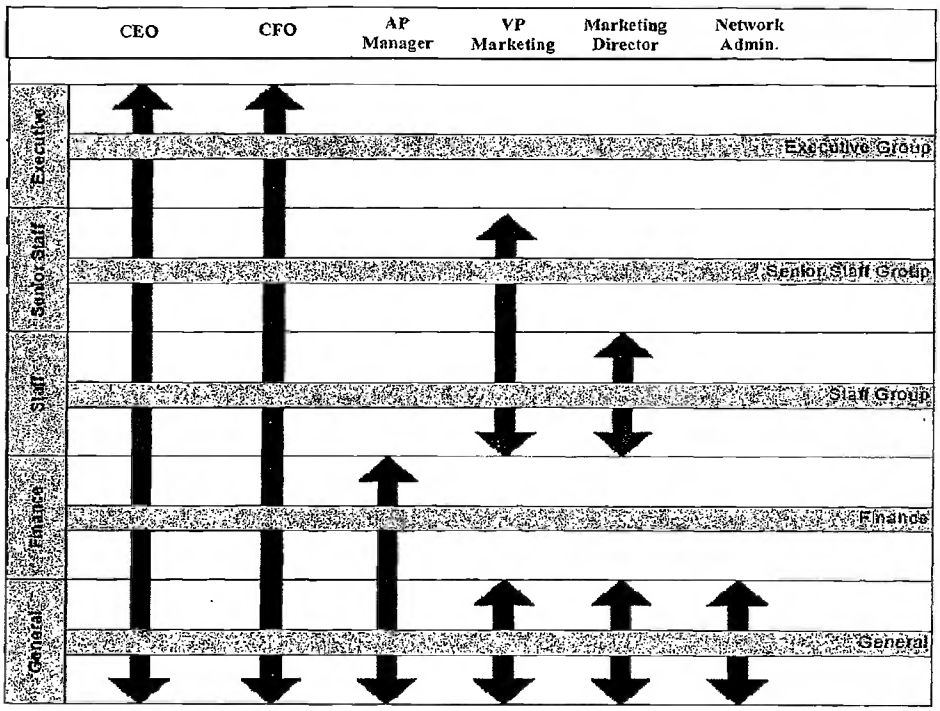

Data to be parsed
Access Parser Master Key
Generate Session Master Key
Session Master Key to be secured
Intermediary Key (Parser Master XOR Session Master)
Encrypt
Encrypt data with Intermediary Key
Session key management
Store Transaction ID / Session Key
Generate Transaction ID
Access Intermediary Key
Split data according to Intermediary Key
Access Parser Master key
Split Transaction ID according to Parser Master key
Parse
Encrypted data share 1
Trans ID share 1
Encrypted data share 2
Trans ID share 2
Encrypted data share 3
Trans ID share 3
Encrypted data share n
Trans ID share n
Generate share 1 key
Generate share 2 key
Generate share 3 key
Generate share n key
Encrypt Pieces
Encrypt share 1 data with share 1 key
Encrypt share 2 data with share 2 key
Encrypt share 3 data with share 3 key
Encrypt share n data with share n key
Encrypted share 1 (data / trans ID)
Key n
Encrypted share 3 (data / trans ID)
Key 2
Obfuscate
Encrypted share 2 (data / trans ID)
Key 1
Encrypted share n (data / trans ID)
Key 3

Figure 25


CEO
CFO
AP Manager
VP Marketing
Marketing Director
Network Admin.
Executive
Executive Group
Senior Staff
Senior Staff Group
Staff
Staff Group
Finance
Finance
General
General

PTO/SB/25 (07-09)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

TERMINAL DISCLAIMER TO OBVIATE A PROVISIONAL DOUBLE PATENTING REJECTION OVER A PENDING "REFERENCE" APPLICATION	Docket Number (Optional) SFC-2

In re Application of: Orsini et al.

Application No.: 11/258,839

Filed: October 25, 2005

For: SECURE DATA PARSER METHOD AND SYSTEM

The owner*, Security First Corp., of 100 percent interest in the instant application hereby disclaims, except as provided below, the terminal part of the statutory term of any patent granted on the instant application which would extend beyond the expiration date of the full statutory term of any patent granted on pending **reference** Application Number 12/148,365, filed on April 18, 2008, as such term is defined in 35 U.S.C. 154 and 173, and as the term of any patent granted on said **reference** application may be shortened by any terminal disclaimer filed prior to the grant of any patent on the pending **reference** application. The owner hereby agrees that any patent so granted on the instant application shall be enforceable only for and during such period that it and any patent granted on the **reference** application are commonly owned. This agreement runs with any patent granted on the instant application and is binding upon the grantee, its successors or assigns.

In making the above disclaimer, the owner does not disclaim the terminal part of any patent granted on the instant application that would extend to the expiration date of the full statutory term as defined in 35 U.S.C. 154 and 173 of any patent granted on said **reference** application, "as the term of any patent granted on said **reference** application may be shortened by any terminal disclaimer filed prior to the grant of any patent on the pending **reference** application," in the event that: any such patent: granted on the pending **reference** application: expires for failure to pay a maintenance fee, is held unenforceable, is found invalid by a court of competent jurisdiction, is statutorily disclaimed in whole or terminally disclaimed under 37 CFR 1.321, has all claims canceled by a reexamination certificate, is reissued, or is in any manner terminated prior to the expiration of its full statutory term as shortened by any terminal disclaimer filed prior to its grant.

Check either box 1 or 2 below, if appropriate.

1. ☐ For submissions on behalf of a business/organization (e.g., corporation, partnership, university, government agency, etc.), the undersigned is empowered to act on behalf of the business/organization.

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

2. ☒ The undersigned is an attorney or agent of record. Reg. No. 64,813

/Laura Zager/ — Signature

June 30, 2011 — Date

Laura Zager — Typed or printed name

(617) 951-7069 — Telephone Number

☒ Terminal disclaimer fee under 37 CFR 1.20(d) is included.

WARNING: Information on this form may become public. Credit card information should not be included on this form. Provide credit card information and authorization on PTO-2038.

*Statement under 37 CFR 3.73(b) is required if terminal disclaimer is signed by the assignee (owner). Form PTO/SB/96 may be used for making this statement. See MPEP § 324.

27088056_1

PTO/SB/96 (07-09)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

STATEMENT UNDER 37 CFR 3.73(b)

Applicant/Patent Owner: Security First Corp

Application No./Patent No.: 11/258,839 Filed/Issue Date: October 25, 2005

Titled: SECURE DATA PARSER METHOD AND SYSTEM

Security First Corp (Name of Assignee), a corporation (Type of Assignee: e.g., corporation, partnership, university, government agency, etc.)

states that it is:

1. [X] the assignee of the entire right, title, and interest in;

2. [ ] an assignee of less than the entire right, title, and interest in
(The extent (by percentage) of its ownership interest is _____ %); or

3. [ ] an assignee of an undivided interest in the entirety of (a complete assignment from one of the joint inventors was made)
the patent application/patent identified above by virtue of either:

A. [X] An assignment from the inventor(s) of the patent application/patent identified above. The assignment was recorded in the United States Patent and Trademark Office at Reel 017501, Frame 0277, or for which a copy thereof is attached.

OR

B. [ ] A chain of title from the inventor(s), of the patent application/patent identified above, to the current assignee as follows:

1. From: _____ To: _____
The document was recorded in the United States Patent and Trademark Office at
Reel _____, Frame _____, or for which a copy thereof is attached.

2. From: _____ To: _____
The document was recorded in the United States Patent and Trademark Office at
Reel _____, Frame _____, or for which a copy thereof is attached.

3. From: _____ To: _____
The document was recorded in the United States Patent and Trademark Office at
Reel _____, Frame _____, or for which a copy thereof is attached.

[ ] Additional documents in the chain of title are listed on a supplemental sheet(s).

[X] As required by 37 CFR 3.73(b)(1)(i), the documentary evidence of the chain of title from the original owner to the assignee was, or concurrently is being, submitted for recordation pursuant to 37 CFR 3.11.

[NOTE: A separate copy (i.e., a true copy of the original assignment document(s)) must be submitted to Assignment Division in accordance with 37 CFR Part 3, to record the assignment in the records of the USPTO. See MPEP 302.08]

The undersigned (whose title is supplied below) is authorized to act on behalf of the assignee.

Signature — Date: 6-29-11

Mark S. O'Hare (Printed or Typed Name) — President & CEO (Title)

Electronic Patent Application Fee Transmittal

Application Number:	11258839
Filing Date:	25-Oct-2005
Title of Invention:	Secure data parser method and system
First Named Inventor/Applicant Name:	Rick L. Orsini
Filer:	Laura A. Zager/Joanne Ryan
Attorney Docket Number:	SFC-2

Filed as Small Entity

Utility under 35 USC 111(a) Filing Fees

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Basic Filing:				
Pages:				
Claims:				
Miscellaneous-Filing:				
Petition:				
Patent-Appeals-and-Interference:				
Post-Allowance-and-Post-Issuance:				
Extension-of-Time:				
Extension - 3 months with $0 paid	2253	1	555	555

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Miscellaneous:				
Submission- Information Disclosure Stmt	1806	1	180	180
Statutory or terminal disclaimer	2814	2	70	140
		Total in USD ($)		**875**

PTO/SB/08a (01-10)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT

(Not for submission under 37 CFR 1.99)

Application Number	11258839
Filing Date	2005-10-25
First Named Inventor	Rick L. Orsini
Art Unit	2432
Examiner Name	Samson B. Lemma
Attorney Docket Number	SFC-2

U.S.PATENTS

Examiner Initial*	Cite No	Patent Number	Kind Code[1]	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	7117365	B1	2006-10-03	Rump et al.	
	2	7649992	B2	2010-01-19	Raju et al.	
	3	7218736	B1	2007-05-15	Nishimura et al.	
	4	6684330	B1	2004-01-27	Wack et al.	
	5	7133845	B1	2006-11-07	Ginter et al.	
	6	6959383	B1	2005-10-25	Terada et al.	
	7	7050580	B1	2006-05-23	Ferre Herrero	
	8	5974144		1999-10-26	Brandman	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	11258839
	Filing Date	2005-10-25
	First Named Inventor	Rick L. Orsini
	Art Unit	2432
	Examiner Name	Samson B. Lemma
	Attorney Docket Number	SFC-2

	9	7869597	B2	2011-01-11	Nakai et al.	
	10	7225158	B2	2007-05-29	Toshikage et al.	
	11	6952684	B2	2005-10-04	Toshikage et al.	
	12	6856383	B1	2005-02-15	Vachris et al.	
	13	6631201	B1	2003-10-07	Dickinson et al.	
	14	6789189	B2	2004-09-07	Wheeler et al.	
	15	7096494	B1	2006-08-22	Chen	
	16	7627125	B2	2009-12-01	Lumsden et al.	
	17	5982322		1999-11-09	Bickley et al.	

If you wish to add additional U.S. Patent citation information please click the Add button.

U.S.PATENT APPLICATION PUBLICATIONS

Examiner Initial*	Cite No	Publication Number	Kind Code[1]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	11258839
	Filing Date	2005-10-25
	First Named Inventor	Rick L. Orsini
	Art Unit	2432
	Examiner Name	Samson B. Lemma
	Attorney Docket Number	SFC-2

	1	20100154053	A1	2010-06-17	Dodgson et al.	
	2	20100153703	A1	2010-06-17	Dodgson et al.	
	3	20100153670	A1	2010-06-17	Dodgson et al.	
	4	20100150341	A1	2010-06-17	Dodgson et al.	
	5	20060177061	A1	2006-08-10	Orsini et al.	
	6	20040111608	A1	2004-06-10	Oom Temudo de Castro et al.	
	7	20070192586	A1	2007-08-16	McNeely	
	8	20080199003	A1	2008-08-21	Hennessey et al.	
	9	20090077379	A1	2009-03-19	Geyzel et al.	
	10	20020172358	A1	2002-11-21	Hurich	
	11	20050273686	A1	2005-12-08	Turner et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	11258839
	Filing Date	2005-10-25
	First Named Inventor	Rick L. Orsini
	Art Unit	2432
	Examiner Name	Samson B. Lemma
	Attorney Docket Number	SFC-2

	12	20060184764	A1	2006-08-17	Osaki	
	13	20090177894	A1	2009-07-09	Orsini et al.	
	14	20090254750	A1	2009-10-08	Bono et al.	
	15	20020108037	A1	2002-08-08	Baker	
	16	20020141590	A1	2002-10-03	Montgomery	
	17	20030068041	A1	2003-04-10	Wee et al.	
	18	20040156508	A1	2004-08-12	Tada et al.	
	19	20060294378	A1	2006-12-28	Lumsden et al.	
	20	20080170693	A1	2008-07-17	Spies et al.	

If you wish to add additional U.S. Published Application citation information please click the Add button.

FOREIGN PATENT DOCUMENTS

Examiner Initial*	Cite No	Foreign Document Number[3]	Country Code[2] i	Kind Code[4]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T[5]

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	11258839
	Filing Date	2005-10-25
	First Named Inventor	Rick L. Orsini
	Art Unit	2432
	Examiner Name	Samson B. Lemma
	Attorney Docket Number	SFC-2

	1	2004248616	AU	B2	2004-12-23	Security First Corporation		☐
	2	2008054406	WO	A2	2008-05-08	Orsini et al.		☐

If you wish to add additional Foreign Patent Document citation information please click the Add button

NON-PATENT LITERATURE DOCUMENTS

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T[5]
	1	Hugo Krawczyk "Secret sharing made Short" IBM T.J. Watson Research Center, [Online] 1998, retrieved from the Internet: URL:http://www.cs.cornell.edu/courses/cs75 4/2001fa/secretshort.pdf> [retrieved on 11-24-2008]	☐
	2	Mitchell C.J. " Making Serial Number Based Authentication Robust Against Loss of State" Operating Systems Review, ACM, New York, NY, Vol. 34, No. 3 July 1, 2000, pgs. 56-59, XP001096714, ISSN: 0163-5980	☐
	3	Damgard, Ivan; Groth, Jens; Non-interactive and Reusable Non-malleable Commitment Schemes; June 9-11, 2003; ACM; STOC '03; Pages 427-428.	☐

If you wish to add additional non-patent literature document citation information please click the Add button

EXAMINER SIGNATURE

Examiner Signature		Date Considered	

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

AMENDMENTS TO THE CLAIMS

The following listing of claims will replace all prior versions, and listings, of claims in the above-identified patent application.

Listing of Claims

1. (Currently Amended) A method for securing streaming of a data set from a first location to a second location, the method comprising:

generating, using processing circuitry at the first location, at least two portions of data from the data set, wherein each of the at least two portions of data respectively contains a substantially random distribution of a respective subset of the data set;

streaming from the first location to the second location, as at least two streaming portions of data, the at least two portions of data over at least one communications path, wherein the at least two portions of data ~~are communicated~~ stream separately from each other; and

restoring, using processing circuitry at the second location, data from at least a subset of the at least two streaming portions of data, wherein the data is restored as the at least two streaming portions of data are being received at the second location.

2. (Previously Presented) The method of claim 1 wherein the streaming comprises streaming the at least two portions of data on separate communications paths in parallel.

3. (Currently amended) The method of claim 1 wherein the streaming comprises streaming ~~that~~ the at least two portions of data on a single communications path in a serial transmission.

4. (Original) The method of claim 1 wherein the generating is accomplished using an email application.

5. (Original) The method of claim 1 wherein the generating is accomplished using a video broadcasting application.

6. (Previously Presented) The method of claim 1 wherein the streaming over the at least one communications path comprises streaming over a wireless communications path.

7. (Previously Presented) The method of claim 6, wherein the streaming over the wireless communications path comprises streaming over the wireless communications path using an 802.11 protocol.

8. (Previously Presented) The method of claim 6, wherein the streaming over the wireless communications path comprises streaming over the wireless communications path using Bluetooth.

9. (Previously Presented) The method of claim 6, wherein the streaming over the wireless communications path comprises streaming over a wireless LAN.

10. (Previously Presented) The method of claim 6, wherein the streaming over the wireless communications path comprises streaming over a cellular network.

11. (Previously Presented) The method of claim 6, wherein the streaming over the wireless communications path comprises streaming over a peer to peer wireless network.

12. (Previously Presented) The method of claim 6, wherein the streaming over the wireless communications path comprises streaming over a hub based wireless network.

13. (Previously Presented) The method of claim 6, wherein the streaming over the wireless communications path comprises streaming over a satellite transmission.

14. (Original) The method of claim 1 wherein at least one of the first location and the second location is a computer peripheral device.

15. (Original) The method of claim 1 wherein at least one of the first location and the second location is a PDA device.

16. (Original) The method of claim 1 wherein at least one of the first location and the second location is a storage device.

17. (Previously Presented) The method of claim 1 wherein the streaming over the at least one communications path comprises streaming over a wired communications path.

18. (Previously Presented) A method for securing a data set, the method comprising:

generating, using processing circuitry, at least two portions of data from the data set, wherein each of the at least two portions of data respectively contains a substantially random distribution of a respective subset of the data set;

storing the at least two portions of data on at least two data depositories, wherein the at least two data depositories are physically separate from each other; and

using at least one of the at least two portions of data stored on at least one of the at least two data depositories as an authentication key stored in an authentication depository in order to restore the data set.

19. (Original) The method of claim 18 wherein the authentication depository is a physical token device.

20. (Original) The method of claim 18 further comprising restoring the data set when the authentication depository is coupled to at least a subset of the at least two data depositories.

21. (Original) The method of claim 20 further comprising removing access to the data set when the authentication depository is no longer coupled to the at least a subset of the at least two data depositories.

22. (Previously Presented) A method for securing a data set, the method comprising:

generating, using processing circuitry, at least two portions of data from the data set, wherein each of the at least two portions of data respectively contains a substantially random distribution of a respective subset of the data set;

generating respective integrity information for each of the at least two portions of data;

writing the respective integrity information to each of the at least two portions of data;

storing each of the at least two portions of data separately; and

restoring the data set from at least a subset of the at least two portions of data when the respective integrity

information of the at least a subset of the at least two portions of data has been verified.

23. (Original) The method of claim 22, wherein the integrity information is generated using a hash algorithm.

24. (Original) The method of claim 22, wherein the integrity information is generated using a MAC signature.

25. (Original) The method of claim 22, wherein the integrity information is generated using a digital signature.

26. (Previously Presented) A method for securing a data set, the method comprising:

generating, using processing circuitry, at least two portions of data from the data set, wherein each of the at least two portions of data respectively contains a substantially random distribution of a respective subset of the data set;

storing each of the at least two portions of data separately; and

restoring the data set from at least a subset of the at least two portions of data, wherein at least one particular portion of data is required in order to restore the data set.

27. (Original) The method of claim 26 further comprising encrypting at least one of the at least two portions of data, the at least one of the at least two portions of data comprising the at least one particular portion of data.

28. (Original) The method of claim 27 wherein the encrypting comprises encrypting at least one of the at least two portions of data according to a first encryption key and at least

one other of the at two least portions of data according to a second encryption key.

29. (Original) The method of claim 27 wherein the encrypting comprises encrypting at least one of the at least two portion of data using a public key that may be decrypted using a private key.

30. (Previously Presented) A method for securing a data set, the method comprising:

generating, using processing circuitry, at least two portions of data from the data set, wherein each of the at least two portions of data respectively contains a substantially random distribution of a respective subset of the data set, and wherein the probability that data from the data set is distributed to at least one of the at least two portions of data is less than the probability that data from the data set is distributed to a remainder of the at least two portions of data;

storing each of the at least two portions of data separately; and

restoring the data set from at least a subset of the at least two portions of data.

31. (Previously Presented) A method for securing a data set, the method comprising:

generating, using processing circuitry, at least two portions of data from the data set, wherein each of the at least two portions of data respectively contains a substantially random distribution of a respective subset of the data set, and wherein at least one of the at least two portions of data has a predetermined size;

storing each of the at least two portions of data separately; and

restoring the data set from at least a subset of the at least two portions of data.

32. (Original) The method of claim 31 wherein the generating the at least two portions of data comprises using a splitting key having a splitting key size and wherein the predetermined size is the splitting key size.

33. (Original) The method of claim 31 further comprising encrypting the data set with an encryption key having an encryption key size prior to the generating the at least two portions of data and wherein the predetermined size is the encryption key size.

34. (Previously Presented) A method for securing a data set, the method comprising:

generating, using processing circuitry, at least two portions of data from the data set, wherein each of the at least two portions of data respectively contains a substantially random distribution of a respective subset of the data set, and wherein the probability that data from the data set is distributed to each of the at least two portions of data is substantially equal;

storing each of the at least two portions of data separately; and

restoring the data set from at least a subset of the at least two portions of data.

35. (Previously Presented) A method for securing a data set, the method comprising:

generating, using processing circuitry, according to a splitting key, at least two portions of data from the data set, wherein each of the at least two portions of data respectively

contains a substantially random distribution of a respective subset of the data set;

generating, according to a Shamir algorithm, splitting key information that is capable of being used to generate the splitting key, wherein the splitting key information is distributed among the at least two portions of data;

storing each of the at least two portions of data separately; and

restoring the data set from at least a subset of the at least two portions of data.

36. (Original) The method of claim 35 wherein restoring the data set comprises restoring the splitting key from the splitting key information from at least a subset of the at least two portions of data.

37-40. (Cancelled)

41. (Previously Presented) A method for securing a data set, the method comprising:

determining a number of portions of data into which to distribute the data set;

generating a substantially random number;

determining, using processing circuitry, into which of the portions of data to distribute each unit of data from the data set based on corresponding data from the substantially random number and distributing accordingly;

storing the portions of data separately; and

restoring the data set from at least a subset of the portions of data, wherein the determining into which of the portions of data to distribute each unit of data from the data set comprises determining into which of the portions of data to

distribute each unit of data from the data set based on a hash function of corresponding data from the substantially random number.

42. (Cancelled)

43. (Original) The method of claim 41 wherein the unit of data is a bit.

44. (Original) The method of claim 41 wherein the size of the substantially random number is a multiple, M, of the size of the data set and wherein the determining into which of the portions of data to distribute each unit of data from the data set comprises determining into which of the portions of data to distribute each unit of data from the data set based on a hash function of a corresponding M bits from the substantially random number.

45. (Original) The method of claim 44 wherein the multiple, M, is 8 and wherein the determining into which of the portions of data to distribute each unit of data from the data set based on a hash function of a corresponding M bits from the substantially random number comprises determining into which of the portions of data to distribute each bit of data from the data set based on a hash function of a corresponding byte from the substantially random number.

46. (Original) The method of claim 41 further comprising appending redundancy data to each of the portions of data.

47. (Previously Presented) A method for securing a data set, the method comprising:

determining a number of portions of data into which to distribute the data set;

generating a substantially random number;

for each unit of data of the data set, splitting the unit of data into a left segment and a right segment by determining, based on corresponding data from the substantially random number, a splitting location wherein bits to the left of the splitting location represent the left segment and bits to the right of the splitting location represent the right segment;

determining, using processing circuitry, into which of the portions of data to distribute each of the left segment and the right segment of each unit of data from the data set based on corresponding data from the substantially random number and distributing accordingly;

storing the portions of data separately; and

restoring the data set from at least a subset of the portions of data.

48. (Original) The method of claim 47 wherein the splitting the unit of data into the left segment and the right segment comprises splitting the unit of data into the left segment and the right segment based on a hash function of a corresponding data from the substantially random number.

49. (Original) The method of claim 47 wherein the unit of data is a byte.

50. (Original) The method of claim 47 wherein generating the substantially random number comprises generating the substantially random number of size equal to the size of the data set.

51. (Original) The method of claim 50 wherein the unit of data is a byte and wherein the splitting the unit of data into the left segment and the right segment comprises splitting the unit

of data into the left segment and the right segment based on a hash function of a corresponding unit of data from the substantially random number.

52. (Original) The method of claim 47 wherein the determining into which of the portions of data to distribute each of the left segment and the right segment comprises determining into which of the portions of data to distribute each of the left segment and the right segment of each unit of data from the data set based on corresponding data from the substantially random number and based on a table containing an array of all possible distributions of the right segment and the left.

53. (Original) The method of 47 wherein the determining into which of the portions of data to distribute each of the left segment and the right segment comprises determining into which of the portions of data to distribute each of the left segment and the right segment of each unit of data from the data set based on a hash function of corresponding data from the substantially random number and based on a table containing an array of all possible distributions of the right segment and the left segment, wherein the hash function of the corresponding data from the substantially random number indicates which entry in the table to use, the entry indicating a destination portion for the left segment and a destination portion for the right segment.

54. (Original) The method of claim 53 wherein:

generating the substantially random number comprises generating the substantially random number of size equal to the size of the data set;

the unit of data is a byte; and

the corresponding data is a corresponding unit of data.

55-68. (Cancelled)

69. (Currently amended) A system for securing streaming of a data set from a first location to a second location, the system comprising:

a first processor for generating at the first location at least two portions of data from the data set, wherein each of the at least two portions of data respectively contains a substantially random distribution of a respective subset of the data set;

at least one communications path over which are streamed, as at least two streaming portions of data, the at least two portions of data from the first location to the second location, wherein the at least two portions of data ~~are communicated~~ stream separately from each other; and

a second processor for restoring at the second location data from at least a subset of the at least two streaming portions of data, wherein the data is restored as the at least two streaming portions of data are being received at the second location.

70. (Previously Presented) The system of claim 69 wherein the at least two streaming portions of data are streamed on separate communications paths in parallel.

71. (Previously Presented) The system of claim 69 wherein the at least two streaming portions of data are streamed on a single communications path in a serial transmission.

72. (Original) The system of claim 69 wherein the first processor is used in conjunction with an email application.

73. (Original) The system of claim 69 wherein the first processor is used in conjunction with a video broadcasting application.

74. (Original) The system of claim 69 wherein the at least one communications path comprises a wireless communications path.

75. (Original) The system of claim 74, wherein the wireless communications path comprises an 802.11 protocol communications path.

76. (Original) The system of claim 74, wherein the wireless communications path comprises a Bluetooth communications path.

77. (Original) The system of claim 74, wherein the wireless communications path comprises a wireless LAN.

78. (Original) The system of claim 74, wherein the wireless communications path comprises a cellular network.

79. (Original) The system of claim 74, wherein the wireless communications path comprises a peer to peer wireless network.

80. (Original) The system of claim 74, wherein the wireless communications path comprises a hub based wireless network.

81. (Original) The system of claim 74, wherein wireless communications path comprises a satellite communications path.

82. (Original) The system of claim 69 wherein at least one of the first location and the second location is a computer peripheral device.

83. (Original) The system of claim 69 wherein at least one of the first location and the second location is a PDA device.

84. (Original) The system of claim 69 wherein at least one of the first location and the second location is a storage device.

85. (Original) The system of claim 69 wherein the at least one communications path comprises a wired communications path.

PTO/SB/22 (07-09)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless if displays a valid OMB control number.

PETITION FOR EXTENSION OF TIME UNDER 37 CFR 1.136(a) FY 2009 *(Fees pursuant to the Consolidated Appropriations Act, 2005 (H.R. 4818).)*	Docket Number (Optional) SFC-2
Application Number 11/258,839	Filed October 25, 2005
For SECURE DATA PARSER METHOD AND SYSTEM	
Art Unit 2432	Examiner S. B. Lemma

This is a request under the provisions of 37 CFR 1.136(a) to extend the period for filing a reply in the above identified application.

The requested extension and fee are as follows (check time period desired and enter the appropriate fee below):

		Fee	Small Entity Fee	
[ ]	One month (37 CFR 1.17(a)(1))	$130	$65	$
[ ]	Two months (37 CFR 1.17(a)(2))	$490	$245	$
[x]	Three months (37 CFR 1.17(a)(3))	$1110	$555	$ 555
[ ]	Four months (37 CFR 1.17(a)(4))	$1730	$865	$
[ ]	Five months (37 CFR 1.17(a)(5))	$2350	$1175	$

[x] Applicant claims small entity status. See 37 CFR 1.27.

[ ] A check in the amount of the fee is enclosed.

[ ] Payment by credit card. Form PTO-2038 is attached.

[x] The Director has already been authorized to charge fees in this application to a Deposit Account.

[x] The Director is hereby authorized to charge any fees which may be required, or credit any overpayment, to Deposit Account Number 06-1075 .

WARNING: Information on this form may become public. Credit card information should not be included on this form. Provide credit card information and authorization on PTO-2038.

I am the

[ ] applicant/inventor.

[ ] assignee of record of the entire interest. See 37 CFR 3.71.
Statement under 37 CFR 3.73(b) is enclosed. (Form PTO/SB/96).

[x] attorney or agent of record. Registration Number 64,813

[ ] attorney or agent under 37 CFR 1.34.
Registration number if acting under 37 CFR 1.34 _____

/Laura Zager/	June 30, 2011
Signature	Date
Laura Zager	(617) 951-7069
Typed or printed name	Telephone Number

NOTE: Signatures of all the inventors or assignees of record of the entire interest or their representative(s) are required. Submit multiple forms if more than one signature is required, see below.

[x] Total of 1 forms are submitted.

27091106_1

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NUMBER	FILING OR 371(C) DATE	FIRST NAMED APPLICANT	ATTY. DOCKET NO./TITLE
11/258,839	10/25/2005	Rick L. Orsini	SFC-2

CONFIRMATION NO. 2420

POA ACCEPTANCE LETTER

1473
ROPES & GRAY LLP
PATENT DOCKETING 39/361
1211 AVENUE OF THE AMERICAS
NEW YORK, NY 10036-8704

OC00000048645779

Date Mailed: 07/13/2011

NOTICE OF ACCEPTANCE OF POWER OF ATTORNEY

This is in response to the Power of Attorney filed 06/30/2011.

The Power of Attorney in this application is accepted. Correspondence in this application will be mailed to the above address as provided by 37 CFR 1.33.

/sibrahim/

Office of Data Management, Application Assistance Unit (571) 272-4000, or (571) 272-4200, or 1-888-786-0101

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NUMBER	FILING OR 371(C) DATE	FIRST NAMED APPLICANT	ATTY. DOCKET NO./TITLE
11/258,839	10/25/2005	Rick L. Orsini	SFC-2

CONFIRMATION NO. 2420

POWER OF ATTORNEY NOTICE

1473
ROPES & GRAY LLP
PATENT DOCKETING 39/361
1211 AVENUE OF THE AMERICAS
NEW YORK, NY 10036-8704

OC000000048645651

Date Mailed: 07/13/2011

NOTICE REGARDING CHANGE OF POWER OF ATTORNEY

This is in response to the Power of Attorney filed 06/30/2011.

- The Power of Attorney to you in this application has been revoked by the assignee who has intervened as provided by 37 CFR 3.71. Future correspondence will be mailed to the new address of record(37 CFR 1.33).

/sibrahim/

Office of Data Management, Application Assistance Unit (571) 272-4000, or (571) 272-4200, or 1-888-786-0101

Application Number	Application/Control No.	Applicant(s)/Patent under Reexamination	
	11/258,839	ORSINI ET AL.	

Document Code - DISQ	Internal Document – DO NOT MAIL

TERMINAL DISCLAIMER	☒ APPROVED	☐ DISAPPROVED
Date Filed : 30 JUN 2011	This patent is subject to a Terminal Disclaimer	

Approved/Disapproved by:

TWO TDS FILED AND APPROVED

JAB

U.S. Patent and Trademark Office

Index of Claims	Application/Control No.	Applicant(s)/Patent Under Reexamination
	11258839	ORSINI ET AL.
	Examiner	**Art Unit**
	Samson B Lemma	2432

✓	Rejected	-	Cancelled	N	Non-Elected	A	Appeal
=	Allowed	÷	Restricted	I	Interference	O	Objected

☐ Claims renumbered in the same order as presented by applicant ☐ CPA ☐ T.D. ☐ R.1.47

CLAIM		DATE								
Final	Original	12/29/2009	04/15/2010	12/20/2010	10/01/2011					
	1	÷	✓	✓	O					
	2	÷	✓	✓	O					
	3	÷	✓	✓	O					
	4	÷	✓	✓	O					
	5	÷	✓	✓	O					
	6	÷	✓	✓	O					
	7	÷	✓	✓	O					
	8	÷	✓	✓	O					
	9	÷	✓	✓	O					
	10	÷	✓	✓	O					
	11	÷	✓	✓	O					
	12	÷	✓	✓	O					
	13	÷	✓	✓	O					
	14	÷	✓	✓	O					
	15	÷	✓	✓	O					
	16	÷	✓	✓	O					
	17	÷	✓	✓	O					
	18	÷	✓	✓	=					
	19	÷	✓	✓	=					
	20	÷	✓	✓	=					
	21	÷	✓	✓	=					
	22	÷	✓	✓	=					
	23	÷	✓	✓	=					
	24	÷	✓	✓	=					
	25	÷	✓	✓	=					
	26	÷	✓	✓	=					
	27	÷	✓	✓	=					
	28	÷	✓	✓	=					
	29	÷	✓	✓	=					
	30	÷	✓	✓	=					
	31	÷	✓	✓	=					
	32	÷	✓	✓	=					
	33	÷	✓	✓	=					
	34	÷	✓	✓	=					
	35	÷	✓	✓	=					
	36	÷	✓	✓	=					

Index of Claims	Application/Control No.	Applicant(s)/Patent Under Reexamination
	11258839	ORSINI ET AL.
	Examiner	**Art Unit**
	Samson B Lemma	2432

✓	Rejected	-	Cancelled	N	Non-Elected	A	Appeal
=	Allowed	÷	Restricted	I	Interference	O	Objected

☐ Claims renumbered in the same order as presented by applicant ☐ CPA ☐ T.D. ☐ R.1.47

CLAIM		DATE								
Final	Original	12/29/2009	04/15/2010	12/20/2010	10/01/2011					
	37	÷	-	-	-					
	38	÷	-	-	-					
	39	÷	-	-	-					
	40	÷	-	-	-					
	41	÷	✓	✓	=					
	42	÷	✓	-	-					
	43	÷	✓	✓	=					
	44	÷	✓	✓	=					
	45	÷	✓	✓	=					
	46	÷	✓	✓	=					
	47	÷	✓	✓	=					
	48	÷	✓	✓	=					
	49	÷	✓	✓	=					
	50	÷	✓	✓	=					
	51	÷	✓	✓	=					
	52	÷	✓	✓	=					
	53	÷	✓	✓	=					
	54	÷	✓	✓	=					
	55	÷	-	-	-					
	56	÷	-	-	-					
	57	÷	-	-	-					
	58	÷	-	-	-					
	59	÷	-	-	-					
	60	÷	-	-	-					
	61	÷	-	-	-					
	62	÷	-	-	-					
	63	÷	-	-	-					
	64	÷	-	-	-					
	65	÷	-	-	-					
	66	÷	-	-	-					
	67	÷	-	-	-					
	68	÷	-	-	-					
	69	÷	✓	✓	O					
	70	÷	✓	✓	O					
	71	÷	✓	✓	O					
	72	÷	✓	✓	O					

Index of Claims	Application/Control No.	Applicant(s)/Patent Under Reexamination
	11258839	ORSINI ET AL.
	Examiner	**Art Unit**
	Samson B Lemma	2432

✓	Rejected	-	Cancelled	N	Non-Elected	A	Appeal
=	Allowed	÷	Restricted	I	Interference	O	Objected

☐ Claims renumbered in the same order as presented by applicant ☐ CPA ☐ T.D. ☐ R.1.47

CLAIM		DATE								
Final	Original	12/29/2009	04/15/2010	12/20/2010	10/01/2011					
	73	÷	✓	✓	O					
	74	÷	✓	✓	O					
	75	÷	✓	✓	O					
	76	÷	✓	✓	O					
	77	÷	✓	✓	O					
	78	÷	✓	✓	O					
	79	÷	✓	✓	O					
	80	÷	✓	✓	O					
	81	÷	✓	✓	O					
	82	÷	✓	✓	O					
	83	÷	✓	✓	O					
	84	÷	✓	✓	O					
	85	÷	✓	✓	O					

INFORMATION DISCLOSURE STATEMENT BY APPLICANT

(Not for submission under 37 CFR 1.99)

Application Number	11258839
Filing Date	2005-10-25
First Named Inventor	Rick L. Orsini
Art Unit	2432
Examiner Name	Samson B. Lemma
Attorney Docket Number	SFC-2

U.S.PATENTS

Examiner Initial*	Cite No	Patent Number	Kind Code[1]	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	7117365	B1	2006-10-03	Rump et al.	
	2	7649992	B2	2010-01-19	Raju et al.	
	3	7218736	B1	2007-05-15	Nishimura et al.	
	4	6684330	B1	2004-01-27	Wack et al.	
	5	7133845	B1	2006-11-07	Ginter et al.	
	6	6959383	B1	2005-10-25	Terada et al.	
	7	7050580	B1	2006-05-23	Ferre Herrero	
	8	5974144		1999-10-26	Brandman	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.L

Receipt date: 06/30/2011	Application Number	11258839	11258839 - GAU: 2432
INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Filing Date	2005-10-25	
	First Named Inventor	Rick L. Orsini	
	Art Unit	2432	
	Examiner Name	Samson B. Lemma	
	Attorney Docket Number	SFC-2	

	9	7869597	B2	2011-01-11	Nakai et al.	
	10	7225158	B2	2007-05-29	Toshikage et al.	
	11	6952684	B2	2005-10-04	Toshikage et al.	
	12	6856383	B1	2005-02-15	Vachris et al.	
	13	6631201	B1	2003-10-07	Dickinson et al.	
	14	6789189	B2	2004-09-07	Wheeler et al.	
	15	7096494	B1	2006-08-22	Chen	
	16	7627125	B2	2009-12-01	Lumsden et al.	
	17	5982322		1999-11-09	Bickley et al.	

If you wish to add additional U.S. Patent citation information please click the Add button.

U.S.PATENT APPLICATION PUBLICATIONS

Examiner Initial*	Cite No	Publication Number	Kind Code[1]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear

EFS Web 2.1.17

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.L

Receipt date: 06/30/2011

INFORMATION DISCLOSURE STATEMENT BY APPLICANT
(Not for submission under 37 CFR 1.99)

Application Number	11258839 11258839 - GAU: 2432
Filing Date	2005-10-25
First Named Inventor	Rick L. Orsini
Art Unit	2432
Examiner Name	Samson B. Lemma
Attorney Docket Number	SFC-2

	1	20100154053	A1	2010-06-17	Dodgson et al.	
	2	20100153703	A1	2010-06-17	Dodgson et al.	
	3	20100153670	A1	2010-06-17	Dodgson et al.	
	4	20100150341	A1	2010-06-17	Dodgson et al.	
	5	20060177061	A1	2006-08-10	Orsini et al.	
	6	20040111608	A1	2004-06-10	Oom Temudo de Castro et al.	
	7	20070192586	A1	2007-08-16	McNeely	
	8	20080199003	A1	2008-08-21	Hennessey et al.	
	9	20090077379	A1	2009-03-19	Geyzel et al.	
	10	20020172358	A1	2002-11-21	Hurich	
	11	20050273686	A1	2005-12-08	Turner et al.	

EFS Web 2.1.17

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.L

Receipt date: 06/30/2011	Application Number	11258839	11258839 - GAU: 2432
INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Filing Date	2005-10-25	
	First Named Inventor	Rick L. Orsini	
	Art Unit	2432	
	Examiner Name	Samson B. Lemma	
	Attorney Docket Number	SFC-2	

	12	20060184764	A1	2006-08-17	Osaki	
	13	20090177894	A1	2009-07-09	Orsini et al.	
	14	20090254750	A1	2009-10-08	Bono et al.	
	15	20020108037	A1	2002-08-08	Baker	
	16	20020141590	A1	2002-10-03	Montgomery	
	17	20030068041	A1	2003-04-10	Wee et al.	
	18	20040156508	A1	2004-08-12	Tada et al.	
	19	20060294378	A1	2006-12-28	Lumsden et al.	
	20	20080170693	A1	2008-07-17	Spies et al.	

If you wish to add additional U.S. Published Application citation information please click the Add button.

FOREIGN PATENT DOCUMENTS

Examiner Initial*	Cite No	Foreign Document Number[3]	Country Code[2] i	Kind Code[4]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T[5]

EFS Web 2.1.17

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.L

Receipt date: 06/30/2011	Application Number	11258839	11258839 - GAU: 2432
INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**	Filing Date	2005-10-25	
	First Named Inventor	Rick L. Orsini	
	Art Unit	2432	
	Examiner Name	Samson B. Lemma	
	Attorney Docket Number	SFC-2	

	1	2004248616	AU	B2	2004-12-23	Security First Corporation		☐
	2	2008054406	WO	A2	2008-05-08	Orsini et al.		☐

If you wish to add additional Foreign Patent Document citation information please click the Add button

NON-PATENT LITERATURE DOCUMENTS

Examiner Initials*	Cite No	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc), date, pages(s), volume-issue number(s), publisher, city and/or country where published.	T[5]
	1	Hugo Krawczyk "Secret sharing made Short" IBM T.J. Watson Research Center, [Online] 1998, retrieved from the Internet: URL:http://www.cs.cornell.edu/courses/cs75 4/2001fa/secretshort.pdf> [retrieved on 11-24-2008]	☐
	2	Mitchell C.J. " Making Serial Number Based Authentication Robust Against Loss of State" Operating Systems Review, ACM, New York, NY, Vol. 34, No. 3 July 1, 2000, pgs. 56-59, XP001096714, ISSN: 0163-5980	☐
	3	Damgard, Ivan; Groth, Jens; Non-interactive and Reusable Non-malleable Commitment Schemes; June 9-11, 2003; ACM; STOC '03; Pages 427-428.	☐

If you wish to add additional non-patent literature document citation information please click the Add button

EXAMINER SIGNATURE

Examiner Signature	/Samson Lemma/	Date Considered	10/10/2011

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through a citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.L

Search Notes	Application/Control No.	Applicant(s)/Patent Under Reexamination
	11258839	ORSINI ET AL.
	Examiner	**Art Unit**
	Samson B Lemma	2432

SEARCHED

Class	Subclass	Date	Examiner
713	176, 186,156	10/01/2011	SL
380	33, 268, 270, 247, 286	10/01/2011	SL

SEARCH NOTES

Search Notes	Date	Examiner
713/$, 380/$ (With text search)	10/01/2011	SL
EAST (See attached)	10/01/2011	SL
NPL (IEEE, ACM DIGITAL library, google, citeseer,...)	10/01/2011	SL
Inventor's name search	10/01/2011	SL

INTERFERENCE SEARCH

Class	Subclass	Date	Examiner

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
11/258,839	10/25/2005	Rick L. Orsini	104093-0002-101	2420

1473 7590 10/17/2011

ROPES & GRAY LLP
PATENT DOCKETING 39/361
1211 AVENUE OF THE AMERICAS
NEW YORK, NY 10036-8704

EXAMINER	
LEMMA, SAMSON B	

ART UNIT	PAPER NUMBER
2432	

NOTIFICATION DATE	DELIVERY MODE
10/17/2011	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

USPatentMail@ropesgray.com
USPatentMail2@ropesgray.com

Office Action Summary	Application No.	Applicant(s)	
	11/258,839	ORSINI ET AL.	
	Examiner	Art Unit	
	SAMSON LEMMA	2432	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *30 June 2011*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ An election was made by the applicant in response to a restriction requirement set forth during the interview on _____; the restriction requirement and election have been incorporated into this action.

4)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

5)☒ Claim(s) *1-36,41,43-54 and 69-85* is/are pending in the application.

5a) Of the above claim(s) _____ is/are withdrawn from consideration.

6)☒ Claim(s) *18-36, 41, 43-54* is/are allowed.

7)☐ Claim(s) _____ is/are rejected.

8)☒ Claim(s) *1-17 and 69-85* is/are objected to.

9)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

10)☐ The specification is objected to by the Examiner.

11)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

12)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

13)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. This office action is in response to an amendment filed on June 30, 2011. Claims 37-40, 42 and 55-68 are canceled. Thus claims 1-36, 41, 43-54 and 69-85 are pending of which claims 1, 18, 22, 26, 30, 31, 34, 35, 41, 47 and 69 are independent. Claims 1, 3 and 69 are amended.

2. The amendment made to independent claims 1 and 69 overcomes the 112 rejection set forth in the previous office action. Thus this particular 112 rejection is withdrawn.

3. The two terminal disclaimers filed on 06/30/2011 overcomes the double patent rejections set forth in the previous office action. Thus, both double patent rejections are withdrawn.

Priority

4. This application claims priority from provisional application 60/622,146 filed on 10/25/2004. Therefore, the effective filling date for the subject matter defined in the pending claims of this application is October 25, 2004.

Information Disclosure Statement

5. The information disclosure statements (IDS) submitted on 06/30/2011 has been considered. The submission is in compliance with the provisions of 37 CFR 1.97. Form PTO-1449 is signed and attached hereto.

Claim Objections

6. Independent claims 1 is objected to because of the following informalities: The amended independent claim 1 recites the following limitations, "streaming from the first location to the second location, as at least two streaming portions of data..." This underlined amended limitation should have been, "at least two streaming portions of data..." The limitation "as" makes the claim ambiguous and should be deleted. Appropriate correction is required.

7. Independent claims 69 is objected to because of the following informalities: The amended independent claim 69 recites the following limitations, "at least one communications path over which are streamed, as at least two streaming portions of data..." This underlined amended limitation should have been, "at least two streaming portions of data..." The limitation "as" makes the claim ambiguous and should be deleted. Appropriate correction is required.

8. Dependent claims 2-17 fully incorporating the deficiencies of the above independent claim 1 are likewise objected.

9. Dependent claims 70-85 fully incorporating the deficiencies of the above independent claim 69 are likewise objected.

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Samson B Lemma whose telephone number is 571-272-3806. The examiner can normally be reached on Monday-Friday (8:00 am---4: 30 pm).

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, BARRON JR GILBERTO can be reached on 571-272-3799. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

/SAMSON LEMMA/

Primary Examiner, Art Unit 2432

PTO/SB/22 (09-11)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless if displays a valid OMB control number.

PETITION FOR EXTENSION OF TIME UNDER 37 CFR 1.136(a)	Docket Number (Optional) 104093-0002-101
Application Number 11/258,839	Filed October 25, 2005
For SECURE DATA PARSER METHOD AND SYSTEM	
Art Unit 2432	Examiner S. B. Lemma

This is a request under the provisions of 37 CFR 1.136(a) to extend the period for filing a reply in the above identified application.

The requested extension and fee are as follows (check time period desired and enter the appropriate fee below):

		Fee	Small Entity Fee	
[x]	One month (37 CFR 1.17(a)(1))	$150	$75	$ 75.00
[ ]	Two months (37 CFR 1.17(a)(2))	$560	$280	$
[ ]	Three months (37 CFR 1.17(a)(3))	$1270	$635	$
[ ]	Four months (37 CFR 1.17(a)(4))	$1980	$990	$
[ ]	Five months (37 CFR 1.17(a)(5))	$2690	$1345	$

[x] Applicant claims small entity status. See 37 CFR 1.27.

[ ] A check in the amount of the fee is enclosed.

[ ] Payment by credit card. Form PTO-2038 is attached.

[x] The Director has already been authorized to charge fees in this application to a Deposit Account.

[x] The Director is hereby authorized to charge any fees which may be required, or credit any overpayment, to Deposit Account Number 06-1075 .

WARNING: Information on this form may become public. Credit card information should not be included on this form. Provide credit card information and authorization on PTO-2038.

I am the

[ ] applicant/inventor.

[ ] assignee of record of the entire interest. See 37 CFR 3.71.
Statement under 37 CFR 3.73(b) is enclosed. (Form PTO/SB/96).

[x] attorney or agent of record. Registration Number 54,130

[ ] attorney or agent under 37 CFR 1.34.
Registration number if acting under 37 CFR 1.34 _____

/Edward A. Gordon/	February 10, 2012
Signature	Date
Edward A. Gordon	(617) 951-7066
Typed or printed name	Telephone Number

NOTE: Signatures of all the inventors or assignees of record of the entire interest or their representative(s) are required. Submit multiple forms if more than one signature is required, see below.

[ ] Total of 1 forms are submitted.

Electronic Acknowledgement Receipt

EFS ID:	12055079
Application Number:	11258839
International Application Number:	
Confirmation Number:	2420
Title of Invention:	Secure data parser method and system
First Named Inventor/Applicant Name:	Rick L. Orsini
Customer Number:	1473
Filer:	Edward A. Gordon/Lily Jiang
Filer Authorized By:	Edward A. Gordon
Attorney Docket Number:	104093-0002-101
Receipt Date:	10-FEB-2012
Filing Date:	25-OCT-2005
Time Stamp:	21:08:38
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	yes
Payment Type	Deposit Account
Payment was successfully received in RAM	$75
RAM confirmation Number	6306
Deposit Account	061075
Authorized User	

The Director of the USPTO is hereby authorized to charge indicated fees and credit any overpayment as follows:

Charge any Additional Fees required under 37 C.F.R. Section 1.16 (National application filing, search, and examination fees)

Charge any Additional Fees required under 37 C.F.R. Section 1.17 (Patent application and reexamination processing fees)

Charge any Additional Fees required under 37 C.F.R. Section 1.19 (Document supply fees)

Charge any Additional Fees required under 37 C.F.R. Section 1.20 (Post Issuance fees)

Charge any Additional Fees required under 37 C.F.R. Section 1.21 (Miscellaneous fees and charges)

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Extension of Time	104093-0002-101_EOT.pdf	73684 18e673849890e8a5fde40f16c25cbdc39e4e78fc	no	1
Warnings:					
Information:					
2	Fee Worksheet (SB06)	fee-info.pdf	29722 3a26c11e1c6173ca865a71e42f4928bdb13bacaf	no	2
Warnings:					
Information:					
		Total Files Size (in bytes):	103406		

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111

If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.

National Stage of an International Application under 35 U.S.C. 371

If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office

If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

PTO/SB/22 (09-11)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless if displays a valid OMB control number.

PETITION FOR EXTENSION OF TIME UNDER 37 CFR 1.136(a)	Docket Number (Optional) 104093-0002-101
Application Number 11/258,839	Filed October 25, 2005
For SECURE DATA PARSER METHOD AND SYSTEM	
Art Unit 2432	Examiner S. B. Lemma

This is a request under the provisions of 37 CFR 1.136(a) to extend the period for filing a reply in the above identified application.

The requested extension and fee are as follows (check time period desired and enter the appropriate fee below):

		Fee	Small Entity Fee	
[x]	One month (37 CFR 1.17(a)(1))	$150	$75	$ 75.00
[ ]	Two months (37 CFR 1.17(a)(2))	$560	$280	$
[ ]	Three months (37 CFR 1.17(a)(3))	$1270	$635	$
[ ]	Four months (37 CFR 1.17(a)(4))	$1980	$990	$
[ ]	Five months (37 CFR 1.17(a)(5))	$2690	$1345	$

[x] Applicant claims small entity status. See 37 CFR 1.27.

[ ] A check in the amount of the fee is enclosed.

[ ] Payment by credit card. Form PTO-2038 is attached.

[x] The Director has already been authorized to charge fees in this application to a Deposit Account.

[x] The Director is hereby authorized to charge any fees which may be required, or credit any overpayment, to Deposit Account Number 06-1075 .

WARNING: Information on this form may become public. Credit card information should not be included on this form. Provide credit card information and authorization on PTO-2038.

I am the

[ ] applicant/inventor.

[ ] assignee of record of the entire interest. See 37 CFR 3.71.
Statement under 37 CFR 3.73(b) is enclosed. (Form PTO/SB/96).

[x] attorney or agent of record. Registration Number 54,130

[ ] attorney or agent under 37 CFR 1.34.
Registration number if acting under 37 CFR 1.34 _____

/Edward A. Gordon/ — Signature
February 10, 2012 — Date

Edward A. Gordon — Typed or printed name
(617) 951-7066 — Telephone Number

NOTE: Signatures of all the inventors or assignees of record of the entire interest or their representative(s) are required. Submit multiple forms if more than one signature is required, see below.

[ ] Total of 1 forms are submitted.

29534614_1

Electronic Patent Application Fee Transmittal

Application Number:	11258839
Filing Date:	25-Oct-2005
Title of Invention:	Secure data parser method and system
First Named Inventor/Applicant Name:	Rick L. Orsini
Filer:	Edward A. Gordon/Lily Jiang
Attorney Docket Number:	104093-0002-101

Filed as Small Entity

Utility under 35 USC 111(a) Filing Fees

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Basic Filing:				
Pages:				
Claims:				
Miscellaneous-Filing:				
Petition:				
Patent-Appeals-and-Interference:				
Post-Allowance-and-Post-Issuance:				
Extension-of-Time:				
Extension - 1 month with $0 paid	2251	1	75	75

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Miscellaneous:				
			Total in USD ($)	**75**

Electronic Acknowledgement Receipt

EFS ID:	12103992
Application Number:	11258839
International Application Number:	
Confirmation Number:	2420
Title of Invention:	Secure data parser method and system
First Named Inventor/Applicant Name:	Rick L. Orsini
Customer Number:	1473
Filer:	Edward A. Gordon/Lily Jiang
Filer Authorized By:	Edward A. Gordon
Attorney Docket Number:	104093-0002-101
Receipt Date:	17-FEB-2012
Filing Date:	25-OCT-2005
Time Stamp:	16:12:41
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1		104093-0002-101_Reply.pdf	92529 e3585f15482f7cee24580da5faaa4a8677e9b364	yes	12

	Multipart Description/PDF files in .zip description		
	Document Description	**Start**	**End**
	Amendment/Req. Reconsideration-After Non-Final Reject	1	1
	Claims	2	11
	Applicant Arguments/Remarks Made in an Amendment	12	12
Warnings:			
Information:			
	Total Files Size (in bytes):	92529	

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.

National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

REMARKS

Claims 1-36, 41, 43-54, and 69-85 were pending in the above-identified application. Applicants note with appreciation the indication of allowable claims 18-36, 41, and 43-54. In the Office Action, the Examiner objected to claims 1 and 69 as reciting the allegedly ambiguous limitation "as at least two streaming portions of data." Applicants respectfully traverse the objection in light of the amendment below.

Applicants wish to thank Examiner Lemma for the courtesies extended during the telephonic interview conducted on Thursday, February 2, 2012. As agreed upon during the interview, to improve clarity, claims 1 and 69 have been amended to recite, "streaming from the first location to the second location the at least two portions of data over at least one communications path, wherein the at least two portions of data stream separately from each other as at least two streaming portions of data."

For at least the forgoing reasons, Applicants respectfully submit that independent claims 1 and 69 are allowable. Claims 2-16 and 70-85 depend from one of independent claims 1 and 69 and add further limitations thereto. Applicants therefore request reconsideration and withdrawal of the objections for claims 1-16 and 69-85.

In view of the above amendment, Applicants believe the pending application is in condition for allowance.

Dated: February 17, 2012

Respectfully submitted,

By /Edward A. Gordon/
Edward A. Gordon
Registration No.: 54,130
ROPES & GRAY LLP
Prudential Tower
800 Boylston Street
Boston, Massachusetts 02199
(617) 951-7000
(617) 235-9492 (Fax)
Attorneys/Agents For Applicant

Docket No.: 104093-0002-101
(PATENT)

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Patent Application of:
Orsini et al.

Application No.: 11/258,839 | Confirmation No.: 2420

Filed: October 25, 2005 | Art Unit: 2432

For: SECURE DATA PARSER METHOD AND SYSTEM | Examiner: S. B. Lemma

AMENDMENT IN RESPONSE TO NON-FINAL OFFICE ACTION UNDER 37 C.F.R. 1.111

MS Amendment
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Madam:

INTRODUCTORY COMMENTS

In response to the Office Action dated October 17, 2011, please amend the above-identified U.S. patent application as follows. The deadline for response was extended to February 17, 2012 via the petition for extension of time filed on February 10, 2012.

Amendments to the Claims are reflected in the listing of claims which begins on page 2 of this paper.

Remarks/Arguments begin on page 12 of this paper.

PTO/SB/06 (07-06)
Approved for use through 1/31/2007. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

PATENT APPLICATION FEE DETERMINATION RECORD

Substitute for Form PTO-875

Application or Docket Number	Filing Date	☐ To be Mailed
11/258,839	10/25/2005	

APPLICATION AS FILED – PART I

FOR	(Column 1) NUMBER FILED	(Column 2) NUMBER EXTRA	SMALL ENTITY ☒ RATE ($)	SMALL ENTITY ☒ FEE ($)	OR	OTHER THAN SMALL ENTITY RATE ($)	OTHER THAN SMALL ENTITY FEE ($)
☐ BASIC FEE (37 CFR 1.16(a), (b), or (c))	N/A	N/A	N/A			N/A	
☐ SEARCH FEE (37 CFR 1.16(k), (i), or (m))	N/A	N/A	N/A			N/A	
☐ EXAMINATION FEE (37 CFR 1.16(o), (p), or (q))	N/A	N/A	N/A			N/A	
TOTAL CLAIMS (37 CFR 1.16(i))	minus 20 =	*	X $ =		OR	X $ =	
INDEPENDENT CLAIMS (37 CFR 1.16(h))	minus 3 =	*	X $ =			X $ =	
☐ APPLICATION SIZE FEE (37 CFR 1.16(s))	If the specification and drawings exceed 100 sheets of paper, the application size fee due is $250 ($125 for small entity) for each additional 50 sheets or fraction thereof. See 35 U.S.C. 41(a)(1)(G) and 37 CFR 1.16(s).						
☐ MULTIPLE DEPENDENT CLAIM PRESENT (37 CFR 1.16(j))							
* If the difference in column 1 is less than zero, enter "0" in column 2.			TOTAL			TOTAL	

APPLICATION AS AMENDED – PART II

AMENDMENT 02/17/2012	(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA	SMALL ENTITY RATE ($)	SMALL ENTITY ADDITIONAL FEE ($)	OR	OTHER THAN SMALL ENTITY RATE ($)	OTHER THAN SMALL ENTITY ADDITIONAL FEE ($)
Total (37 CFR 1.16(i))	* 67	Minus	** 85	= 0	X $30 =	0	OR	X $ =	
Independent (37 CFR 1.16(h))	* 11	Minus	***21	= 0	X $125 =	0	OR	X $ =	
☐ Application Size Fee (37 CFR 1.16(s))									
☐ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))							OR		
					TOTAL ADD'L FEE	0	OR	TOTAL ADD'L FEE	

AMENDMENT	(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA	RATE ($)	ADDITIONAL FEE ($)	OR	RATE ($)	ADDITIONAL FEE ($)
Total (37 CFR 1.16(i))	*	Minus	**	=	X $ =		OR	X $ =	
Independent (37 CFR 1.16(h))	*	Minus	***	=	X $ =		OR	X $ =	
☐ Application Size Fee (37 CFR 1.16(s))									
☐ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))							OR		
					TOTAL ADD'L FEE		OR	TOTAL ADD'L FEE	

* If the entry in column 1 is less than the entry in column 2, write "0" in column 3.

** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, enter "20".

*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, enter "3".

The "Highest Number Previously Paid For" (Total or Independent) is the highest number found in the appropriate box in column 1.

Legal Instrument Examiner:
/TERRY MALLOY/

This collection of information is required by 37 CFR 1.16. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 12 minutes to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

If you need assistance in completing the form, call 1-800-PTO-9199 and select option 2.

AMENDMENTS TO THE CLAIMS

The following listing of claims will replace all prior versions, and listings, of claims in the above-identified patent application.

Listing of Claims

1. (Currently Amended) A method for securing streaming of a data set from a first location to a second location, the method comprising:

generating, using processing circuitry at the first location, at least two portions of data from the data set, wherein each of the at least two portions of data respectively contains a substantially random distribution of a respective subset of the data set;

streaming from the first location to the second location~~, as at least two streaming portions of data,~~ the at least two portions of data over at least one communications path, wherein the at least two portions of data stream separately from each other as at least two streaming portions of data; and

restoring, using processing circuitry at the second location, data from at least a subset of the at least two streaming portions of data, wherein the data is restored as the at least two streaming portions of data are being received at the second location.

2. (Previously Presented) The method of claim 1 wherein the streaming comprises streaming the at least two portions of data on separate communications paths in parallel.

3. (Previously Presented) The method of claim 1 wherein the streaming comprises streaming the at least two portions of data on a single communications path in a serial transmission.

4. (Original) The method of claim 1 wherein the generating is accomplished using an email application.

5. (Original) The method of claim 1 wherein the generating is accomplished using a video broadcasting application.

6. (Previously Presented) The method of claim 1 wherein the streaming over the at least one communications path comprises streaming over a wireless communications path.

7. (Previously Presented) The method of claim 6, wherein the streaming over the wireless communications path comprises streaming over the wireless communications path using an 802.11 protocol.

8. (Previously Presented) The method of claim 6, wherein the streaming over the wireless communications path comprises streaming over the wireless communications path using Bluetooth.

9. (Previously Presented) The method of claim 6, wherein the streaming over the wireless communications path comprises streaming over a wireless LAN.

10. (Previously Presented) The method of claim 6, wherein the streaming over the wireless communications path comprises streaming over a cellular network.

11. (Previously Presented) The method of claim 6, wherein the streaming over the wireless communications path comprises streaming over a peer to peer wireless network.

12. (Previously Presented) The method of claim 6, wherein the streaming over the wireless communications path comprises streaming over a hub based wireless network.

13. (Previously Presented) The method of claim 6, wherein the streaming over the wireless communications path comprises streaming over a satellite transmission.

14. (Original) The method of claim 1 wherein at least one of the first location and the second location is a computer peripheral device.

15. (Original) The method of claim 1 wherein at least one of the first location and the second location is a PDA device.

16. (Original) The method of claim 1 wherein at least one of the first location and the second location is a storage device.

17. (Previously Presented) The method of claim 1 wherein the streaming over the at least one communications path comprises streaming over a wired communications path.

18. (Previously Presented) A method for securing a data set, the method comprising:

generating, using processing circuitry, at least two portions of data from the data set, wherein each of the at least two portions of data respectively contains a substantially random distribution of a respective subset of the data set;

storing the at least two portions of data on at least two data depositories, wherein the at least two data depositories are physically separate from each other; and

using at least one of the at least two portions of data stored on at least one of the at least two data depositories as an authentication key stored in an authentication depository in order to restore the data set.

19. (Original) The method of claim 18 wherein the authentication depository is a physical token device.

20. (Original) The method of claim 18 further comprising restoring the data set when the authentication depository is coupled to at least a subset of the at least two data depositories.

21. (Original) The method of claim 20 further comprising removing access to the data set when the authentication depository is no longer coupled to the at least a subset of the at least two data depositories.

22. (Previously Presented) A method for securing a data set, the method comprising:

generating, using processing circuitry, at least two portions of data from the data set, wherein each of the at least two portions of data respectively contains a substantially random distribution of a respective subset of the data set;

generating respective integrity information for each of the at least two portions of data;

writing the respective integrity information to each of the at least two portions of data;

storing each of the at least two portions of data separately; and

restoring the data set from at least a subset of the at least two portions of data when the respective integrity information of the at least a subset of the at least two portions of data has been verified.

23. (Original) The method of claim 22, wherein the integrity information is generated using a hash algorithm.

24. (Original) The method of claim 22, wherein the integrity information is generated using a MAC signature.

25. (Original) The method of claim 22, wherein the integrity information is generated using a digital signature.

26. (Previously Presented) A method for securing a data set, the method comprising:

generating, using processing circuitry, at least two portions of data from the data set, wherein each of the at least two portions of data respectively contains a substantially random distribution of a respective subset of the data set;

storing each of the at least two portions of data separately; and

restoring the data set from at least a subset of the at least two portions of data, wherein at least one particular portion of data is required in order to restore the data set.

27. (Original) The method of claim 26 further comprising encrypting at least one of the at least two portions of data, the at least one of the at least two portions of data comprising the at least one particular portion of data.

28. (Original) The method of claim 27 wherein the encrypting comprises encrypting at least one of the at least two portions of data according to a first encryption key and at least one other of the at two least portions of data according to a second encryption key.

29. (Original) The method of claim 27 wherein the encrypting comprises encrypting at least one of the at least two portion of data using a public key that may be decrypted using a private key.

30. (Previously Presented) A method for securing a data set, the method comprising:

generating, using processing circuitry, at least two portions of data from the data set, wherein each of the at least two portions of data respectively contains a substantially random distribution of a respective subset of the data set, and wherein the probability that data from the data set is distributed to at least one of the at least two portions of data is less than the probability that data from the data set is distributed to a remainder of the at least two portions of data;

storing each of the at least two portions of data separately; and

restoring the data set from at least a subset of the at least two portions of data.

31. (Previously Presented) A method for securing a data set, the method comprising:

generating, using processing circuitry, at least two portions of data from the data set, wherein each of the at least two portions of data respectively contains a substantially random distribution of a respective subset of the data set, and wherein at least one of the at least two portions of data has a predetermined size;

storing each of the at least two portions of data separately; and

restoring the data set from at least a subset of the at least two portions of data.

32. (Original) The method of claim 31 wherein the generating the at least two portions of data comprises using a splitting key having a splitting key size and wherein the predetermined size is the splitting key size.

33. (Original) The method of claim 31 further comprising encrypting the data set with an encryption key having an encryption key size prior to the generating the at least two portions of data and wherein the predetermined size is the encryption key size.

34. (Previously Presented) A method for securing a data set, the method comprising:

generating, using processing circuitry, at least two portions of data from the data set, wherein each of the at least two portions of data respectively contains a substantially random distribution of a respective subset of the data set, and wherein the probability that data from the data set is distributed to each of the at least two portions of data is substantially equal;

storing each of the at least two portions of data separately; and

restoring the data set from at least a subset of the at least two portions of data.

35. (Previously Presented) A method for securing a data set, the method comprising:

generating, using processing circuitry, according to a splitting key, at least two portions of data from the data set, wherein each of the at least two portions of data respectively contains a substantially random distribution of a respective subset of the data set;

generating, according to a Shamir algorithm, splitting key information that is capable of being used to generate the splitting key, wherein the splitting key information is distributed among the at least two portions of data;

storing each of the at least two portions of data separately; and

restoring the data set from at least a subset of the at least two portions of data.

36. (Original) The method of claim 35 wherein restoring the data set comprises restoring the splitting key from the splitting key information from at least a subset of the at least two portions of data.

37-40. (Cancelled)

41. (Previously Presented) A method for securing a data set, the method comprising:

determining a number of portions of data into which to distribute the data set;

generating a substantially random number;

determining, using processing circuitry, into which of the portions of data to distribute each unit of data from the data set based on corresponding data from the substantially random number and distributing accordingly;

storing the portions of data separately; and

restoring the data set from at least a subset of the portions of data, wherein the determining into which of the portions of data to distribute each unit of data from the data set comprises determining into which of the portions of data to distribute each unit of data from the data set based on a hash function of corresponding data from the substantially random number.

42. (Cancelled)

43. (Original) The method of claim 41 wherein the unit of data is a bit.

44. (Original) The method of claim 41 wherein the size of the substantially random number is a multiple, M, of the size of the data set and wherein the determining into which of the portions of data to distribute each unit of data from the data set comprises determining into which of the portions of data to distribute each unit of data from the data set based on a hash function of a corresponding M bits from the substantially random number.

45. (Original) The method of claim 44 wherein the multiple, M, is 8 and wherein the determining into which of the portions of data to distribute each unit of data from the data set based on a hash function of a corresponding M bits from the substantially random number comprises determining into which of the portions of data to distribute each bit of data from the data set based on a hash function of a corresponding byte from the substantially random number.

46. (Original) The method of claim 41 further comprising appending redundancy data to each of the portions of data.

47. (Previously Presented) A method for securing a data set, the method comprising:

determining a number of portions of data into which to distribute the data set;

generating a substantially random number;

for each unit of data of the data set, splitting the unit of data into a left segment and a right segment by determining, based on corresponding data from the substantially random number, a splitting location wherein bits to the left of the splitting location represent the left segment and bits to the right of the splitting location represent the right segment;

determining, using processing circuitry, into which of the portions of data to distribute each of the left segment and the right segment of each unit of data from the data set based on corresponding data from the substantially random number and distributing accordingly;

storing the portions of data separately; and

restoring the data set from at least a subset of the portions of data.

48. (Original) The method of claim 47 wherein the splitting the unit of data into the left segment and the right segment comprises splitting the unit of data into the left segment and the right segment based on a hash function of a corresponding data from the substantially random number.

49. (Original) The method of claim 47 wherein the unit of data is a byte.

50. (Original) The method of claim 47 wherein generating the substantially random number comprises generating the substantially random number of size equal to the size of the data set.

51. (Original) The method of claim 50 wherein the unit of data is a byte and wherein the splitting the unit of data into the left segment and the right segment comprises splitting the unit of data into the left segment and the right segment based on a hash function of a corresponding unit of data from the substantially random number.

52. (Original) The method of claim 47 wherein the determining into which of the portions of data to distribute each of the left segment and the right segment comprises determining into which of the portions of data to distribute each of the left segment and the right segment of each unit of data from the data set based on corresponding data from the substantially random number and based on a table containing an array of all possible distributions of the right segment and the left.

53. (Original) The method of 47 wherein the determining into which of the portions of data to distribute each of the left segment and the right segment comprises determining into which of the portions of data to distribute each of the left segment and the right segment of each unit of data from the data set based on a hash function of corresponding data from the substantially random number and based on a table containing an array of all possible distributions of the right segment and the left segment, wherein the hash function of the corresponding data from the substantially random number indicates which entry in the table to use, the entry indicating a destination portion for the left segment and a destination portion for the right segment.

54. (Original) The method of claim 53 wherein:

generating the substantially random number comprises generating the substantially random number of size equal to the size of the data set;

the unit of data is a byte; and

the corresponding data is a corresponding unit of data.

55-68. (Cancelled)

69. (Currently Amended) A system for securing streaming of a data set from a first location to a second location, the system comprising:

a first processor for generating at the first location at least two portions of data from the data set, wherein each of the at least two portions of data respectively contains a substantially random distribution of a respective subset of the data set;

at least one communications path over which are streamed~~, as at least two streaming portions of data,~~ the at least two portions of data from the first location to the second location, wherein the at least two portions of data stream separately from each other as at least two streaming portions of data; and

a second processor for restoring at the second location data from at least a subset of the at least two streaming portions of data, wherein the data is restored as the at least two streaming portions of data are being received at the second location.

70. (Previously Presented) The system of claim 69 wherein the at least two streaming portions of data are streamed on separate communications paths in parallel.

71. (Previously Presented) The system of claim 69 wherein the at least two streaming portions of data are streamed on a single communications path in a serial transmission.

72. (Original) The system of claim 69 wherein the first processor is used in conjunction with an email application.

73. (Original) The system of claim 69 wherein the first processor is used in conjunction with a video broadcasting application.

74. (Original) The system of claim 69 wherein the at least one communications path comprises a wireless communications path.

75. (Original) The system of claim 74, wherein the wireless communications path comprises an 802.11 protocol communications path.

76. (Original) The system of claim 74, wherein the wireless communications path comprises a Bluetooth communications path.

77. (Original) The system of claim 74, wherein the wireless communications path comprises a wireless LAN.

78. (Original) The system of claim 74, wherein the wireless communications path comprises a cellular network.

79. (Original) The system of claim 74, wherein the wireless communications path comprises a peer to peer wireless network.

80. (Original) The system of claim 74, wherein the wireless communications path comprises a hub based wireless network.

81. (Original) The system of claim 74, wherein wireless communications path comprises a satellite communications path.

82. (Original) The system of claim 69 wherein at least one of the first location and the second location is a computer peripheral device.

83. (Original) The system of claim 69 wherein at least one of the first location and the second location is a PDA device.

84. (Original) The system of claim 69 wherein at least one of the first location and the second location is a storage device.

85. (Original) The system of claim 69 wherein the at least one communications path comprises a wired communications path.

Electronic Patent Application Fee Transmittal

Application Number:	11258839
Filing Date:	25-Oct-2005
Title of Invention:	Secure data parser method and system
First Named Inventor/Applicant Name:	Rick L. Orsini
Filer:	Edward A. Gordon/Joanne Ryan
Attorney Docket Number:	104093-0002-101

Filed as Large Entity

Utility under 35 USC 111(a) Filing Fees

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Basic Filing:				
Pages:				
Claims:				
Miscellaneous-Filing:				
Petition:				
Patent-Appeals-and-Interference:				
Post-Allowance-and-Post-Issuance:				
Extension-of-Time:				

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Miscellaneous:				
Submission- Information Disclosure Stmt	1806	1	180	180
		Total in USD ($)		**180**

PATENT COOPERATION TREATY

DEC 19 2008

ROPES & GRAY LLP - IP DOCKETING
REFERRED TO
NOTED BY

From the INTERNATIONAL SEARCHING AUTHORITY

PCT

To:

ROPES & GRAY LLP
Attn. Ingerman, Jeffrey H.
1211 Avenue of the Americas
New York NY 10036
ETATS-UNIS D'AMERIQUE

NOTIFICATION OF TRANSMITTAL OF THE INTERNATIONAL SEARCH REPORT AND THE WRITTEN OPINION OF THE INTERNATIONAL SEARCHING AUTHORITY, OR THE DECLARATION

(PCT Rule 44.1)

	Date of mailing *(day/month/year)* 16/12/2008
Applicant's or agent's file reference SEC-5 PCT	**FOR FURTHER ACTION** See paragraphs 1 and 4 below
International application No. PCT/US2007/023626	International filing date *(day/month/year)* 07/11/2007
Applicant BELLARE, MIHIR	

1. [x] The applicant is hereby notified that the international search report and the written opinion of the International Searching Authority have been established and are transmitted herewith.

   Filing of amendments and statement under Article 19:
   The applicant is entitled, if he so wishes, to amend the claims of the International Application (see Rule 46):

   When? The time limit for filing such amendments is normally **two** months from the date of transmittal of the International Search Report.

   Where? Directly to the International Bureau of WIPO, 34 chemin des Colombettes 1211 Geneva 20, Switzerland, Fascimile No.: (41-22) 338.82.70

   For more detailed instructions, see the notes on the accompanying sheet.

2. [ ] The applicant is hereby notified that no international search report will be established and that the declaration under Article 17(2)(a) to that effect and the written opinion of the International Searching Authority are transmitted herewith.

3. [ ] **With regard to the protest** against payment of (an) additional fee(s) under Rule 40.2, the applicant is notified that:

   [ ] the protest together with the decision thereon has been transmitted to the International Bureau together with the applicant's request to forward the texts of both the protest and the decision thereon to the designated Offices.

   [ ] no decision has been made yet on the protest; the applicant will be notified as soon as a decision is made.

4. **Reminders**

   Shortly after the expiration of **18 months** from the priority date, the international application will be published by the International Bureau. If the applicant wishes to avoid or postpone publication, a notice of withdrawal of the international application, or of the priority claim, must reach the International Bureau as provided in Rules 90*bis*.1 and 90*bis*.3, respectively, before the completion of the technical preparations for international publication.

   The applicant may submit comments on an informal basis on the written opinion of the International Searching Authority to the International Bureau. The International Bureau will send a copy of such comments to all designated Offices unless an international preliminary examination report has been or is to be established. These comments would also be made available to the public but not before the expiration of 30 months from the priority date.

   Within **19 months** from the priority date, but only in respect of some designated Offices, a demand for international preliminary examination must be filed if the applicant wishes to postpone the entry into the national phase **until 30 months** from the priority date (in some Offices even later); otherwise, the applicant must, **within 20 months** from the priority date, perform the prescribed acts for entry into the national phase before those designated Offices.

   In respect of other designated Offices, the time limit of **30 months** (or later) will apply even if no demand is filed within 19 months.

   See the Annex to Form PCT/IB/301 and, for details about the applicable time limits, Office by Office, see the *PCT Applicant's Guide*, Volume II, National Chapters and the WIPO Internet site.

Name and mailing address of the International Searching Authority	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL-2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Lucía Toscano Oliveros

Form PCT/ISA/220 (October 2005) *(See notes on accompanying sheet)*

NOTES TO FORM PCT/ISA/220

These Notes are intended to give the basic instructions concerning the filing of amendments under article 19. The Notes are based on the requirements of the Patent Cooperation Treaty, the Regulations and the Administrative Instructions under that Treaty. In case of discrepancy between these Notes and those requirements, the latter are applicable. For more detailed information, see also the *PCT Applicant's Guide*, a publication of WIPO.

In these Notes, "Article", "Rule", and "Section" refer to the provisions of the PCT, the PCT Regulations and the PCT Administrative Instructions, respectively.

INSTRUCTIONS CONCERNING AMENDMENTS UNDER ARTICLE 19

The applicant has, after having received the international search report and the written opinion of the International Searching Authority, one opportunity to amend the claims of the international application. It should however be emphasized that, since all parts of the international application (claims,description and drawings) may be amended during the international preliminary examination procedure, there is usually no need to file amendments of the claims under Article 19 except where, e.g. the applicant wants the latter to be published for the purposes of provisional protection or has another reason for amending the claims before international publication. Furthermore, it should be emphasized that provisional protection is available in some States only (see *PCT Applicant's Guide*, Volume I/A, Annexes B1 and B2).

The attention of the applicant is drawn to the fact that amendments to the claims under Article 19 are not allowed where the International Searching Authority has declared, under Article 17(2), that no international search report would be established (see *PCT Applicant's Guide*, Volume I/A, paragraph 296).

What parts of the international application may be amended?

Under Article 19, only the claims may be amended.

During the international phase, the claims may also be amended (or further amended) under Article 34 before the International Preliminary Examining Authority. The description and drawings may only be amended under Article 34 before the International Examining Authority.

Upon entry into the national phase, all parts of the international application may be amended under Article 28 or, where applicable, Article 41.

When?

Within 2 months from the date of transmittal of the international search report or 16 months from the priority date, whichever time limit expires later. It should be noted, however, that the amendments will be considered as having been received on time if they are received by the International Bureau after the expiration of the applicable time limit but before the completion of the technical preparations for international publication (Rule 46.1).

Where not to file the amendments?

The amendments may only be filed with the International Bureau and not with the receiving Office or the International Searching Authority (Rule 46.2).

Where a demand for international preliminary examination has been/is filed, see below.

How?

Either by cancelling one or more entire claims, by adding one or more new claims or by amending the text of one or more of the claims as filed.

A replacement sheet must be submitted for each sheet of the claims which, on account of an amendment or amendments, differs from the sheet originally filed.

All the claims appearing on a replacement sheet must be numbered in Arabic numerals. Where a claim is cancelled, no renumbering of the other claims is required. In all cases where claims are renumbered, they must be renumbered consecutively (Section 205(b)).

The amendments must be made in the language in which the international application is to be published.

What documents must/may accompany the amendments?

Letter (Section 205(b)):

The amendments must be submitted with a letter.

The letter will not be published with the international application and the amended claims. It should not be confused with the "Statement under Article 19(1)" (see below, under "Statement under Article 19(1)").

The letter must be in English or French, at the choice of the applicant. However, if the language of the international application is English, the letter must be in English; if the language of the international application is French, the letter must be in French.

The letter must indicate the differences between the claims as filed and the claims as amended. It must, in particular, indicate, in connection with each claim appearing in the international application (it being understood that identical indications concerning several claims may be grouped),whether

(i) the claim is unchanged;

(ii) the claim is cancelled;

(iii) the claim is new;

(iv) the claim replaces one or more claims as filed;

(v) the claim is the result of the division of a claim as filed.

The following examples illustrate the manner in which amendments must be explained in the accompanying letter:

1. [Where originally there were 48 claims and after amendment of some claims there are 51]:
"Claims 1 to 29, 31, 32, 34, 35, 37 to 48 replaced by amended claims bearing the same numbers; claims 30, 33 and 36 unchanged; new claims 49 to 51 added."

2. [Where originally there were 15 claims and after amendment of all claims there are 11]:
"Claims 1 to 15 replaced by amended claims 1 to 11."

3. [Where originally there were 14 claims and the amendments consist in cancelling some claims and in adding new claims]:
"Claims 1 to 6 and 14 unchanged; claims 7 to 13 cancelled; new claims 15, 16 and 17 added." or
"Claims 7 to 13 cancelled; new claims 15, 16 and 17 added; all other claims unchanged."

4. [Where various kinds of amendments are made]:
"Claims 1–10 unchanged; claims 11 to 13, 18 and 19 cancelled; claims 14, 15 and 16 replaced by amended claim 14; claim 17 subdivided into amended claims 15, 16 and 17; new claims 20 and 21 added."

"Statement under article 19(1)" (Rule 46.4)

The amendments may be accompanied by a statement explaining the amendments and indicating any impact that such amendments might have on the description and the drawings (which cannot be amended under Article 19(1)).

The statement will be published with the international application and the amended claims.

It must be in the language in which the international application is to be published.

It must be brief, not exceeding 500 words if in English or if translated into English.

It should not be confused with and does not replace the letter indicating the differences between the claims as filed and as amended. It must be filed on a separate sheet and must be identified as such by a heading, preferably by using the words "Statement under Article 19(1)."

It may not contain any disparaging comments on the international search report or the relevance of citations contained in that report. Reference to citations, relevant to a given claim, contained in the international search report may be made only in connection with an amendment of that claim.

Consequence if a demand for international preliminary examination has already been filed

If, at the time of filing any amendments and any accompanying statement, under Article 19, a demand for international preliminary examination has already been submitted, the applicant must preferably, at the time of filing the amendments (and any statement) with the International Bureau, also file with the International Preliminary Examining Authority a copy of such amendments (and of any statement) and, where required, a translation of such amendments for the procedure before that Authority (see Rules 55.3(a) and 62.2, first sentence). For further information, see the Notes to the demand form (PCT/IPEA/401).

If a demand for international preliminary examination is made, the written opinion of the International Searching Authority will, except in certain cases where the International Preliminary Examining Authority did not act as International Searching Authority and where it has notified the International Bureau under Rule 66.1*bis*(b), be considered to be a written opinion of the International Preliminary Examining Authority. If a demand is made, the applicant may submit to the International Preliminary Examining Authority a reply to the written opinion together, where appropriate, with amendments before the expiration of 3 months from the date of mailing of Form PCT/ISA/220 or before the expiration of 22 months from the priority date, whichever expires later (Rule 43*bis*.1(c)).

Consequence with regard to translation of the international application for entry into the national phase

The applicant's attention is drawn to the fact that, upon entry into the national phase, a translation of the claims as amended under Article 19 may have to be furnished to the designated/elected Offices, instead of, or in addition to, the translation of the claims as filed.

For further details on the requirements of each designated/elected Office, see the *PCT Applicant's Guide,* Volume II.

PATENT COOPERATION TREATY

PCT

INTERNATIONAL SEARCH REPORT

(PCT Article 18 and Rules 43 and 44)

Applicant's or agent's file reference	**FOR FURTHER ACTION**	see Form PCT/ISA/220 as well as, where applicable, item 5 below.
SEC-5 PCT		
International application No.	International filing date *(day/month/year)*	(Earliest) Priority Date *(day/month/year)*
PCT/US2007/023626	07/11/2007	07/11/2006
Applicant		
BELLARE, MIHIR		

This international search report has been prepared by this International Searching Authority and is transmitted to the applicant according to Article 18. A copy is being transmitted to the International Bureau.

This international search report consists of a total of 3 sheets.

[X] It is also accompanied by a copy of each prior art document cited in this report.

1. **Basis of the report**
   a. With regard to the **language,** the international search was carried out on the basis of:
      [X] the international application in the language in which it was filed
      [ ] a translation of the international application into _____, which is the language of a translation furnished for the purposes of international search (Rules 12.3(a) and 23.1(b))
   b. [ ] This international search report has been established taking into account the **rectification of an obvious mistake** authorized by or notified to this Authority under Rule 91 (Rule 43.6*bis*(a)).
   c. [ ] With regard to any **nucleotide and/or amino acid sequence** disclosed in the international application, see Box No. I.

2. [ ] **Certain claims were found unsearchable** (See Box No. II)

3. [ ] **Unity of invention is lacking** (see Box No III)

4. With regard to the **title,**
   [X] the text is approved as submitted by the applicant
   [ ] the text has been established by this Authority to read as follows:

5. With regard to the **abstract,**
   [X] the text is approved as submitted by the applicant
   [ ] the text has been established, according to Rule 38.2(b), by this Authority as it appears in Box No. IV. The applicant may, within one month from the date of mailing of this international search report, submit comments to this Authority

6. With regard to the **drawings,**
   a. the figure of the **drawings** to be published with the abstract is Figure No. 7
      [X] as suggested by the applicant
      [ ] as selected by this Authority, because the applicant failed to suggest a figure
      [ ] as selected by this Authority, because this figure better characterizes the invention
   b. [ ] none of the figures is to be published with the abstract

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2007/023626

A. CLASSIFICATION OF SUBJECT MATTER
INV. H04L9/08

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
H04L G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

EPO-Internal, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	HUGO KRAWCZYK: "Secret Sharing Made Short" IBM T.J. WATSON RESEARCH CENTER, [Online] 1998, XP002505270 USA Retrieved from the Internet: URL:http://www.cs.cornell.edu/courses/cs754/2001fa/secretshort.pdf> [retrieved on 2008-11-24] the whole document ----- -/--	1-20

[X] Further documents are listed in the continuation of Box C. [ ] See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier document but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
1 December 2008	16/12/2008

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL – 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Apostolescu, Radu

4

Form PCT/ISA/210 (second sheet) (April 2005)

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	RABIN M O: "EFFICIENT DISPERSAL OF INFORMATION FOR SECURITY, LOAD BALANCING, AND FAULT TOLERANCE" JOURNAL OF THE ACM, ACM, NEW YORK, NY, US, vol. 36, no. 2, 1 April 1989 (1989-04-01), pages 335-348, XP000570108 ISSN: 0004-5411 the whole document	1-20
A	CRESCENZO DI G ET AL: "NON-INTERACTIVE AND NON-MALLEABLE COMMITMENT" PROCEEDINGS OF THE 30TH ANNUAL ACM SYMPOSIUM ON THEORY OF COMPUTING. DALLAS, TX, MAY 23 - 26, 1998; [PROCEEDINGS OF THE ANNUAL ACM SYMPOSIUM ON THEORY OF COMPUTING], NEW YORK, NY : ACM, US, 23 May 1998 (1998-05-23), pages 141-150, XP000970902 ISBN: 978-0-89791-962-3 the whole document	1-20
A	SHAMIR A: "How to Share a Secret" COMMUNICATIONS OF THE ASSOCIATION FOR COMPUTING MACHINERY, ACM, NEW YORK, NY, US, vol. 22, no. 11, 1 November 1979 (1979-11-01), pages 1-4, XP002241399 ISSN: 0001-0782 the whole document	1-20
A	CACHIN: "On-line Secret Sharing" CRYPTOGRAPHY AND CODING. IMA CONFERENCE. PROCEEDINGS, XX, XX, 18 December 1995 (1995-12-18), pages 190-198, XP002137681 the whole document	1-20
A	MAYER A ET AL: "Generalized secret sharing and group-key distribution using short keys" COMPRESSION AND COMPLEXITY OF SEQUENCES 1997. PROCEEDINGS SALERNO, ITALY 11-13 JUNE 1997, LOS ALAMITOS, CA, USA,IEEE COMPUT. SOC, US, 11 June 1997 (1997-06-11), pages 30-44, XP010274905 ISBN: 978-0-8186-8132-5 the whole document	1-20

4

PATENT COOPERATION TREATY

From the
INTERNATIONAL SEARCHING AUTHORITY

To:

see form PCT/ISA/220

PCT

WRITTEN OPINION OF THE INTERNATIONAL SEARCHING AUTHORITY

(PCT Rule 43*bis*.1)

Date of mailing *(day/month/year)*	see form PCT/ISA/210 (second sheet)
Applicant's or agent's file reference	**FOR FURTHER ACTION** See paragraph 2 below

International application No.	International filing date *(day/month/year)*	Priority date *(day/month/year)*
PCT/US2007/023626	07.11.2007	07.11.2006

International Patent Classification (IPC) or both national classification and IPC
INV. H04L9/08

Applicant
BELLARE, MIHIR

1. This opinion contains indications relating to the following items:

- ☒ Box No. I Basis of the opinion
- ☒ Box No. II Priority
- ☐ Box No. III Non-establishment of opinion with regard to novelty, inventive step and industrial applicability
- ☐ Box No. IV Lack of unity of invention
- ☒ Box No. V Reasoned statement under Rule 43*bis*.1(a)(i) with regard to novelty, inventive step or industrial applicability; citations and explanations supporting such statement
- ☐ Box No. VI Certain documents cited
- ☐ Box No. VII Certain defects in the international application
- ☐ Box No. VIII Certain observations on the international application

2. **FURTHER ACTION**

If a demand for international preliminary examination is made, this opinion will usually be considered to be a written opinion of the International Preliminary Examining Authority ("IPEA") except that this does not apply where the applicant chooses an Authority other than this one to be the IPEA and the chosen IPEA has notifed the International Bureau under Rule 66.1*bis*(b) that written opinions of this International Searching Authority will not be so considered.

If this opinion is, as provided above, considered to be a written opinion of the IPEA, the applicant is invited to submit to the IPEA a written reply together, where appropriate, with amendments, before the expiration of 3 months from the date of mailing of Form PCT/ISA/220 or before the expiration of 22 months from the priority date, whichever expires later.

For further options, see Form PCT/ISA/220.

3. For further details, see notes to Form PCT/ISA/220.

Name and mailing address of the ISA:	Date of completion of this opinion	Authorized Officer
European Patent Office D-80298 Munich Tel. +49 89 2399 - 0 Tx: 523656 epmu d Fax: +49 89 2399 - 4465	see form PCT/ISA/210	Apostolescu, Radu Telephone No. +49 89 2399-7950

Form PCT/ISA/237 (Cover Sheet) (April 2005)

Box No. I Basis of the opinion

1. With regard to the **language**, this opinion has been established on the basis of:

   ☒ the international application in the language in which it was filed

   ☐ a translation of the international application into , which is the language of a translation furnished for the purposes of international search (Rules 12.3(a) and 23.1 (b)).

2. ☐ This opinion has been established taking into account the **rectification of an obvious mistake** authorized by or notified to this Authority under Rule 91 (Rule 43bis.1(a))

3. With regard to any **nucleotide and/or amino acid sequence** disclosed in the international application and necessary to the claimed invention, this opinion has been established on the basis of:

   a. type of material:

      ☐ a sequence listing

      ☐ table(s) related to the sequence listing

   b. format of material:

      ☐ on paper

      ☐ in electronic form

   c. time of filing/furnishing:

      ☐ contained in the international application as filed.

      ☐ filed together with the international application in electronic form.

      ☐ furnished subsequently to this Authority for the purposes of search.

4. ☐ In addition, in the case that more than one version or copy of a sequence listing and/or table relating thereto has been filed or furnished, the required statements that the information in the subsequent or additional copies is identical to that in the application as filed or does not go beyond the application as filed, as appropriate, were furnished.

5. Additional comments:

Box No. II Priority

1. ☒ The validity of the priority claim has not been considered because the International Searching Authority does not have in its possession a copy of the earlier application whose priority has been claimed or, where required, a translation of that earlier application. This opinion has nevertheless been established on the assumption that the relevant date (Rules 43*bis*.1 and 64.1) is the claimed priority date.

2. ☐ This opinion has been established as if no priority had been claimed due to the fact that the priority claim has been found invalid (Rules 43*bis*.1 and 64.1). Thus for the purposes of this opinion, the international filing date indicated above is considered to be the relevant date.

3. Additional observations, if necessary:

Box No. V Reasoned statement under Rule 43*bis*.1(a)(i) with regard to novelty, inventive step or industrial applicability; citations and explanations supporting such statement

1. Statement

Novelty (N)	Yes:	Claims	1-20
	No:	Claims	
Inventive step (IS)	Yes:	Claims	1-20
	No:	Claims	
Industrial applicability (IA)	Yes:	Claims	1-20
	No:	Claims	

2. Citations and explanations

see separate sheet

Re Item V

Reasoned statement with regard to novelty, inventive step or industrial applicability; citations and explanations supporting such statement

Reference is made to the following documents:

D1: HUGO KRAWCZYK: "Secret Sharing Made Short" IBM T.J. WATSON RESEARCH CENTER, [Online] 1998, XP002505270 USA Retrieved from the Internet: URL:http://www.cs.cornell.edu/courses/cs75 4/2001fa/secretshort.pdf> [retrieved on 2008-11-24]

D2: RABIN M O: "EFFICIENT DISPERSAL OF INFORMATION FOR SECURITY, LOAD BALANCING, AND FAULT TOLERANCE" JOURNAL OF THE ACM, ACM, NEW YORK, NY, US, vol. 36, no. 2, 1 April 1989 (1989-04-01), pages 335-348, XP000570108 ISSN: 0004-5411

D3: CRESCENZO DI G ET AL: "NON-INTERACTIVE AND NON-MALLEABLE COMMITMENT" PROCEEDINGS OF THE 30TH ANNUAL ACM SYMPOSIUM ON THEORY OF COMPUTING. DALLAS, TX, MAY 23 - 26, 1998; [PROCEEDINGS OF THE ANNUAL ACM SYMPOSIUM ON THEORY OF COMPUTING], NEW YORK, NY : ACM, US, 23 May 1998 (1998-05-23), pages 141-150, XP000970902 ISBN: 978-0-89791-962-3

D4: SHAMIR A: "How to Share a Secret" COMMUNICATIONS OF THE ASSOCIATION FOR COMPUTING MACHINERY, ACM, NEW YORK, NY, US, vol. 22, no. 11, 1 November 1979 (1979-11-01), pages 1-4, XP002241399 ISSN: 0001-0782

D5: CACHIN: "On-line Secret Sharing" CRYPTOGRAPHY AND CODING. IMA CONFERENCE. PROCEEDINGS, XX, XX, 18 December 1995 (1995-12-18), pages 190-198, XP002137681

D6: MAYER A ET AL: "Generalized secret sharing and group-key distribution using short keys" COMPRESSION AND COMPLEXITY OF SEQUENCES 1997. PROCEEDINGS SALERNO, ITALY 11-13 JUNE 1997, LOS ALAMITOS, CA, USA,IEEE COMPUT. SOC, US, 11 June 1997 (1997-06-11), pages 30-44, XP010274905 ISBN: 978-0-8186-8132-5

1. The document D1 is regarded as being the closest prior art to the subject-matter of claim 1, and describes, in order to achieve a space efficient secret sharing scheme, a combination of an information dispersal scheme with a secure encryption scheme and a perfect secret sharing scheme. The information dispersal scheme is intended for the distribution of a piece of information among n active processors, in such a way that the recovery of the information is possible in the presence of m active processors (i.e. out of m fragments), where m and n are parameters satisfying 1<=m<=n. The scheme assumes that active processors behave honestly, i.e. returned fragments are unmodified. The basic idea is to add to the information, say a file F, some amount of redundancy and then to partition it into n fragments, each transmitted to one of the parties. Reconstruction of F is possible out of m (legitimate) fragments. Further the author of D1 assumes:
- a generic information dispersal algorithm (IDA) which works for parameters n (number of file fragments) and m (number of required fragments to reconstruct the file);
- a secure private key encryption function (ENC); and
- a perfect (n,m)-secret sharing scheme (PSS).
An (n,m)-secret sharing scheme consists of a distribution and a reconstruction process. The distribution process gets as input the secret S (and the values of n and m) and generates n shares S_1, S_2, ..., S_n, which are privately delivered to the system participants. The reconstruction process reconstructs the correct secret when input with any subset of m shares. Given a particular secret sharing algorithm A, the n resultant shares S_1, S_2, ..., S_n shares are denoted by A(S), whereas A(S) is a random variable depending on the internal random coins of A.

The difference between independent claim 1 of the present invention and the prior art resides mainly in generating a collection of fragments from the data to be secured, each fragment in the collection being produced by combining: 1) a share from a collection of shares produced from a CSS (computational secret sharing) scheme applied to the data and 2) committal and decommittal values computed from a random value and the collection of shares, each fragment being stored on at least one data repository.
Hence the subject-matter of claim 1 is new (Article 33(2) PCT).

The objective problem solved by the above-identified differences is to improve the securing of data from unauthorized access or use.

The independent claim 1 solve the technical problem in a way (producing fragments of data by combining a share with committal and decommittal values obtained from random value and collection of shares, each fragment being stored on at least one data repository), which is not disclosed or suggested in the cited prior art (D1-D6), and therefore it is considered to be inventive in accordance with Article 33(3) PCT.

It is considered that also independent claims 10 and 17 relate to new and inventive subject-matter (Articles 33 (2) and (3) PCT), since the cited prior art (D1-D6) does not disclose or suggest the specifically claimed steps of producing data fragments comprising a share, committal and decommittal values, and of storing each fragment on at least one data repository.

It is considered that also independent claim 20 relates to new and inventive subject-matter (Articles 33 (2) and (3) PCT), since the cited prior art (D1-D6) does not disclose or suggest the specifically claimed features of the method claim 20.

Claims 2-9, 11-16, 18 and 19 are dependent on claim 1, 10 and respectively 17 and as such also meet the requirements of the PCT with respect to novelty and inventive step.

2. Claims 1, 10, 17 and 20 have been drafted as separate independent method claims. These claims lack conciseness and as such do not meet the requirements of Article 6 PCT.
Moreover it is not clear which are the corresponding special technical features that link said claims, such that the requirements of Rule 13.1 PCT are fulfilled.

3. The features in claim 20 of separating each purported share into a content portion and a vector of checksums, and using the vector of checksums associated with the designated purported share to classify each share of the collection of purported shares as an actual share or as a bogus share, is not referred to in the description. Claim 20 is therefore not supported by the description as required by Article 6 PCT.
Moreover the requirement of Article 5 PCT is not fulfilled.

4. The phrase "which is hereby incorporated by reference herein" in par. 0001, 0040 and 0088 should be deleted as the application should be self-contained; such referenced documents are not regarded as part of the disclosure unless they contain matter essential to the invention, in which case the subject-matter in question would have to be incorporated into the description. This however is not the case here (see Guidelines 4.26 PCT)

Reviewed & Docketed

PATENT COOPERATION TREATY

From the INTERNATIONAL SEARCHING AUTHORITY

PCT

To:
Ingerman, Jeffrey H.
ROPES & GRAY LLP
1211 Avenue of the Americas
New York NY 10036
ETATS-UNIS D'AMERIQUE

RECEIVED
DEC 16 2010
ROPES & GRAY LLP - IP DOCKETING
RECEIVED IN NY
FILED IN NY

NOTIFICATION OF TRANSMITTAL OF THE INTERNATIONAL SEARCH REPORT AND THE WRITTEN OPINION OF THE INTERNATIONAL SEARCHING AUTHORITY, OR THE DECLARATION

(PCT Rule 44.1)

Applicant's or agent's file reference 1040930012W1	Date of mailing *(day/month/year)* 14 December 2010 (14-12-2010) **FOR FURTHER ACTION** See paragraphs 1 and 4 below
International application No. PCT/US2010/035377	International filing date *(day/month/year)* 19 May 2010 (19-05-2010)
Applicant SECURITY FIRST CORP.	

1. [X] The applicant is hereby notified that the international search report and the written opinion of the International Searching Authority have been established and are transmitted herewith.

   Filing of amendments and statement under Article 19:
   The applicant is entitled, if he so wishes, to amend the claims of the International Application (see Rule 46):

   When? The time limit for filing such amendments is normally two months from the date of transmittal of the International Search Report.

   Where? Directly to the International Bureau of WIPO, 34 chemin des Colombettes 1211 Geneva 20, Switzerland, Fascimile No.: (41-22) 338.82.70

   For more detailed instructions, see *PCT Applicant's Guide*, International Phase, paragraphs 9.004 - 9.011.

2. [ ] The applicant is hereby notified that no international search report will be established and that the declaration under Article 17(2)(a) to that effect and the written opinion of the International Searching Authority are transmitted herewith.

3. [ ] **With regard to any protest** against payment of (an) additional fee(s) under Rule 40.2, the applicant is notified that:

   [ ] the protest together with the decision thereon has been transmitted to the International Bureau together with the applicant's request to forward the texts of both the protest and the decision thereon to the designated Offices.

   [ ] no decision has been made yet on the protest; the applicant will be notified as soon as a decision is made.

4. **Reminders**

   The applicant may submit comments on an informal basis on the written opinion of the International Searching Authority to the International Bureau. The International Bureau will send a copy of such comments to all designated Offices unless an international preliminary examination report has been or is to be established. Following the expiration of 30 months from the priority date, these comments will also be made available to the public.

   Shortly after the expiration of **18 months** from the priority date, the international application will be published by the International Bureau. If the applicant wishes to avoid or postpone publication, a notice of withdrawal of the international application, or of the priority claim, must reach the International Bureau before completion of the technical preparations for international publication (Rules 90*bis*.1 and 90*bis*.3).

   Within **19 months** from the priority date, but only in respect of some designated Offices, a demand for international preliminary examination must be filed if the applicant wishes to postpone the entry into the national phase **until 30 months** from the priority date (in some Offices even later); otherwise, the applicant must, **within 20 months** from the priority date, perform the prescribed acts for entry into the national phase before those designated Offices.

   In respect of other designated Offices, the time limit of **30 months** (or later) will apply even if no demand is filed within 19 months.

   For details about the applicable time limits, Office by Office, see www.wipo.int/pct/en/texts/time_limits.html and the *PCT Applicant's Guide*, National Chapters.

Name and mailing address of the International Searching Authority	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL-2280 HV Rijswijk Tel. (+31-70) 340-2040 Fax: (+31-70) 340-3016	SCHIEßL, Wolfgang-Peter Tel: +49 (0)89 2399-2860

Form PCT/ISA/220 (July 2010)

PATENT COOPERATION TREATY

PCT

INTERNATIONAL SEARCH REPORT

(PCT Article 18 and Rules 43 and 44)

Applicant's or agent's file reference	**FOR FURTHER ACTION**	see Form PCT/ISA/220 as well as, where applicable, item 5 below.
1040930012W1		
International application No.	International filing date *(day/month/year)*	(Earliest) Priority Date *(day/month/year)*
PCT/US2010/035377	19/05/2010	19/05/2009
Applicant		
SECURITY FIRST CORP.		

This international search report has been prepared by this International Searching Authority and is transmitted to the applicant according to Article 18. A copy is being transmitted to the International Bureau.

This international search report consists of a total of 5 sheets.

☐ It is also accompanied by a copy of each prior art document cited in this report.

1. **Basis of the report**

a. With regard to the **language,** the international search was carried out on the basis of:

☒ the international application in the language in which it was filed

☐ a translation of the international application into _____, which is the language of a translation furnished for the purposes of international search (Rules 12.3(a) and 23.1(b))

b. ☐ This international search report has been established taking into account the **rectification of an obvious mistake** authorized by or notified to this Authority under Rule 91 (Rule 43.6*bis*(a)).

c. ☐ With regard to any **nucleotide and/or amino acid sequence** disclosed in the international application, see Box No. I.

2. ☐ **Certain claims were found unsearchable** (See Box No. II)

3. ☒ **Unity of invention is lacking** (see Box No III)

4. With regard to the **title,**

☒ the text is approved as submitted by the applicant

☐ the text has been established by this Authority to read as follows:

5. With regard to the **abstract,**

☒ the text is approved as submitted by the applicant

☐ the text has been established, according to Rule 38.2(b), by this Authority as it appears in Box No. IV. The applicant may, within one month from the date of mailing of this international search report, submit comments to this Authority

6. With regard to the **drawings,**

a. the figure of the **drawings** to be published with the abstract is Figure No. 42

☒ as suggested by the applicant

☐ as selected by this Authority, because the applicant failed to suggest a figure

☐ as selected by this Authority, because this figure better characterizes the invention

b. ☐ none of the figures is to be published with the abstract

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2010/035377

A. CLASSIFICATION OF SUBJECT MATTER
INV. H04L29/06 H04L29/08

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)
EPO-Internal, COMPENDEX, INSPEC, PAJ, IBM-TDB, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2006/177061 A1 (ORSINI RICK L [US] ET AL) 10 August 2006 (2006-08-10) abstract paragraph [0118] - paragraph [0128] figures 7, 8	1-10, 14-23
X	US 2007/160198 A1 (ORSINI RICK [US] ET AL) 12 July 2007 (2007-07-12) abstract paragraph [0124] - paragraph [0134] figures 7, 8	1-10, 14-23

☐ Further documents are listed in the continuation of Box C.

☒ See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
22 September 2010	14/12/2010

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL – 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Horn, Marc-Philipp

3

Form PCT/ISA/210 (second sheet) (April 2005)

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2010/035377

Box No. II Observations where certain claims were found unsearchable (Continuation of item 2 of first sheet)

This international search report has not been established in respect of certain claims under Article 17(2)(a) for the following reasons:

1. ☐ Claims Nos.:
because they relate to subject matter not required to be searched by this Authority, namely:

2. ☐ Claims Nos.:
because they relate to parts of the international application that do not comply with the prescribed requirements to such an extent that no meaningful international search can be carried out, specifically:

3. ☐ Claims Nos.:
because they are dependent claims and are not drafted in accordance with the second and third sentences of Rule 6.4(a).

Box No. III Observations where unity of invention is lacking (Continuation of item 3 of first sheet)

This International Searching Authority found multiple inventions in this international application, as follows:

see additional sheet

1. ☐ As all required additional search fees were timely paid by the applicant, this international search report covers all searchable claims.

2. ☐ As all searchable claims could be searched without effort justifying an additional fees, this Authority did not invite payment of additional fees.

3. ☐ As only some of the required additional search fees were timely paid by the applicant, this international search report covers only those claims for which fees were paid, specifically claims Nos.:

4. ☒ No required additional search fees were timely paid by the applicant. Consequently, this international search report is restricted to the invention first mentioned in the claims; it is covered by claims Nos.:

1-10, 14-23

Remark on Protest

☐ The additional search fees were accompanied by the applicant's protest and, where applicable, the payment of a protest fee.

☐ The additional search fees were accompanied by the applicant's protest but the applicable protest fee was not paid within the time limit specified in the invitation.

☐ No protest accompanied the payment of additional search fees.

FURTHER INFORMATION CONTINUED FROM PCT/ISA/ 210

This International Searching Authority found multiple (groups of) inventions in this international application, as follows:

1. claims: 1-10, 14-23

   Splitting data and storing the data on a pluraltiy of remote storage devices in a cloud computing storage network

   ---

2. claims: 11-13, 24-26

   Sending split data from a first location to a second location over a plurality of orthogonal frequency-division multiplexing channels.

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/US2010/035377

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2006177061 A1	10-08-2006	NONE	
US 2007160198 A1	12-07-2007	AU 2006350252 A1	08-05-2008
		CA 2629015 A1	08-05-2008
		CN 101401341 A	01-04-2009
		EP 1952575 A2	06-08-2008
		WO 2008054406 A2	08-05-2008

Form PCT/ISA/210 (patent family annex) (April 2005)

PATENT COOPERATION TREATY

From the
INTERNATIONAL SEARCHING AUTHORITY

To:

see form PCT/ISA/220

PCT

WRITTEN OPINION OF THE INTERNATIONAL SEARCHING AUTHORITY

(PCT Rule 43*bis*.1)

Date of mailing *(day/month/year)* see form PCT/ISA/210 (second sheet)	
Applicant's or agent's file reference see form PCT/ISA/220	**FOR FURTHER ACTION** See paragraph 2 below

International application No.	International filing date *(day/month/year)*	Priority date *(day/month/year)*
PCT/US2010/035377	19.05.2010	19.05.2009

International Patent Classification (IPC) or both national classification and IPC
INV. H04L29/06 H04L29/08

Applicant
SECURITY FIRST CORP.

1. This opinion contains indications relating to the following items:

- ☒ Box No. I Basis of the opinion
- ☐ Box No. II Priority
- ☒ Box No. III Non-establishment of opinion with regard to novelty, inventive step and industrial applicability
- ☒ Box No. IV Lack of unity of invention
- ☒ Box No. V Reasoned statement under Rule 43*bis*.1(a)(i) with regard to novelty, inventive step and industrial applicability; citations and explanations supporting such statement
- ☐ Box No. VI Certain documents cited
- ☐ Box No. VII Certain defects in the international application
- ☒ Box No. VIII Certain observations on the international application

2. **FURTHER ACTION**

If a demand for international preliminary examination is made, this opinion will usually be considered to be a written opinion of the International Preliminary Examining Authority ("IPEA") except that this does not apply where the applicant chooses an Authority other than this one to be the IPEA and the chosen IPEA has notifed the International Bureau under Rule 66.1*bis*(b) that written opinions of this International Searching Authority will not be so considered.

If this opinion is, as provided above, considered to be a written opinion of the IPEA, the applicant is invited to submit to the IPEA a written reply together, where appropriate, with amendments, before the expiration of 3 months from the date of mailing of Form PCT/ISA/220 or before the expiration of 22 months from the priority date, whichever expires later.

For further options, see Form PCT/ISA/220.

3. For further details, see notes to Form PCT/ISA/220.

Name and mailing address of the ISA:	Date of completion of this opinion	Authorized Officer
European Patent Office D-80298 Munich Tel. +49 89 2399 - 0 Fax: +49 89 2399 - 4465	see form PCT/ISA/210	Horn, Marc-Philipp Telephone No. +49 89 2399-7579

Form PCT/ISA/237 (Cover Sheet) (July 2009)

Box No. I Basis of the opinion

1. With regard to the **language**, this opinion has been established on the basis of:

   ☒ the international application in the language in which it was filed

   ☐ a translation of the international application into , which is the language of a translation furnished for the purposes of international search (Rules 12.3(a) and 23.1 (b)).

2. ☐ This opinion has been established taking into account the **rectification of an obvious mistake** authorized by or notified to this Authority under Rule 91 (Rule 43bis.1(a))

3. With regard to any **nucleotide and/or amino acid sequence** disclosed in the international application, this opinion has been established on the basis of a sequence listing filed or furnished:

   a. (means)

      ☐ on paper

      ☐ in electronic form

   b. (time)

      ☐ in the international application as filed

      ☐ together with the international application in electronic form

      ☐ subsequently to this Authority for the purposes of search

4. ☐ In addition, in the case that more than one version or copy of a sequence listing has been filed or furnished, the required statements that the information in the subsequent or additional copies is identical to that in the application as filed or does not go beyond the application as filed, as appropriate, were furnished.

5. Additional comments:

Box No. III Non-establishment of opinion with regard to novelty, inventive step and industrial applicability

The questions whether the claimed invention appears to be novel, to involve an inventive step (to be non obvious), or to be industrially applicable have not been examined in respect of

☐ the entire international application

☒ claims Nos. 11-13, 24-26

because:

☐ the said international application, or the said claims Nos. relate to the following subject matter which does not require an international search *(specify)*:

☐ the description, claims or drawings *(indicate particular elements below)* or said claims Nos. are so unclear that no meaningful opinion could be formed *(specify)*:

☐ the claims, or said claims Nos. are so inadequately supported by the description that no meaningful opinion could be formed *(specify)*:

☒ no international search report has been established for the whole application or for said claims Nos. 11-13, 24-26

☐ a meaningful opinion could not be formed without the sequence listing; the applicant did not, within the prescribed time limit:

- ☐ furnish a sequence listing on paper complying with the standard provided for in Annex C of the Administrative Instructions, and such listing was not available to the International Searching Authority in a form and manner acceptable to it.
- ☐ furnish a sequence listing in electronic form complying with the standard provided for in Annex C of the Administrative Instructions, and such listing was not available to the International Searching Authority in a form and manner acceptable to it.
- ☐ pay the required late furnishing fee for the furnishing of a sequence listing in response to an invitation under Rules 13*ter*.1(a) or (b).

☐ See Supplemental Box for further details

Box No. IV Lack of unity of invention

1. ☒ In response to the invitation (Form PCT/ISA/206) to pay additional fees, the applicant has, within the applicable time limit:

   ☐ paid additional fees

   ☐ paid additional fees under protest and, where applicable, the protest fee

   ☐ paid additional fees under protest but the applicable protest fee was not paid

   ☒ not paid additional fees

2. ☐ This Authority found that the requirement of unity of invention is not complied with and chose not to invite the applicant to pay additional fees.

3. This Authority considers that the requirement of unity of invention in accordance with Rule 13.1, 13.2 and 13.3 is

   ☐ complied with

   ☒ not complied with for the following reasons:

   see separate sheet

4. Consequently, this report has been established in respect of the following parts of the international application:

   ☐ all parts.

   ☒ the parts relating to claims Nos. 1-10, 14-23

Box No. V Reasoned statement under Rule 43*bis*.1(a)(i) with regard to novelty, inventive step or industrial applicability; citations and explanations supporting such statement

1. Statement

Novelty (N)	Yes:	Claims	
	No:	Claims	1-10, 14-23
Inventive step (IS)	Yes:	Claims	
	No:	Claims	1-10, 14-23
Industrial applicability (IA)	Yes:	Claims	1-10, 14-23
	No:	Claims	

2. Citations and explanations

   see separate sheet

Box No. VIII Certain observations on the international application

The following observations on the clarity of the claims, description, and drawings or on the question whether the claims are fully supported by the description, are made:

see separate sheet

Re Item IV

Lack of unity of invention

The international application does not fulfill the requirement of Rule 13 PCT with respect to unity à priori.

There are two distinct groups of subject-matter:

I) Claims 1-10 and 14-23

II) Claims 11-13 and 24-26

Both groups have the following technical features in common:

Generating at a first system/location at least two portions of data from a data set. This feature is generally known in the field of data processing.

The technical problem of the subject-matter defined by the claims of group I is to securely store data. The problem is solved by the special technical features of communicating the at least two portions of data from the first system over a communications network to the cloud computing storage network, wherein the cloud computing storage network includes at least two respectively remote storage devices; and storing the at least two portions of data within the cloud computing storage network on the at least two networked storage devices, whereby the data set is restorable from at least a subset of the at least two portions of data by recombining data from the subset of the at least two portions of data.

The technical problem of the subject-matter defined by the claims of group II is to securely transmit data from a first location to a second location. The problem is solved by the special technical features of communicating from the first location to a second location the at least two portions of data over at least two OFDM channels, wherein different portions of data are communicated over different OFDM channels, whereby the data set is restorable at the second location from at least a subset of the at least two portions of data by recombining data from the subset of the at least two portions of data received over the at least two OFDM channels.

These special technical features are neither the same nor corresponding.

Re Item V

Reasoned statement under Rule 66.2(a)(ii) with regard to novelty, inventive step or industrial applicability; citations and explanations supporting such statement

1 Reference is made to the following documents:

D1 US 2006/177061 A1 (ORSINI RICK L [US] ET AL) 10 August 2006 (2006-08-10)

D2 US 2007/160198 A1 (ORSINI RICK [US] ET AL) 12 July 2007 (2007-07-12)

2 Claims 1 and 14 do not fulfill the requirements of Article 33(2) PCT because their subject-matter is not new.

2.1 Document D1 discloses, in terms of claim 1, a method for securing data using a cloud computing storage network (paragraphs 120-123, figures 7, 8), the method comprising:

generating at a first system at least two portions of data from a data set (paragraph 122);

communicating the at least two portions of data from the first system over a communications network to the cloud computing storage network, wherein the cloud computing storage network includes at least two respectively remote storage devices (paragraph 123; figure 8, step 835); and

storing the at least two portions of data within the cloud computing storage network on the at least two networked storage devices, whereby the data set is restorable from at least a subset of the at least two portions of data by recombining data from the subset of the at least two portions of data (paragraph 123; figure 8, step 835).

2.2 The same opinion and reasoning applies mutatis mutandis to independent claim 14.

2.3 The same opinion and reasoning can also be formulated when taking document D2 as evidence.

3 As far as the dependent claims 2 to 10 and 15 to 23 are clear, their subject-matter is known from documents D1 and D2 (Article 33(2) PCT).

Re Item VIII

Certain observations on the international application

1 The subject-matter of claims 2 and 15 is not clear (Article 6 PCT) because the expression "generating the at least two portions of data and communicating the at least two portions of data to the cloud computing network is substantially transparent to the first system" leaves the reader in doubt as to its meaning. In particular, the question arises how a system can generate and communicate data in a manner "transparent" (meaning "unnoticed") to itself.

2 The subject-matter of claims 6, 7, 19 and 20 is not clear (Article 6 PCT) because it defines a result to be achieved. It is doubted that the skilled reader is readily aware of how to "generate a virtual machine image" which is capable to "access the data set within the cloud computing storage network".

3 The subject-matter of claims 9, 10, 22 and 23 is not clear (Article 6 PCT) because the skilled reader is left in doubt to what the term "resilient" refers. For example, the reader is left in doubt if this term refers to data security (e.g. protection against network attacks) or a data safety (e.g. protection against fire, flood or earth quakes).

Possible steps after receipt of the international search report (ISR) and written opinion of the International Searching Authority (WO-ISA)

General information	For all international applications filed on or after 01/01/2004 the competent ISA will establish an ISR. It is accompanied by the WO-ISA. Unlike the former written opinion of the IPEA (Rule 66.2 PCT), the WO-ISA is not meant to be responded to, but to be taken into consideration for further procedural steps. This document explains about the possibilities.
Amending claims under Art. 19 PCT	Within 2 months after the date of mailing of the ISR and the WO-ISA the applicant may file amended claims under Art. 19 PCT directly with the International Bureau of WIPO. The PCT reform of 2004 did not change this procedure. For further information please see Rule 46 PCT as well as form PCT/ISA/220 and the corresponding Notes to form PCT/ISA/220.
Filing a demand for international preliminary examination	In principle, the WO-ISA will be considered as the written opinion of the IPEA. This should, in many cases, make it unnecessary to file a demand for international preliminary examination. If the applicant nevertheless wishes to file a demand this must be done before expiry of 3 months after the date of mailing of the ISR/ WO-ISA or 22 months after priority date, whichever expires later (Rule 54bis PCT). Amendments under Art. 34 PCT can be filed with the IPEA as before, normally at the same time as filing the demand (Rule 66.1 (b) PCT). If a demand for international preliminary examination is filed and no comments/amendments have been received the WO-ISA will be transformed by the IPEA into an IPRP (International Preliminary Report on Patentability) which would merely reflect the content of the WO-ISA. The demand can still be withdrawn (Art. 37 PCT).
Filing informal comments	After receipt of the ISR/WO-ISA the applicant may file informal comments on the WO-ISA directly with the International Bureau of WIPO. These will be communicated to the designated Offices together with the IPRP (International Preliminary Report on Patentability) at 30 months from the priority date. Please also refer to the next box.
End of the international phase	At the end of the international phase the International Bureau of WIPO will transform the WO-ISA or, if a demand was filed, the written opinion of the IPEA into the IPRP, which will then be transmitted together with possible informal comments to the designated Offices. The IPRP replaces the former IPER (international preliminary examination report).
Relevant PCT Rules and more information	Rule 43 PCT, Rule 43bis PCT, Rule 44 PCT, Rule 44bis PCT, PCT Newsletter 12/2003, OJ 11/2003, OJ 12/2003

BNSDOCID: <XS_____200704010CK_I_>

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
27 August 2009 (27.08.2009)

PCT

(10) International Publication Number
WO 2009/105280 A2

(51) International Patent Classification:
H04L 29/06 (2006.01)

(21) International Application Number:
PCT/US2009/001158

(22) International Filing Date:
23 February 2009 (23.02.2009)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
61/066,699 22 February 2008 (22.02.2008) US
61/147,961 28 January 2009 (28.01.2009) US

(71) Applicant *(for all designated States except US)*: **SECURITY FIRST CORPORATION** [US/US]; 22362 Gilberto #130, Rancho Santa Margarita, CA 92688 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **BONO, Stephen, C.** [US/US]; 1020 Park Avenue, Apt. 207, Baltimore, MD 21201 (US). **GREEN, Matthew, D.** [US/US]; 1017 W. 38th Street, Baltimore, MD 21211 (US). **LANDAU, Gabriel, D.** [US/US]; 6623 Bonnie Ridge Drive, Apt. 201, Baltimore, MD 21209 (US). **ORSINI, Rick, L.** [US/US]; 2100 Kings Forest Lane, Flower Mound, TX 75028 (US). **O'HARE, Mark, S.** [US/US]; 8 Kennedy Court, Coto De Caza, CA 92679 (US). **DAVENPORT, Roger** [US/US]; 3204 Cr 4119, Campbell, TX 75422 (US).

(74) Agents: **INGERMAN, Jeffrey, H.** et al.; Ropes & Gray LLP, 1211 Avenue Of The Americas, New York, NY 10036-8704 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *without international search report and to be republished upon receipt of that report (Rule 48.2(g))*

(54) Title: SYSTEMS AND METHODS FOR SECURE WORKGROUP MANAGEMENT AND COMMUNICATION

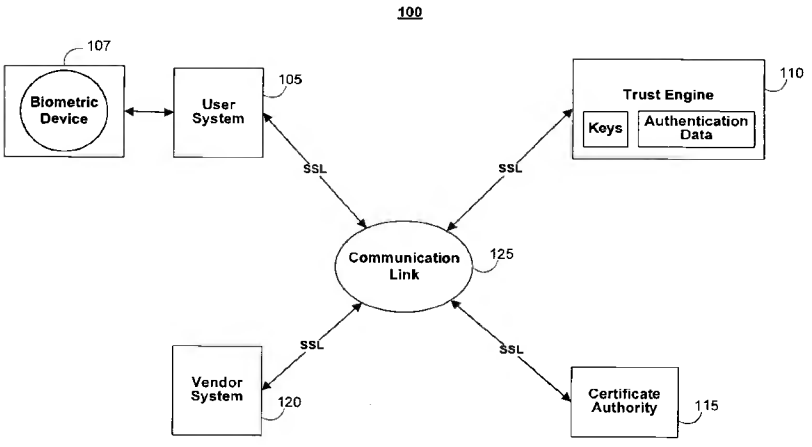


FIG. 1

(57) Abstract: A secure data parser is provided that may be integrated into any suitable system for securely storing and communicating data. The secure data parser may split or share a data set into multiple portions that are stored or communicated distinctly. Encryption of the original data, the portions of data, or both may be employed for additional security. The secure data parser may be used to protect data in motion by splitting an original data set into portions of data that may be communicated using one or more communications paths. Secure workgroup communication is supported through the secure distribution and management of a workgroup key for use with the secure data parser.

WO 2009/105280 A2

SYSTEMS AND METHODS FOR SECURE WORKGROUP MANAGEMENT AND COMMUNICATION

Cross-Reference to Related Applications

[0001] This claims the benefit of United States Provisional Patent Application Nos. 61/066,699, entitled "Systems and Methods for Secure Workgroup Management" filed February 22, 2008, and 61/147,961, entitled "Workgroup Key Management and Distribution" filed January 28, 2009, both of which are hereby incorporated by reference herein in their entireties.

Field of the Invention

[0002] The present invention relates in general to improved systems and methods for securing data in motion (*i.e.*, data being transferred from one location to another) within a workgroup. The systems and methods described herein may be used in conjunction with other systems and methods described in commonly-owned U.S. Patent No. 7,391,865 and commonly-owned U.S. Patent Application Nos. 10/458,928, filed June 11, 2003, 11/258,839, filed October 25, 2005, 11/602,667, filed November 20, 2006, 11/983,355, filed November 7, 2007, 11/999,575, filed December 5, 2007, 12/209,703, filed September 12, 2008, and 12/349,897, filed January 7, 2009, all of which are hereby incorporated by reference herein in their entireties.

Background of the Invention

[0003] In today's society, individuals and businesses conduct an ever-increasing amount of activities on and over computer systems. These computer systems, including proprietary and non-proprietary computer networks, are often storing,

archiving, and transmitting all types of sensitive information. Thus, an ever-increasing need exists for ensuring data stored and transmitted over these systems cannot be read or otherwise compromised.

[0004] One common solution for securing computer systems is to provide login and password functionality. However, password management has proven to be quite costly with a large percentage of help desk calls relating to password issues. Moreover, passwords provide little security in that they are generally stored in a file susceptible to inappropriate access, through, for example, brute-force attacks.

[0005] Another solution for securing computer systems is to provide cryptographic infrastructures. Cryptography, in general, refers to protecting data by transforming, or encrypting, it into an unreadable format. Only those who possess the key(s) to the encryption can decrypt the data into a useable format. Cryptography may be used to identify users, *e.g.*, authentication, to allow access privileges, *e.g.*, authorization, to create digital certificates and signatures, and the like. One popular cryptography system is a public key system that uses two keys, a public key known to everyone and a private key known only to the individual or business owner thereof. Generally, the data encrypted with one key is decrypted with the other and neither key is recreatable from the other.

[0006] Unfortunately, even the foregoing typical public-key cryptographic systems are still highly reliant on the user for security. For example, cryptographic systems issue the private key to the user, for example, through the user's browser. Unsophisticated users then generally store the private key on a hard drive accessible to others through an open computer system, such as, for example, the Internet. On the other hand, users may choose poor names for files containing their private key, such as, for example, "key." The result of the foregoing and other acts is to allow the key or keys to be susceptible to compromise.

[0007] Traditional solutions for securing data in motion also fail to provide robust and secure workgroup support. For example, the members of a workgroup may wish to communicate securely (*e.g.*, in a private and authenticated manner) with other workgroup members over one or more secure channels. Typical secure workgroup communication protocols based on pre-shared workgroup encryption keys, public key broadcast encryption, and the like often become unstable and

insecure because of insecure and inefficient management and distribution of the shared workgroup key. In addition, these secure workgroup communication protocols generally include only limited workgroup client revocation support. For example, the communication privileges of any one workgroup member generally cannot be efficiently revoked, for example, after compromise of the member's identity or shared workgroup key, or after the workgroup member leaves a workgroup. Moreover, with traditional approaches, there is generally no way to quickly and efficiently renew a workgroup security mechanism either on-demand, periodically, or automatically after the communication privileges of any one workgroup member has been revoked (or the workgroup member leaves the workgroup).

Summary of the Invention

[0008] Based on the foregoing, a need exists to provide a more robust and secure workgroup security scheme for use with data in motion solutions.

[0009] In some embodiments, a workgroup management module is provided to manage workgroup encryption keys. The module may be integrated with third-party user authentication, user logon, or directory access services, such as Microsoft's® Active Directory (AD) directory service. The directory service may enforce security policies for users (and groups of users), as well as manage and verify workgroup membership and track workgroup and user activities.

[0010] The workgroup management module may be integrated with any of the secure data parsers described in the above-referenced U.S. patent and patent applications in order to secure data in motion between any two or more end points (*e.g.*, clients or services) belonging to the same workgroup. In some embodiments, users are authenticated through the verification of a shared secret. For example, Digest Authentication may be used to transform a user's ASCII password into a cryptographic key. The client device may then request a challenge value, which is encrypted using the password-based key. The resulting ciphertext may then be transmitted to the authentication server. In response to receiving the ciphertext, the authentication server may perform the same encryption on the challenge value (using a stored copy of the user's password-based key) and verify that the received

ciphertext and the result of the server's encryption are a match. If there is a match, the client may be authenticated. Otherwise, the authentication request may be denied.

[0011] After a user is authenticated, further requests for services and communications on the network may be handled by a Kerberos-based subsystem of the authentication server. Users or client devices may be supplied a Ticket Granting Ticket (TGT) that serves as a token to be given to a Key Distribution Center (KDC) whenever a secure service or secure communication with another workgroup member is desired. In some embodiments, the TGT may include a block or string of data indicative of one or more of the identity of the recipient, a timestamp, and any other credentials. The TGT may be encrypted under the KDC's Master Key so that only the KDC may read its contents.

[0012] To communication securely with a service (*e.g.*, email or web server), a client may pass the TGT to the KDC and request a Service Ticket (ST). If the TGT is valid, the KDC may return two STs to the client, each containing the credentials of the parties involved in the communication, a timestamp, and a unique and fresh Session Key (SK) to be used as a temporary shared key between both parties for encryption. One of the STs may be encrypted using the shared password-based key between the user and the KDC, and the other ST may be encrypted using the shared password-based key between the service being accessed and the KDC. The ST encrypted for the client making the request may immediately be decrypted and the session key uncovered. The user may then send the second ST and a message encrypted using the SK to the targeted service. The service may then decrypt the ST with its password-based key to obtain the SK, and subsequently decrypt the message sent by the user. Through this process, the client and the service may now share the SK.

[0013] The SK may then be used as a Workgroup Key (WK) for use with the secure data parser. The WK may be stored locally to a client's KeyStore. The WK may also be used to wrap internally created session keys, including the Encryption Session Key (ESK) and the MAC Session Key (MK). After being wrapped with the WK, the ciphertexts may be securely shared among each header generated by the secure data parser, where a single share of each wrapped key is stored with

each header. Accessing the ESK or MK now may require obtaining the WK and some minimum quorum of shares sufficient to restore the wrapped ESK and MK ciphertexts.

[0014] In some embodiments, the WK may be used to wrap an internally generated ESK. Once wrapped with the WK, the ciphertext may be again wrapped with a freshly generated Key Encrypting Key (KEK), resulting in an ESK that has been doubly wrapped and that can only be recovered with knowledge of both the WK and the KEK. The KEK may be securely shared among each header generated by the secure data parser, and the doubly wrapped ESK may be stored within each header. Recovering the ESK may now require obtaining WK and some minimum quorum of shares sufficient to rebuild the KEK.

[0015] In some embodiments, online key updates are supported. Each time a client wishes to update its key (or check the availability of an update), the client may interact with an online key server. This online key server may then distribute and share key material according to a workgroup security policy.

[0016] In some embodiments, online key updates may be made available on-demand, periodically on some schedule defined by the workgroup security policy, or automatically after some event (*e.g.*, after a client's communication privileges are revoked or a client leaves a workgroup). The key update system may be managed by a Workgroup Authority (WA), which may control workgroup membership for multiple client devices and users. Each client or user in the system may be initially provisioned with a public/private keypair. Public keys may be registered with the WA, which may also record or store the public keys. Alternatively, in some embodiments, a symmetric encryption scheme may be used where each client or user in the system is initially provisioned with a secret key instead of a public/private keypair. Secret keys may be registered with the WA, which may also record or store the secret keys. The WA (or some other workgroup key server) may provide each device with the key server's public key certificate, which has been signed by a trusted Certificate Authority (CA).

[0017] To initiate an outgoing connection or receive an incoming connection, each client may contact its WA, which may, in some embodiments, interface with a third-party authentication or user logon service (*e.g.*, an Active Directory server)

to determine the client's workgroup membership. The client may then proves its identity to the WA using public-key or secret key techniques. The current WK may then be encrypted using a public-key or secret key encryption scheme such that a receiving client can decrypt the resulting ciphertext to obtain the current WK. The resulting ciphertext may also be signed by the key server (using the signing key corresponding to its certified public key) and transmitted to the requesting client.

[0018] In some embodiments, key updates may be supported even when the key server cannot be reached by a client (*i.e.*, the client is untethered or partially untethered). Renewability may still be achieved by incorporating a mechanism to periodically update the shared workgroup key. On a periodic schedule, key updates may be initiated by the key server, which may interface with a third-party authentication or user logon service to determine group membership. Each key update may contain a fresh WK and its identifier. The WK may be encrypted using a public-key broadcast encryption scheme such that only the current workgroup members can decrypt the resulting ciphertext. The resulting ciphertext may then be signed by the key server (using the signing key corresponding to its certified public key) and transmitted to the non-revoked devices via a semi-trusted channel (*e.g.*, a website). For example, when a client is prepared to go offline, the key server can pre-provision the client with key updates for a limited number of future time periods. Additionally or alternatively, for clients with limited network access, key revocation and update messages can be posted to a highly-available location, *e.g.*, a TLS-secured public website.

[0019] To join a workgroup, in some embodiments,
a client is first registered with the WA, and the client is then provisioned with a secret key. The WA may store the client's identity and secret key in a secure location (*e.g.*, in an encrypted lookup table or keystore).

[0020] To initiate connections, in some embodiments, a client may connect to the WA using its secret key and request the WK. The WA may look up the client and use the stored key to decrypt the message. It may then responds with the WK encrypted with the client's secret key. The client may now use the WK to initiate a connection to another client in the workgroup. In some embodiments, the WA also

indicates to the client an approximate duration of validity for the WK. Security policies in force on the client may require the client to contact the WA after key expiration for a key update (but the client may also check for key updates before key expiration if desired). When receiving a connection, security policies in force on the client may require a fresh and unexpired WK, verifying the status of the WK by the WA if necessary.

[0021] To revoke the communication privileges of a single client within a workgroup, the WA may refuse further key updates to the revoked client. A compromised client, therefore, may only have communication access to the workgroup during the period of time that the current WK is valid. Revocation may only affect new connections, or both new connections and existing connections (by terminating stale connections or forcing reconnection when a connection exceeds some system-defined length of time or number of connections).

[0022] To initiate a connection with another workgroup member, in some embodiments, the initiator may send the headers generated by the secure data parser to a recipient workgroup member. The headers may include one or more of the workgroup key, a workgroup key identifier, and a workgroup key version number. The recipient workgroup member may then check the workgroup key identifier to verify that the identifier is the correct identifier used for the current workgroup. The recipient workgroup member may also check the workgroup key version number to verify that the version number is the same as the recipient workgroup member's current workgroup key version number (and optionally if the current time is greater than the workgroup key refresh alarm time). If the workgroup key identifier is valid and the workgroup key version numbers are identical (and optionally if the current time is not greater than the workgroup key refresh alarm time), then the recipient workgroup member may report a successful connection attempt.

Brief Description of the Drawings

[0023] The present invention is described in more detail below in connection with the attached drawings, which are meant to illustrate and not to limit the invention, and in which:

[0024] FIGURE 1 illustrates a block diagram of a cryptographic system, according to aspects of an embodiment of the invention;

[0025] FIGURE 2 illustrates a block diagram of the trust engine of FIGURE 1, according to aspects of an embodiment of the invention;

[0026] FIGURE 3 illustrates a block diagram of the transaction engine of FIGURE 2, according to aspects of an embodiment of the invention;

[0027] FIGURE 4 illustrates a block diagram of the depository of FIGURE 2, according to aspects of an embodiment of the invention;

[0028] FIGURE 5 illustrates a block diagram of the authentication engine of FIGURE 2, according to aspects of an embodiment of the invention;

[0029] FIGURE 6 illustrates a block diagram of the cryptographic engine of FIGURE 2, according to aspects of an embodiment of the invention;

[0030] FIGURE 7 is a block diagram of an illustrative data in motion system in accordance with one embodiment of the present invention;

[0031] FIGURE 8 is a block diagram of another illustrative data in motion system in accordance with one embodiment of the present invention;

[0032] FIGURES 9A and 9B are simplified and illustrative process flow diagrams for header generation and data splitting for data in motion, that may be used in any suitable combination, with any suitable additions, deletions, or modifications in accordance with one embodiment of the present invention;

[0033] FIGURE 10 is a simplified block diagram of an illustrative share format, that may be used in any suitable combination, with any suitable additions, deletions, or modifications in accordance with one embodiment of the present invention;

[0034] FIGURE 11 is a simplified protocol illustrating the message sequence for authenticating a client in a workgroup in accordance with one embodiment of the present invention;

[0035] FIGURE 12 is a simplified protocol illustrating the message sequence for obtaining service tickets in a workgroup in accordance with one embodiment of the present invention;

[0036] FIGURE 13 is a simplified protocol illustrating the message sequence for using a workgroup key to secure data in motion in accordance with one embodiment of the present invention;

[0037] FIGURE 14 is a simplified protocol illustrating the message sequence for kernel and user space interaction in accordance with one embodiment of the present invention;

[0038] FIGURE 15 is a simplified block diagram of the initial setup of Workgroup Authorities within a domain in accordance with one embodiment of the present invention;

[0039] FIGURE 16 is a simplified protocol illustrating the connection sequence for two clients within a workgroup in accordance with one embodiment of the present invention;

[0040] FIGURE 17 is an illustrative process flow diagram for joining a workgroup in accordance with one embodiment of the present invention;

[0041] FIGURES 18A, 18B, and 18C are illustrative process flow diagrams for initiating a connection between two workgroup clients in accordance with one embodiment of the present invention;

[0042] FIGURE 19 is an illustrative process flow diagram for revoking a client's communication privileges within a workgroup in accordance with one embodiment of the present invention; and

[0043] FIGURE 20 is an illustrative process flow diagram for updating a workgroup key in accordance with one embodiment of the present invention.

Detailed Description

[0044] The present invention provides a cryptographic system where one or more secure servers, or a trust engine, stores cryptographic keys and user authentication data. Users may access the functionality of conventional cryptographic systems through network access to the trust engine, however, the trust engine does not release actual keys and other authentication data and therefore, the keys and data remain secure. This server-centric storage of keys and authentication data provides for user-independent security, portability, availability, and straightforwardness.

[0045] Because users can be confident in, or trust, the cryptographic system to perform user and document authentication and other cryptographic functions, a wide variety of functionality may be incorporated into the system. For example, the trust engine provider can ensure against agreement repudiation by, for example, authenticating the agreement participants, digitally signing the agreement on behalf of or for the participants, and storing a record of the agreement digitally signed by each participant. In addition, the cryptographic system may monitor agreements and determine to apply varying degrees of authentication, based on, for example, price, user, vendor, geographic location, place of use, or the like.

[0046] To facilitate a complete understanding of the invention, the remainder of the detailed description describes the invention with reference to the figures, wherein like elements are referenced with like numerals throughout.

[0047] FIGURE 1 illustrates a block diagram of a cryptographic system 100, according to aspects of an embodiment of the invention. As shown in FIGURE 1, the cryptographic system 100 includes a user system 105, a trust engine 110, a certificate authority 115, and a vendor system 120, communicating through a communication link 125.

[0048] According to one embodiment of the invention, the user system 105 comprises a conventional general-purpose computer having one or more microprocessors, such as, for example, an Intel-based processor. Moreover, the user system 105 includes an appropriate operating system, such as, for example, an operating system capable of including graphics or windows, such as Windows, Unix, Linux, or the like. As shown in FIGURE 1, the user system 105 may include a biometric device 107. The biometric device 107 may advantageously capture a user's biometric and transfer the captured biometric to the trust engine 110. According to one embodiment of the invention, the biometric device may advantageously comprise a device having attributes and features similar to those disclosed in U.S. Patent 6,856,383, entitled *"RELIEF OBJECT IMAGE GENERATOR,"* U.S. Patent Application No. 09/558,634, filed on April 26, 2000, entitled *"IMAGING DEVICE FOR A RELIEF OBJECT AND SYSTEM AND METHOD OF USING THE IMAGE DEVICE,"* U.S. Patent No. 6,631,201, entitled *"RELIEF OBJECT SENSOR ADAPTOR,"* and U.S. Patent Application

No. 09/477,943, filed on January 5, 2000, entitled *"PLANAR OPTICAL IMAGE SENSOR AND SYSTEM FOR GENERATING AN ELECTRONIC IMAGE OF A RELIEF OBJECT FOR FINGERPRINT READING,"* all of which are owned by the instant assignee, and all of which are hereby incorporated by reference herein.

[0049] In addition, the user system 105 may connect to the communication link 125 through a conventional service provider, such as, for example, a dial up, digital subscriber line (DSL), cable modem, fiber connection, or the like. According to another embodiment, the user system 105 connects the communication link 125 through network connectivity such as, for example, a local or wide area network. According to one embodiment, the operating system includes a TCP/IP stack that handles all incoming and outgoing message traffic passed over the communication link 125.

[0050] Although the user system 105 is disclosed with reference to the foregoing embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize from the disclosure herein, a wide number of alternatives embodiments of the user system 105, including almost any computing device capable of sending or receiving information from another computer system. For example, the user system 105 may include, but is not limited to, a computer workstation, an interactive television, an interactive kiosk, a personal mobile computing device, such as a digital assistant, mobile or cellular telephone, laptop, or the like, a wireless communications device, a smartcard, an embedded computing device, or the like, which can interact with the communication link 125. In such alternative systems, the operating systems will likely differ and be adapted for the particular device. However, according to one embodiment, the operating systems advantageously continue to provide the appropriate communications protocols needed to establish communication with the communication link 125.

[0051] FIGURE 1 illustrates the trust engine 110. According to one embodiment, the trust engine 110 comprises one or more secure servers for accessing and storing sensitive information, which may be any type or form of data, such as, but not limited to text, audio, video, user authentication data and public and private cryptographic keys. According to one embodiment, the authentication data includes data designed to uniquely identify a user of the

cryptographic system 100. For example, the authentication data may include a user identification number, one or more biometrics, and a series of questions and answers generated by the trust engine 110 or the user, but answered initially by the user at enrollment. The foregoing questions may include demographic data, such as place of birth, address, anniversary, or the like, personal data, such as mother's maiden name, favorite ice cream, or the like, or other data designed to uniquely identify the user. The trust engine 110 compares a user's authentication data associated with a current transaction, to the authentication data provided at an earlier time, such as, for example, during enrollment. The trust engine 110 may advantageously require the user to produce the authentication data at the time of each transaction, or, the trust engine 110 may advantageously allow the user to periodically produce authentication data, such as at the beginning of a string of transactions or the logging onto a particular vendor website.

[0052] According to the embodiment where the user produces biometric data, the user provides a physical characteristic, such as, but not limited to, facial scan, hand scan, ear scan, iris scan, retinal scan, vascular pattern, DNA, a fingerprint, writing or speech, to the biometric device 107. The biometric device advantageously produces an electronic pattern, or biometric, of the physical characteristic. The electronic pattern is transferred through the user system 105 to the trust engine 110 for either enrollment or authentication purposes.

[0053] Once the user produces the appropriate authentication data and the trust engine 110 determines a positive match between that authentication data (current authentication data) and the authentication data provided at the time of enrollment (enrollment authentication data), the trust engine 110 provides the user with complete cryptographic functionality. For example, the properly authenticated user may advantageously employ the trust engine 110 to perform hashing, digitally signing, encrypting and decrypting (often together referred to only as encrypting), creating or distributing digital certificates, and the like. However, the private cryptographic keys used in the cryptographic functions will not be available outside the trust engine 110, thereby ensuring the integrity of the cryptographic keys.

[0054] According to one embodiment, the trust engine 110 generates and stores cryptographic keys. According to another embodiment, at least one cryptographic key is associated with each user. Moreover, when the cryptographic keys include public-key technology, each private key associated with a user is generated within, and not released from, the trust engine 110. Thus, so long as the user has access to the trust engine 110, the user may perform cryptographic functions using his or her private or public key. Such remote access advantageously allows users to remain completely mobile and access cryptographic functionality through practically any Internet connection, such as cellular and satellite phones, kiosks, laptops, hotel rooms and the like.

[0055] According to another embodiment, the trust engine 110 performs the cryptographic functionality using a key pair generated for the trust engine 110. According to this embodiment, the trust engine 110 first authenticates the user, and after the user has properly produced authentication data matching the enrollment authentication data, the trust engine 110 uses its own cryptographic key pair to perform cryptographic functions on behalf of the authenticated user.

[0056] A skilled artisan will recognize from the disclosure herein that all the cryptographic keys may advantageously include one or more of symmetric encryption keys (*e.g.*, private keys) and asymmetric encryption keys (*e.g.*, public/private keypairs). In addition, a skilled artisan will recognize from the disclosure herein that the foregoing keys may be implemented with a wide number of algorithms available from commercial technologies, such as, for example, RSA, ELGAMAL, or the like. When an asymmetric cryptographic scheme is described herein, this scheme may be substituted with a symmetric scheme, and *vice versa*.

[0057] FIGURE 1 also illustrates the certificate authority 115. According to one embodiment, the certificate authority 115 may advantageously comprise a trusted third-party organization or company that issues digital certificates, such as, for example, VeriSign, Baltimore, Entrust, or the like. The trust engine 110 may advantageously transmit requests for digital certificates, through one or more conventional digital certificate protocols, such as, for example, PKCS10, to the certificate authority 115. In response, the certificate authority 115 will issue a digital certificate in one or more of a number of differing protocols, such as, for

example, PKCS7. According to one embodiment of the invention, the trust engine 110 requests digital certificates from several or all of the prominent certificate authorities 115 such that the trust engine 110 has access to a digital certificate corresponding to the certificate standard of any requesting party.

[0058] According to another embodiment, the trust engine 110 internally performs certificate issuances. In this embodiment, the trust engine 110 may access a certificate system for generating certificates and/or may internally generate certificates when they are requested, such as, for example, at the time of key generation or in the certificate standard requested at the time of the request. The trust engine 110 will be disclosed in greater detail below.

[0059] FIGURE 1 also illustrates the vendor system 120. According to one embodiment, the vendor system 120 advantageously comprises a Web server. Typical Web servers generally serve content over the Internet using one of several internet markup languages or document format standards, such as the Hyper-Text Markup Language (HTML) or the Extensible Markup Language (XML). The Web server accepts requests from browsers like Netscape and Internet Explorer and then returns the appropriate electronic documents. A number of server or client-side technologies can be used to increase the power of the Web server beyond its ability to deliver standard electronic documents. For example, these technologies include Common Gateway Interface (CGI) scripts, Secure Sockets Layer (SSL) security, and Active Server Pages (ASPs). The vendor system 120 may advantageously provide electronic content relating to commercial, personal, educational, or other transactions.

[0060] Although the vendor system 120 is disclosed with reference to the foregoing embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize from the disclosure herein that the vendor system 120 may advantageously comprise any of the devices described with reference to the user system 105 or combination thereof.

[0061] FIGURE 1 also illustrates the communication link 125 connecting the user system 105, the trust engine 110, the certificate authority 115, and the vendor system 120. According to one embodiment, the communication link 125 preferably comprises the Internet. The Internet, as used throughout this disclosure

is a global network of computers. The structure of the Internet, which is well known to those of ordinary skill in the art, includes a network backbone with networks branching from the backbone. These branches, in turn, have networks branching from them, and so on. Routers move information packets between network levels, and then from network to network, until the packet reaches the neighborhood of its destination. From the destination, the destination network's host directs the information packet to the appropriate terminal, or node. In one advantageous embodiment, the Internet routing hubs comprise domain name system (DNS) servers using Transmission Control Protocol/Internet Protocol (TCP/IP) as is well known in the art. The routing hubs connect to one or more other routing hubs via high-speed communication links.

[0062] One popular part of the Internet is the World Wide Web. The World Wide Web contains different computers, which store documents capable of displaying graphical and textual information. The computers that provide information on the World Wide Web are typically called "websites." A website is defined by an Internet address that has an associated electronic page. The electronic page can be identified by a Uniform Resource Locator (URL). Generally, an electronic page is a document that organizes the presentation of text, graphical images, audio, video, and so forth.

[0063] Although the communication link 125 is disclosed in terms of its preferred embodiment, one of ordinary skill in the art will recognize from the disclosure herein that the communication link 125 may include a wide range of interactive communications links. For example, the communication link 125 may include interactive television networks, telephone networks, wireless data transmission systems, two-way cable systems, customized private or public computer networks, interactive kiosk networks, automatic teller machine networks, direct links, satellite or cellular networks, and the like.

[0064] FIGURE 2 illustrates a block diagram of the trust engine 110 of FIGURE 1 according to aspects of an embodiment of the invention. As shown in FIGURE 2, the trust engine 110 includes a transaction engine 205, a depository 210, an authentication engine 215, and a cryptographic engine 220. According to one embodiment of the invention, the trust engine 110 also includes mass storage 225.

As further shown in FIGURE 2, the transaction engine 205 communicates with the depository 210, the authentication engine 215, and the cryptographic engine 220, along with the mass storage 225. In addition, the depository 210 communicates with the authentication engine 215, the cryptographic engine 220, and the mass storage 225. Moreover, the authentication engine 215 communicates with the cryptographic engine 220. According to one embodiment of the invention, some or all of the foregoing communications may advantageously comprise the transmission of XML documents to IP addresses that correspond to the receiving device. As mentioned in the foregoing, XML documents advantageously allow designers to create their own customized document tags, enabling the definition, transmission, validation, and interpretation of data between applications and between organizations. Moreover, some or all of the foregoing communications may include conventional SSL technologies.

[0065] According to one embodiment, the transaction engine 205 comprises a data routing device, such as a conventional Web server available from Netscape, Microsoft, Apache, or the like. For example, the Web server may advantageously receive incoming data from the communication link 125. According to one embodiment of the invention, the incoming data is addressed to a front-end security system for the trust engine 110. For example, the front-end security system may advantageously include a firewall, an intrusion detection system searching for known attack profiles, and/or a virus scanner. After clearing the front-end security system, the data is received by the transaction engine 205 and routed to one of the depository 210, the authentication engine 215, the cryptographic engine 220, and the mass storage 225. In addition, the transaction engine 205 monitors incoming data from the authentication engine 215 and cryptographic engine 220, and routes the data to particular systems through the communication link 125. For example, the transaction engine 205 may advantageously route data to the user system 105, the certificate authority 115, or the vendor system 120.

[0066] According to one embodiment, the data is routed using conventional HTTP routing techniques, such as, for example, employing URLs or Uniform Resource Indicators (URIs). URIs are similar to URLs, however, URIs typically

indicate the source of files or actions, such as, for example, executables, scripts, and the like. Therefore, according to the one embodiment, the user system 105, the certificate authority 115, the vendor system 120, and the components of the trust engine 210, advantageously include sufficient data within communication URLs or URIs for the transaction engine 205 to properly route data throughout the cryptographic system.

[0067] Although the data routing is disclosed with reference to its preferred embodiment, a skilled artisan will recognize a wide number of possible data routing solutions or strategies. For example, XML or other data packets may advantageously be unpacked and recognized by their format, content, or the like, such that the transaction engine 205 may properly route data throughout the trust engine 110. Moreover, a skilled artisan will recognize that the data routing may advantageously be adapted to the data transfer protocols conforming to particular network systems, such as, for example, when the communication link 125 comprises a local network.

[0068] According to yet another embodiment of the invention, the transaction engine 205 includes conventional SSL encryption technologies, such that the foregoing systems may authenticate themselves, and vise-versa, with transaction engine 205, during particular communications. As will be used throughout this disclosure, the term "½ SSL" refers to communications where a server but not necessarily the client, is SSL authenticated, and the term "FULL SSL" refers to communications where the client and the server are SSL authenticated. When the instant disclosure uses the term "SSL", the communication may comprise ½ or FULL SSL.

[0069] As the transaction engine 205 routes data to the various components of the cryptographic system 100, the transaction engine 205 may advantageously create an audit trail. According to one embodiment, the audit trail includes a record of at least the type and format of data routed by the transaction engine 205 throughout the cryptographic system 100. Such audit data may advantageously be stored in the mass storage 225.

[0070] FIGURE 2 also illustrates the depository 210. According to one embodiment, the depository 210 comprises one or more data storage facilities, such

as, for example, a directory server, a database server, or the like. As shown in FIGURE 2, the depository 210 stores cryptographic keys and enrollment authentication data. The cryptographic keys may advantageously correspond to the trust engine 110 or to users of the cryptographic system 100, such as the user or vendor. The enrollment authentication data may advantageously include data designed to uniquely identify a user, such as, user ID, passwords, answers to questions, biometric data, or the like. This enrollment authentication data may advantageously be acquired at enrollment of a user or another alternative later time. For example, the trust engine 110 may include periodic or other renewal or reissue of enrollment authentication data.

[0071] According to one embodiment, the communication from the transaction engine 205 to and from the authentication engine 215 and the cryptographic engine 220 comprises secure communication, such as, for example conventional SSL technology. In addition, as mentioned in the foregoing, the data of the communications to and from the depository 210 may be transferred using URLs, URIs, HTTP or XML documents, with any of the foregoing advantageously having data requests and formats embedded therein.

[0072] As mentioned above, the depository 210 may advantageously comprises a plurality of secure data storage facilities. In such an embodiment, the secure data storage facilities may be configured such that a compromise of the security in one individual data storage facility will not compromise the cryptographic keys or the authentication data stored therein. For example, according to this embodiment, the cryptographic keys and the authentication data are mathematically operated on so as to statistically and substantially randomize the data stored in each data storage facility. According to one embodiment, the randomization of the data of an individual data storage facility renders that data undecipherable. Thus, compromise of an individual data storage facility produces only a randomized undecipherable number and does not compromise the security of any cryptographic keys or the authentication data as a whole.

[0073] FIGURE 2 also illustrates the trust engine 110 including the authentication engine 215. According to one embodiment, the authentication engine 215 comprises a data comparator configured to compare data from the

transaction engine 205 with data from the depository 210. For example, during authentication, a user supplies current authentication data to the trust engine 110 such that the transaction engine 205 receives the current authentication data. As mentioned in the foregoing, the transaction engine 205 recognizes the data requests, preferably in the URL or URI, and routes the authentication data to the authentication engine 215. Moreover, upon request, the depository 210 forwards enrollment authentication data corresponding to the user to the authentication engine 215. Thus, the authentication engine 215 has both the current authentication data and the enrollment authentication data for comparison.

[0074] According to one embodiment, the communications to the authentication engine comprise secure communications, such as, for example, SSL technology. Additionally, security can be provided within the trust engine 110 components, such as, for example, super-encryption using public key technologies. For example, according to one embodiment, the user encrypts the current authentication data with the public key of the authentication engine 215. In addition, the depository 210 also encrypts the enrollment authentication data with the public key of the authentication engine 215. In this way, only the authentication engine's private key can be used to decrypt the transmissions.

[0075] As shown in FIGURE 2, the trust engine 110 also includes the cryptographic engine 220. According to one embodiment, the cryptographic engine comprises a cryptographic handling module, configured to advantageously provide conventional cryptographic functions, such as, for example, public-key infrastructure (PKI) functionality. For example, the cryptographic engine 220 may advantageously issue public and private keys for users of the cryptographic system 100. In this manner, the cryptographic keys are generated at the cryptographic engine 220 and forwarded to the depository 210 such that at least the private cryptographic keys are not available outside of the trust engine 110. According to another embodiment, the cryptographic engine 220 randomizes and splits at least the private cryptographic key data, thereby storing only the randomized split data. Similar to the splitting of the enrollment authentication data, the splitting process ensures the stored keys are not available outside the cryptographic engine 220.

According to another embodiment, the functions of the cryptographic engine can be combined with and performed by the authentication engine 215.

[0076] According to one embodiment, communications to and from the cryptographic engine include secure communications, such as SSL technology. In addition, XML documents may advantageously be employed to transfer data and/or make cryptographic function requests.

[0077] FIGURE 2 also illustrates the trust engine 110 having the mass storage 225. As mentioned in the foregoing, the transaction engine 205 keeps data corresponding to an audit trail and stores such data in the mass storage 225. Similarly, according to one embodiment of the invention, the depository 210 keeps data corresponding to an audit trail and stores such data in the mass storage device 225. The depository audit trail data is similar to that of the transaction engine 205 in that the audit trail data comprises a record of the requests received by the depository 210 and the response thereof. In addition, the mass storage 225 may be used to store digital certificates having the public key of a user contained therein.

[0078] Although the trust engine 110 is disclosed with reference to its preferred and alternative embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize in the disclosure herein, a wide number of alternatives for the trust engine 110. For example, the trust engine 110, may advantageously perform only authentication, or alternatively, only some or all of the cryptographic functions, such as data encryption and decryption. According to such embodiments, one of the authentication engine 215 and the cryptographic engine 220 may advantageously be removed, thereby creating a more straightforward design for the trust engine 110. In addition, the cryptographic engine 220 may also communicate with a certificate authority such that the certificate authority is embodied within the trust engine 110. According to yet another embodiment, the trust engine 110 may advantageously perform authentication and one or more cryptographic functions, such as, for example, digital signing.

[0079] FIGURE 3 illustrates a block diagram of the transaction engine 205 of FIGURE 2, according to aspects of an embodiment of the invention. According to this embodiment, the transaction engine 205 comprises an operating system 305

having a handling thread and a listening thread. The operating system 305 may advantageously be similar to those found in conventional high volume servers, such as, for example, Web servers available from Apache. The listening thread monitors the incoming communication from one of the communication link 125, the authentication engine 215, and the cryptographic engine 220 for incoming data flow. The handling thread recognizes particular data structures of the incoming data flow, such as, for example, the foregoing data structures, thereby routing the incoming data to one of the communication link 125, the depository 210, the authentication engine 215, the cryptographic engine 220, or the mass storage 225. As shown in FIGURE 3, the incoming and outgoing data may advantageously be secured through, for example, SSL technology.

[0080] FIGURE 4 illustrates a block diagram of the depository 210 of FIGURE 2 according to aspects of an embodiment of the invention. According to this embodiment, the depository 210 comprises one or more lightweight directory access protocol (LDAP) servers. LDAP directory servers are available from a wide variety of manufacturers such as Netscape, ISO, Microsoft, and others. FIGURE 4 also shows that the directory server preferably stores data 405 corresponding to the cryptographic keys and data 410 corresponding to the enrollment authentication data. According to one embodiment, the depository 210 comprises a single logical memory structure indexing authentication data and cryptographic key data to a unique user ID. The single logical memory structure preferably includes mechanisms to ensure a high degree of trust, or security, in the data stored therein. For example, the physical location of the depository 210 may advantageously include a wide number of conventional security measures, such as limited employee access, modern surveillance systems, and the like. In addition to, or in lieu of, the physical securities, the computer system or server may advantageously include software solutions to protect the stored data. For example, the depository 210 may advantageously create and store data 415 corresponding to an audit trail of actions taken. In addition, the incoming and outgoing communications may advantageously be encrypted with public key encryption coupled with conventional SSL technologies.

[0081] According to another embodiment, the depository 210 may include distinct and physically separated data storage facilities. For example, depository 210 may include more than one physical storage device (*e.g.*, tape drive, hard disk drive, optical drive, or any combination thereof) housed in geographically separated locations.

[0082] FIGURE 5 illustrates a block diagram of the authentication engine 215 of FIGURE 2 according to aspects of an embodiment of the invention. Similar to the transaction engine 205 of FIGURE 3, the authentication engine 215 comprises an operating system 505 having at least a listening and a handling thread of a modified version of a conventional Web server, such as, for example, Web servers available from Apache. As shown in FIGURE 5, the authentication engine 215 includes access to at least one private key 510. The private key 510 may advantageously be used for example, to decrypt data from the transaction engine 205 or the depository 210, which was encrypted with a corresponding public key of the authentication engine 215.

[0083] FIGURE 5 also illustrates the authentication engine 215 comprising a comparator 515, a data splitting module 520, and a data assembling module 525. According to an embodiment of the invention, the comparator 515 includes technology capable of comparing potentially complex patterns related to the foregoing biometric authentication data. The technology may include hardware, software, or combined solutions for pattern comparisons, such as, for example, those representing finger print patterns or voice patterns. In addition, according to one embodiment, the comparator 515 of the authentication engine 215 may advantageously compare conventional hashes of documents in order to render a comparison result. According to one embodiment of the invention, the comparator 515 includes the application of heuristics 530 to the comparison. The heuristics 530 may advantageously address circumstances surrounding an authentication attempt, such as, for example, the time of day, IP address or subnet mask, purchasing profile, email address, processor serial number or ID, or the like.

[0084] Moreover, the nature of biometric data comparisons may result in varying degrees of confidence being produced from the matching of current biometric authentication data to enrollment data. For example, unlike a traditional password

which may only return a positive or negative match, a fingerprint may be determined to be a partial match, *e.g.* a 90% match, a 75% match, or a 10% match, rather than simply being correct or incorrect. Other biometric identifiers such as voice print analysis or face recognition may share this property of probabilistic authentication, rather than absolute authentication.

[0085] When working with such probabilistic authentication or in other cases where an authentication is considered less than absolutely reliable, it is desirable to apply the heuristics 530 to determine whether the level of confidence in the authentication provided is sufficiently high to authenticate the transaction which is being made.

[0086] It will sometimes be the case that the transaction at issue is a relatively low value transaction where it is acceptable to be authenticated to a lower level of confidence. This could include a transaction which has a low dollar value associated with it (*e.g.*, a $10 purchase) or a transaction with low risk (*e.g.*, admission to a members-only web site).

[0087] Conversely, for authenticating other transactions, it may be desirable to require a high degree of confidence in the authentication before allowing the transaction to proceed. Such transactions may include transactions of large dollar value (*e.g.*, signing a multi-million dollar supply contract) or transaction with a high risk if an improper authentication occurs (*e.g.*, remotely logging onto a government computer).

[0088] The use of the heuristics 530 in combination with confidence levels and transactions values may be used as will be described below to allow the comparator to provide a dynamic context-sensitive authentication system.

[0089] According to another embodiment of the invention, the comparator 515 may advantageously track authentication attempts for a particular transaction. For example, when a transaction fails, the trust engine 110 may request the user to re-enter his or her current authentication data. The comparator 515 of the authentication engine 215 may advantageously employ an attempt limiter 535 to limit the number of authentication attempts, thereby prohibiting brute-force attempts to impersonate a user's authentication data. According to one embodiment, the attempt limiter 535 comprises a software module monitoring

transactions for repeating authentication attempts and, for example, limiting the authentication attempts for a given transaction to three. Thus, the attempt limiter 535 will limit an automated attempt to impersonate an individual's authentication data to, for example, simply three "guesses." Upon three failures, the attempt limiter 535 may advantageously deny additional authentication attempts. Such denial may advantageously be implemented through, for example, the comparator 515 returning a negative result regardless of the current authentication data being transmitted. On the other hand, the transaction engine 205 may advantageously block any additional authentication attempts pertaining to a transaction in which three attempts have previously failed.

[0090] The authentication engine 215 also includes the data splitting module 520 and the data assembling module 525. The data splitting module 520 advantageously comprises a software, hardware, or combination module having the ability to mathematically operate on various data so as to substantially randomize (*e.g.*, randomize or pseudo-randomize) and split the data into individual portions (also called shares). In some embodiments, to substantially randomize data, data splitting module 520 may generate random or pseudo-random numbers. The random or pseudo-random numbers may then be associated with units (*e.g.*, any number of bits, bytes, or blocks) of data from a data set. The random or pseudo-random numbers may also be associated with the shares. Data splitting module 520 may determine into which share to store each unit of data from the data set based on, at least in part, the association of the random or pseudo-random numbers with the units of data and the shares.

[0091] According to one embodiment, original data is not recreatable from an individual portion or user share. The data assembling module 525 advantageously comprises a software, hardware, or combination module configured to mathematically operate on the foregoing substantially randomized portions, such that the combination thereof provides the original deciphered data. According to one embodiment, the authentication engine 215 employs the data splitting module 520 to randomize and split enrollment authentication data into portions, and employs the data assembling module 525 to reassemble the portions into usable enrollment authentication data.

[0092] FIGURE 6 illustrates a block diagram of the cryptographic engine 220 of the trust engine 200 of FIGURE 2 according to aspects of one embodiment of the invention. Similar to the transaction engine 205 of FIGURE 3, the cryptographic engine 220 comprises an operating system 605 having at least a listening and a handling thread of a modified version of a conventional Web server, such as, for example, Web servers available from Apache. As shown in FIGURE 6, the cryptographic engine 220 comprises a data splitting module 610 and a data assembling module 620 that function similar to those of FIGURE 5. However, according to one embodiment, the data splitting module 610 and the data assembling module 620 process cryptographic key data, as opposed to the foregoing enrollment authentication data. Although, a skilled artisan will recognize from the disclosure herein that the data splitting module 910 and the data splitting module 620 may be combined with those of the authentication engine 215.

[0093] The cryptographic engine 220 also comprises a cryptographic handling module 625 configured to perform one, some or all of a wide number of cryptographic functions. According to one embodiment, the cryptographic handling module 625 may comprise software modules or programs, hardware, or both. According to another embodiment, the cryptographic handling module 625 may perform data comparisons, data parsing, data splitting, data separating, data hashing, data encryption or decryption, digital signature verification or creation, digital certificate generation, storage, or requests, cryptographic key generation, or the like. Moreover, a skilled artisan will recognize from the disclosure herein that the cryptographic handling module 825 may advantageously comprises a public-key infrastructure, such as Pretty Good Privacy (PGP), an RSA-based public-key system, or a wide number of alternative key management systems. In addition, the cryptographic handling module 625 may perform public-key encryption, symmetric-key encryption, or both. In addition to the foregoing, the cryptographic handling module 625 may include one or more computer programs or modules, hardware, or both, for implementing seamless, transparent, interoperability functions.

[0094] A skilled artisan will also recognize from the disclosure herein that the cryptographic functionality may include a wide number or variety of functions generally relating to cryptographic key management systems.

[0095] The secure data parser of the present invention may be integrated into an operating system kernel (*e.g.*, Linux, Unix, or any other suitable commercial or proprietary operating system). This integration may be used to protect data at the device level whereby, for example, data that would ordinarily be stored in one or more devices is separated into a certain number of portions by the secure data parser integrated into the operating system and stored among the one or more devices. When original data is attempted to be accessed, the appropriate software, also integrated into the operating system, may recombine the parsed data portions into the original data in a way that may be transparent to the end user.

[0096] The secure data parser of the present invention may be integrated into a volume manager or any other suitable component of a storage system to protect local and networked data storage across any or all supported platforms. For example, with the secure data parser integrated, a storage system may make use of the redundancy offered by the secure data parser (*i.e.*, which is used to implement the feature of needing fewer than all separated portions of data in order to reconstruct the original data) to protect against data loss. The secure data parser also allows all data written to storage devices, whether using redundancy or not, to be in the form of multiple portions that are generated according to the parsing of the present invention. When original data is attempted to be accessed, the appropriate software, also integrated into the volume manager or other suitable component of the storage system, may recombine the parsed data portions into the original data in a way that may be transparent to the end user.

[0097] In one suitable approach, the secure data parser of the present invention may be integrated into a RAID controller (as either hardware or software). This allows for the secure storage of data to multiple drives while maintaining fault tolerance in case of drive failure.

[0098] The secure data parser of the present invention may be integrated into a database in order to, for example, protect sensitive table information. For example, in one suitable approach, data associated with particular cells of a database table

(*e.g.*, individual cells, one or more particular columns, one or more particular rows, any combination thereof, or an entire database table) may be parsed and separated according to the present invention (*e.g.*, where the different portions are stored on one or more storage devices at one or more locations or on a single storage device). Access to recombine the portions in order to view the original data may be granted by traditional authentication methods (*e.g.*, username and password query).

[0099] The secure parser of the present invention may be integrated in any suitable system that involves data in motion (*i.e.*, transfer of data from one location to another). Such systems include, for example, network file servers and file systems, email servers, web servers, streaming data broadcasts, and wireless (*e.g.*, WiFi) communications. With respect to email, in one suitable approach, the secure parser may be used to parse outgoing messages (*i.e.*, containing text, binary data, or both (*e.g.*, files attached to an email message)) and sending the different portions of the parsed data along different paths thus creating multiple streams of data. If any one of these streams of data is compromised, the original message remains secure because the system may require that more than one of the portions be combined, in accordance with the present invention, in order to generate the original data. In another suitable approach, the different portions of data may be communicated along one path sequentially so that if one portion is obtained, it may not be sufficient to generate the original data. The different portions arrive at the intended recipient's location and may be combined to generate the original data in accordance with the present invention.

[0100] FIGURES 7 and 8 are illustrative block diagrams of such email systems. FIGURE 7 shows a sender system 700, which may include any suitable hardware, such as a computer terminal, personal computer, handheld device (*e.g.*, PDA, Blackberry), cellular telephone, computer network, any other suitable hardware, or any combination thereof. Sender system 700 is used to generate and/or store a message 704, which may be, for example, a network packet, an email message, a binary data file (*e.g.*, graphics, voice, video, etc.), or any combination of the foregoing. Message 704 is parsed and split by secure data parser 702 in accordance with the present invention. The resultant data portions may be communicated across one or more separate communications paths 706 over

network 708 (*e.g.*, the Internet, an intranet, a LAN, WiFi, Bluetooth, any other suitable hard-wired or wireless communications means, or any combination thereof) to recipient system 710. The data portions may be communicated parallel in time or alternatively, according to any suitable time delay between the communication of the different data portions. Recipient system 710 may be any suitable hardware as described above with respect to sender system 700. The separate data portions carried along communications paths 706 are recombined at recipient system 710 to generate the original message or data in accordance with the present invention.

[0101] FIGURE 8 shows a sender system 800, which may include any suitable hardware, such as a computer terminal, personal computer, handheld device (*e.g.*, PDA), cellular telephone, computer network, any other suitable hardware, or any combination thereof. Sender system 800 is used to generate and/or store a message 804, which may be, for example, a network packet, an email message, a binary data file (*e.g.*, graphics, voice, video, etc.), or any combination of the foregoing. Message 804 is parsed and split by secure data parser 802 in accordance with the present invention. The resultant data portions may be communicated across a single communications paths 806 over network 808 (*e.g.*, the Internet, an intranet, a LAN, WiFi, Bluetooth, any other suitable communications means, or any combination thereof) to recipient system 810. The data portions may be communicated serially across communications path 806 with respect to one another. Recipient system 810 may be any suitable hardware as described above with respect to sender system 800. The separate data portions carried along communications path 806 are recombined at recipient system 810 to generate the original message or data in accordance with the present invention.

[0102] It will be understood that the arrangement of FIGURES 7 and 8 are merely illustrative. Any other suitable arrangement may be used. For example, in another suitable approach, the features of the systems of FIGURES 7 and 8 may be combined whereby the multi-path approach of FIGURE 7 is used and in which one or more of communications paths 706 are used to carry more than one portion of data as communications path 806 does in the context of FIGURE 8.

[0103] The secure data parser may be integrated at any suitable level of a data-in motion system. For example, in the context of an email system, the secure parser may be integrated at the user-interface level (*e.g.*, into Microsoft® Outlook), in which case the user may have control over the use of the secure data parser features when using email. Alternatively, the secure parser may be implemented in a back-end component such as at the exchange server, in which case messages may be automatically parsed, split, and communicated along different paths in accordance with the present invention without any user intervention. In some embodiments, the integration may be transparent to the user.

[0104] Similarly, in the case of streaming broadcasts of data (*e.g.*, audio, video), the outgoing data may be parsed and separated into multiple streams each containing a portion of the parsed data. The multiple streams may be transmitted along one or more paths and recombined at the recipient's location in accordance with the present invention. One of the benefits of this approach is that it avoids the relatively large overhead associated with traditional encryption of data followed by transmission of the encrypted data over a single communications channel. The secure parser of the present invention allows data in motion to be sent in multiple parallel streams, increasing speed and efficiency.

[0105] It will be understand that the secure data parser may be integrated for protection of and fault tolerance of any type of data in motion through any transport medium, including, for example, wired, wireless, or physical. For example, voice over Internet protocol (VoIP) applications may make use of the secure data parser of the present invention. Wireless or wired data transport from or to any suitable personal digital assistant (PDA) devices such as Blackberries and SmartPhones may be secured using the secure data parser of the present invention. Communications using wireless 802.11 protocols for peer to peer and hub based wireless networks, satellite communications, point to point wireless communications, Internet client/server communications, or any other suitable communications may involve the data in motion capabilities of the secure data parser in accordance with the present invention. Data communication between computer peripheral device (*e.g.*, printer, scanner, monitor, keyboard, network router, biometric authentication device (*e.g.*, fingerprint scanner), or any other

suitable peripheral device) between a computer and a computer peripheral device, between a computer peripheral device and any other suitable device, or any combination thereof may make use of the data in motion features of the present invention.

[0106] The data in motion features of the present invention may also apply to physical transportation of secure shares using for example, separate routes, vehicles, methods, any other suitable physical transportation, or any combination thereof. For example, physical transportation of data may take place on digital/magnetic tapes, floppy disks, optical disks, physical tokens, USB drives, removable hard drives, consumer electronic devices with flash memory (*e.g.*, Apple IPODs or other MP3 players), flash memory, any other suitable medium used for transporting data, or any combination thereof.

[0107] The secure data parser of the present invention may provide security with the ability for disaster recovery. According to the present invention, fewer than all portions of the separated data generated by the secure data parser may be necessary in order to retrieve the original data. That is, out of m portions stored, n may be the minimum number of these m portions necessary to retrieve the original data, where n <= m. For example, if each of four portions is stored in a different physical location relative to the other three portions, then, if n=2 in this example, two of the locations may be compromised whereby data is destroyed or inaccessible, and the original data may still be retrieved from the portions in the other two locations. Any suitable value for n or m may be used.

[0108] In addition, the n of m feature of the present invention may be used to create a "two man rule" whereby in order to avoid entrusting a single individual or any other entity with full access to what may be sensitive data, two or more distinct entities, each with a portion of the separated data parsed by the secure parser of the present invention may need to agree to put their portions together in order to retrieve the original data.

[0109] The secure data parser of the present invention may be used to provide a group of entities with a group-wide key that allows the group members to access particular information authorized to be accessed by that particular group. The group key may be one of the data portions generated by the secure parser in

accordance with the present invention that may be required to be combined with another portion centrally stored, for example in order to retrieve the information sought. This feature allows for, for example, secure collaboration among a group. It may be applied in for example, dedicated networks, virtual private networks, intranets, or any other suitable network.

[0110] Specific applications of this use of the secure parser include, for example, coalition information sharing in which, for example, multi-national friendly government forces are given the capability to communicate operational and otherwise sensitive data on a security level authorized to each respective country over a single network or a dual network (*i.e.*, as compared to the many networks involving relatively substantial manual processes currently used). This capability is also applicable for companies or other organizations in which information needed to be known by one or more specific individuals (within the organization or without) may be communicated over a single network without the need to worry about unauthorized individuals viewing the information.

[0111] Another specific application includes a multi-level security hierarchy for government systems. That is, the secure parser of the present invention may provide for the ability to operate a government system at different levels of classified information (*e.g.*, unclassified, classified, secret, top secret) using a single network. If desired, more networks may be used (*e.g.*, a separate network for top secret), but the present invention allows for substantially fewer than current arrangement in which a separate network is used for each level of classification.

[0112] It will be understood that any combination of the above described applications of the secure parser of the present invention may be used. For example, the group key application can be used together with the data in motion security application (*i.e.*, whereby data that is communicated over a network can only be accessed by a member of the respective group and where, while the data is in motion, it is split among multiple paths (or sent in sequential portions) in accordance with the present invention).

[0113] The secure data parser of the present invention may be integrated into any middleware application to enable applications to securely store data to different database products or to different devices without modification to either the

applications or the database. Middleware is a general term for any product that allows two separate and already existing programs to communicate. For example, in one suitable approach, middleware having the secure data parser integrated, may be used to allow programs written for a particular database to communicate with other databases without custom coding.

[0114] The secure data parser of the present invention may be implemented having any combination of any suitable capabilities, such as those discussed herein. In some embodiments of the present invention, for example, the secure data parser may be implemented having only certain capabilities whereas other capabilities may be obtained through the use of external software, hardware, or both interfaced either directly or indirectly with the secure data parser.

[0115] The secure data parser of the present invention may be advantageously used in a number of applications and technologies. For example, email system, RAID systems, video broadcasting systems, database systems, tape backup systems, or any other suitable system may have the secure data parser integrated at any suitable level. As previously discussed, it will be understand that the secure data parser may also be integrated for protection and fault tolerance of any type of data in motion through any transport medium, including, for example, wired, wireless, or physical transport mediums. As one example, voice over Internet protocol (VoIP) applications may make use of the secure data parser of the present invention to solve problems relating to echoes and delays that are commonly found in VoIP. The need for network retry on dropped packets may be eliminated by using fault tolerance, which guarantees packet delivery even with the loss of a predetermined number of shares. Packets of data (*e.g.*, network packets) may also be efficiently split and restored "on-the-fly" with minimal delay and buffering, resulting in a comprehensive solution for various types of data in motion. The secure data parser may act on network data packets, network voice packets, file system data blocks, or any other suitable unit of information. In addition to being integrated with a VoIP application, the secure data parser may be integrated with a file-sharing application (*e.g.*, a peer-to-peer file-sharing application), a video broadcasting application, an electronic voting or polling application (which may implement an electronic voting protocol and blind signatures, such as the Sensus

protocol), an email application, or any other network application that may require or desire secure communication.

[0116] In some embodiments, support for network data in motion may be provided by the secure data parser of the present invention in two distinct phases -- a header generation phase and a data partitioning phase. Simplified header generation process 900 and simplified data partitioning process 910 are shown in FIGURES 9A and 9B, respectively. One or both of these processes may be performed on network packets, file system blocks, or any other suitable information.

[0117] In some embodiments, header generation process 900 shown in FIGURE 9A may be performed one time at the initiation of a network packet stream. At step 902, a random (or pseudo-random) split encryption key, K, may be generated. The split encryption key, K, may then be optionally encrypted (*e.g.*, using the workgroup key described above) at AES key wrap step 904. Although an AES key wrap may be used in some embodiments, any suitable key encryption or key wrap algorithm may be used in other embodiments. AES key wrap step 904 may operate on the entire split encryption key, K, or the split encryption key may be parsed into several blocks (*e.g.*, 64-bit blocks). AES key wrap step 904 may then operate on blocks of the split encryption key, if desired.

[0118] At step 906, a secret sharing algorithm (*e.g.*, Shamir) may be used to split the split encryption key, K, into key shares. Each key share may then be embedded into one of the output shares (*e.g.*, in the share headers). Finally, a share integrity block and (optionally) a post-authentication tag (*e.g.*, MAC) may be appended to the header block of each share. Each header block may be designed to fit within a single data packet.

[0119] After header generation is complete (*e.g.*, using simplified header generation process 900), the secure data parser may enter the data partitioning phase shown in FIGURE 9B using simplified data splitting process 910. Each incoming data packet or data block in the stream is encrypted using the split encryption key, K, at step 912. At step 914, share integrity information (*e.g.*, a hash H) may be computed on the resulting ciphertext from step 912. For example, a SHA-256 hash may be computed. At step 916, the data packet or data block may

then be partitioned into two or more data shares using one of the data splitting algorithms described above in accordance with the present invention. In some embodiments, the data packet or data block may be split so that each data share contains a substantially random distribution of the encrypted data packet or data block. The integrity information (*e.g.*, hash H) may then be appended to each data share. An optional post-authentication tag (*e.g.*, MAC) may also be computed and appended to each data share in some embodiments.

[0120] Each data share may include metadata, which may be necessary to permit correct reconstruction of the data blocks or data packets. This information may be included in the share header. The metadata may include such information as cryptographic key shares, key identities or identifiers, share nonces, signatures/MAC values, and integrity blocks. In order to maximize bandwidth efficiency, the metadata may be stored in a compact binary format.

[0121] For example, in some embodiments, the share header may include a cleartext header chunk, which is not encrypted and may include such elements as the Shamir key share, per-session nonce, per-share nonce, and key identifiers (*e.g.*, a workgroup key identifier and a post-authentication key identifier). The share header may also include an encrypted header chunk, which is encrypted using the split encryption key. An integrity header chunk, which may include integrity checks for any number of the previous blocks (*e.g.*, the previous two blocks) may also be included in the header. Any other suitable values or information may also be included in the share header in other embodiments.

[0122] As shown in illustrative share format 1000 of FIGURE 10, header block 1002 may be associated with two or more output blocks 1004. Each header block, such as header block 1002, may be designed to fit within a single network data packet. In some embodiments, after header block 1002 is transmitted from a first location to a second location, the output blocks may then be transmitted. Alternatively, header block 1002 and output blocks 1004 may be transmitted at the same time in parallel. The transmission may occur over one or more similar or dissimilar communications paths.

[0123] Each output block may include data portion 1006 and integrity/authenticity portion 1008. As described above, each data share may be

secured using a share integrity portion including share integrity information (*e.g.*, a SHA-256 hash) of the encrypted, pre-partitioned data. To verify the integrity of the outputs blocks at recovery time, the secure data parser may compare the share integrity blocks of each share and then invert the split algorithm. The hash of the recovered data may then be verified against the share hash.

[0124] As described above, the secure data parser of the present invention may also be used to secure data in motion (or data being communicated from one system or service to another) within a workgroup. In some embodiments, a centralized server handles all user, service, and workgroup management for a single domain (or across multiple domains). The centralized server may include one server or a cluster of servers (for fault-tolerance) and may include an LDAP server, such as a server running Microsoft® Server 2003 and Active Directory.

[0125] Seamless user authentication within a workgroup may be provided using challenge/response protocol 1100 as shown in FIGURE 11. Using protocol 1100, workgroup client 1102 may authenticate itself to authentication server 1104. Seamless user authentication may be granted through the verification of a shared secret (for example, using Digest Authentication). When a user logs onto workgroup client 1102, a password entered by the user may be put through a cryptographic function transforming the ASCII text of the password into a cryptographic key. This password-based key (labeled "PBK" in the example of FIGURE 11) may also be stored on, or accessible by, authentication server 1104.

[0126] To be authenticated, workgroup client 1102 may identify itself to authentication server 1104 with message 1106. For example, message 1106 may include data indicative of the user's identity, such as, for example, the user's workgroup name, logon username, network address, MAC address, or any combination of the foregoing. In response to message 1106, authentication server 1104 may send workgroup client 1102 a challenge value in message 1108. The challenge value may include any number of bits or bytes which, after received by workgroup client 1102, may be encrypted using the password-based key. The resulting ciphertext (labeled "Y" in the example of FIGURE 11) may then be returned to the server in message 1110. Authentication server 1104 may then perform the same encryption on the challenge value using the stored password-

based key for the workgroup client and verify that its own encryption and the received ciphertext match. If there is a match, the user has been authenticated. Otherwise, the authentication request may be denied.

[0127] FIGURE 12 shows illustrative process 1200 for communicating with a service on the network. Process 1200 may be implemented using a Kerberos subsystem, which allows two workgroup clients (such as clients 1202 and 1206) on a network to prove their authenticity to one another through the use of a central server sometimes referred to as Key Distribution Center (KDC) 1204.

[0128] After being authenticated (*e.g.*, using a challenge/response protocol as shown by messages 1208), a user may be given a Ticket Granting Ticket (TGT) in message 1210. This ticket may serve as a token to be given to KDC 1204 whenever client 1202 intends to communicate with another user or service on the network (such as a web server, email server, or client 1206). The TGT may include, for example, a block of information containing the name or identity of the recipient, a timestamp, and any other necessary credentials. The TGT may also be encrypted under the KDC's Master Key so that only the KDC can read its contents.

[0129] When a user with a TGT wants to communicate with a service on the network, it may pass the TGT to the KDC and request a valid Service Ticket (ST). For example, as shown in FIGURE 12, workgroup client 1202 has requested a ST for client 1206 via message 1212. If the TGT is valid, KDC 1204 may return to client 1202 two STs via message 1214, each ST containing the credentials of the parties involved in the communication, a timestamp, and a unique and fresh session key (SK) to be used as a temporary shared key between both parties for encryption. One of the STs may be encrypted under the shared password-based key between the user and the KDC, and the other ST may be encrypted under the shared password-based key between the service or client being contacted (*e.g.*, client 1206) and the KDC.

[0130] The ST encrypted for client 1202 making the request can immediately be decrypted by client 1202 and the SK uncovered. Client 1202 may then send the second ST and a message encrypted under the SK to the targeted service or client (*e.g.*, client 1206) via message 1216. In a final step, the service or client (*e.g.*, client 1206) may then decrypt the ST with its password-based key to obtain the

SK, and subsequently decrypt the message sent by client 1202. Through this process, clients 1202 and 1206 now share the SK.

[0131] In practice, one or more steps shown in process 1200 may be combined with other steps, performed in any suitable order, performed in parallel (*e.g.*, simultaneously or substantially simultaneously), or removed.

[0132] FIGURE 13 shows illustrative process 1300 for using shared keys with the secure data parser. The SK described above may be used as a shared workgroup key (WK) to support secure workgroup communication using the secure data parser. In some embodiments, the secure data parser WK may be used to wrap internally generated session keys, including the Encryption Session Key (ESK) and the MAC session key (MK), as described in more detail in U.S. Patent No. 7,391,865 and U.S. Patent Application Nos. 10/458,928, filed June 11, 2003, 11/258,839, filed October 25, 2005, 11/602,667, filed November 20, 2006, 11/983,355, filed November 7, 2007, 11/999,575, filed December 5, 2007, 12/209,703, filed September 12, 2008, and 12/349,897, filed January 7, 2009.

[0133] After wrapped with the WK, the ciphertexts may be securely shared (*e.g.*, using a secret sharing scheme, such as Shamir) among each header generated by the secure data parser, where a single share of each wrapped key is stored with each header. Obtaining the ESK or MK may now require obtaining WK and a quorum of shares sufficient to rebuild the wrapped ESK and MK ciphertexts. In some embodiments, the secure data parser WK may be used to wrap an internally generated Encryption Session Key (ESK). Once wrapped with the WK, the ciphertext may be again wrapped with a freshly generated Key Encrypting Key (KEK). Now the ESK has been doubly wrapped and may only be recovered with knowledge of both the WK and the KEK. The KEK may be securely shared among each header generated by the secure data parser, and the doubly wrapped ESK may be stored within each header. Obtaining the ESK now may require obtaining the WK and a quorum of shares sufficient to rebuild the KEK. In a typical communication session, two users or workgroup clients may obtain tickets encrypted only for each client individually, the tickets containing the WK value. The WK value may then be imported in to the KeyStore for each of the two users

or clients and used as a normal WK in the context of a secure data parser communication session.

[0134] For example, client 1302 may import a shared WK into its KeyStore and generate secure data parser headers, as described above in regard to FIGURES 9 and 10. The secure data parser headers may include one or more of the WK, the workgroup-key ID, and the WK version number (*e.g.*, the WK, key ID, and version number may be shared out to the headers using a secret sharing scheme, such as Shamir, in some embodiments). Client 1302 may then send these headers to client 1304 via messages 1306. Client 1304 may reconstruct the WK, the workgroup-key ID, and the WK version number from received headers and import the WK into its KeyStore. After each client has stored the WK in its respective KeyStore, a secure data parser communications session (*e.g.*, parsing a data set and restoring the data set) may then take place using data messages 1308 and 1310.

[0135] In practice, one or more steps shown in process 1300 may be combined with other steps, performed in any suitable order, performed in parallel (*e.g.*, simultaneously or substantially simultaneously), or removed.

[0136] In order for the above mechanisms to be integrated with a kernel-level network driver, in some embodiments, authentication server modifications are provided to distribute the session keys. Workstation modifications may also be provided, in some embodiments, to request TGTs and STs. Process 1400, as shown in FIGURE 14, illustrates an exemplary message sequence. For example, Kerberos functionality and communication with KDC 1406 generally cannot occur in kernel space 1404. Instead, the network driver may call back to user space 1402 to perform the Kerberos key exchanges, then pass back down any data that is used by the network driver (or sent to the original destination). For example, after client 1401 wishes to communicate with client 1419, message 1408 may be generated and delivered from user space 1402 to the network driver in kernel space 1404. The network driver may determine whether or not there is already a secure date parser session with the desired destination IP address (or other network identifier), and, if so, may use that channel. If there is no existing session, the network driver may pass the IP address (or other network identifier) up to a user-level service to obtain STs using message 1410.

[0137] The service may then determine whether or not the STs for the given IP address (or other network identifier) are currently cached, and, if not, the user-level service may obtain the STs from KDC 1406 via messages 1412. Next, the user-level service may decrypt the embedded WK in the ST for client 1401 and pass the WK and the ST for client 1419 back to the network driver in kernel space 1404 via message 1414. The network driver may then use the WK to generate secure data parser headers and send them, along with the ST for client 1419, to client 1419 via messages 1416. After receiving the headers and the ST at the network driver of client 1419, the ST may then be passed up to its own user-level service via message 1418. The user-level service may decrypt the ST and extract the WK, which is then passed back down to the network driver via message 1420. Client 1419 may then use WK to restore the secure data parser headers it has received and complete the channel.

[0138] In practice, one or more steps shown in process 1400 may be combined with other steps, performed in any suitable order, performed in parallel (*e.g.*, simultaneously or substantially simultaneously), or removed.

[0139] FIGURE 15 shows illustrative network topology 1500. Certificate Authority (CA) 1502 may enroll each Workgroup Authority (WA) in the network or domain. Although in the example of FIGURE 15 three WAs are shown (*i.e.*, WA 1504, WA 1506, and WA 1508), in actual implementations there may be more or less than three WAs in a single network or domain. CA 1502 may enroll each of WA 1504, WA 1506, and WA 1508 by issuing each WA a WA certificate for their respective workgroup. The certificates may be digitally signed by CA 1502 to prevent tampering.

[0140] After being enrolled by CA 1602, WA 1504, WA 1506, and WA 1508 may begin enrolling clients (*e.g.*, users, machines, and services) into their respective workgroups. For example, in some embodiments, enrollment may be based, at least in part, on pre-exiting Active Directory (or other LDAP) membership semantics. Enrolled clients may be issued certificates signed by their respective WA, as well as the certificates of the WA, CA, or both.

[0141] FIGURE 16 shows illustrative connection setup 1600. Initiator 1604 may first ensure that it is using the latest WK by performing key update 1608 with WA

1602. Initiator 1604 may provide a nonce to WA 1602 and prove that it possesses a valid certificate. WA 1602 may then provide an encrypted and signed WK that includes the initiator's nonce. In some embodiments, encryption is performed using RSA PKCS #1 v2 with RSA-PSS or ECDSA signatures. In other embodiments, AES-GCM authenticated encryption is used under pre-shared symmetric keys.

[0142] Initiator 1604 then initiates a connection with recipient 1606 using the latest WK with request message 1610. After receiving the connection request, recipient 1606 may ensure that it has the latest WK. Key update 1612 may be performed in the same encrypted, authenticated replay-resistant manner as performed by initiator 1604. If recipient 1606 belongs to more then one workgroup, recipient 1606 selects the appropriate WA based, at least in part, on the workgroup-key ID supplied in request message 1610. If initiator 1604 and recipient 1606 retrieve the same WK from WA 1602, then recipient 1606 may send successful connection message 1614 to initiator 1604.

[0143] One or both of the key update steps shown in connection setup 1600 may be performed upon initial connection with another workgroup client, periodically after a workgroup key refresh alarm time has been reached, or periodically on any other suitable schedule. In some embodiments, as described in more detail below, key updates are performed automatically before every new communication session is established. System or user control of the workgroup key TTL and refresh alarm may provide customized levels of renewability. This may be beneficial, for example, after a client's communication privileges have been revoked and immediate revocability is desired.

[0144] The need for certificate revocation lists (CRLs) may also be reduced or even altogether eliminated using this approach. Workgroup membership may be maintained by the authentication or LDAP server. In some embodiments, WA revocation is also supported using the Online Certificate Status Protocol (OCSP). For example, an OSCP responder may determine the status of a WA's certificate without checking CRL published by the CA.

[0145] In practice, one or more steps shown in connection setup 1700 may be combined with other steps, performed in any suitable order, performed in parallel (*e.g.*, simultaneously or substantially simultaneously), or removed.

[0146] FIGURES 17-20 show illustrative processes for supporting the secure workgroup key management and distribution of the present invention. FIGURE 17 shows illustrative process 1700 for joining a new workgroup. At step 1702, client *n* may indicate a desire to join a workgroup. At step 1704, the WA may generate a secret key E_n for the client. In some embodiments, E_n may be a key for a symmetric cipher. In other embodiments, a public-key encryption scheme using an asymmetric cipher may be used and E_n may be a key (*e.g.*, a private key) for an asymmetric cipher. For example, the Naor, Naor and Lotspeich subset cover broadcast encryption approach may be used in place of the symmetric scheme described below. This public-key encryption algorithm may be built on top of any other secure public-key encryption scheme.

[0147] Key generation using the public-key approach, in some embodiments, may be performed centrally by the WA. If M is the maximum workgroup size, then the WA may generate $2M - 1$ distinct public/private keypairs. If a binary tree exists of span M, with each keypair associated with an internal node in the tree, then for each device or client in the workgroup, the i^{th} leaf in the tree could be chosen and the device or client i could be provided with each keypair associated with the i^{th} leaf upward to the root of the tree. The total size of the device or client's key material may then be equal to $\log M + 1$. Other approaches for key generation using a public-key or secret key approach may be used in other embodiments.

[0148] At step 1706, the WA may store one or more of the identity of the client *n*, the client's secret key E_n, and a list of authorized workgroup-key IDs. At step 1708, client *n* may store its secret key E_n. For example, the client may store its secret key locally in a key store. At step 1710, client *n* may also store the workgroup-key ID (*e.g.*, public knowledge and/or string identifier) and set the workgroup key refresh alarm to 0 (*i.e.*, immediate) and key version to 0.

[0149] A single client may belong to one or more workgroups. Each workgroup (and hence current WK for that workgroup) may be associated with a unique

workgroup identifier. The unique workgroup identifiers and associated workgroup-key IDs may help the client and WA determine to which workgroup or workgroups the client currently belongs as well as determine which workgroup key to use to establish a communication session with another client who also belongs to one or more workgroups.

[0150] In practice, one or more steps shown in process 1700 may be combined with other steps, performed in any suitable order, performed in parallel (*e.g.*, simultaneously or substantially simultaneously), or removed.

[0151] FIGURES 18A, 18B, and 18C show illustrative process 1800 for initiating a connection between two clients. At step 1802, client 1 may initiate a communication session with client 2. For example, client 1 may send its identity (*e.g.*, network address or username) to client 2 and request a communication session with client 2. At step 1804, client 1 may check its workgroup key refresh alarm time. At step 1804, client 1 may also verify that any time to live (TTL) value associated with client 1's WK has not expired. For example, in order to determine if client 1's WK is expired, client 1 may compare the TTL value associated with client 1's WK to the current time or the WK timestamp. The WK timestamp may indicate when the WK was generated or delivered to client 1. If, at step 1806, the refresh alarm time is greater than the current time (or WK timestamp), illustrative process 1800 may continue in FIGURE 18B. If, at step 1806, the refresh alarm time is not greater than the current time (or WK timestamp), at step 1808, client 1 may contact client 2 with the secure data parser's generated headers. These headers may include, for example, the WK and WK version number. At step 1810, client 2 may read the workgroup-key ID. If, at step 1812, the workgroup-key ID is not valid for this workgroup (or a workgroup to which client 2 currently belongs), client 2 may report the error to the calling application at step 1814. In some embodiments, the calling application may determine the response to the signaled error. If, at step 1812, the workgroup-key ID is valid for this workgroup (or a workgroup to which client 2 currently belongs), illustrative process 1800 may continue in FIGURE 18C.

[0152] FIGURE 18B continues illustrative process 1800 after it is determined, at step 1806, that the refresh alarm time is greater than the current time (or WK

timestamp). In some embodiments, this may indicate a stale WK. In order for client 1 to continue, in some embodiments, it must contact the WA and refresh its WK. At step 1816, client 1 may generate a nonce N_1. At step 1816, client 1 may also send a message to the WA. In some embodiments, the message may be formatted as follows: {Client 1 || E_1{UP-TO-DATE? || N_1 || workgroup-key ID || client workgroup key version}}, where Client 1 is the identity of client 1, E_1 is a cryptographic function based, at least in part, on client 1's secret key, and UP-TO-DATE? is the message type.

[0153] At step 1818, the WA may look up client 1 and access E_1, client 1's secret key. As previous described, the WA may store each workgroup member's secret key in a key store, key repository, or database. At step 1818, the WA may also decrypt the message sent by client 1 at step 1816. At step 1820, the WA may determine if client 1 is authorized by validating that the received workgroup-key ID is authorized for this client. The WA may also check to see if there is a newer version of the WK and calculate how long until the WK expires (*e.g.*, a "time to live" or TTL value for the WK may be computed). If, at step 1820, the WA determines that the client is not found or not authorized for this workgroup, the WA may send back a not authorized message at step 1822. Otherwise, at step 1824, the WA may send back a message encrypted with client 1's secret key. The message may be in the following format in some embodiments: E_1{UP-TO-DATE? || N_1 || workgroup-key ID || current workgroup key version || TTL}.

[0154] At step 1826, client 1 may decrypt the message and compare is WK version with the current WK version. If client 1's WK version is earlier that the current WK version, client 1 may update its WK at step 1830. An illustrative process a client may use to update its WK is described with regard to FIGURE 20 below. If client 1's WK version is not earlier than the current WK version (*e.g.*, it is the current version), then client 1 may update its WK refresh alarm to the current time plus the TTL calculated by the WA at step 1828.

[0155] Referring now to FIGURE 18C, if the workgroup-key ID is valid for the particular workgroup, client 2 may then determine if the version number of client 1's WK is less than client 2's current version number at step 1832. If so, client 2 may report an error to the calling application at step 1834. In some embodiments,

the calling application may determine the response to the signaled error (*e.g.*, the calling application may automatically perform a key update to obtain a newer version of the WK and retry the connection). If, at step 1832, client 2 determines that client 1's WK version number is not greater than client 2's version, client 2 may then determine if the current time is greater than the WK refresh alarm time at step 1836. If the current time is greater than the refresh alarm time, client 2 may contact the WA using the UP-TO-DATE? (described above) and/or KEY-REQUEST (described in regard to FIGURE 20 below) message types to verify the freshness of the current workgroup key, request a new workgroup key, or both. It should be noted that the UP-TO-DATE? message type may be an optimization for checking that a workgroup key is fresh. Nevertheless, the KEY-REQUEST message type can be used without a preceding UP-TO-DATE? message type in order to reduce latency. In other embodiments, the WA may force clients to check the freshness of keys on every connection attempt by setting the TTL to 0.

[0156] After updating its key at step 1838 (or if the current time was not greater than the refresh alarm time at step 1836), client 2 may then determine, at step 1840, if the version number of client 1's WK is not equal to client 2's now up-to-date version number. If the two version numbers are not the same, client 2 may report an error to the calling application at step 1842. If the workgroup-key ID is valid and the version numbers are identical, client 2 may report a successful connection attempt to the calling application at step 1844.

[0157] In practice, one or more steps shown in process 1800 may be combined with other steps, performed in any suitable order, performed in parallel (*e.g.*, simultaneously or substantially simultaneously), or removed.

[0158] FIGURE 19 shows illustrative process 1900 for revoking the communication privileges of a workgroup client. As previously described, traditional secure workgroup communication protocols may require the use of CRLs and may not support immediate revocability and renewability of the workgroup security mechanism. The present invention, however, may support the immediate discovery of revoked clients by renewing the security mechanism periodically (or at every connection attempt). For example, to revoke a workgroup client, the WA may start sending a new WK to all non-revoked users. By using a

versioned WK, connections using a stale WK may be refused, depending on the security policy in force at the receiving client.

[0159] A workgroup client revocation request may be received at step 1902. For example, one or more users may be removed from a workgroup (or their communication privileges revoked). As previously described, workgroup membership may be maintained by an LDAP (*e.g.*, Microsoft® Active Directory) server in some embodiments. Certain users (*e.g.*, workgroup administrators) may be authorized to remove specific users (or groups of users), machines (*e.g.*, computers and printers), services (*e.g.*, web services and print/file services), or any combination thereof from a workgroup using a workgroup administration interface. At step 1904, the server managing the workgroup membership may determine if the revocation request is valid. Valid revocation requests may be authenticated (*e.g.*, singed) by an authorized user or other entity (*e.g.*, WA or CA) and indicate a valid workgroup user, machine, or service to revoke. If the revocation request is not valid, an error may be returned to the calling application at step 1906 and the revocation request ignored.

[0160] If the revocation request is valid at step 1904, however, the revoked workgroup-key ID may be removed from the workgroup client's list at step 1908. For example, the WA may remove the client's workgroup-key ID from a workgroup-key ID table in a relational database stored at the WA or CA. At step 1910, the WA may determine if a key update should be triggered. In some embodiments, at step 1910, a key update is automatically triggered in response to each valid revocation request. In other embodiments, key updates may be triggered only after a predetermined number of valid revocation requests (as defined by the workgroup security policy), a predetermined number of currently outstanding revoked clients, periodically after a predetermined length of time since the last key update, on-demand by the WA or an authorized user, automatically after one or more users leave the workgroup, or at any combination of the foregoing times. Triggering a key update may implicitly deny access to all revoked users because the revoked users may not have access to the new WK. Communication using stale WKs may be controlled, limited, or completely

prohibited by customizing WK version numbers, TTL values computed by the WA, and WK refresh alarm times.

[0161] If a key update is not triggered at step 1910, illustrative process 1900 may return to step 1902 to await another revocation request. If a key update is triggered at step 1910, the WA may increment its current WK version number and generate a new WK at step 1912. To generate a new WK, in some embodiments, the WA may evaluate the workgroup name and current update number using its Master Key Generation Key (MKGK) through a secure Key Derivation Function (KDF). Other approaches for generating a new WK may be used in other embodiments. The newly generated WK may then be encrypted by the WA and a signed key update message may be generated at step 1914. In some embodiments, the key update message generated at step 1914 may include some or all of the following fields: (1) WA Identifier and Certificate Field, which may include identifying information for the WA authoring the update, including its CA-signed certificate; (2) Workgroup Identifier Field, which may include a unique identifier specifying the name of the workgroup to which this update applies; (3) Sequence Identifier Field, which may include a sequential counter indicating the number of the update; (4) Update Time Period Field, which may include an indication of the time period during which the key material will be valid; (5) Update Payload Field, which may include a ciphertext containing the updated key material; (6) Auxiliary Data Field, which may include additional information required by the client (*e.g.*, a key policy and the location where further updates may be found); (7) Re-keying Flag, which, when set, may instruct the client to regenerate its secret key and contact the WA for re-certification; and (8) Signature Filed, which may include a digital signature computed over the previous fields using the WA's signing key.

[0162] If a public-key scheme is used, the key update message may be encrypted to all of the non-revoked users by marking each leaf in the binary tree described above in regard to FIGURE 17 according to whether the device at that position has been revoked. Then, all nodes which are parents (or coexistent) to the leaves associated with non-revoked devices may be identified (but *not* revoked ones). For example, the root of the full tree is parent to all of the leaves; thus, this node is sufficient when no devices are revoked. Next, the message may be encrypted

under each of the keys associated with those nodes. The result may include a list identifying these nodes, as well as a list of the ciphertexts produced. When this encryption scheme is used, each non-revoked user may possess at least one of the keypairs that was used to encrypt the message. It may identify this keypair, decrypt the appropriate ciphertext, and output the message.

[0163] After the key update message has been generated, it may be published to all non-revoked users via a semi-trusted channel (*e.g.*, website) at step 1916. Untethered (and semi-tethered) operation may also be supported. When a client or device is prepared to go offline, the WA may pre-provision the client or device with key updates for a limited number of future time periods. Additionally or alternatively, for devices with some network access, key revocation and key update messages may be posted to a highly-available location, *e.g.*, a TLS-secured public website.

[0164] In practice, one or more steps shown in process 1900 may be combined with other steps, performed in any suitable order, performed in parallel (*e.g.*, simultaneously or substantially simultaneously), or removed.

[0165] FIGURE 20 shows illustrative process 2000 for requesting a WK update. At step 2002, a client may send the WA a key request message encrypted with the client's secret key (or private key, if an asymmetric encryption scheme is used). For example, at step 2002, the client may send a message in the following format to the WA: {Client X $\|$ E_X{KEY-REQUEST $\|$ N_2 $\|$ workgroup-key ID $\|$ requested key version}}, where X is the identity of the requesting client, E_X is client X's secret key (or private key, if an asymmetric encryption scheme is used), KEY-REQUEST represents the key request message type, and N_2 is a nonce generated by the requesting client. At step 2004, the WA may receive the encrypted message and look up the client to obtain the client's secret encryption key E_X. At step 2004, the WA may also decrypt the key request message and, at step 2006, verify that the workgroup-key ID is valid for this client. If the workgroup-key ID is not valid for this client, at step 2008 the WA may send back a not authorized message and illustrative process 2000 may return to step 2002. If the workgroup-key ID is valid for this client, however, the WA may access the newest (*i.e.*, most current) version of the key (it may be newer than the version

requested). The WA may then create a response to the key request message at step 2012 and encrypt the response using the client's secret key E_X. In some embodiments, the WA's response to the key request message may take the following format: E_X\{KEY-REQUEST || N_2 || workgroup-key ID || current key version || TTL || key\}. After receiving the response from the WA, the client may update its WK at step 2014. Finally, at step 2016, the client may update its WK refresh alarm time to the current time plus the received TTL value.

[0166] In practice, one or more steps shown in process 2000 may be combined with other steps, performed in any suitable order, performed in parallel (*e.g.*, simultaneously or substantially simultaneously), or removed.

[0167] A number of key distribution systems and protocols (*e.g.*, the Kerberos protocol) have been developed that support key distribution in a reasonably efficient manner. However, these prior systems and protocols fail to meet the requirements for secure workgroup key distribution and management for use in securing data in motion in conjunction with the secure data parser of the present invention. Some of these requirements may include, for example: (1) Identification Unavailable For Connection – The system should support communication between two clients when one or both do not have the full identity of the other. An immediate consequence of this requirement may be to exclude per-peer session keys; (2) Revocation – The system should be able to revoke the communication privileges of any client. The time required for the revocation to be completed should be user-configurable. In some embodiments, revocation may only affect new connections, but not existing ones; (3) Parameterized Network Chatter – This system should support a centralized authority that can validate and distribute keys. However, communication with this server may allow for a distributed key to be usable or valid only for some user-configurable or predetermined period of time; (4) Minimal Participant State – Clients may only need to store a small amount of information to communicate. In particular, the state stored by a client should not include per-peer information. All of the above requirements may be met using the systems and methods described herein.

[0168] In some embodiments, the following client functions may be used to support the processes and systems of the present invention at a workgroup client.

The functions described below may be implemented using software, firmware, hardware (*e.g.*, embedded systems), or any combination of the foregoing. Arguments in the function below may be indicated except for "helper" arguments such as the length field to a buffer. Also, only some notable error codes are mentioned. In other embodiments, additional error codes, arguments, or both, may be used.

[0169] 1. Function wa_init – This function may return type ErrorCode, take clientId as input, and output type ctx. This function may be used to initialize the client-side key distribution service. The ctx variable may be initialized and hold the state on all the keys currently in use, their versions, and other key-specific information. This function may also set up an event thread for handling asynchronous communications. The clientId should be the same identifier given to the WA.

[0170] 2. Function wa_newKeyId – This function may return type ErrorCode, take types ctx, wgKeyId, waAddress, and privateKey as inputs, and have no output. This function may be used to add a workgroup key ID to the ctx context for managing. waAddress may be the network address for the WA that is responsible for this particular ID. privateKey may be used for encrypting communication with the WA and should be the same key provided to the WA.

[0171] 3. Function wa_teardown – This function may return type ErrorCode, take type ctx as input, and have no output. This function may be used to tear down all of the states managed by the service and deallocate the memory for the context ctx. This function may also cancel any pending requests (*e.g.*, tickets).

[0172] 4. Fucntion wa_keyUpToDate_async – This function may return type ErrorCode, take types ctx, wgKeyId, and callback as inputs, and output type Ticket t. This function may be used to check that the workgroup key identified by wgKeyId is up-to-date. There may be no need to specify the version number in some embodiments because the version number may be handled internally by the managing context. This call may be asynchronous and may return immediately. When the event loop receives the response from the server, the event loop may call the callback function. It may then pass to this callback function an error code as well as a status code, although the status code may be meaningful if there was no

error. If the client is not authorized to communicate with this workgroup any longer, callback may receive ERROR_WORKGROUP_AUTHORIZATION. If there is no error and the key is still valid, it may receive STATUS_KEY_OK. Otherwise, it may receive STATUS_KEY_STALE. When this function terminates, the ticket t may serve as an identifier for the waiting request listener.

[0173] 5. Function wa_keyRequest_async – This function may return type ErrorCode, take types ctx, wgKeyId, and callback as inputs and return type Ticket t. This function may be used to refresh the key material and version for this key identifier. A call to this function may be asynchronous and may return immediately. When the event loop receives the response, the event loop may call the callback function. It may pass to this callback function an error code indicating whether the key update was successful or not. If the client is not authorized to communicate with this workgroup any longer, callback may receive ERROR_WORKGROUP_AUTHORIZATION. When this function terminates, the ticket t may serve as an identifier for the waiting request listener.

[0174] 6. Function wa_keyValidate_async – This function may return type ErrorCode, take ctx, wgKeyId, wgKeyIDVersion, and callback as inputs, and output type Ticket t. This function may be used to validate a key received from another party. This function may update its own key (as per wa_keyRequest_async) as part of the operation. This function may also be asynchronous and may return immediately. When the event loop receives the response, the event loop may call the callback function. It may pass to this callback function an error code indicating whether the key check was successful and a status code indicating the validity of the key. The three possible status codes may include, for example, STATUS_KEY_OK, STATUS_KEY_STALE, and STATUS_KEY_INVALID. If the workgroup key is not one currently managed by the context ctx, this function may return ERROR_UNKNOWN_KEY. This error code may also be passed to the callback function as an optimization.

[0175] 7. Function wa_currentVersion – This function may return type ErrorCode, take ctx and wgKeyId as inputs, and output version. This function may be used to return the current version as managed by the context ctx for the given workgroup-key ID.

[0176] 8. Function wa_nextRefresh – This function may return type ErrorCode, take ctx and wgKeyId as inputs and output timeLeft. This function may be used to return the number of seconds remaining before the key should be refreshed. This function may return 0 if the next refresh time has already expired.

[0177] 9. Function wa_status – This function may return type, take Ticket t as input, and output type status. This function may be used to provide, for example, STATUS_COMPLETE or STATUS_WAITING for the given ticket to status. If the ticket is unknown, this function may return ERROR_UNKNOWN_TICKET.

[0178] 10. Function wa_timeWaiting – This function may return type ErrorCode, take Ticket t as input, and output seconds. This function may be used to provide the number of seconds that the request has been pending to seconds.

[0179] 11. Function wa_cancel – This function may return type ErrorCode, take Ticket t as input , and have no output. This function may be used to cancel a pending request.

[0180] The above functions are merely illustrative. Many other functions may be defined at a workgroup client for use in distributing and managing workgroup keys in accordance with the present invention. In addition, although some applications of the secure data parser and workgroup management module are described above, it should be clearly understood that the present invention may be integrated with any network application in order to increase security, fault-tolerance, anonymity, or any suitable combination of the foregoing.

[0181] Additionally, other combinations, additions, substitutions and modifications will be apparent to the skilled artisan in view of the disclosure herein. For example, although many of the processes illustrated in the figures refer to a client's secret key and/or symmetric key cryptography, a skilled artisan will recognize that a public/private keypair could alternatively be used in a corresponding asymmetric cryptographic scheme.

What is Claimed is:

1. A method for secure workgroup communication, the method comprising:

receiving, at a first workgroup client, an encrypted message from a workgroup key server, wherein the encrypted message comprises a workgroup key, a workgroup key version number, and a time to live (TTL) value for the workgroup key; and

initiating a communication session with a second workgroup client, wherein initiating the communication session comprises:

determining, at the first workgroup client, if the workgroup key is expired based, at least in part, on the TTL value for the workgroup key;

in response to determining that the workgroup key is expired, checking the availability of a new workgroup key from the key server;

in response to determining that the workgroup key is not expired, sending, to the second workgroup client, a plurality of share headers, wherein the share headers include the workgroup key and the workgroup key version number; and

verifying, at the second workgroup client, that the second workgroup client's workgroup key version matches the first workgroup client's workgroup key version.

2. The method of claim 1 further comprising generating the plurality of share headers before sending the plurality of share headers, wherein generating the plurality of share headers comprises distributing the workgroup key into the plurality of share headers using a secret sharing scheme.

3. The method of claim 1 wherein the encrypted message is encrypted under a symmetric cipher using the first workgroup client's secret key.

4. The method of claim 1 wherein the encrypted message is encrypted under an asymmetric cipher using the first workgroup client's public key.

5. The method of claim 1 wherein the share headers additionally include a workgroup identifier, the method further comprising verifying, at the second workgroup client, that the workgroup identifier is valid for a workgroup to which the second workgroup client belongs.

6. The method of claim 1 further comprising:

receiving, at the first workgroup client, a workgroup key update message, the workgroup key update message including a new workgroup key, a new workgroup key version number, and a new TTL value for the new workgroup key; and

updating, at the first workgroup client, a workgroup key refresh alarm based, at least in part, on the new TTL value.

7. The method of claim 6 wherein updating the workgroup key refresh alarm comprises setting the workgroup key refresh alarm to the current time plus the new TTL value.

8. A method for secure workgroup communication, the method comprising:

provisioning each client in a workgroup with a public key and a private key;

recording, at a key server, each client's public key;

generating a workgroup key update message, wherein the workgroup key update message includes a workgroup key, a workgroup key version number, and a time to live (TTL) value for the workgroup key;

encrypting the workgroup key update message; and

broadcasting the encrypted workgroup key update message to the workgroup.

9. The method of claim 8 wherein encrypting the workgroup key update message comprising encrypting the workgroup key update message using a public-key broadcast encryption scheme.

10. The method of claim 9 wherein encrypting the workgroup key update message using a public-key broadcast encryption scheme comprises:

generating a binary tree of span M, wherein M is the maximum size of the workgroup;

associating the public key of each client of the workgroup with a unique leaf of the binary tree;

identifying all nodes of the binary tree that are coexistent to or parents of leaves associated with non-revoked clients of the workgroup; and

encrypting the workgroup key update message under each of the public keys associated with the identified nodes.

11. The method of claim 8 wherein the generating and broadcasting are performed periodically on a predefined schedule.

12. The method of claim 8 wherein the generating and broadcasting are performed automatically in response to the communication privileges of a client of the workgroup being revoked.

13. The method of claim 8 wherein broadcasting the encrypted workgroup key update message to the workgroup comprises posting the key update message to a website.

14. A system for secure workgroup communication, the system comprising:

a first workgroup client configured to:

receive an encrypted message from a workgroup key server, wherein the encrypted message comprises a workgroup key, a workgroup key version number, and a time to live (TTL) value for the workgroup key; and

initiate a communication session with a second workgroup client, wherein the first workgroup client is configured to:

determine if the workgroup key is expired based, at least in part, on the TTL value for the workgroup key;

in response to determining that the workgroup key is expired, check the availability of a new workgroup key from the key server;

in response to determining that the workgroup key is not expired, send, to the second workgroup client, a plurality of share headers, wherein the share headers include the workgroup key and the workgroup key version number; and

a second workgroup client configured to verify that the second workgroup client's workgroup key version matches the first workgroup client's workgroup key version.

15. The system of claim 14 wherein the first workgroup client is configured to generate the plurality of share headers by distributing the workgroup key into the plurality of share headers using a secret sharing scheme.

16. The system of claim 14 wherein the encrypted message is encrypted under a symmetric cipher using the first workgroup client's secret key.

17. The system of claim 14 wherein the encrypted message is encrypted under an asymmetric cipher using the first workgroup client's public key.

18. The system of claim 14 wherein the share headers additionally include a workgroup identifier, the second workgroup client further configured to verify that the workgroup identifier is valid for a workgroup to which the second workgroup client belongs.

19. The system of claim 14 wherein the first workgroup client is configured to:

receive a workgroup key update message, the workgroup key update message including a new workgroup key, a new workgroup key version number, and a new TTL value for the new workgroup key; and

update a workgroup key refresh alarm based, at least in part, on the new TTL value.

20. The system of claim 19 wherein the first workgroup client is configured to update the workgroup key refresh alarm by setting the workgroup key refresh alarm to the current time plus the new TTL value.

21. A system for secure workgroup communication, the system comprising:

a workgroup key server configured to:

provision each client in a workgroup with a public key and a private key;

record each client's public key;

generate a workgroup key update message, wherein the workgroup key update message includes a workgroup key, a workgroup key version number, and a time to live (TTL) value for the workgroup key;

encrypt the workgroup key update message; and

broadcast the encrypted workgroup key update message to the workgroup.

22. The system of claim 21 wherein the workgroup key server is configured to encrypt the workgroup key update message using a public-key broadcast encryption scheme.

23. The system of claim 21 wherein the workgroup key server is configured to encrypt the workgroup key update message using a public-key broadcast encryption scheme by:

generating a binary tree of span M, wherein M is the maximum size of the workgroup;

associating the public key of each client of the workgroup with a unique leaf of the binary tree;

identifying all nodes of the binary tree that are coexistent to or parents of leaves associated with non-revoked clients of the workgroup; and

encrypting the workgroup key update message under each of the public keys associated with the identified nodes.

24. The system of claim 21 wherein the workgroup key server is configured to generate the workgroup key update message and broadcast the encrypted workgroup key update message periodically on a predefined schedule.

25. The system of claim 21 wherein the workgroup key server is configured to generate the workgroup key update message and broadcast the encrypted workgroup key update message automatically in response to the communication privileges of a client of the workgroup being revoked.

26. The system of claim 21 wherein the workgroup key server is configured to broadcast the encrypted workgroup key update message to the workgroup by posting the key update message to a website.

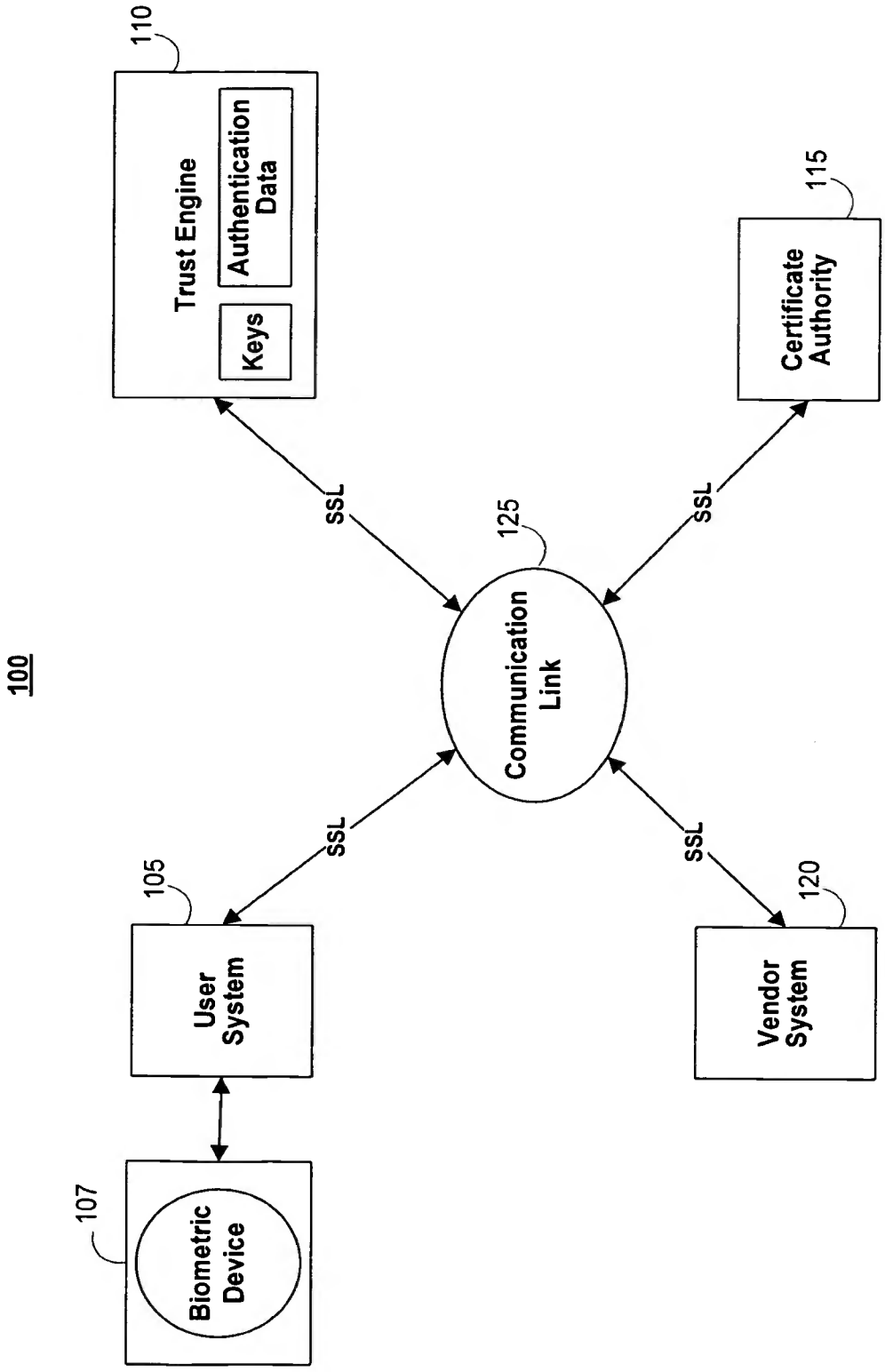

100
110
Trust Engine
Authentication Data
Keys
115
Certificate Authority
SSL
SSL
125
Communication Link
SSL
SSL
105
User System
120
Vendor System
107
Biometric Device

FIG. 1

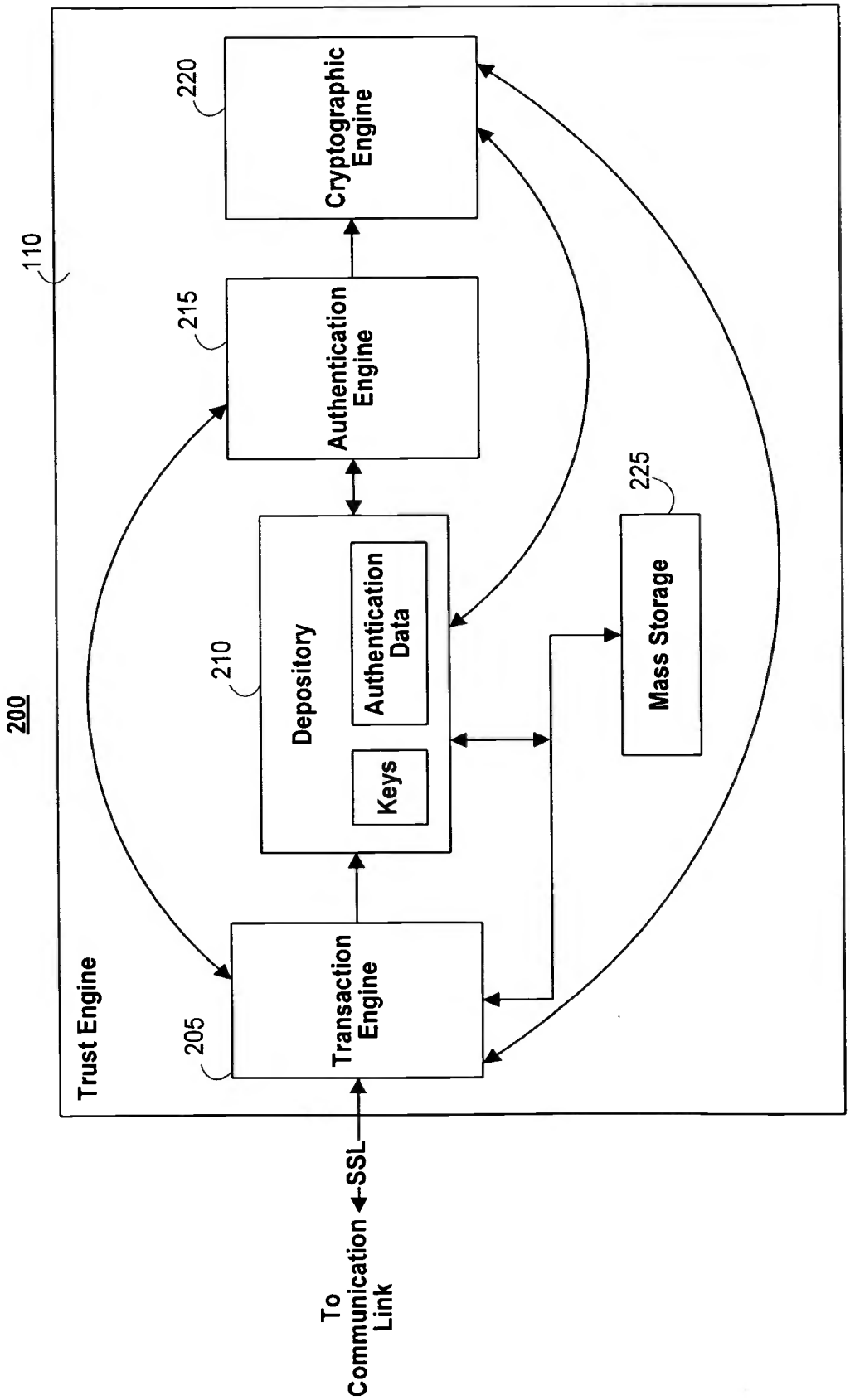

200
110
Trust Engine
205
Transaction Engine
210
Depository
Authentication Data
Keys
215
Authentication Engine
220
Cryptographic Engine
225
Mass Storage
SSL
To Communication Link

FIG. 2

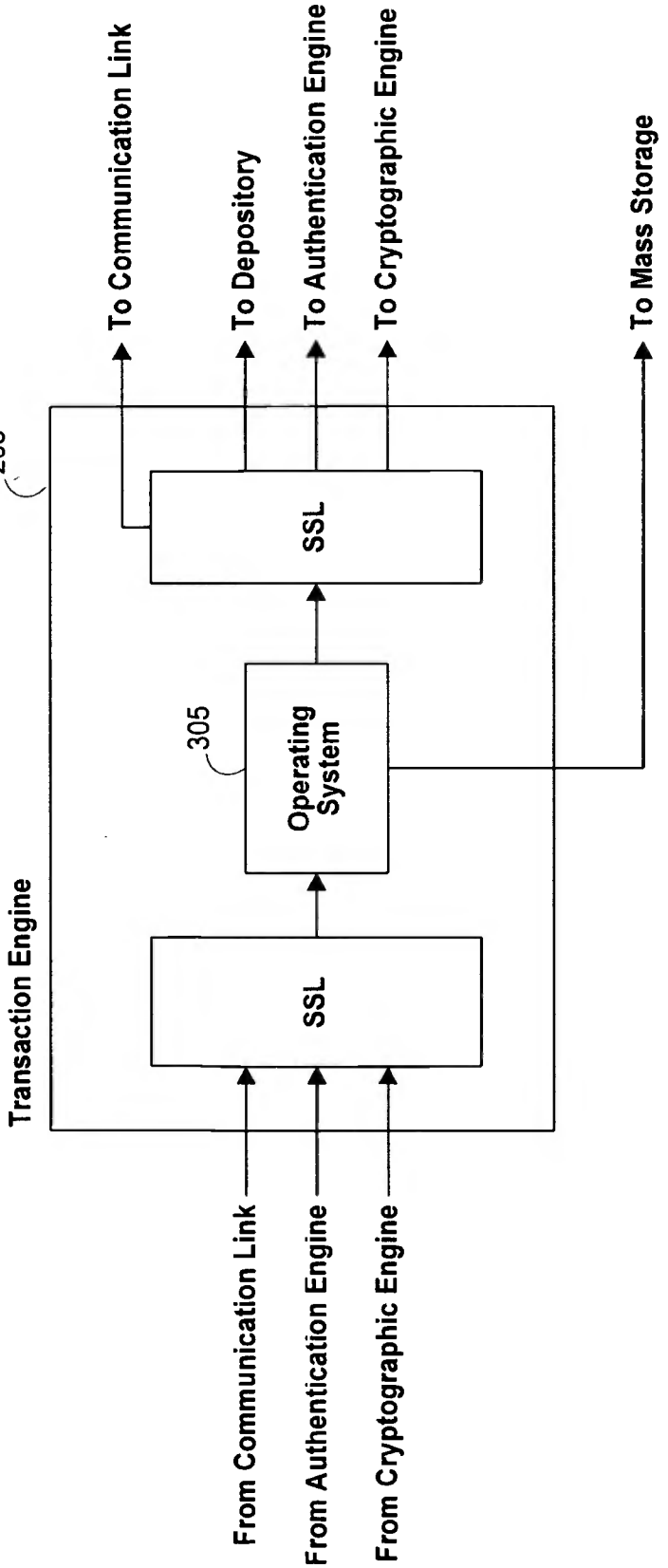

To Communication Link
To Depository
To Authentication Engine
To Cryptographic Engine
To Mass Storage
205
SSL
305
Operating System
SSL
Transaction Engine
From Communication Link
From Authentication Engine
From Cryptographic Engine

FIG. 3

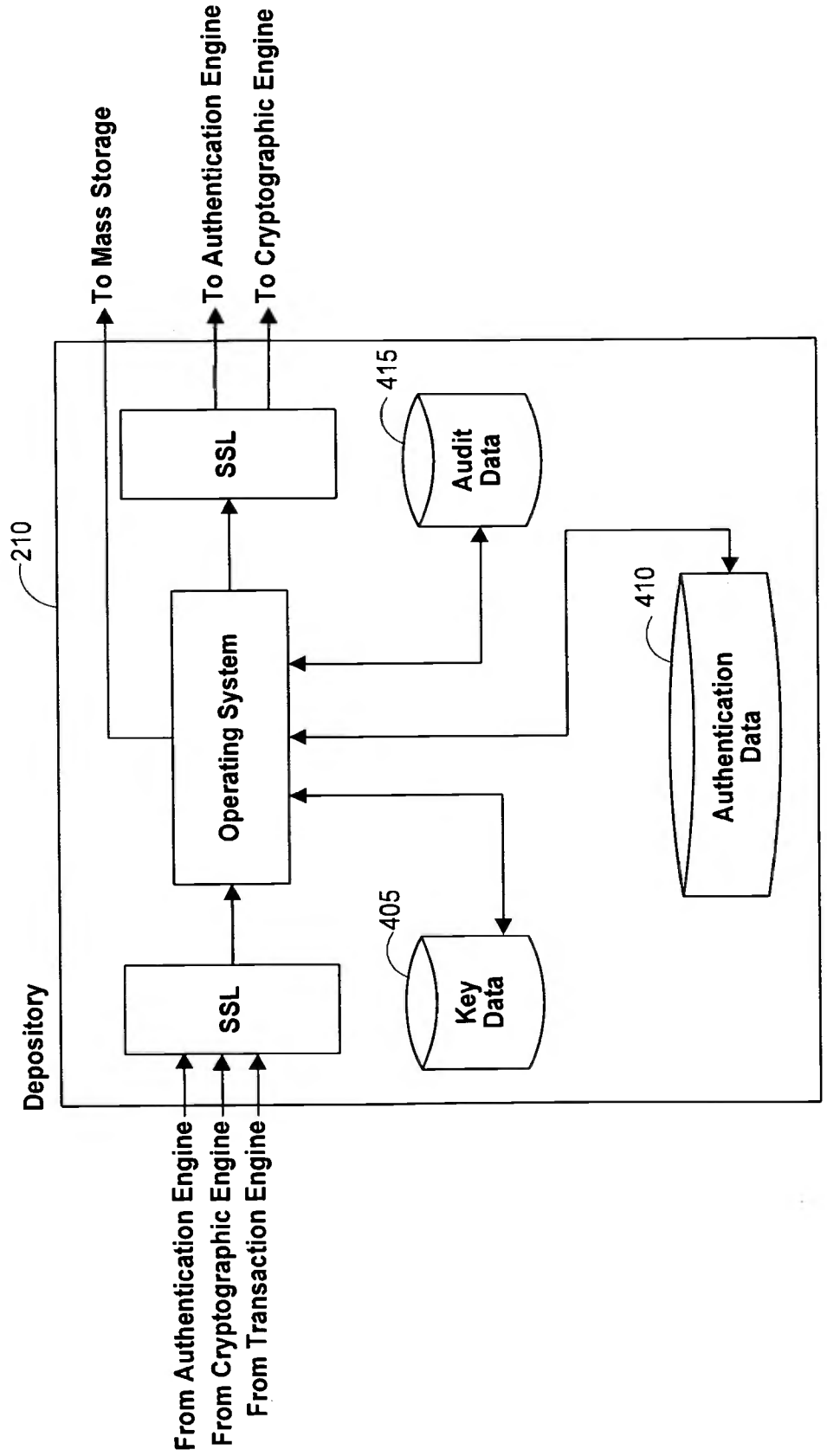

To Mass Storage
To Authentication Engine
To Cryptographic Engine
SSL
415
Audit Data
210
Operating System
410
Authentication Data
405
Key Data
SSL
Depository
From Authentication Engine
From Cryptographic Engine
From Transaction Engine

FIG. 4

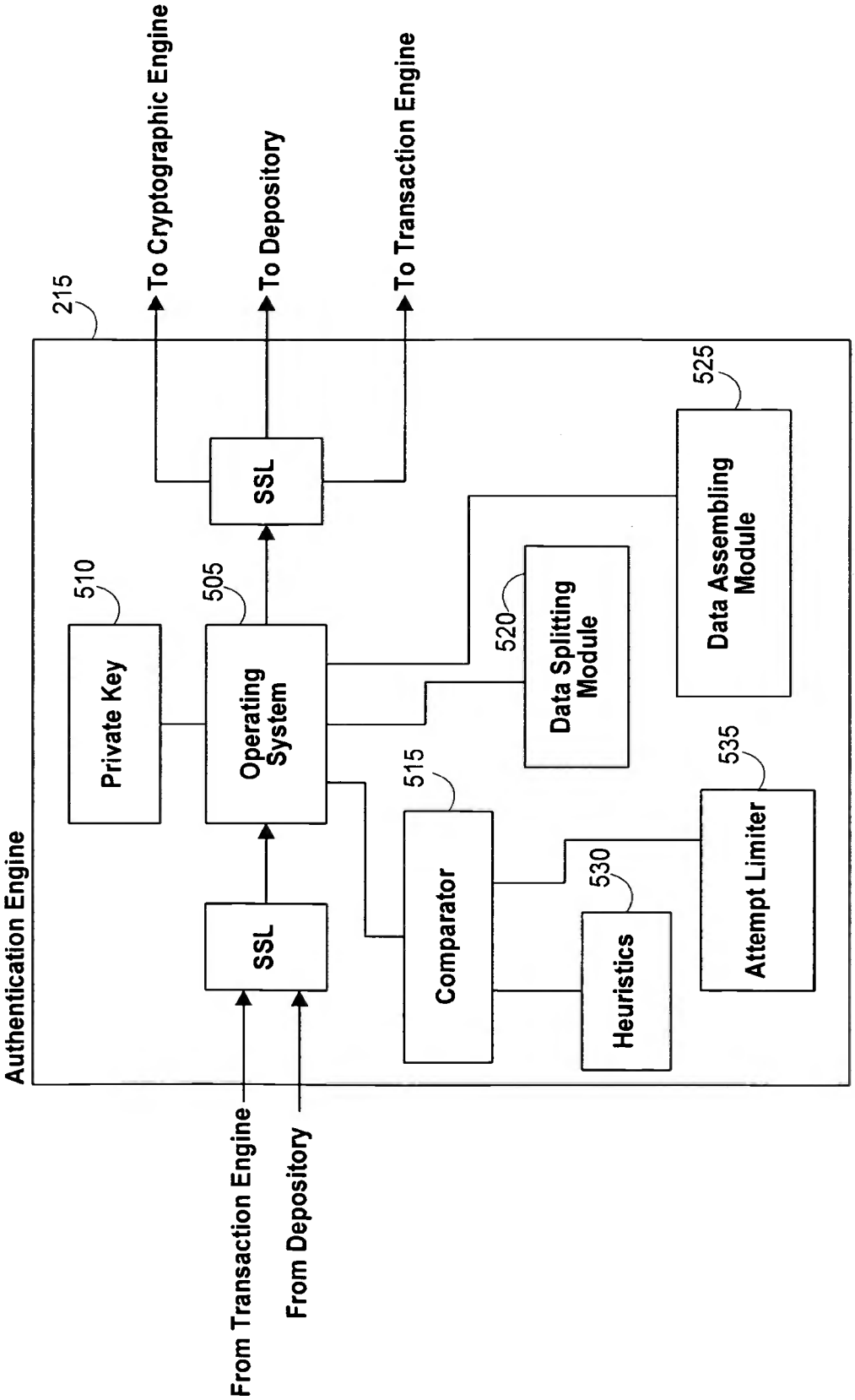

Authentication Engine
215
To Cryptographic Engine
To Depository
To Transaction Engine
SSL
SSL
Private Key
510
Operating System
505
Data Splitting Module
520
Data Assembling Module
525
Comparator
515
Heuristics
530
Attempt Limiter
535
From Transaction Engine
From Depository

FIG. 5

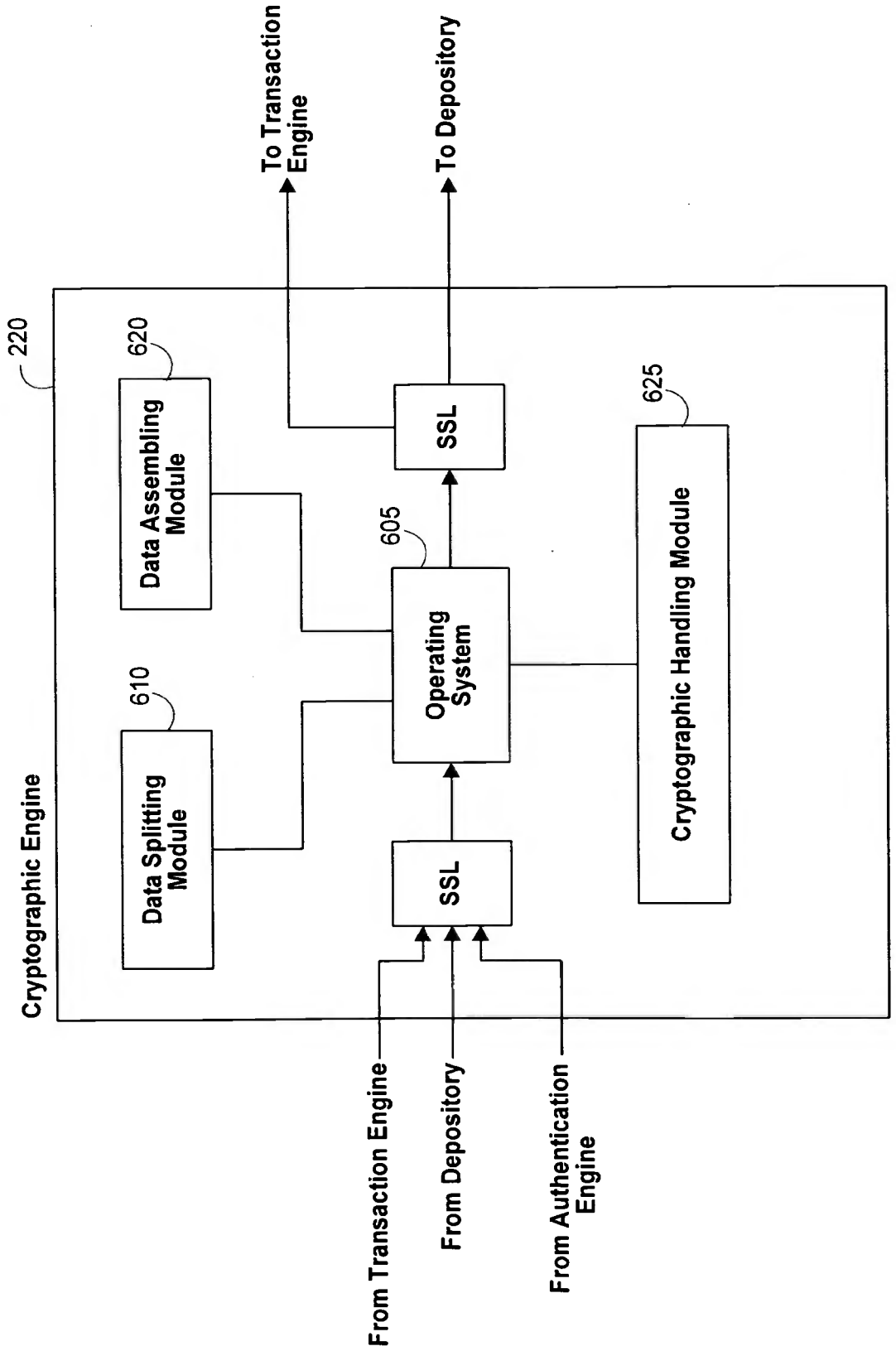

To Transaction Engine
To Depository
220
620
Data Assembling Module
SSL
625
Cryptographic Handling Module
605
Operating System
610
Data Splitting Module
SSL
Cryptographic Engine
From Transaction Engine
From Depository
From Authentication Engine

FIG. 6

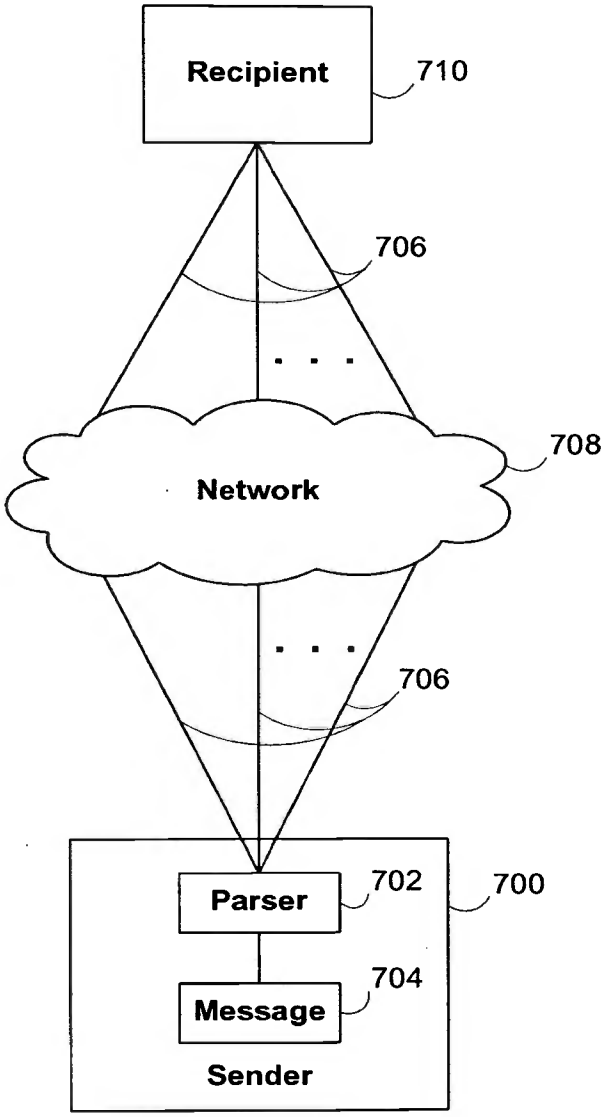

Recipient
710
706
Network
708
706
Parser
702
700
Message
704
Sender

FIG. 7

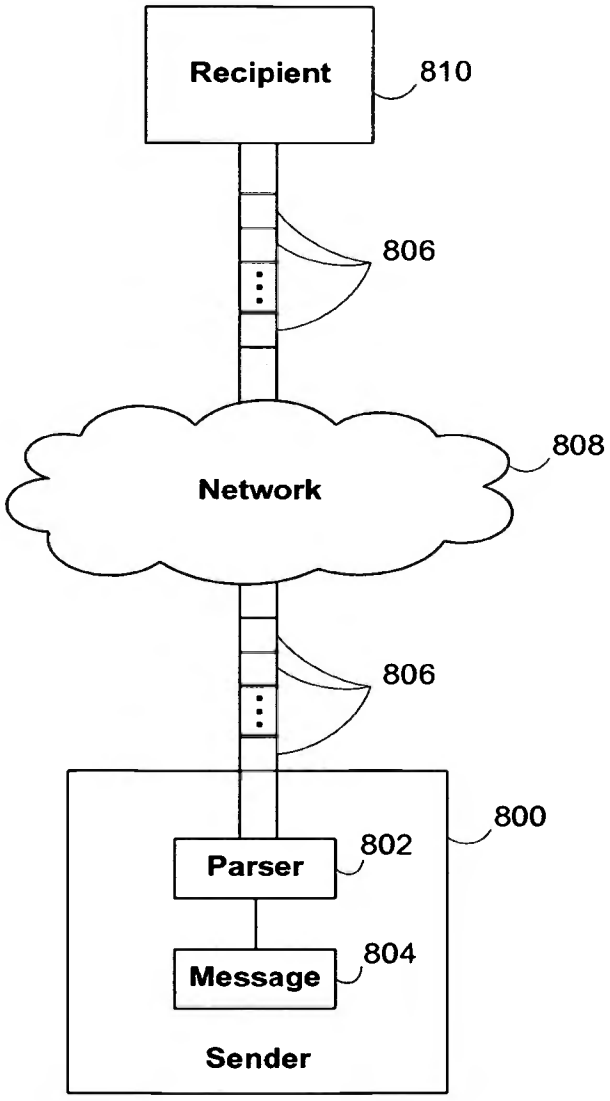

Recipient
810
806
808
Network
806
800
Parser
802
Message
804
Sender

FIG. 8

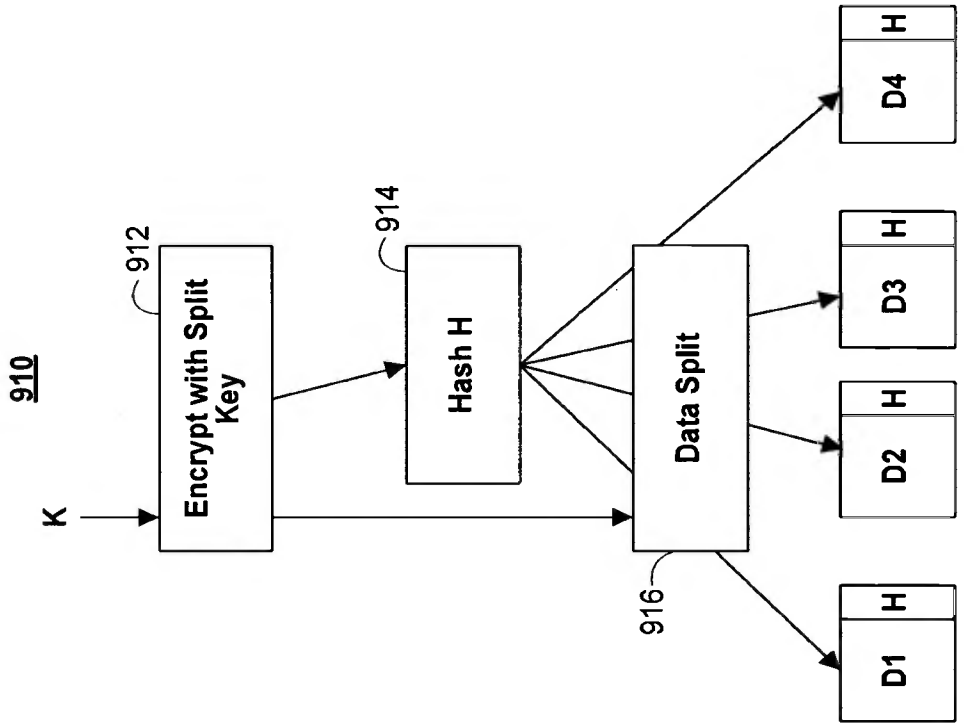

910
K
912
Encrypt with Split Key
914
Hash H
Data Split
916
D1
H
D2
H
D3
H
D4
H

FIG. 9B

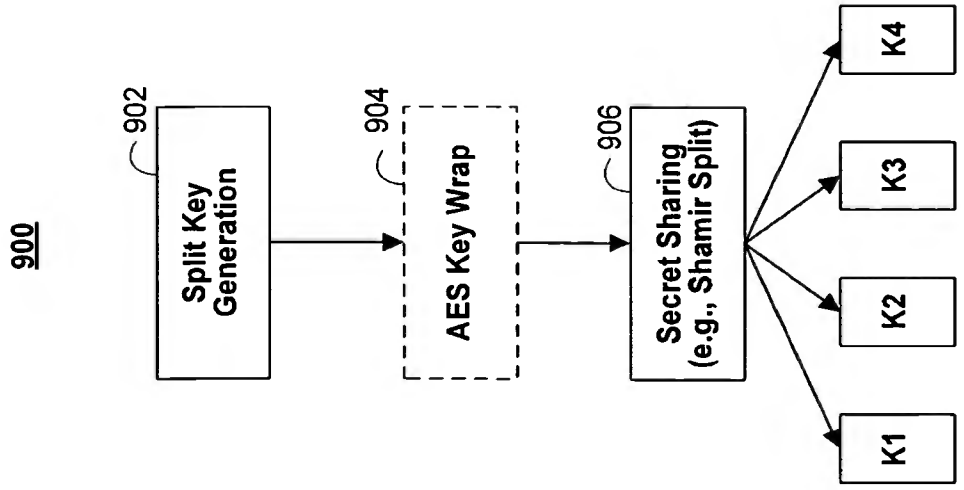

900
902
Split Key Generation
904
AES Key Wrap
906
Secret Sharing (e.g., Shamir Split)
K1
K2
K3
K4

FIG. 9A

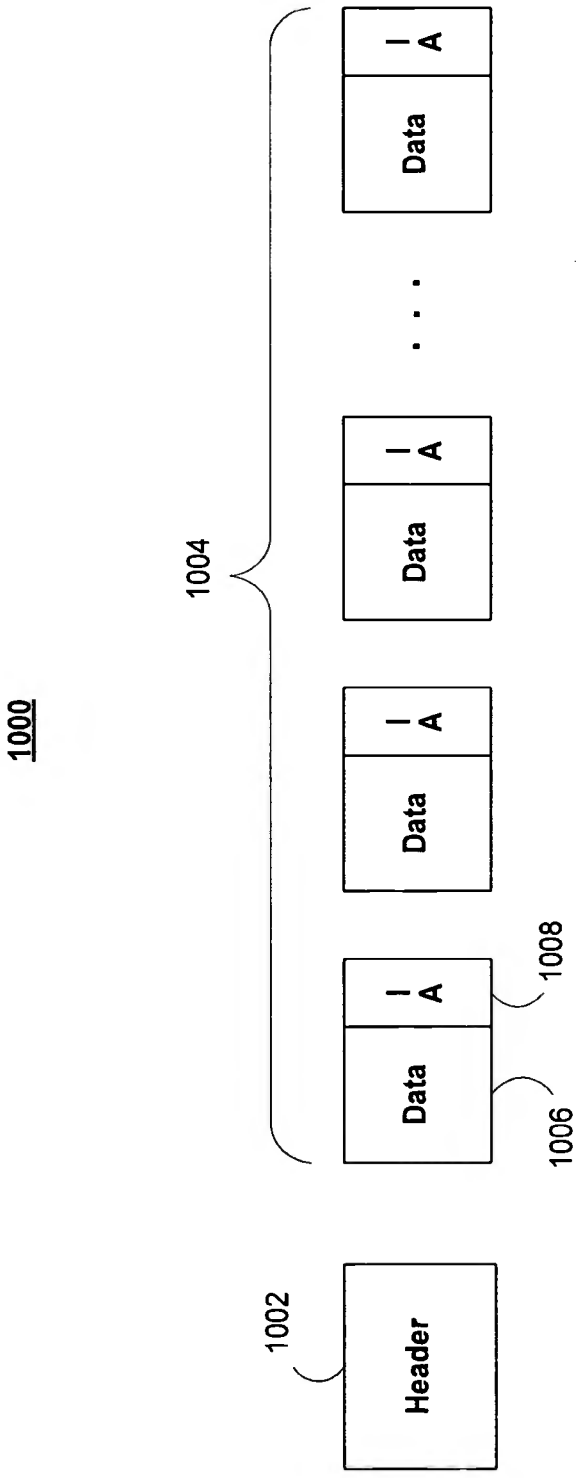

1000
1004
Header
1002
Data
I A
1006
1008
Data
I A
Data
I A
Data
I A
. . .
Data
I A

FIG. 10

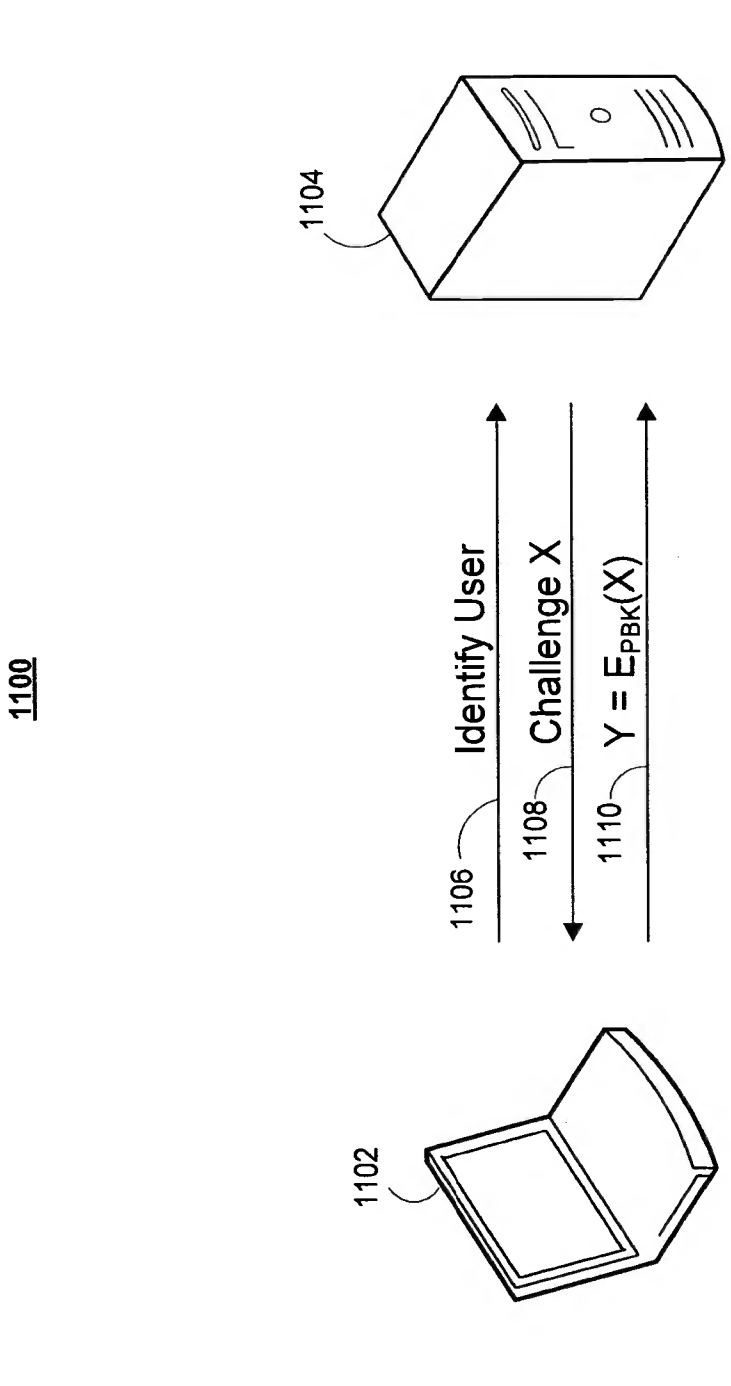

1100
1102
1104
1106
1108
1110
Identify User
Challenge X
Y = E_PBK(X)
PBK = Digest(password)
PBK = Look up (User)
Compute Z = E_PBK(X)
Compare Y == Z?

FIG. 11

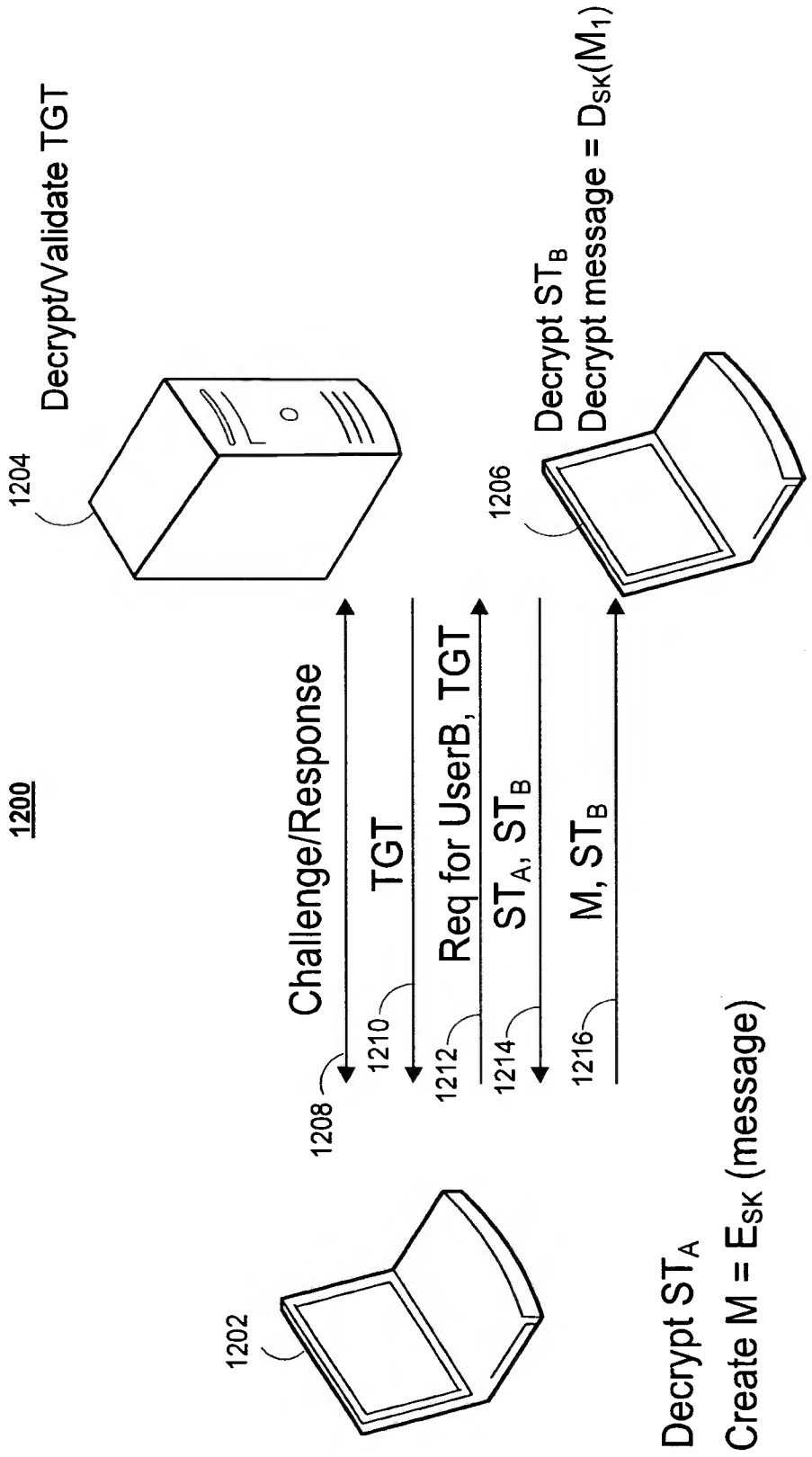

1200
1202
1204
1206
Decrypt/Validate TGT
Challenge/Response
TGT
Req for UserB, TGT
ST_A, ST_B
M, ST_B
Decrypt ST_B
Decrypt message = $D_{SK}(M_1)$
Decrypt ST_A
Create M = E_{SK} (message)
1208
1210
1212
1214
1216

FIG. 12

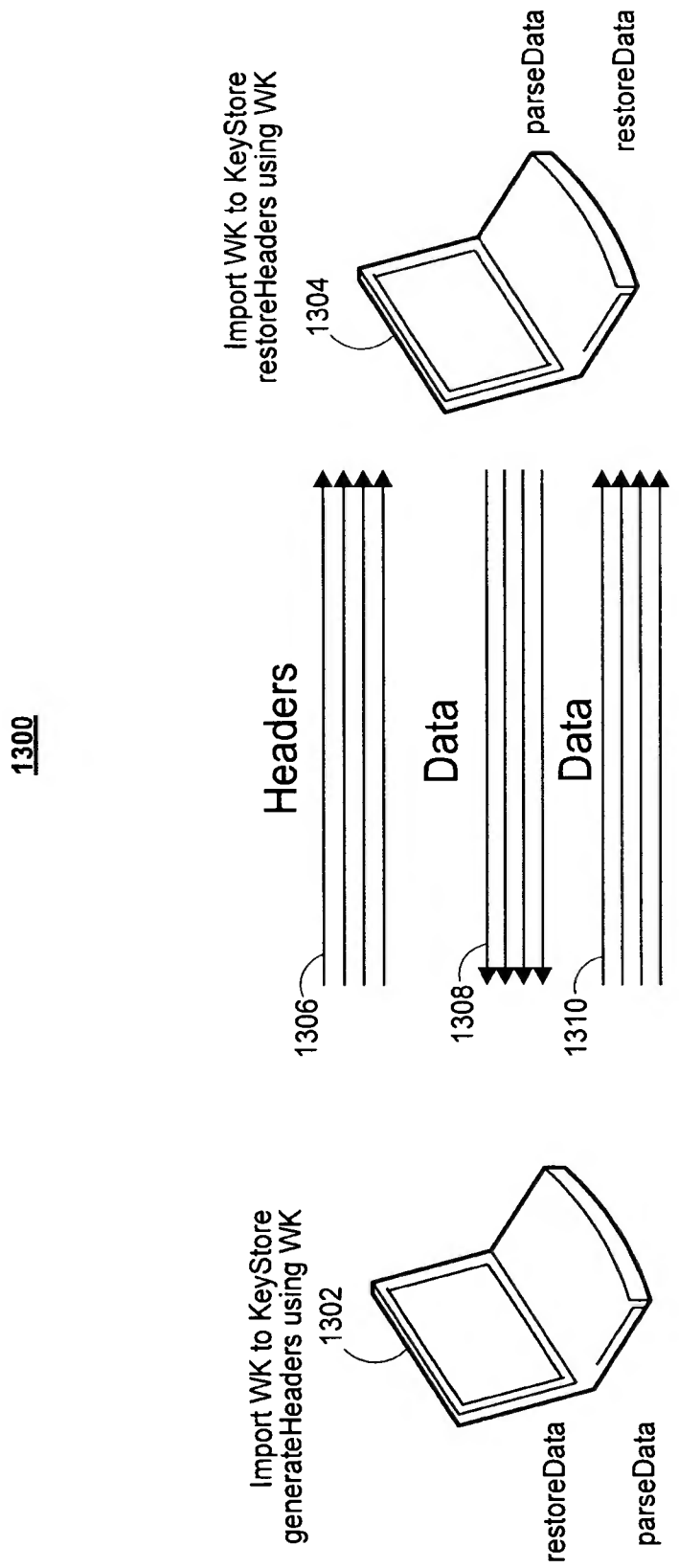

1300
Import WK to KeyStore
generateHeaders using WK
1302
restoreData
parseData
Headers
Data
Data
1306
1308
1310
Import WK to KeyStore
restoreHeaders using WK
1304
parseData
restoreData

FIG. 13

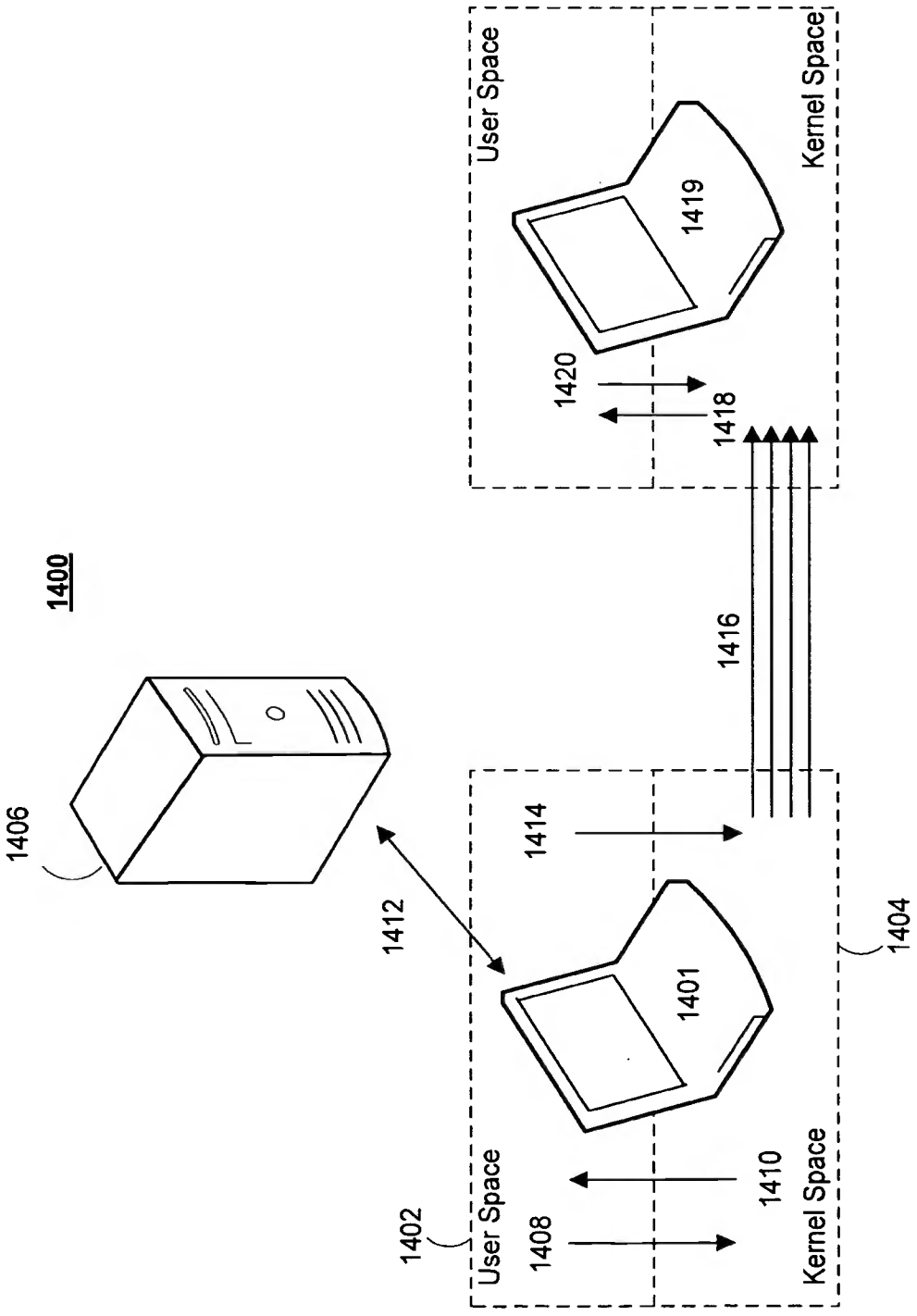

1400
1406
1412
1402
User Space
1408
1410
Kernel Space
1404
1401
1414
1416
1418
1420
User Space
1419
Kernel Space

FIG. 14

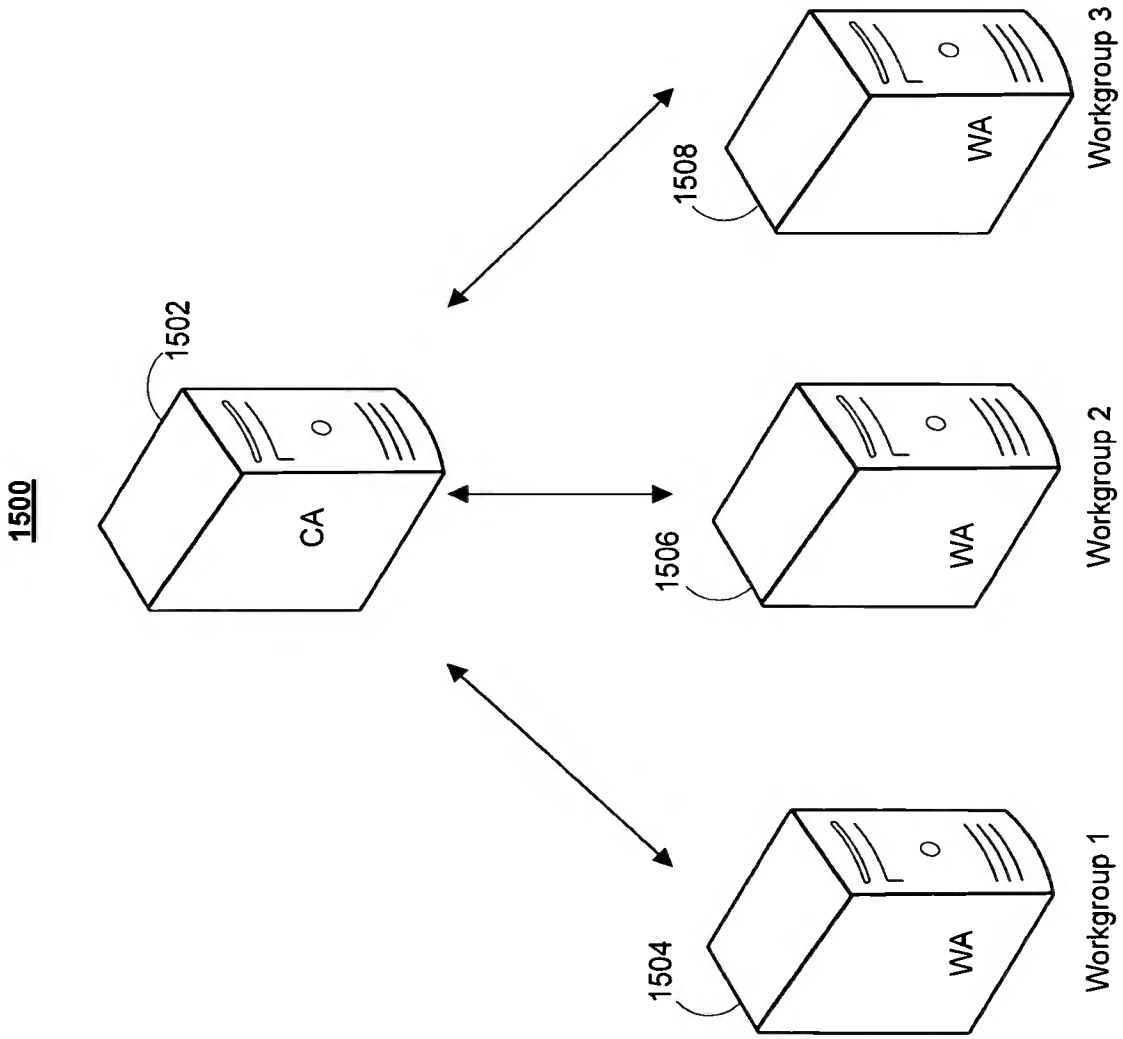

1500
1502
CA
1504
WA
Workgroup 1
1506
WA
Workgroup 2
1508
WA
Workgroup 3

FIG. 15

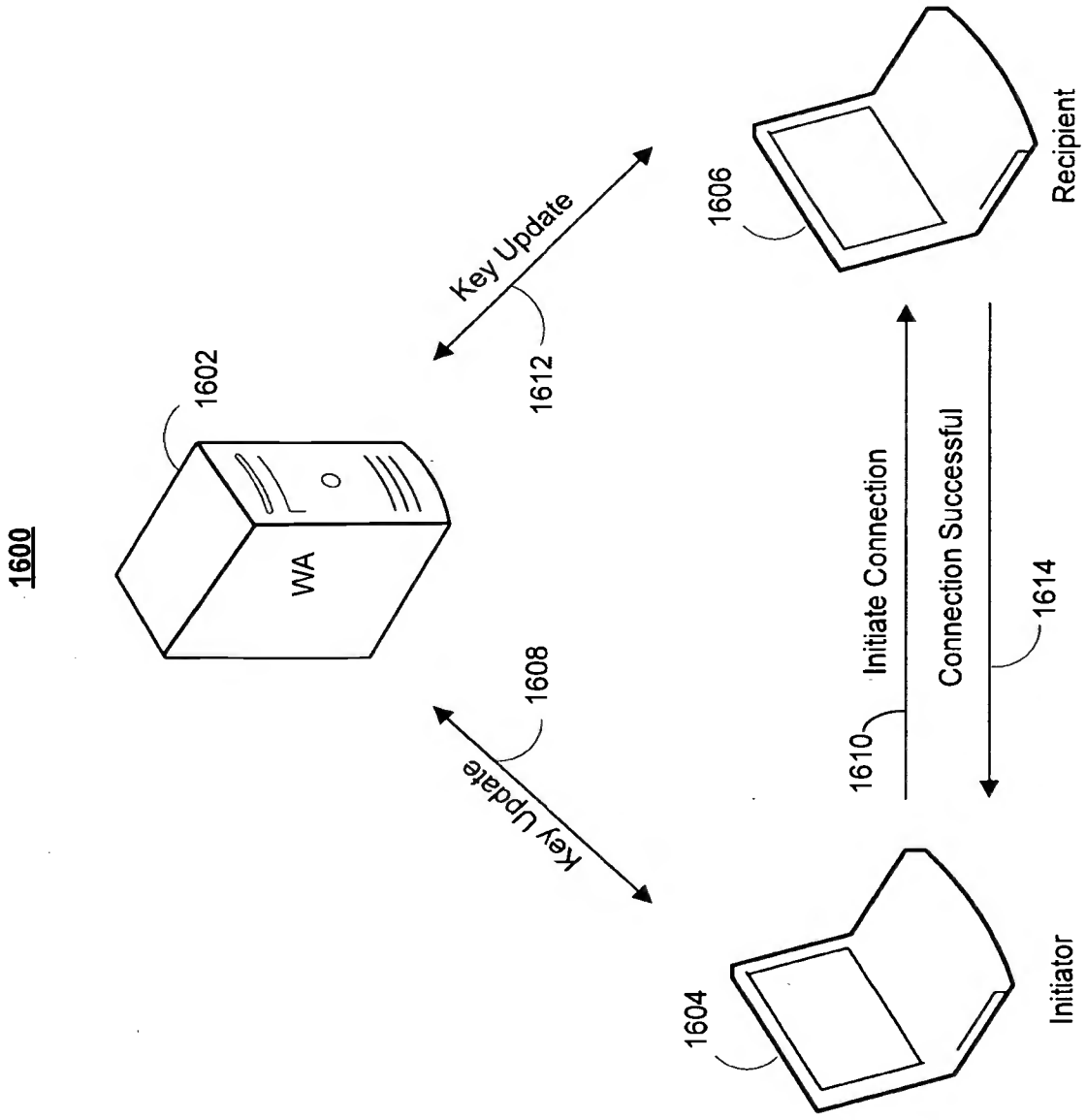

1600
1602
WA
Key Update
1612
1606
Recipient
Initiate Connection
Connection Successful
1610
1614
1608
Key Update
1604
Initiator

FIG. 16

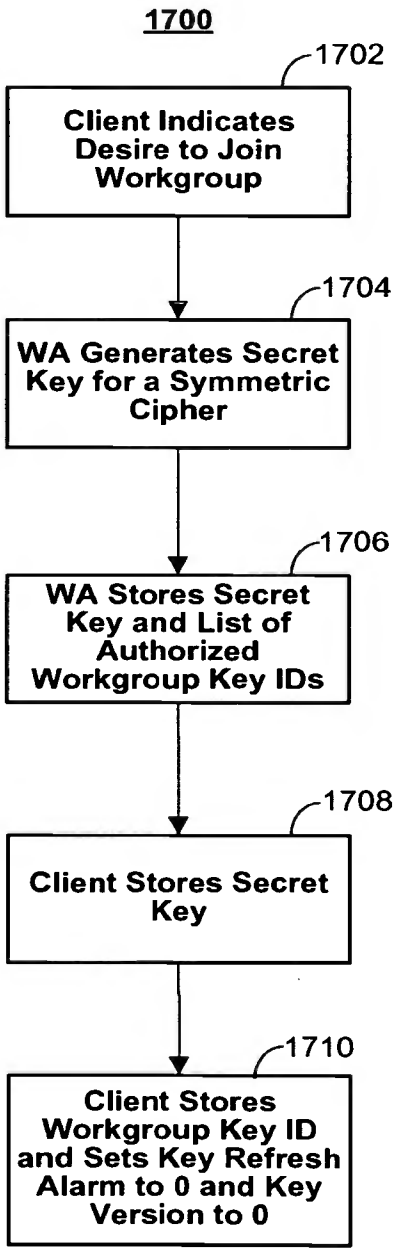

1700
1702
Client Indicates Desire to Join Workgroup
1704
WA Generates Secret Key for a Symmetric Cipher
1706
WA Stores Secret Key and List of Authorized Workgroup Key IDs
1708
Client Stores Secret Key
1710
Client Stores Workgroup Key ID and Sets Key Refresh Alarm to 0 and Key Version to 0

FIG. 17

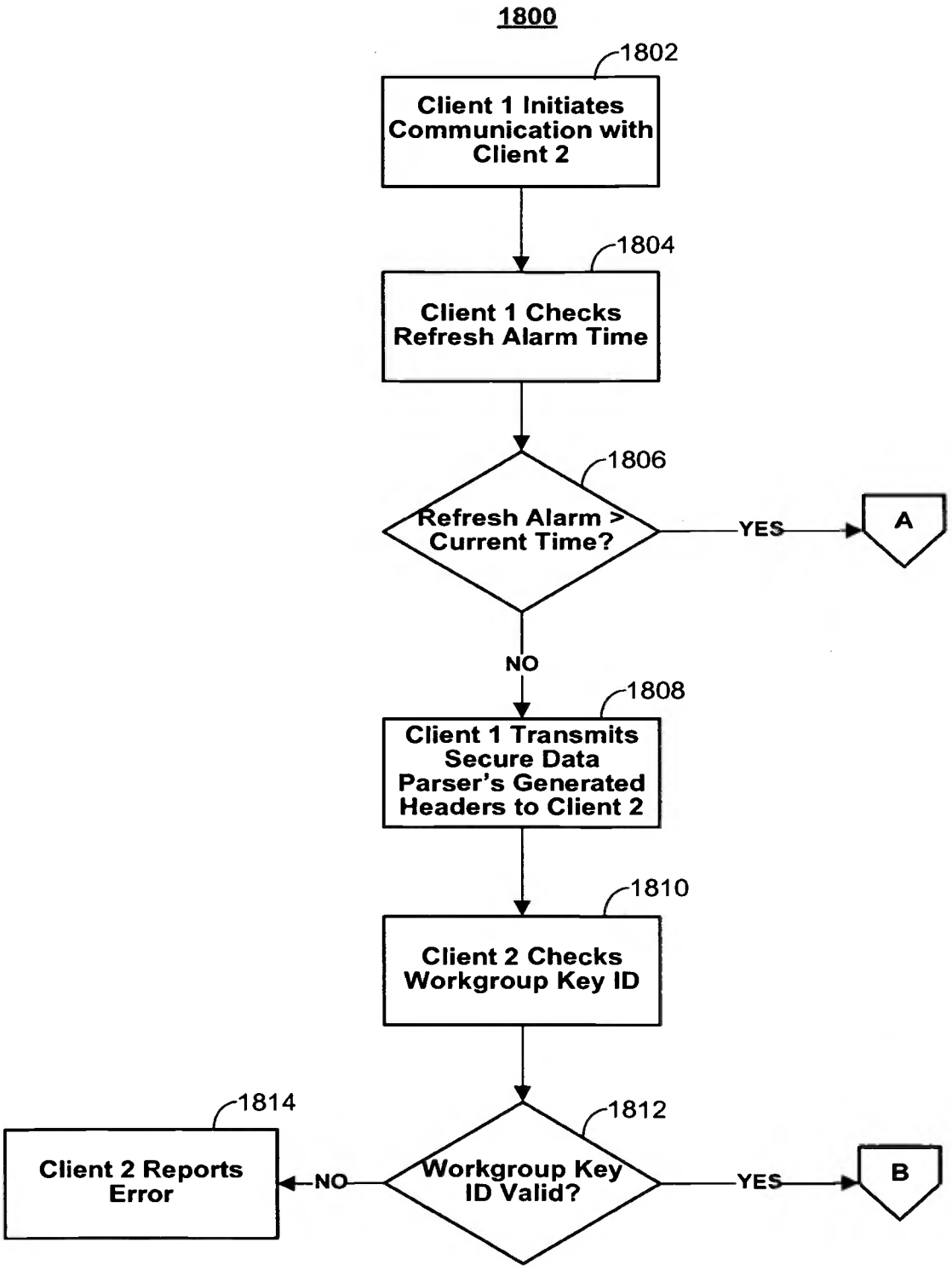

1800
1802
Client 1 Initiates Communication with Client 2
1804
Client 1 Checks Refresh Alarm Time
1806
Refresh Alarm > Current Time?
YES
A
NO
1808
Client 1 Transmits Secure Data Parser's Generated Headers to Client 2
1810
Client 2 Checks Workgroup Key ID
1812
Workgroup Key ID Valid?
NO
1814
Client 2 Reports Error
YES
B

FIG. 18A

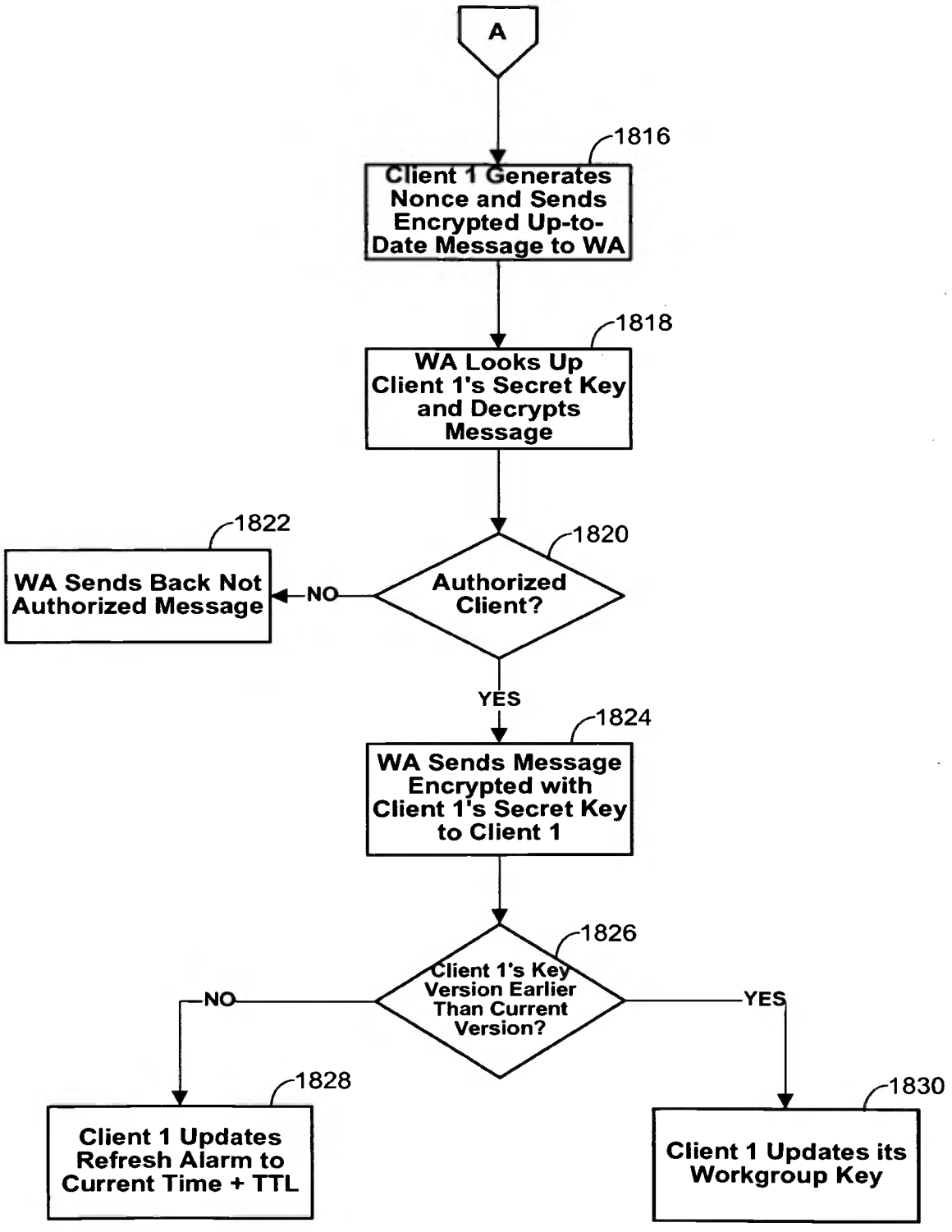

A
1816
Client 1 Generates Nonce and Sends Encrypted Up-to-Date Message to WA
1818
WA Looks Up Client 1's Secret Key and Decrypts Message
1820
Authorized Client?
NO
1822
WA Sends Back Not Authorized Message
YES
1824
WA Sends Message Encrypted with Client 1's Secret Key to Client 1
1826
Client 1's Key Version Earlier Than Current Version?
NO
YES
1828
Client 1 Updates Refresh Alarm to Current Time + TTL
1830
Client 1 Updates its Workgroup Key

FIG. 18B

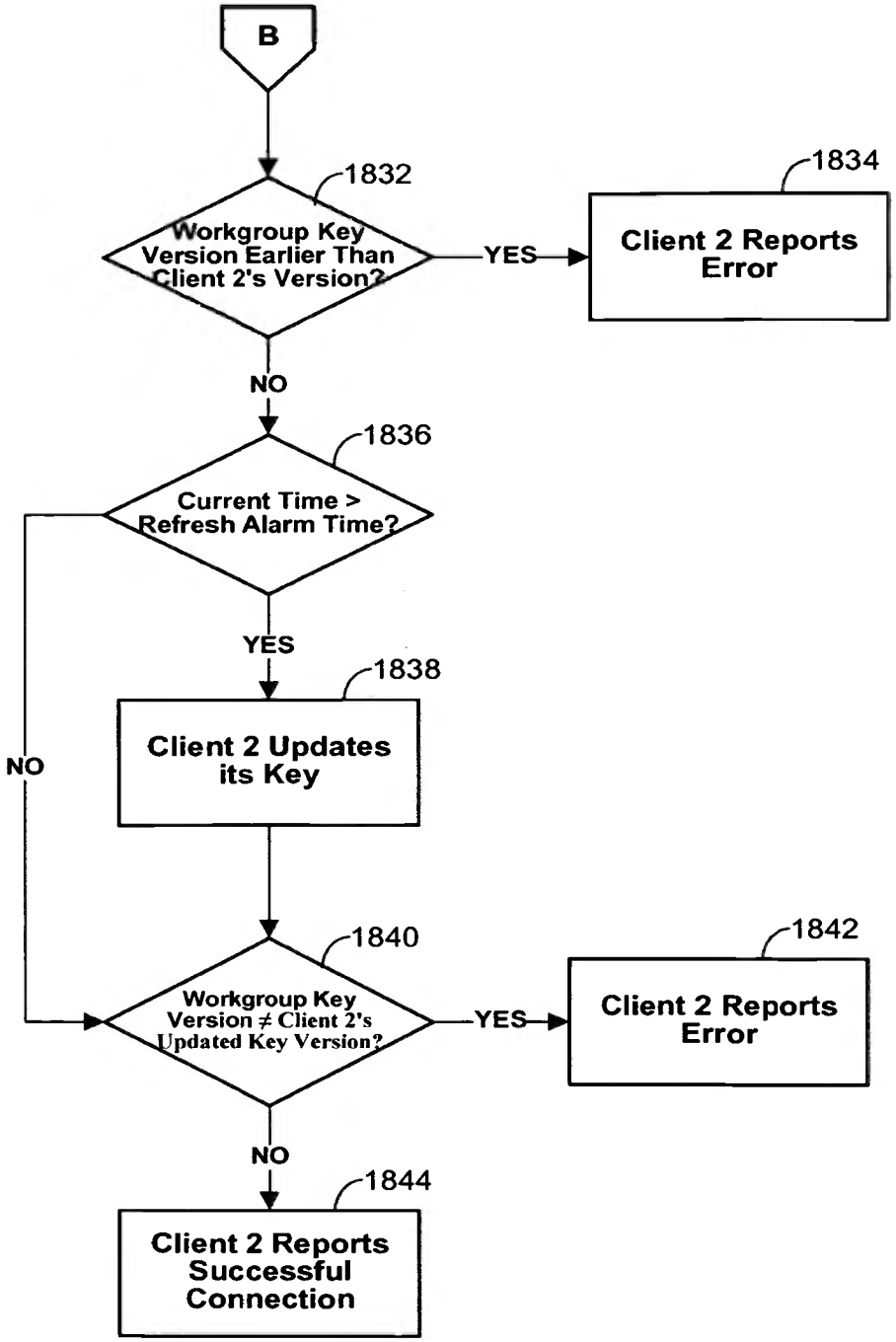

B
1832
Workgroup Key Version Earlier Than Client 2's Version?
YES
1834
Client 2 Reports Error
NO
1836
Current Time > Refresh Alarm Time?
YES
1838
Client 2 Updates its Key
NO
1840
Workgroup Key Version ≠ Client 2's Updated Key Version?
YES
1842
Client 2 Reports Error
NO
1844
Client 2 Reports Successful Connection

FIG. 18C

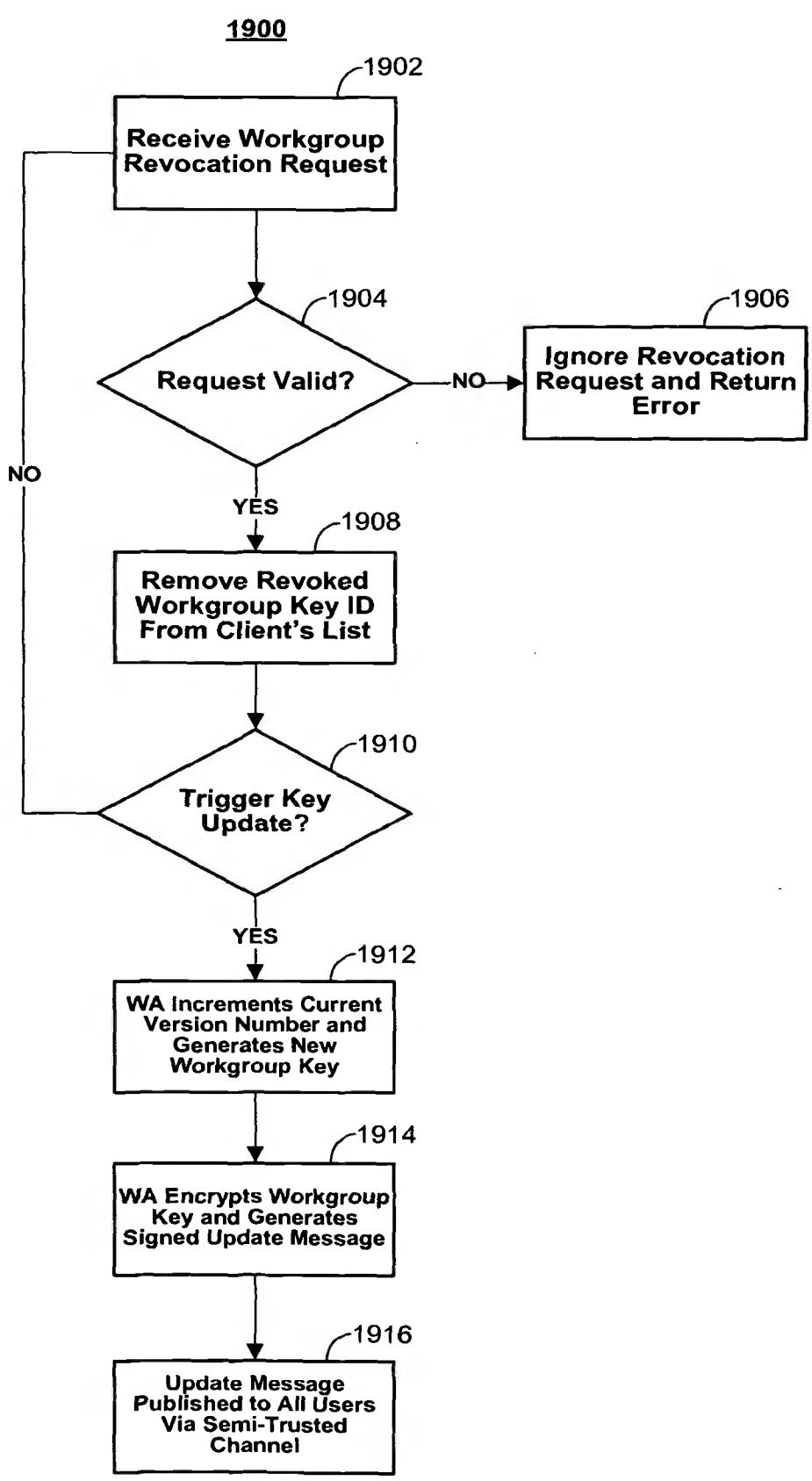

1900
1902
Receive Workgroup Revocation Request
1904
Request Valid?
NO
1906
Ignore Revocation Request and Return Error
YES
NO
1908
Remove Revoked Workgroup Key ID From Client's List
1910
Trigger Key Update?
YES
1912
WA Increments Current Version Number and Generates New Workgroup Key
1914
WA Encrypts Workgroup Key and Generates Signed Update Message
1916
Update Message Published to All Users Via Semi-Trusted Channel

FIG. 19

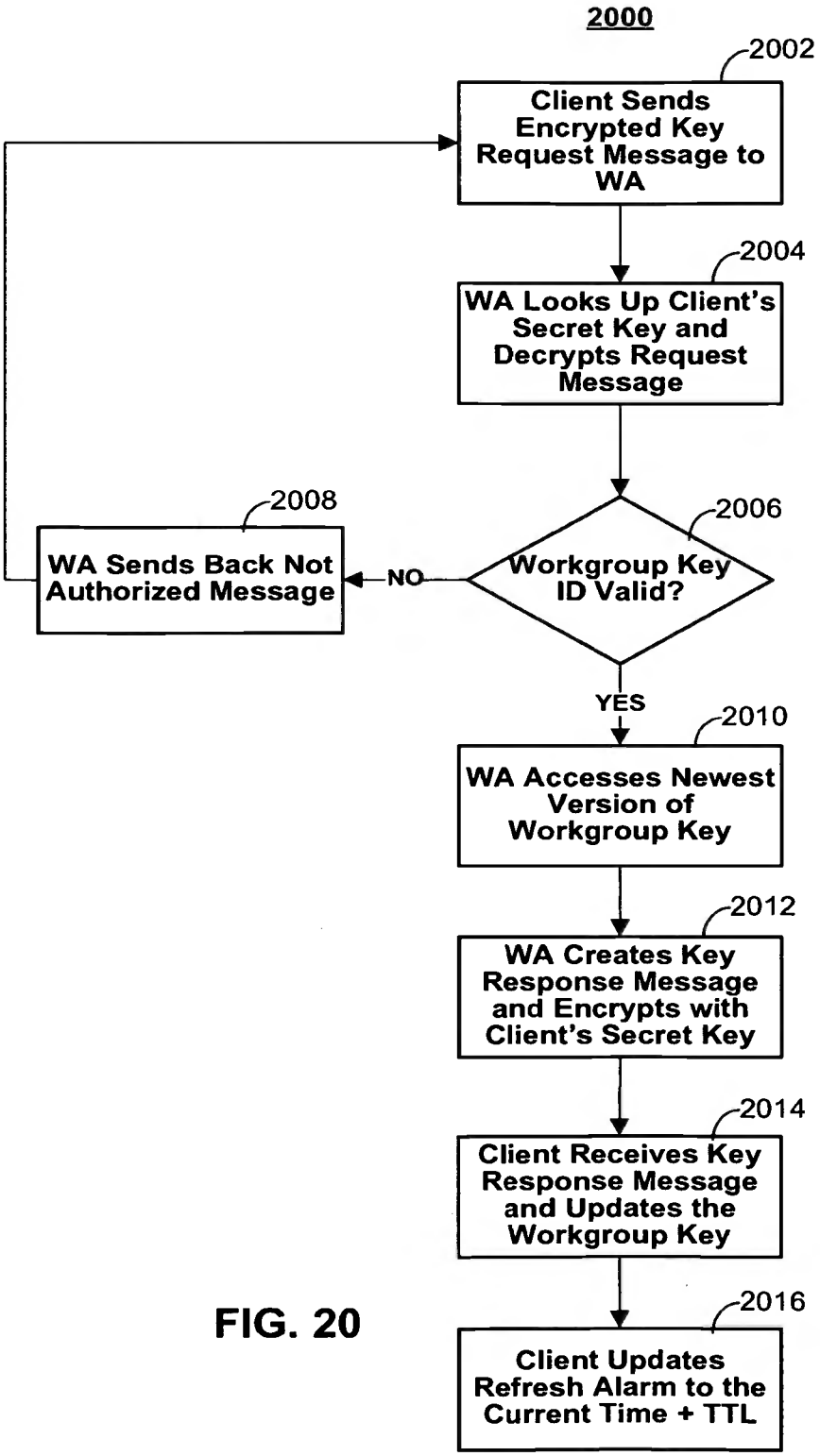

2000
2002
Client Sends Encrypted Key Request Message to WA
2004
WA Looks Up Client's Secret Key and Decrypts Request Message
2006
Workgroup Key ID Valid?
NO
2008
WA Sends Back Not Authorized Message
YES
2010
WA Accesses Newest Version of Workgroup Key
2012
WA Creates Key Response Message and Encrypts with Client's Secret Key
2014
Client Receives Key Response Message and Updates the Workgroup Key
2016
Client Updates Refresh Alarm to the Current Time + TTL

FIG. 20

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
27 August 2009 (27.08.2009)

PCT

(10) International Publication Number
WO 2009/105280 A3

(51) International Patent Classification:
H04L 29/06 (2006.01)

(21) International Application Number:
PCT/US2009/001158

(22) International Filing Date:
23 February 2009 (23.02.2009)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
61/066,699 22 February 2008 (22.02.2008) US
61/147,961 28 January 2009 (28.01.2009) US

(71) Applicant *(for all designated States except US)*: **SECURITY FIRST CORPORATION** [US/US]; 22362 Gilberto #130, Rancho Santa Margarita, CA 92688 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **BONO, Stephen, C.** [US/US]; 1020 Park Avenue, Apt. 207, Baltimore, MD 21201 (US). **GREEN, Matthew, D.** [US/US]; 1017 W. 38th Street, Baltimore, MD 21211 (US). **LANDAU, Gabriel, D.** [US/US]; 6623 Bonnie Ridge Drive, Apt. 201, Baltimore, MD 21209 (US). **ORSINI, Rick, L.** [US/US]; 2100 Kings Forest Lane, Flower Mound, TX 75028 (US). **O'HARE, Mark, S.** [US/US]; 8 Kennedy Court, Coto De Caza, CA 92679 (US). **DAVENPORT, Roger** [US/US]; 3204 Cr 4119, Campbell, TX 75422 (US).

(74) Agents: **INGERMAN, Jeffrey, H.** et al.; Ropes & Gray LLP, 1211 Avenue Of The Americas, New York, NY 10036-8704 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, SE, SI, SK, TR),

[Continued on next page]

(54) Title: SYSTEMS AND METHODS FOR SECURE WORKGROUP MANAGEMENT AND COMMUNICATION

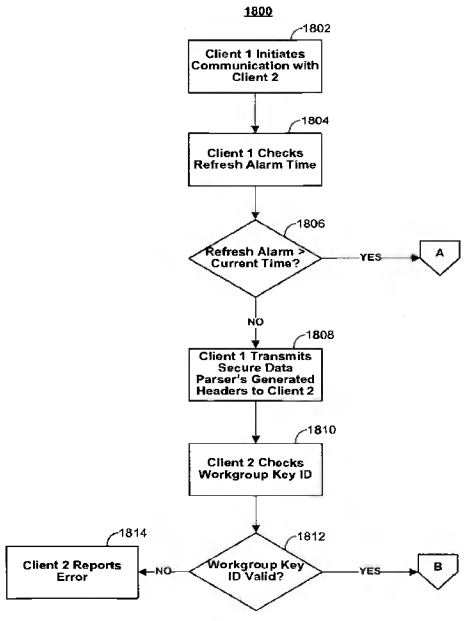


FIG. 18A

(57) Abstract: A secure data parser is provided that may be integrated into any suitable system for securely storing and communicating data. The secure data parser may split or share a data set into multiple portions that are stored or communicated distinctly. Encryption of the original data, the portions of data, or both may be employed for additional security. The secure data parser may be used to protect data in motion by splitting an original data set into portions of data that may be communicated using one or more communications paths. Secure workgroup communication is supported through the secure distribution and management of a workgroup key for use with the secure data parser.

WO 2009/105280 A3

OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))*

(88) Date of publication of the international search report:
22 October 2009

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2009/001158

A. CLASSIFICATION OF SUBJECT MATTER
INV. H04L29/06

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
H04L G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)
EPO-Internal

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2007/160198 A1 (ORSINI RICK [US] ET AL) 12 July 2007 (2007-07-12) abstract paragraph [0442] - paragraph [0455] paragraph [0477] paragraph [0525] - paragraph [0545]	1-7, 14-20
A	MITCHELL C J: "MAKING SERIAL NUMBER BASED AUTHENTICATION ROBUST AGAINST LOSS OF STATE" OPERATING SYSTEMS REVIEW, ACM, NEW YORK, NY, US, vol. 34, no. 3, 1 July 2000 (2000-07-01), pages 56-59, XP001096714 ISSN: 0163-5980 paragraphs [0002], [0004]	1-7, 14-20

☐ Further documents are listed in the continuation of Box C.

☒ See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
9 June 2009	08/09/2009

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL – 2280 HV Rijswijk Tel. (+31–70) 340–2040, Fax: (+31–70) 340–3016	Figiel, Barbara

Form PCT/ISA/210 (second sheet) (April 2005)

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2009/001158

Box No. II Observations where certain claims were found unsearchable (Continuation of item 2 of first sheet)

This international search report has not been established in respect of certain claims under Article 17(2)(a) for the following reasons:

1. ☐ Claims Nos.:
because they relate to subject matter not required to be searched by this Authority, namely:

2. ☐ Claims Nos.:
because they relate to parts of the international application that do not comply with the prescribed requirements to such an extent that no meaningful international search can be carried out, specifically:

3. ☐ Claims Nos.:
because they are dependent claims and are not drafted in accordance with the second and third sentences of Rule 6.4(a).

Box No. III Observations where unity of invention is lacking (Continuation of item 3 of first sheet)

This International Searching Authority found multiple inventions in this international application, as follows:

see additional sheet

1. ☐ As all required additional search fees were timely paid by the applicant, this international search report covers allsearchable claims.

2. ☐ As all searchable claims could be searched without effort justifying an additional fees, this Authority did not invite payment of additional fees.

3. ☐ As only some of the required additional search fees were timely paid by the applicant, this international search reportcovers only those claims for which fees were paid, specifically claims Nos.:

4. ☒ No required additional search fees were timely paid by the applicant. Consequently, this international search report is restricted to the invention first mentioned in the claims; it is covered by claims Nos.:

1-7, 14-20

Remark on Protest

☐ The additional search fees were accompanied by the applicant's protest and, where applicable, the payment of a protest fee.

☐ The additional search fees were accompanied by the applicant's protest but the applicable protest fee was not paid within the time limit specified in the invitation.

☐ No protest accompanied the payment of additional search fees.

FURTHER INFORMATION CONTINUED FROM PCT/ISA/ 210

This International Searching Authority found multiple (groups of) inventions in this international application, as follows:

1. claims: 1-7, 14-20

initiating a communication session with a second workgroup client

---

2. claims: 8-13, 21-26

update of the a workgroup key

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/US2009/001158

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2007160198 A1	12-07-2007	AU 2006350252 A1	08-05-2008
		CA 2629015 A1	08-05-2008
		CN 101401341 A	01-04-2009
		EP 1952575 A2	06-08-2008
		WO 2008054406 A2	08-05-2008

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
25 November 2010 (25.11.2010)

PCT

(10) International Publication Number
WO 2010/135412 A2

(51) International Patent Classification:
H04L 29/06 (2006.01) *H04L 29/08* (2006.01)

(21) International Application Number:
PCT/US2010/035377

(22) International Filing Date:
19 May 2010 (19.05.2010)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
61/179,481 19 May 2009 (19.05.2009) US

(71) Applicant *(for all designated States except US)*: **SECURITY FIRST CORP.** [US/US]; 22362 Gilberto #130, Rancho Santa Margarita, California 92688 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **ORSINI, Rick, L.** [US/US]; 2100 Kings Forest Lane, Flower Mound, Texas 75028 (US). **O'HARE, Mark, S.** [US/US]; 8 Kennedy Court, Coto De Caza, California 92679 (US). **MARTIN, Don** [US/US]; 11006 Sterling Cover Drive, Chesterfield, Virginia 23838 (US).

(74) Agents: **INGERMAN, Jeffrey, H.** et al.; Ropes & Gray LLP, 1211 Avenue of the Americas, New York, New York 10036-8704 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

[Continued on next page]

(54) Title: SYSTEMS AND METHODS FOR SECURING DATA IN THE CLOUD

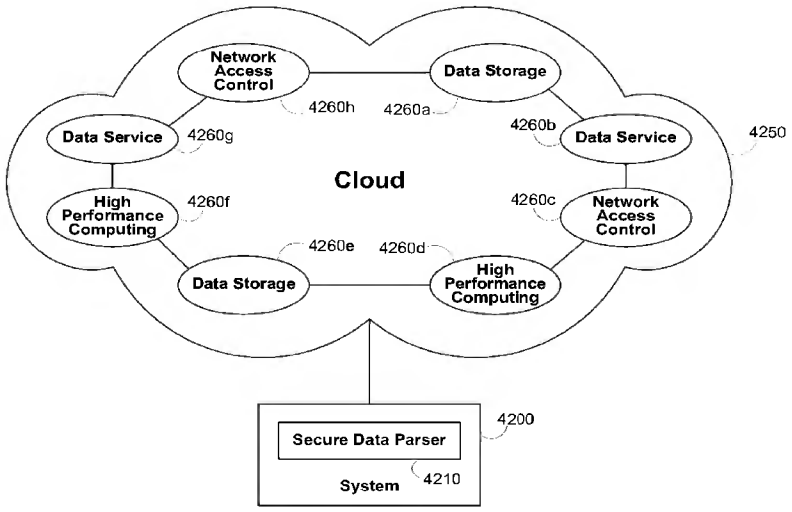


FIG. 42

(57) Abstract: A secure data parser is provided that may be integrated into any suitable system for securely storing data in and communicating data with cloud computing resources. The secure data parser parses data and then splits the data into multiple portions that are stored or communicated distinctly. Encryption of the original data, the portions of data, or both may be employed for additional security.

WO 2010/135412 A2

Published:

— *without international search report and to be republished upon receipt of that report (Rule 48.2(g))*

SYSTEMS AND METHODS FOR SECURING DATA IN THE CLOUD

Cross-Reference to Related Applications

[0001] This claims the benefit of United States Provisional Patent Application No. 61/179,481, entitled "Systems and Methods for Securing Data in SOA, Cloud, and High Performance Environments" filed May 19, 2009, which is hereby incorporated by reference herein in its entirety.

Field of the Invention

[0002] The present invention relates in general to systems and methods for securing data in the cloud. The systems and methods described herein may be used in conjunction with other systems and methods described in commonly-owned U.S. Patent No. 7,391,865 and commonly-owned U.S. Patent Application Nos. 11/258,839, filed October 25, 2005, 11/602,667, filed November 20, 2006, 11/983,355, filed November 7, 2007, 11/999,575, filed December 5, 2007, 12/148,365, filed April 18, 2008, 12/209,703, filed September 12, 2008, 12/349,897, filed January 7, 2009, 12/391,025, filed February 23, 2009, all of which are hereby incorporated by reference herein in their entireties.

Background of the Invention

[0003] In today's society, individuals and businesses conduct an ever-increasing amount of activities on and over computer systems. These computer systems, including proprietary and non-proprietary computer networks, are often storing, archiving, and transmitting all types of sensitive information. Thus, an ever-increasing need exists for ensuring data stored and transmitted over these systems cannot be read or otherwise compromised.

[0004] One common solution for securing computer systems is to provide login and password functionality. However, password management has proven to be quite costly with a large percentage of help desk calls relating to password issues. Moreover, passwords provide little security in that they are generally stored in a file susceptible to inappropriate access, through, for example, brute-force attacks.

[0005] Another solution for securing computer systems is to provide cryptographic infrastructures. Cryptography, in general, refers to protecting data by transforming, or encrypting, it into an unreadable format. Only those who possess the key(s) to the encryption can decrypt the data into a useable format. Cryptography is used to identify users, e.g., authentication, to allow access privileges, e.g., authorization, to create digital certificates and signatures, and the like. One popular cryptography system is a public key system that uses two keys, a public key known to everyone and a private key known only to the individual or business owner thereof. Generally, the data encrypted with one key is decrypted

with the other and neither key is recreatable from the other.

[0006] Unfortunately, even the foregoing typical public-key cryptographic systems are still highly reliant on the user for security. For example, cryptographic systems issue the private key to the user, for example, through the user's browser. Unsophisticated users then generally store the private key on a hard drive accessible to others through an open computer system, such as, for example, the Internet. On the other hand, users may choose poor names for files containing their private key, such as, for example, "key." The result of the foregoing and other acts is to allow the key or keys to be susceptible to compromise.

[0007] In addition to the foregoing compromises, a user may save his or her private key on a computer system configured with an archiving or backup system, potentially resulting in copies of the private key traveling through multiple computer storage devices or other systems. This security breach is often referred to as "key migration." Similar to key migration, many applications provide access to a user's private key through, at most, simple login and password access. As mentioned in the foregoing, login and password access often does not provide adequate security.

[0008] One solution for increasing the security of the foregoing cryptographic systems is to include biometrics as part of the authentication or authorization. Biometrics generally include measurable physical characteristics, such as, for example, finger prints or speech that can be checked by an automated system, such as, for example, pattern matching or

recognition of finger print patterns or speech patterns. In such systems, a user's biometric and/or keys may be stored on mobile computing devices, such as, for example, a smartcard, laptop, personal digital assistant, or mobile phone, thereby allowing the biometric or keys to be usable in a mobile environment.

[0009] The foregoing mobile biometric cryptographic system still suffers from a variety of drawbacks. For example, the mobile user may lose or break the smartcard or portable computing device, thereby having his or her access to potentially important data entirely cut-off. Alternatively, a malicious person may steal the mobile user's smartcard or portable computing device and use it to effectively steal the mobile user's digital credentials. On the other hand, the portable-computing device may be connected to an open system, such as the Internet, and, like passwords, the file where the biometric is stored may be susceptible to compromise through user inattentiveness to security or malicious intruders.

Summary of the Invention

[0010] Based on the foregoing, a need exists to provide a cryptographic system whose security is user-independent while still supporting mobile users.

[0011] Accordingly, one aspect of the present invention is to provide a method for securing virtually any type of data from unauthorized access or use. The method comprises one or more steps of parsing, splitting and/or separating the data to be secured into two or more parts or portions. The method also comprises encrypting the data to be secured. Encryption of the data may be performed prior to or

after the first parsing, splitting and/or separating of the data. In addition, the encrypting step may be repeated for one or more portions of the data. Similarly, the parsing, splitting and/or separating steps may be repeated for one or more portions of the data. The method also optionally comprises storing the parsed, split and/or separated data that has been encrypted in one location or in multiple locations. This method also optionally comprises reconstituting or re-assembling the secured data into its original form for authorized access or use. This method may be incorporated into the operations of any computer, server, engine or the like, that is capable of executing the desired steps of the method.

[0012] Another aspect of the present invention provides a system for securing virtually any type of data from unauthorized access or use. This system comprises a data splitting module, a cryptographic handling module, and, optionally, a data assembly module. The system may, in one embodiment, further comprise one or more data storage facilities where secure data may be stored.

[0013] Accordingly, one aspect of the invention is to provide a secure server, or trust engine, having server-centric keys, or in other words, storing cryptographic keys and user authentication data on a server. According to this embodiment, a user accesses the trust engine in order to perform authentication and cryptographic functions, such as, but not limited to, for example, authentication, authorization, digital signing and generation, storage, and retrieval of certificates, encryption, notary-like and power-of-attorney-like actions, and the like.

[0014] Another aspect of the invention is to provide a reliable, or trusted, authentication process. Moreover, subsequent to a trustworthy positive authentication, a wide number of differing actions may be taken, from providing cryptographic technology, to system or device authorization and access, to permitting use or control of one or a wide number of electronic devices.

[0015] Another aspect of the invention is to provide cryptographic keys and authentication data in an environment where they are not lost, stolen, or compromised, thereby advantageously avoiding a need to continually reissue and manage new keys and authentication data. According to another aspect of the invention, the trust engine allows a user to use one key pair for multiple activities, vendors, and/or authentication requests. According to yet another aspect of the invention, the trust engine performs at least one step of cryptographic processing, such as, but not limited to, encrypting, authenticating, or signing, on the server side, thereby allowing clients or users to possess only minimal computing resources.

[0016] According to yet another aspect of the invention, the trust engine includes one or multiple depositories for storing portions of each cryptographic key and authentication data. The portions are created through a data splitting process that prohibits reconstruction without a predetermined portion from more than one location in one depository or from multiple depositories. According to another embodiment, the multiple depositories may be geographically remote such that a rogue employee or

otherwise compromised system at one depository will not provide access to a user's key or authentication data.

[0017] According to yet another embodiment, the authentication process advantageously allows the trust engine to process multiple authentication activities in parallel. According to yet another embodiment, the trust engine may advantageously track failed access attempts and thereby limit the number of times malicious intruders may attempt to subvert the system.

[0018] According to yet another embodiment, the trust engine may include multiple instantiations where each trust engine may predict and share processing loads with the others. According to yet another embodiment, the trust engine may include a redundancy module for polling a plurality of authentication results to ensure that more than one system authenticates the user.

[0019] Therefore, one aspect of the invention includes a secure cryptographic system, which may be remotely accessible, for storing data of any type, including, but not limited to, a plurality of private cryptographic keys to be associated with a plurality of users. The cryptographic system associates each of the plurality of users with one or more different keys from the plurality of private cryptographic keys and performs cryptographic functions for each user using the associated one or more different keys without releasing the plurality of private cryptographic keys to the users. The cryptographic system comprises a depository system having at least one server which stores the data to be secured, such as a plurality of private cryptographic keys and a plurality of enrollment authentication data. Each enrollment

authentication data identifies one of multiple users and each of the multiple users is associated with one or more different keys from the plurality of private cryptographic keys. The cryptographic system also may comprise an authentication engine which compares authentication data received by one of the multiple users to enrollment authentication data corresponding to the one of multiple users and received from the depository system, thereby producing an authentication result. The cryptographic system also may comprise a cryptographic engine which, when the authentication result indicates proper identification of the one of the multiple users, performs cryptographic functions on behalf of the one of the multiple users using the associated one or more different keys received from the depository system. The cryptographic system also may comprise a transaction engine connected to route data from the multiple users to the depository server system, the authentication engine, and the cryptographic engine.

[0020] Another aspect of the invention includes a secure cryptographic system that is optionally remotely accessible. The cryptographic system comprises a depository system having at least one server which stores at least one private key and any other data, such as, but not limited to, a plurality of enrollment authentication data, wherein each enrollment authentication data identifies one of possibly multiple users. The cryptographic system may also optionally comprise an authentication engine which compares authentication data received by users to enrollment authentication data corresponding to the user and received from the depository system, thereby producing

an authentication result. The cryptographic system also comprises a cryptographic engine which, when the authentication result indicates proper identification of the user, performs cryptographic functions on behalf of the user using at least said private key, which may be received from the depository system. The cryptographic system may also optionally comprise a transaction engine connected to route data from the users to other engines or systems such as, but not limited to, the depository server system, the authentication engine, and the cryptographic engine.

[0021] Another aspect of the invention includes a method of facilitating cryptographic functions. The method comprises associating a user from multiple users with one or more keys from a plurality of private cryptographic keys stored on a secure location, such as a secure server. The method also comprises receiving authentication data from the user, and comparing the authentication data to authentication data corresponding to the user, thereby verifying the identity of the user. The method also comprises utilizing the one or more keys to perform cryptographic functions without releasing the one or more keys to the user.

[0022] Another aspect of the invention includes an authentication system for uniquely identifying a user through secure storage of the user's enrollment authentication data. The authentication system comprises one or more data storage facilities, wherein each data storage facility includes a computer accessible storage medium which stores at least one of portions of enrollment authentication data. The authentication system also comprises an authentication

engine which communicates with the data storage facility or facilities. The authentication engine comprises a data splitting module which operates on the enrollment authentication data to create portions, a data assembling module which processes the portions from at least one of the data storage facilities to assemble the enrollment authentication data, and a data comparator module which receives current authentication data from a user and compares the current authentication data with the assembled enrollment authentication data to determine whether the user has been uniquely identified.

[0023] Another aspect of the invention includes a cryptographic system. The cryptographic system comprises one or more data storage facilities, wherein each data storage facility includes a computer accessible storage medium which stores at least one portion of one ore more cryptographic keys. The cryptographic system also comprises a cryptographic engine which communicates with the data storage facilities. The cryptographic engine also comprises a data splitting module which operate on the cryptographic keys to create portions, a data assembling module which processes the portions from at least one of the data storage facilities to assemble the cryptographic keys, and a cryptographic handling module which receives the assembled cryptographic keys and performs cryptographic functions therewith.

[0024] Another aspect of the invention includes a method of storing any type of data, including, but not limited to, authentication data in geographically remote secure data storage facilities thereby protecting the data against composition of any

individual data storage facility. The method comprises receiving data at a trust engine, combining at the trust engine the data with a first substantially random value to form a first combined value, and combining the data with a second substantially random value to form a second combined value. The method comprises creating a first pairing of the first substantially random value with the second combined value, creating a second pairing of the first substantially random value with the second substantially random value, and storing the first pairing in a first secure data storage facility. The method comprises storing the second pairing in a second secure data storage facility remote from the first secure data storage facility.

[0025] Another aspect of the invention includes a method of storing any type of data, including, but not limited to, authentication data comprising receiving data, combining the data with a first set of bits to form a second set of bits, and combining the data with a third set of bits to form a fourth set of bits. The method also comprises creating a first pairing of the first set of bits with the third set of bits. The method also comprises creating a second pairing of the first set of bits with the fourth set of bits, and storing one of the first and second pairings in a first computer accessible storage medium. The method also comprises storing the other of the first and second pairings in a second computer accessible storage medium.

[0026] Another aspect of the invention includes a method of storing cryptographic data in geographically remote secure data storage facilities thereby protecting the cryptographic data against comprise of

any individual data storage facility. The method comprises receiving cryptographic data at a trust engine, combining at the trust engine the cryptographic data with a first substantially random value to form a first combined value, and combining the cryptographic data with a second substantially random value to form a second combined value. The method also comprises creating a first pairing of the first substantially random value with the second combined value, creating a second pairing of the first substantially random value with the second substantially random value, and storing the first pairing in a first secure data storage facility. The method also comprises storing the second pairing in a secure second data storage facility remote from the first secure data storage facility.

[0027] Another aspect of the invention includes a method of storing cryptographic data comprising receiving authentication data and combining the cryptographic data with a first set of bits to form a second set of bits. The method also comprises combining the cryptographic data with a third set of bits to form a fourth set of bits, creating a first pairing of the first set of bits with the third set of bits, and creating a second pairing of the first set of bits with the fourth set of bits. The method also comprises storing one of the first and second pairings in a first computer accessible storage medium, and storing the other of the first and second pairings in a second computer accessible storage medium.

[0028] Another aspect of the invention includes a method of handling sensitive data of any type or form in a cryptographic system, wherein the sensitive data exists in a useable form only during actions by

authorized users, employing the sensitive data. The method also comprises receiving in a software module, substantially randomized or encrypted sensitive data from a first computer accessible storage medium, and receiving in the software module, substantially randomized or encrypted data which may or may not be sensitive data, from one or more other computer accessible storage medium. The method also comprises processing the substantially randomized pre-encrypted sensitive data and the substantially randomized or encrypted data which may or may not be sensitive data, in the software module to assemble the sensitive data and employing the sensitive data in a software engine to perform an action. The action includes, but is not limited to, one of authenticating a user and performing a cryptographic function.

[0029] Another aspect of the invention includes a secure authentication system. The secure authentication system comprises a plurality of authentication engines. Each authentication engine receives enrollment authentication data designed to uniquely identify a user to a degree of certainty. Each authentication engine receives current authentication data to compare to the enrollment authentication data, and each authentication engine determines an authentication result. The secure authentication system also comprises a redundancy system which receives the authentication result of at least two of the authentication engines and determines whether the user has been uniquely identified.

[0030] Another aspect of the invention includes a secure data in motion system whereby data may be transmitted in different portions that are secured in

accordance with the present invention such that any one portion becoming compromised shall not provide sufficient data to restore the original data. This may be applied to any transmission of data, whether it be wired, wireless, or physical.

[0031] Another aspect of the invention includes integration of the secure data parser of the present invention into any suitable system where data is stored or communicated. For example, email system, RAID systems, video broadcasting systems, database systems, or any other suitable system may have the secure data parser integrated at any suitable level.

[0032] Another aspect of the invention includes using any suitable parsing and splitting algorithm to generate shares of data. Either random, pseudo-random, deterministic, or any combination thereof may be employed for parsing and splitting data.

[0033] Another aspect of the invention includes integration of the secure data parser of the present invention into any suitable system where data is stored in or communicated with cloud computing resources. The secure data parser may be used to secure data stored in the cloud and to secure data services provided in the cloud. Network access in the cloud may be secured to facilitate secure communications between users and communities of interest. In some embodiments, virtual machine images may be sent to a user to provide secure access to cloud computing resources.

[0034] Another aspect of the invention includes integration of the secure data parser of the present invention into any suitable system for securing data networks. The secure data parser may be used to secure orthogonal frequency-division multiplexing (OFDM)

networks such as broadband over wireless and broadband over power lines. The secure data parser may also be used to secure access to critical infrastructure systems such as the power grid.

Brief Description of the Drawings

[0035] The present invention is described in more detail below in connection with the attached drawings, which are meant to illustrate and not to limit the invention, and in which:

[0036] FIGURE 1 illustrates a block diagram of a cryptographic system, according to aspects of an embodiment of the invention;

[0037] FIGURE 2 illustrates a block diagram of the trust engine of FIGURE 1, according to aspects of an embodiment of the invention;

[0038] FIGURE 3 illustrates a block diagram of the transaction engine of FIGURE 2, according to aspects of an embodiment of the invention;

[0039] FIGURE 4 illustrates a block diagram of the depository of FIGURE 2, according to aspects of an embodiment of the invention;

[0040] FIGURE 5 illustrates a block diagram of the authentication engine of FIGURE 2, according to aspects of an embodiment of the invention;

[0041] FIGURE 6 illustrates a block diagram of the cryptographic engine of FIGURE 2, according to aspects of an embodiment of the invention;

[0042] FIGURE 7 illustrates a block diagram of a depository system, according to aspects of another embodiment of the invention;

[0043] FIGURE 8 illustrates a flow chart of a data splitting process according to aspects of an embodiment of the invention;

[0044] FIGURE 9, Panel A illustrates a data flow of an enrollment process according to aspects of an embodiment of the invention;

[0045] FIGURE 9, Panel B illustrates a flow chart of an interoperability process according to aspects of an embodiment of the invention;

[0046] FIGURE 10 illustrates a data flow of an authentication process according to aspects of an embodiment of the invention;

[0047] FIGURE 11 illustrates a data flow of a signing process according to aspects of an embodiment of the invention;

[0048] FIGURE 12 illustrates a data flow and an encryption/decryption process according to aspects and yet another embodiment of the invention;

[0049] FIGURE 13 illustrates a simplified block diagram of a trust engine system according to aspects of another embodiment of the invention;

[0050] FIGURE 14 illustrates a simplified block diagram of a trust engine system according to aspects of another embodiment of the invention;

[0051] FIGURE 15 illustrates a block diagram of the redundancy module of FIGURE 14, according to aspects of an embodiment of the invention;

[0052] FIGURE 16 illustrates a process for evaluating authentications according to one aspect of the invention;

[0053] FIGURE 17 illustrates a process for assigning a value to an authentication according to one aspect as shown in FIGURE 16 of the invention;

[0054] FIGURE 18 illustrates a process for performing trust arbitrage in an aspect of the invention as shown in FIGURE 17; and

[0055] FIGURE 19 illustrates a sample transaction between a user and a vendor according to aspects of an embodiment of the invention where an initial web based contact leads to a sales contract signed by both parties.

[0056] FIGURE 20 illustrates a sample user system with a cryptographic service provider module which provides security functions to a user system.

[0057] FIGURE 21 illustrates a process for parsing, splitting and/or separating data with encryption and storage of the encryption master key with the data.

[0058] FIGURE 22 illustrates a process for parsing, splitting and/or separating data with encryption and storing the encryption master key separately from the data.

[0059] FIGURE 23 illustrates the intermediary key process for parsing, splitting and/or separating data with encryption and storage of the encryption master key with the data.

[0060] FIGURE 24 illustrates the intermediary key process for parsing, splitting and/or separating data with encryption and storing the encryption master key separately from the data.

[0061] FIGURE 25 illustrates utilization of the cryptographic methods and systems of the present invention with a small working group.

[0062] FIGURE 26 is a block diagram of an illustrative physical token security system employing the secure data parser in accordance with one embodiment of the present invention.

[0063] FIGURE 27 is a block diagram of an illustrative arrangement in which the secure data parser is integrated into a system in accordance with one embodiment of the present invention.

[0064] FIGURE 28 is a block diagram of an illustrative data in motion system in accordance with one embodiment of the present invention.

[0065] FIGURE 29 is a block diagram of another illustrative data in motion system in accordance with one embodiment of the present invention.

[0066] FIGURE 30-32 are block diagrams of an illustrative system having the secure data parser integrated in accordance with one embodiment of the present invention.

[0067] FIGURE 33 is a process flow diagram of an illustrative process for parsing and splitting data in accordance with one embodiment of the present invention.

[0068] FIGURE 34 is a process flow diagram of an illustrative process for restoring portions of data into original data in accordance with one embodiment of the present invention.

[0069] FIGURE 35 is a process flow diagram of an illustrative process for splitting data at the bit level in accordance with one embodiment of the present invention.

[0070] FIGURE 36 is a process flow diagram of illustrative steps and features, that may be used in any suitable combination, with any suitable additions, deletions, or modifications in accordance with one embodiment of the present invention.

[0071] FIGURE 37 is a process flow diagram of illustrative steps and features, that may be used in

any suitable combination, with any suitable additions, deletions, or modifications in accordance with one embodiment of the present invention.

[0072] FIGURE 38 is a simplified block diagram of the storage of key and data components within shares, that may be used in any suitable combination, with any suitable additions, deletions, or modifications in accordance with one embodiment of the present invention.

[0073] FIGURE 39 is a simplified block diagram of the storage of key and data components within shares using a workgroup key, that may be used in any suitable combination, with any suitable additions, deletions, or modifications in accordance with one embodiment of the present invention.

[0074] FIGURES 40A and 40B are simplified and illustrative process flow diagrams for header generation and data splitting for data in motion, that may be used in any suitable combination, with any suitable additions, deletions, or modifications in accordance with one embodiment of the present invention.

[0075] FIGURE 41 is a simplified block diagram of an illustrative share format, that may be used in any suitable combination, with any suitable additions, deletions, or modifications in accordance with one embodiment of the present invention.

[0076] FIGURE 42 is a block diagram of an illustrative arrangement in which the secure data parser is integrated into a system connected to cloud computing resources in accordance with one embodiment of the present invention.

[0077] FIGURE 43 is a block diagram of an illustrative arrangement in which the secure data parser is integrated into a system for sending data through the cloud in accordance with one embodiment of the present invention.

[0078] FIGURE 44 is a block diagram of an illustrative arrangement in which the secure data parser is used to secure data services in the cloud in accordance with one embodiment of the present invention.

[0079] FIGURE 45 is a block diagram of an illustrative arrangement in which the secure data parser is used to secure data storage in the cloud in accordance with one embodiment of the present invention.

[0080] FIGURE 46 is a block diagram of an illustrative arrangement in which the secure data parser is used to secure network access control in accordance with one embodiment of the present invention.

[0081] FIGURE 47 is a block diagram of an illustrative arrangement in which the secure data parser is used to secure high performance computing resources in accordance with one embodiment of the present invention.

[0082] FIGURE 48 is a block diagram of an illustrative arrangement in which the secure data parser is used to secure access using virtual machines in accordance with one embodiment of the present invention.

[0083] FIGURES 49 and 50 show block diagrams of alternative illustrative arrangements for securing

access using virtual machines in accordance with embodiments of the present invention.

[0084] FIGURE 51 is a block diagram of an illustrative arrangement in which the secure data parser is used to secure orthogonal frequency-division multiplexing (OFDM) networks in accordance with one embodiment of the present invention.

[0085] FIGURE 52 is a block diagram of an illustrative arrangement in which the secure data parser is used to secure the power grid in accordance with one embodiment of the present invention.

Detailed Description of the Invention

[0086] One aspect of the present invention is to provide a cryptographic system where one or more secure servers, or a trust engine, stores cryptographic keys and user authentication data. Users access the functionality of conventional cryptographic systems through network access to the trust engine, however, the trust engine does not release actual keys and other authentication data and therefore, the keys and data remain secure. This server-centric storage of keys and authentication data provides for user-independent security, portability, availability, and straightforwardness.

[0087] Because users can be confident in, or trust, the cryptographic system to perform user and document authentication and other cryptographic functions, a wide variety of functionality may be incorporated into the system. For example, the trust engine provider can ensure against agreement repudiation by, for example, authenticating the agreement participants, digitally

signing the agreement on behalf of or for the participants, and storing a record of the agreement digitally signed by each participant. In addition, the cryptographic system may monitor agreements and determine to apply varying degrees of authentication, based on, for example, price, user, vendor, geographic location, place of use, or the like.

[0088] To facilitate a complete understanding of the invention, the remainder of the detailed description describes the invention with reference to the figures, wherein like elements are referenced with like numerals throughout.

[0089] FIGURE 1 illustrates a block diagram of a cryptographic system 100, according to aspects of an embodiment of the invention. As shown in FIGURE 1, the cryptographic system 100 includes a user system 105, a trust engine 110, a certificate authority 115, and a vendor system 120, communicating through a communication link 125.

[0090] According to one embodiment of the invention, the user system 105 comprises a conventional general-purpose computer having one or more microprocessors, such as, for example, an Intel-based processor. Moreover, the user system 105 includes an appropriate operating system, such as, for example, an operating system capable of including graphics or windows, such as Windows, Unix, Linux, or the like. As shown in FIGURE 1, the user system 105 may include a biometric device 107. The biometric device 107 may advantageously capture a user's biometric and transfer the captured biometric to the trust engine 110. According to one embodiment of the invention, the biometric device may advantageously comprise a device

having attributes and features similar to those disclosed in U.S. Patent Application No. 08/926,277, filed on September 5, 1997, entitled *"RELIEF OBJECT IMAGE GENERATOR,"* U.S. Patent Application No. 09/558,634, filed on April 26, 2000, entitled *"IMAGING DEVICE FOR A RELIEF OBJECT AND SYSTEM AND METHOD OF USING THE IMAGE DEVICE,"* U.S. Patent Application No. 09/435,011, filed on November 5, 1999, entitled *"RELIEF OBJECT SENSOR ADAPTOR,"* and U.S. Patent Application No. 09/477,943, filed on January 5, 2000, entitled *"PLANAR OPTICAL IMAGE SENSOR AND SYSTEM FOR GENERATING AN ELECTRONIC IMAGE OF A RELIEF OBJECT FOR FINGERPRINT READING,"* all of which are owned by the instant assignee, and all of which are hereby incorporated by reference herein.

[0091] In addition, the user system 105 may connect to the communication link 125 through a conventional service provider, such as, for example, a dial up, digital subscriber line (DSL), cable modem, fiber connection, or the like. According to another embodiment, the user system 105 connects the communication link 125 through network connectivity such as, for example, a local or wide area network. According to one embodiment, the operating system includes a TCP/IP stack that handles all incoming and outgoing message traffic passed over the communication link 125.

[0092] Although the user system 105 is disclosed with reference to the foregoing embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize from the disclosure herein, a wide number of alternatives embodiments of the user system 105, including almost

any computing device capable of sending or receiving information from another computer system. For example, the user system 105 may include, but is not limited to, a computer workstation, an interactive television, an interactive kiosk, a personal mobile computing device, such as a digital assistant, mobile phone, laptop, or the like, a wireless communications device, a smartcard, an embedded computing device, or the like, which can interact with the communication link 125. In such alternative systems, the operating systems will likely differ and be adapted for the particular device. However, according to one embodiment, the operating systems advantageously continue to provide the appropriate communications protocols needed to establish communication with the communication link 125.

[0093] FIGURE 1 illustrates the trust engine 110. According to one embodiment, the trust engine 110 comprises one or more secure servers for accessing and storing sensitive information, which may be any type or form of data, such as, but not limited to text, audio, video, user authentication data and public and private cryptographic keys. According to one embodiment, the authentication data includes data designed to uniquely identify a user of the cryptographic system 100. For example, the authentication data may include a user identification number, one or more biometrics, and a series of questions and answers generated by the trust engine 110 or the user, but answered initially by the user at enrollment. The foregoing questions may include demographic data, such as place of birth, address, anniversary, or the like, personal data, such as mother's maiden name, favorite ice cream, or the

like, or other data designed to uniquely identify the user. The trust engine 110 compares a user's authentication data associated with a current transaction, to the authentication data provided at an earlier time, such as, for example, during enrollment. The trust engine 110 may advantageously require the user to produce the authentication data at the time of each transaction, or, the trust engine 110 may advantageously allow the user to periodically produce authentication data, such as at the beginning of a string of transactions or the logging onto a particular vendor website.

[0094] According to the embodiment where the user produces biometric data, the user provides a physical characteristic, such as, but not limited to, facial scan, hand scan, ear scan, iris scan, retinal scan, vascular pattern, DNA, a fingerprint, writing or speech, to the biometric device 107. The biometric device advantageously produces an electronic pattern, or biometric, of the physical characteristic. The electronic pattern is transferred through the user system 105 to the trust engine 110 for either enrollment or authentication purposes.

[0095] Once the user produces the appropriate authentication data and the trust engine 110 determines a positive match between that authentication data (current authentication data) and the authentication data provided at the time of enrollment (enrollment authentication data), the trust engine 110 provides the user with complete cryptographic functionality. For example, the properly authenticated user may advantageously employ the trust engine 110 to perform hashing, digitally signing, encrypting and decrypting

(often together referred to only as encrypting), creating or distributing digital certificates, and the like. However, the private cryptographic keys used in the cryptographic functions will not be available outside the trust engine 110, thereby ensuring the integrity of the cryptographic keys.

[0096] According to one embodiment, the trust engine 110 generates and stores cryptographic keys. According to another embodiment, at least one cryptographic key is associated with each user. Moreover, when the cryptographic keys include public-key technology, each private key associated with a user is generated within, and not released from, the trust engine 110. Thus, so long as the user has access to the trust engine 110, the user may perform cryptographic functions using his or her private or public key. Such remote access advantageously allows users to remain completely mobile and access cryptographic functionality through practically any Internet connection, such as cellular and satellite phones, kiosks, laptops, hotel rooms and the like.

[0097] According to another embodiment, the trust engine 110 performs the cryptographic functionality using a key pair generated for the trust engine 110. According to this embodiment, the trust engine 110 first authenticates the user, and after the user has properly produced authentication data matching the enrollment authentication data, the trust engine 110 uses its own cryptographic key pair to perform cryptographic functions on behalf of the authenticated user.

[0098] A skilled artisan will recognize from the disclosure herein that the cryptographic keys may

advantageously include some or all of symmetric keys, public keys, and private keys. In addition, a skilled artisan will recognize from the disclosure herein that the foregoing keys may be implemented with a wide number of algorithms available from commercial technologies, such as, for example, RSA, ELGAMAL, or the like.

[0099] FIGURE 1 also illustrates the certificate authority 115. According to one embodiment, the certificate authority 115 may advantageously comprise a trusted third-party organization or company that issues digital certificates, such as, for example, VeriSign, Baltimore, Entrust, or the like. The trust engine 110 may advantageously transmit requests for digital certificates, through one or more conventional digital certificate protocols, such as, for example, PKCS10, to the certificate authority 115. In response, the certificate authority 115 will issue a digital certificate in one or more of a number of differing protocols, such as, for example, PKCS7. According to one embodiment of the invention, the trust engine 110 requests digital certificates from several or all of the prominent certificate authorities 115 such that the trust engine 110 has access to a digital certificate corresponding to the certificate standard of any requesting party.

[0100] According to another embodiment, the trust engine 110 internally performs certificate issuances. In this embodiment, the trust engine 110 may access a certificate system for generating certificates and/or may internally generate certificates when they are requested, such as, for example, at the time of key generation or in the certificate standard requested at

the time of the request. The trust engine 110 will be disclosed in greater detail below.

[0101] FIGURE 1 also illustrates the vendor system 120. According to one embodiment, the vendor system 120 advantageously comprises a Web server. Typical Web servers generally serve content over the Internet using one of several internet markup languages or document format standards, such as the Hyper-Text Markup Language (HTML) or the Extensible Markup Language (XML). The Web server accepts requests from browsers like Netscape and Internet Explorer and then returns the appropriate electronic documents. A number of server or client-side technologies can be used to increase the power of the Web server beyond its ability to deliver standard electronic documents. For example, these technologies include Common Gateway Interface (CGI) scripts, Secure Sockets Layer (SSL) security, and Active Server Pages (ASPs). The vendor system 120 may advantageously provide electronic content relating to commercial, personal, educational, or other transactions.

[0102] Although the vendor system 120 is disclosed with reference to the foregoing embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize from the disclosure herein that the vendor system 120 may advantageously comprise any of the devices described with reference to the user system 105 or combination thereof.

[0103] FIGURE 1 also illustrates the communication link 125 connecting the user system 105, the trust engine 110, the certificate authority 115, and the vendor system 120. According to one embodiment, the

communication link 125 preferably comprises the Internet. The Internet, as used throughout this disclosure is a global network of computers. The structure of the Internet, which is well known to those of ordinary skill in the art, includes a network backbone with networks branching from the backbone. These branches, in turn, have networks branching from them, and so on. Routers move information packets between network levels, and then from network to network, until the packet reaches the neighborhood of its destination. From the destination, the destination network's host directs the information packet to the appropriate terminal, or node. In one advantageous embodiment, the Internet routing hubs comprise domain name system (DNS) servers using Transmission Control Protocol/Internet Protocol (TCP/IP) as is well known in the art. The routing hubs connect to one or more other routing hubs via high-speed communication links.

[0104] One popular part of the Internet is the World Wide Web. The World Wide Web contains different computers, which store documents capable of displaying graphical and textual information. The computers that provide information on the World Wide Web are typically called "websites." A website is defined by an Internet address that has an associated electronic page. The electronic page can be identified by a Uniform Resource Locator (URL). Generally, an electronic page is a document that organizes the presentation of text, graphical images, audio, video, and so forth.

[0105] Although the communication link 125 is disclosed in terms of its preferred embodiment, one of ordinary skill in the art will recognize from the disclosure herein that the communication link 125 may

include a wide range of interactive communications links. For example, the communication link 125 may include interactive television networks, telephone networks, wireless data transmission systems, two-way cable systems, customized private or public computer networks, interactive kiosk networks, automatic teller machine networks, direct links, satellite or cellular networks, and the like.

[0106] FIGURE 2 illustrates a block diagram of the trust engine 110 of FIGURE 1 according to aspects of an embodiment of the invention. As shown in FIGURE 2, the trust engine 110 includes a transaction engine 205, a depository 210, an authentication engine 215, and a cryptographic engine 220. According to one embodiment of the invention, the trust engine 110 also includes mass storage 225. As further shown in FIGURE 2, the transaction engine 205 communicates with the depository 210, the authentication engine 215, and the cryptographic engine 220, along with the mass storage 225. In addition, the depository 210 communicates with the authentication engine 215, the cryptographic engine 220, and the mass storage 225. Moreover, the authentication engine 215 communicates with the cryptographic engine 220. According to one embodiment of the invention, some or all of the foregoing communications may advantageously comprise the transmission of XML documents to IP addresses that correspond to the receiving device. As mentioned in the foregoing, XML documents advantageously allow designers to create their own customized document tags, enabling the definition, transmission, validation, and interpretation of data between applications and between organizations. Moreover, some or all of the foregoing

communications may include conventional SSL technologies.

[0107] According to one embodiment, the transaction engine 205 comprises a data routing device, such as a conventional Web server available from Netscape, Microsoft, Apache, or the like. For example, the Web server may advantageously receive incoming data from the communication link 125. According to one embodiment of the invention, the incoming data is addressed to a front-end security system for the trust engine 110. For example, the front-end security system may advantageously include a firewall, an intrusion detection system searching for known attack profiles, and/or a virus scanner. After clearing the front-end security system, the data is received by the transaction engine 205 and routed to one of the depository 210, the authentication engine 215, the cryptographic engine 220, and the mass storage 225. In addition, the transaction engine 205 monitors incoming data from the authentication engine 215 and cryptographic engine 220, and routes the data to particular systems through the communication link 125. For example, the transaction engine 205 may advantageously route data to the user system 105, the certificate authority 115, or the vendor system 120.

[0108] According to one embodiment, the data is routed using conventional HTTP routing techniques, such as, for example, employing URLs or Uniform Resource Indicators (URIs). URIs are similar to URLs, however, URIs typically indicate the source of files or actions, such as, for example, executables, scripts, and the like. Therefore, according to the one embodiment, the user system 105, the certificate authority 115, the

vendor system 120, and the components of the trust engine 210, advantageously include sufficient data within communication URLs or URIs for the transaction engine 205 to properly route data throughout the cryptographic system.

[0109] Although the data routing is disclosed with reference to its preferred embodiment, a skilled artisan will recognize a wide number of possible data routing solutions or strategies. For example, XML or other data packets may advantageously be unpacked and recognized by their format, content, or the like, such that the transaction engine 205 may properly route data throughout the trust engine 110. Moreover, a skilled artisan will recognize that the data routing may advantageously be adapted to the data transfer protocols conforming to particular network systems, such as, for example, when the communication link 125 comprises a local network.

[0110] According to yet another embodiment of the invention, the transaction engine 205 includes conventional SSL encryption technologies, such that the foregoing systems may authenticate themselves, and vise-versa, with transaction engine 205, during particular communications. As will be used throughout this disclosure, the term "½ SSL" refers to communications where a server but not necessarily the client, is SSL authenticated, and the term "FULL SSL" refers to communications where the client and the server are SSL authenticated. When the instant disclosure uses the term "SSL", the communication may comprise ½ or FULL SSL.

[0111] As the transaction engine 205 routes data to the various components of the cryptographic system 100,

the transaction engine 205 may advantageously create an audit trail. According to one embodiment, the audit trail includes a record of at least the type and format of data routed by the transaction engine 205 throughout the cryptographic system 100. Such audit data may advantageously be stored in the mass storage 225.

[0112] FIGURE 2 also illustrates the depository 210. According to one embodiment, the depository 210 comprises one or more data storage facilities, such as, for example, a directory server, a database server, or the like. As shown in FIGURE 2, the depository 210 stores cryptographic keys and enrollment authentication data. The cryptographic keys may advantageously correspond to the trust engine 110 or to users of the cryptographic system 100, such as the user or vendor. The enrollment authentication data may advantageously include data designed to uniquely identify a user, such as, user ID, passwords, answers to questions, biometric data, or the like. This enrollment authentication data may advantageously be acquired at enrollment of a user or another alternative later time. For example, the trust engine 110 may include periodic or other renewal or reissue of enrollment authentication data.

[0113] According to one embodiment, the communication from the transaction engine 205 to and from the authentication engine 215 and the cryptographic engine 220 comprises secure communication, such as, for example conventional SSL technology. In addition, as mentioned in the foregoing, the data of the communications to and from the depository 210 may be transferred using URLs, URIs, HTTP or XML documents, with any of the foregoing

advantageously having data requests and formats embedded therein.

[0114] As mentioned above, the depository 210 may advantageously comprises a plurality of secure data storage facilities. In such an embodiment, the secure data storage facilities may be configured such that a compromise of the security in one individual data storage facility will not compromise the cryptographic keys or the authentication data stored therein. For example, according to this embodiment, the cryptographic keys and the authentication data are mathematically operated on so as to statistically and substantially randomize the data stored in each data storage facility. According to one embodiment, the randomization of the data of an individual data storage facility renders that data undecipherable. Thus, compromise of an individual data storage facility produces only a randomized undecipherable number and does not compromise the security of any cryptographic keys or the authentication data as a whole.

[0115] FIGURE 2 also illustrates the trust engine 110 including the authentication engine 215. According to one embodiment, the authentication engine 215 comprises a data comparator configured to compare data from the transaction engine 205 with data from the depository 210. For example, during authentication, a user supplies current authentication data to the trust engine 110 such that the transaction engine 205 receives the current authentication data. As mentioned in the foregoing, the transaction engine 205 recognizes the data requests, preferably in the URL or URI, and routes the authentication data to the authentication engine 215. Moreover, upon request, the depository 210

forwards enrollment authentication data corresponding to the user to the authentication engine 215. Thus, the authentication engine 215 has both the current authentication data and the enrollment authentication data for comparison.

[0116] According to one embodiment, the communications to the authentication engine comprise secure communications, such as, for example, SSL technology. Additionally, security can be provided within the trust engine 110 components, such as, for example, super-encryption using public key technologies. For example, according to one embodiment, the user encrypts the current authentication data with the public key of the authentication engine 215. In addition, the depository 210 also encrypts the enrollment authentication data with the public key of the authentication engine 215. In this way, only the authentication engine's private key can be used to decrypt the transmissions.

[0117] As shown in FIGURE 2, the trust engine 110 also includes the cryptographic engine 220. According to one embodiment, the cryptographic engine comprises a cryptographic handling module, configured to advantageously provide conventional cryptographic functions, such as, for example, public-key infrastructure (PKI) functionality. For example, the cryptographic engine 220 may advantageously issue public and private keys for users of the cryptographic system 100. In this manner, the cryptographic keys are generated at the cryptographic engine 220 and forwarded to the depository 210 such that at least the private cryptographic keys are not available outside of the trust engine 110. According to another embodiment, the

cryptographic engine 220 randomizes and splits at least the private cryptographic key data, thereby storing only the randomized split data. Similar to the splitting of the enrollment authentication data, the splitting process ensures the stored keys are not available outside the cryptographic engine 220. According to another embodiment, the functions of the cryptographic engine can be combined with and performed by the authentication engine 215.

[0118] According to one embodiment, communications to and from the cryptographic engine include secure communications, such as SSL technology. In addition, XML documents may advantageously be employed to transfer data and/or make cryptographic function requests.

[0119] FIGURE 2 also illustrates the trust engine 110 having the mass storage 225. As mentioned in the foregoing, the transaction engine 205 keeps data corresponding to an audit trail and stores such data in the mass storage 225. Similarly, according to one embodiment of the invention, the depository 210 keeps data corresponding to an audit trail and stores such data in the mass storage device 225. The depository audit trail data is similar to that of the transaction engine 205 in that the audit trail data comprises a record of the requests received by the depository 210 and the response thereof. In addition, the mass storage 225 may be used to store digital certificates having the public key of a user contained therein.

[0120] Although the trust engine 110 is disclosed with reference to its preferred and alternative embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will

recognize in the disclosure herein, a wide number of alternatives for the trust engine 110. For example, the trust engine 110, may advantageously perform only authentication, or alternatively, only some or all of the cryptographic functions, such as data encryption and decryption. According to such embodiments, one of the authentication engine 215 and the cryptographic engine 220 may advantageously be removed, thereby creating a more straightforward design for the trust engine 110. In addition, the cryptographic engine 220 may also communicate with a certificate authority such that the certificate authority is embodied within the trust engine 110. According to yet another embodiment, the trust engine 110 may advantageously perform authentication and one or more cryptographic functions, such as, for example, digital signing.

[0121] FIGURE 3 illustrates a block diagram of the transaction engine 205 of FIGURE 2, according to aspects of an embodiment of the invention. According to this embodiment, the transaction engine 205 comprises an operating system 305 having a handling thread and a listening thread. The operating system 305 may advantageously be similar to those found in conventional high volume servers, such as, for example, Web servers available from Apache. The listening thread monitors the incoming communication from one of the communication link 125, the authentication engine 215, and the cryptographic engine 220 for incoming data flow. The handling thread recognizes particular data structures of the incoming data flow, such as, for example, the foregoing data structures, thereby routing the incoming data to one of the communication link 125, the depository 210, the authentication engine 215, the

cryptographic engine 220, or the mass storage 225. As shown in FIGURE 3, the incoming and outgoing data may advantageously be secured through, for example, SSL technology.

[0122] FIGURE 4 illustrates a block diagram of the depository 210 of FIGURE 2 according to aspects of an embodiment of the invention. According to this embodiment, the depository 210 comprises one or more lightweight directory access protocol (LDAP) servers. LDAP directory servers are available from a wide variety of manufacturers such as Netscape, ISO, and others. FIGURE 4 also shows that the directory server preferably stores data 405 corresponding to the cryptographic keys and data 410 corresponding to the enrollment authentication data. According to one embodiment, the depository 210 comprises a single logical memory structure indexing authentication data and cryptographic key data to a unique user ID. The single logical memory structure preferably includes mechanisms to ensure a high degree of trust, or security, in the data stored therein. For example, the physical location of the depository 210 may advantageously include a wide number of conventional security measures, such as limited employee access, modern surveillance systems, and the like. In addition to, or in lieu of, the physical securities, the computer system or server may advantageously include software solutions to protect the stored data. For example, the depository 210 may advantageously create and store data 415 corresponding to an audit trail of actions taken. In addition, the incoming and outgoing communications may advantageously be encrypted with

public key encryption coupled with conventional SSL technologies.

[0123] According to another embodiment, the depository 210 may comprise distinct and physically separated data storage facilities, as disclosed further with reference to FIGURE 7.

[0124] FIGURE 5 illustrates a block diagram of the authentication engine 215 of FIGURE 2 according to aspects of an embodiment of the invention. Similar to the transaction engine 205 of FIGURE 3, the authentication engine 215 comprises an operating system 505 having at least a listening and a handling thread of a modified version of a conventional Web server, such as, for example, Web servers available from Apache. As shown in FIGURE 5, the authentication engine 215 includes access to at least one private key 510. The private key 510 may advantageously be used for example, to decrypt data from the transaction engine 205 or the depository 210, which was encrypted with a corresponding public key of the authentication engine 215.

[0125] FIGURE 5 also illustrates the authentication engine 215 comprising a comparator 515, a data splitting module 520, and a data assembling module 525. According to the preferred embodiment of the invention, the comparator 515 includes technology capable of comparing potentially complex patterns related to the foregoing biometric authentication data. The technology may include hardware, software, or combined solutions for pattern comparisons, such as, for example, those representing finger print patterns or voice patterns. In addition, according to one embodiment, the comparator 515 of the authentication

engine 215 may advantageously compare conventional hashes of documents in order to render a comparison result. According to one embodiment of the invention, the comparator 515 includes the application of heuristics 530 to the comparison. The heuristics 530 may advantageously address circumstances surrounding an authentication attempt, such as, for example, the time of day, IP address or subnet mask, purchasing profile, email address, processor serial number or ID, or the like.

[0126] Moreover, the nature of biometric data comparisons may result in varying degrees of confidence being produced from the matching of current biometric authentication data to enrollment data. For example, unlike a traditional password which may only return a positive or negative match, a fingerprint may be determined to be a partial match, e.g. a 90% match, a 75% match, or a 10% match, rather than simply being correct or incorrect. Other biometric identifiers such as voice print analysis or face recognition may share this property of probabilistic authentication, rather than absolute authentication.

[0127] When working with such probabilistic authentication or in other cases where an authentication is considered less than absolutely reliable, it is desirable to apply the heuristics 530 to determine whether the level of confidence in the authentication provided is sufficiently high to authenticate the transaction which is being made.

[0128] It will sometimes be the case that the transaction at issue is a relatively low value transaction where it is acceptable to be authenticated to a lower level of confidence. This could include a

transaction which has a low dollar value associated with it (e.g., a $10 purchase) or a transaction with low risk (e.g., admission to a members-only web site).

[0129] Conversely, for authenticating other transactions, it may be desirable to require a high degree of confidence in the authentication before allowing the transaction to proceed. Such transactions may include transactions of large dollar value (e.g., signing a multi-million dollar supply contract) or transaction with a high risk if an improper authentication occurs (e.g., remotely logging onto a government computer).

[0130] The use of the heuristics 530 in combination with confidence levels and transactions values may be used as will be described below to allow the comparator to provide a dynamic context-sensitive authentication system.

[0131] According to another embodiment of the invention, the comparator 515 may advantageously track authentication attempts for a particular transaction. For example, when a transaction fails, the trust engine 110 may request the user to re-enter his or her current authentication data. The comparator 515 of the authentication engine 215 may advantageously employ an attempt limiter 535 to limit the number of authentication attempts, thereby prohibiting brute-force attempts to impersonate a user's authentication data. According to one embodiment, the attempt limiter 535 comprises a software module monitoring transactions for repeating authentication attempts and, for example, limiting the authentication attempts for a given transaction to three. Thus, the attempt limiter 535 will limit an automated attempt to

impersonate an individual's authentication data to, for example, simply three "guesses." Upon three failures, the attempt limiter 535 may advantageously deny additional authentication attempts. Such denial may advantageously be implemented through, for example, the comparator 515 returning a negative result regardless of the current authentication data being transmitted. On the other hand, the transaction engine 205 may advantageously block any additional authentication attempts pertaining to a transaction in which three attempts have previously failed.

[0132] The authentication engine 215 also includes the data splitting module 520 and the data assembling module 525. The data splitting module 520 advantageously comprises a software, hardware, or combination module having the ability to mathematically operate on various data so as to substantially randomize and split the data into portions. According to one embodiment, original data is not recreatable from an individual portion. The data assembling module 525 advantageously comprises a software, hardware, or combination module configured to mathematically operate on the foregoing substantially randomized portions, such that the combination thereof provides the original deciphered data. According to one embodiment, the authentication engine 215 employs the data splitting module 520 to randomize and split enrollment authentication data into portions, and employs the data assembling module 525 to reassemble the portions into usable enrollment authentication data.

[0133] FIGURE 6 illustrates a block diagram of the cryptographic engine 220 of the trust engine 200 of FIGURE 2 according to aspects of one embodiment of the

invention. Similar to the transaction engine 205 of FIGURE 3, the cryptographic engine 220 comprises an operating system 605 having at least a listening and a handling thread of a modified version of a conventional Web server, such as, for example, Web servers available from Apache. As shown in FIGURE 6, the cryptographic engine 220 comprises a data splitting module 610 and a data assembling module 620 that function similar to those of FIGURE 5. However, according to one embodiment, the data splitting module 610 and the data assembling module 620 process cryptographic key data, as opposed to the foregoing enrollment authentication data. Although, a skilled artisan will recognize from the disclosure herein that the data splitting module 910 and the data splitting module 620 may be combined with those of the authentication engine 215.

[0134] The cryptographic engine 220 also comprises a cryptographic handling module 625 configured to perform one, some or all of a wide number of cryptographic functions. According to one embodiment, the cryptographic handling module 625 may comprise software modules or programs, hardware, or both. According to another embodiment, the cryptographic handling module 625 may perform data comparisons, data parsing, data splitting, data separating, data hashing, data encryption or decryption, digital signature verification or creation, digital certificate generation, storage, or requests, cryptographic key generation, or the like. Moreover, a skilled artisan will recognize from the disclosure herein that the cryptographic handling module 825 may advantageously comprises a public-key infrastructure, such as Pretty Good Privacy (PGP), an RSA-based public-key system, or

a wide number of alternative key management systems. In addition, the cryptographic handling module 625 may perform public-key encryption, symmetric-key encryption, or both. In addition to the foregoing, the cryptographic handling module 625 may include one or more computer programs or modules, hardware, or both, for implementing seamless, transparent, interoperability functions.

[0135] A skilled artisan will also recognize from the disclosure herein that the cryptographic functionality may include a wide number or variety of functions generally relating to cryptographic key management systems.

[0136] FIGURE 7 illustrates a simplified block diagram of a depository system 700 according to aspects of an embodiment of the invention. As shown in FIGURE 7, the depository system 700 advantageously comprises multiple data storage facilities, for example, data storage facilities D1, D2, D3, and D4. However, it is readily understood by those of ordinary skill in the art that the depository system may have only one data storage facility. According to one embodiment of the invention, each of the data storage facilities D1 through D4 may advantageously comprise some or all of the elements disclosed with reference to the depository 210 of FIGURE 4. Similar to the depository 210, the data storage facilities D1 through D4 communicate with the transaction engine 205, the authentication engine 215, and the cryptographic engine 220, preferably through conventional SSL. Communication links transferring, for example, XML documents. Communications from the transaction engine 205 may advantageously include requests for data, wherein the

request is advantageously broadcast to the IP address of each data storage facility D1 through D4. On the other hand, the transaction engine 205 may broadcast requests to particular data storage facilities based on a wide number of criteria, such as, for example, response time, server loads, maintenance schedules, or the like.

[0137] In response to requests for data from the transaction engine 205, the depository system 700 advantageously forwards stored data to the authentication engine 215 and the cryptographic engine 220. The respective data assembling modules receive the forwarded data and assemble the data into useable formats. On the other hand, communications from the authentication engine 215 and the cryptographic engine 220 to the data storage facilities D1 through D4 may include the transmission of sensitive data to be stored. For example, according to one embodiment, the authentication engine 215 and the cryptographic engine 220 may advantageously employ their respective data splitting modules to divide sensitive data into undecipherable portions, and then transmit one or more undecipherable portions of the sensitive data to a particular data storage facility.

[0138] According to one embodiment, each data storage facility, D1 through D4, comprises a separate and independent storage system, such as, for example, a directory server. According to another embodiment of the invention, the depository system 700 comprises multiple geographically separated independent data storage systems. By distributing the sensitive data into distinct and independent storage facilities D1 through D4, some or all of which may be advantageously

geographically separated, the depository system 700 provides redundancy along with additional security measures. For example, according to one embodiment, only data from two of the multiple data storage facilities, D1 through D4, are needed to decipher and reassemble the sensitive data. Thus, as many as two of the four data storage facilities D1 through D4 may be inoperative due to maintenance, system failure, power failure, or the like, without affecting the functionality of the trust engine 110. In addition, because, according to one embodiment, the data stored in each data storage facility is randomized and undecipherable, compromise of any individual data storage facility does not necessarily compromise the sensitive data. Moreover, in the embodiment having geographical separation of the data storage facilities, a compromise of multiple geographically remote facilities becomes increasingly difficult. In fact, even a rogue employee will be greatly challenged to subvert the needed multiple independent geographically remote data storage facilities.

[0139] Although the depository system 700 is disclosed with reference to its preferred and alternative embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize from the disclosure herein, a wide number of alternatives for the depository system 700. For example, the depository system 700 may comprise one, two or more data storage facilities. In addition, sensitive data may be mathematically operated such that portions from two or more data storage facilities are needed to reassemble and decipher the sensitive data.

[0140] As mentioned in the foregoing, the authentication engine 215 and the cryptographic engine 220 each include a data splitting module 520 and 610, respectively, for splitting any type or form of sensitive data, such as, for example, text, audio, video, the authentication data and the cryptographic key data. FIGURE 8 illustrates a flowchart of a data splitting process 800 performed by the data splitting module according to aspects of an embodiment of the invention. As shown in FIGURE 8, the data splitting process 800 begins at step 805 when sensitive data "S" is received by the data splitting module of the authentication engine 215 or the cryptographic engine 220. Preferably, in step 810, the data splitting module then generates a substantially random number, value, or string or set of bits, "A." For example, the random number A may be generated in a wide number of varying conventional techniques available to one of ordinary skill in the art, for producing high quality random numbers suitable for use in cryptographic applications. In addition, according to one embodiment, the random number A comprises a bit length which may be any suitable length, such as shorter, longer or equal to the bit length of the sensitive data, S.

[0141] In addition, in step 820 the data splitting process 800 generates another statistically random number "C." According to the preferred embodiment, the generation of the statistically random numbers A and C may advantageously be done in parallel. The data splitting module then combines the numbers A and C with the sensitive data S such that new numbers "B" and "D" are generated. For example, number B may comprise the

binary combination of A XOR S and number D may comprise the binary combination of C XOR S. The XOR function, or the "exclusive-or" function, is well known to those of ordinary skill in the art. The foregoing combinations preferably occur in steps 825 and 830, respectively, and, according to one embodiment, the foregoing combinations also occur in parallel. The data splitting process 800 then proceeds to step 835 where the random numbers A and C and the numbers B and D are paired such that none of the pairings contain sufficient data, by themselves, to reorganize and decipher the original sensitive data S. For example, the numbers may be paired as follows: AC, AD, BC, and BD. According to one embodiment, each of the foregoing pairings is distributed to one of the depositories D1 through D4 of FIGURE 7. According to another embodiment, each of the foregoing pairings is randomly distributed to one of the depositories D1 through D4. For example, during a first data splitting process 800, the pairing AC may be sent to depository D2, through, for example, a random selection of D2's IP address. Then, during a second data splitting process 800, the pairing AC may be sent to depository D4, through, for example, a random selection of D4's IP address. In addition, the pairings may all be stored on one depository, and may be stored in separate locations on said depository.

[0142] Based on the foregoing, the data splitting process 800 advantageously places portions of the sensitive data in each of the four data storage facilities D1 through D4, such that no single data storage facility D1 through D4 includes sufficient encrypted data to recreate the original sensitive data

S. As mentioned in the foregoing, such randomization of the data into individually unusable encrypted portions increases security and provides for maintained trust in the data even if one of the data storage facilities, D1 through D4, is compromised.

[0143] Although the data splitting process 800 is disclosed with reference to its preferred embodiment, the invention is not intended to be limited thereby. Rather a skilled artisan will recognize from the disclosure herein, a wide number of alternatives for the data splitting process 800. For example, the data splitting process may advantageously split the data into two numbers, for example, random number A and number B and, randomly distribute A and B through two data storage facilities. Moreover, the data splitting process 800 may advantageously split the data among a wide number of data storage facilities through generation of additional random numbers. The data may be split into any desired, selected, predetermined, or randomly assigned size unit, including but not limited to, a bit, bits, bytes, kilobytes, megabytes or larger, or any combination or sequence of sizes. In addition, varying the sizes of the data units resulting from the splitting process may render the data more difficult to restore to a useable form, thereby increasing security of sensitive data. It is readily apparent to those of ordinary skill in the art that the split data unit sizes may be a wide variety of data unit sizes or patterns of sizes or combinations of sizes. For example, the data unit sizes may be selected or predetermined to be all of the same size, a fixed set of different sizes, a combination of sizes, or randomly generates sizes. Similarly, the data units may be

distributed into one or more shares according to a fixed or predetermined data unit size, a pattern or combination of data unit sizes, or a randomly generated data unit size or sizes per share.

[0144] As mentioned in the foregoing, in order to recreate the sensitive data S, the data portions need to be derandomized and reorganized. This process may advantageously occur in the data assembling modules, 525 and 620, of the authentication engine 215 and the cryptographic engine 220, respectively. The data assembling module, for example, data assembly module 525, receives data portions from the data storage facilities D1 through D4, and reassembles the data into useable form. For example, according to one embodiment where the data splitting module 520 employed the data splitting process 800 of FIGURE 8, the data assembling module 525 uses data portions from at least two of the data storage facilities D1 through D4 to recreate the sensitive data S. For example, the pairings of AC, AD, BC, and BD, were distributed such that any two provide one of A and B, or, C and D. Noting that S = A XOR B or S = C XOR D indicates that when the data assembling module receives one of A and B, or, C and D, the data assembling module 525 can advantageously reassemble the sensitive data S. Thus, the data assembling module 525 may assemble the sensitive data S, when, for example, it receives data portions from at least the first two of the data storage facilities D1 through D4 to respond to an assemble request by the trust engine 110.

[0145] Based on the above data splitting and assembling processes, the sensitive data S exists in usable format only in a limited area of the trust engine 110. For example, when the sensitive data S

includes enrollment authentication data, usable, nonrandomized enrollment authentication data is available only in the authentication engine 215. Likewise, when the sensitive data S includes private cryptographic key data, usable, nonrandomized private cryptographic key data is available only in the cryptographic engine 220.

[0146] Although the data splitting and assembling processes are disclosed with reference to their preferred embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize from the disclosure herein, a wide number of alternatives for splitting and reassembling the sensitive data S. For example, public-key encryption may be used to further secure the data at the data storage facilities D1 through D4. In addition, it is readily apparent to those of ordinary skill in the art that the data splitting module described herein is also a separate and distinct embodiment of the present invention that may be incorporated into, combined with or otherwise made part of any pre-existing computer systems, software suites, database, or combinations thereof, or other embodiments of the present invention, such as the trust engine, authentication engine, and transaction engine disclosed and described herein.

[0147] FIGURE 9A illustrates a data flow of an enrollment process 900 according to aspects of an embodiment of the invention. As shown in FIGURE 9A, the enrollment process 900 begins at step 905 when a user desires to enroll with the trust engine 110 of the cryptographic system 100. According to this embodiment, the user system 105 advantageously includes a client-side applet, such as a Java-based, that

queries the user to enter enrollment data, such as demographic data and enrollment authentication data. According to one embodiment, the enrollment authentication data includes user ID, password(s), biometric(s), or the like. According to one embodiment, during the querying process, the client-side applet preferably communicates with the trust engine 110 to ensure that a chosen user ID is unique. When the user ID is nonunique, the trust engine 110 may advantageously suggest a unique user ID. The client-side applet gathers the enrollment data and transmits the enrollment data, for example, through and XML document, to the trust engine 110, and in particular, to the transaction engine 205. According to one embodiment, the transmission is encoded with the public key of the authentication engine 215.

[0148] According to one embodiment, the user performs a single enrollment during step 905 of the enrollment process 900. For example, the user enrolls himself or herself as a particular person, such as Joe User. When Joe User desires to enroll as Joe User, CEO of Mega Corp., then according to this embodiment, Joe User enrolls a second time, receives a second unique user ID and the trust engine 110 does not associate the two identities. According to another embodiment of the invention, the enrollment process 900 provides for multiple user identities for a single user ID. Thus, in the above example, the trust engine 110 will advantageously associate the two identities of Joe User. As will be understood by a skilled artisan from the disclosure herein, a user may have many identities, for example, Joe User the head of household, Joe User the member of the Charitable Foundations, and the like.

Even though the user may have multiple identities, according to this embodiment, the trust engine 110 preferably stores only one set of enrollment data. Moreover, users may advantageously add, edit/update, or delete identities as they are needed.

[0149] Although the enrollment process 900 is disclosed with reference to its preferred embodiment, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize from the disclosure herein, a wide number of alternatives for gathering of enrollment data, and in particular, enrollment authentication data. For example, the applet may be common object model (COM) based applet or the like.

[0150] On the other hand, the enrollment process may include graded enrollment. For example, at a lowest level of enrollment, the user may enroll over the communication link 125 without producing documentation as to his or her identity. According to an increased level of enrollment, the user enrolls using a trusted third party, such as a digital notary. For example, and the user may appear in person to the trusted third party, produce credentials such as a birth certificate, driver's license, military ID, or the like, and the trusted third party may advantageously include, for example, their digital signature in enrollment submission. The trusted third party may include an actual notary, a government agency, such as the Post Office or Department of Motor Vehicles, a human resources person in a large company enrolling an employee, or the like. A skilled artisan will understand from the disclosure herein that a wide

number of varying levels of enrollment may occur during the enrollment process 900.

[0151] After receiving the enrollment authentication data, at step 915, the transaction engine 205, using conventional FULL SSL technology forwards the enrollment authentication data to the authentication engine 215. In step 920, the authentication engine 215 decrypts the enrollment authentication data using the private key of the authentication engine 215. In addition, the authentication engine 215 employs the data splitting module to mathematically operate on the enrollment authentication data so as to split the data into at least two independently undecipherable, randomized, numbers. As mentioned in the foregoing, at least two numbers may comprise a statistically random number and a binary X0Red number. In step 925, the authentication engine 215 forwards each portion of the randomized numbers to one of the data storage facilities D1 through D4. As mentioned in the foregoing, the authentication engine 215 may also advantageously randomize which portions are transferred to which depositories.

[0152] Often during the enrollment process 900, the user will also desire to have a digital certificate issued such that he or she may receive encrypted documents from others outside the cryptographic system 100. As mentioned in the foregoing, the certificate authority 115 generally issues digital certificates according to one or more of several conventional standards. Generally, the digital certificate includes a public key of the user or system, which is known to everyone.

[0153] Whether the user requests a digital certificate at enrollment, or at another time, the request is transferred through the trust engine 110 to the authentication engine 215. According to one embodiment, the request includes an XML document having, for example, the proper name of the user. According to step 935, the authentication engine 215 transfers the request to the cryptographic engine 220 instructing the cryptographic engine 220 to generate a cryptographic key or key pair.

[0154] Upon request, at step 935, the cryptographic engine 220 generates at least one cryptographic key. According to one embodiment, the cryptographic handling module 625 generates a key pair, where one key is used as a private key, and one is used as a public key. The cryptographic engine 220 stores the private key and, according to one embodiment, a copy of the public key. In step 945, the cryptographic engine 220 transmits a request for a digital certificate to the transaction engine 205. According to one embodiment, the request advantageously includes a standardized request, such as PKCS10, embedded in, for example, an XML document. The request for a digital certificate may advantageously correspond to one or more certificate authorities and the one or more standard formats the certificate authorities require.

[0155] In step 950 the transaction engine 205 forwards this request to the certificate authority 115, who, in step 955, returns a digital certificate. The return digital certificate may advantageously be in a standardized format, such as PKCS7, or in a proprietary format of one or more of the certificate authorities 115. In step 960, the digital certificate is received

by the transaction engine 205, and a copy is forwarded to the user and a copy is stored with the trust engine 110. The trust engine 110 stores a copy of the certificate such that the trust engine 110 will not need to rely on the availability of the certificate authority 115. For example, when the user desires to send a digital certificate, or a third party requests the user's digital certificate, the request for the digital certificate is typically sent to the certificate authority 115. However, if the certificate authority 115 is conducting maintenance or has been victim of a failure or security compromise, the digital certificate may not be available.

[0156] At any time after issuing the cryptographic keys, the cryptographic engine 220 may advantageously employ the data splitting process 800 described above such that the cryptographic keys are split into independently undecipherable randomized numbers. Similar to the authentication data, at step 965 the cryptographic engine 220 transfers the randomized numbers to the data storage facilities D1 through D4.

[0157] A skilled artisan will recognize from the disclosure herein that the user may request a digital certificate anytime after enrollment. Moreover, the communications between systems may advantageously include FULL SSL or public-key encryption technologies. Moreover, the enrollment process may issue multiple digital certificates from multiple certificate authorities, including one or more proprietary certificate authorities internal or external to the trust engine 110.

[0158] As disclosed in steps 935 through 960, one embodiment of the invention includes the request for a

certificate that is eventually stored on the trust engine 110. Because, according to one embodiment, the cryptographic handling module 625 issues the keys used by the trust engine 110, each certificate corresponds to a private key. Therefore, the trust engine 110 may advantageously provide for interoperability through monitoring the certificates owned by, or associated with, a user. For example, when the cryptographic engine 220 receives a request for a cryptographic function, the cryptographic handling module 625 may investigate the certificates owned by the requesting user to determine whether the user owns a private key matching the attributes of the request. When such a certificate exists, the cryptographic handling module 625 may use the certificate or the public or private keys associated therewith, to perform the requested function. When such a certificate does not exist, the cryptographic handling module 625 may advantageously and transparently perform a number of actions to attempt to remedy the lack of an appropriate key. For example, FIGURE 9B illustrates a flowchart of an interoperability process 970, which according to aspects of an embodiment of the invention, discloses the foregoing steps to ensure the cryptographic handling module 625 performs cryptographic functions using appropriate keys.

[0159] As shown in FIGURE 9B, the interoperability process 970 begins with step 972 where the cryptographic handling module 925 determines the type of certificate desired. According to one embodiment of the invention, the type of certificate may advantageously be specified in the request for cryptographic functions, or other data provided by the

requestor. According to another embodiment, the certificate type may be ascertained by the data format of the request. For example, the cryptographic handling module 925 may advantageously recognize the request corresponds to a particular type.

[0160] According to one embodiment, the certificate type may include one or more algorithm standards, for example, RSA, ELGAMAL, or the like. In addition, the certificate type may include one or more key types, such as symmetric keys, public keys, strong encryption keys such as 256 bit keys, less secure keys, or the like. Moreover, the certificate type may include upgrades or replacements of one or more of the foregoing algorithm standards or keys, one or more message or data formats, one or more data encapsulation or encoding schemes, such as Base 32 or Base 64. The certificate type may also include compatibility with one or more third-party cryptographic applications or interfaces, one or more communication protocols, or one or more certificate standards or protocols. A skilled artisan will recognize from the disclosure herein that other differences may exist in certificate types, and translations to and from those differences may be implemented as disclosed herein.

[0161] Once the cryptographic handling module 625 determines the certificate type, the interoperability process 970 proceeds to step 974, and determines whether the user owns a certificate matching the type determined in step 974. When the user owns a matching certificate, for example, the trust engine 110 has access to the matching certificate through, for example, prior storage thereof, the cryptographic handling module 825 knows that a matching private key

is also stored within the trust engine 110. For example, the matching private key may be stored within the depository 210 or depository system 700. The cryptographic handling module 625 may advantageously request the matching private key be assembled from, for example, the depository 210, and then in step 976, use the matching private key to perform cryptographic actions or functions. For example, as mentioned in the foregoing, the cryptographic handling module 625 may advantageously perform hashing, hash comparisons, data encryption or decryption, digital signature verification or creation, or the like.

[0162] When the user does not own a matching certificate, the interoperability process 970 proceeds to step 978 where the cryptographic handling module 625 determines whether the users owns a cross-certified certificate. According to one embodiment, cross-certification between certificate authorities occurs when a first certificate authority determines to trust certificates from a second certificate authority. In other words, the first certificate authority determines that certificates from the second certificate authority meets certain quality standards, and therefore, may be "certified" as equivalent to the first certificate authority's own certificates. Cross-certification becomes more complex when the certificate authorities issue, for example, certificates having levels of trust. For example, the first certificate authority may provide three levels of trust for a particular certificate, usually based on the degree of reliability in the enrollment process, while the second certificate authority may provide seven levels of trust. Cross-certification may

advantageously track which levels and which certificates from the second certificate authority may be substituted for which levels and which certificates from the first. When the foregoing cross-certification is done officially and publicly between two certification authorities, the mapping of certificates and levels to one another is often called "chaining."

[0163] According to another embodiment of the invention, the cryptographic handling module 625 may advantageously develop cross-certifications outside those agreed upon by the certificate authorities. For example, the cryptographic handling module 625 may access a first certificate authority's certificate practice statement (CPS), or other published policy statement, and using, for example, the authentication tokens required by particular trust levels, match the first certificate authority's certificates to those of another certificate authority.

[0164] When, in step 978, the cryptographic handling module 625 determines that the users owns a cross-certified certificate, the interoperability process 970 proceeds to step 976, and performs the cryptographic action or function using the cross-certified public key, private key, or both. Alternatively, when the cryptographic handling module 625 determines that the users does not own a cross-certified certificate, the interoperability process 970 proceeds to step 980, where the cryptographic handling module 625 selects a certificate authority that issues the requested certificate type, or a certificate cross-certified thereto. In step 982, the cryptographic handling module 625 determines whether the user enrollment authentication data,

discussed in the foregoing, meets the authentication requirements of the chosen certificate authority. For example, if the user enrolled over a network by, for example, answering demographic and other questions, the authentication data provided may establish a lower level of trust than a user providing biometric data and appearing before a third-party, such as, for example, a notary. According to one embodiment, the foregoing authentication requirements may advantageously be provided in the chosen authentication authority's CPS.

[0165] When the user has provided the trust engine 110 with enrollment authentication data meeting the requirements of chosen certificate authority, the interoperability process 970 proceeds to step 984, where the cryptographic handling module 825 acquires the certificate from the chosen certificate authority. According to one embodiment, the cryptographic handling module 625 acquires the certificate by following steps 945 through 960 of the enrollment process 900. For example, the cryptographic handling module 625 may advantageously employ one or more public keys from one or more of the key pairs already available to the cryptographic engine 220, to request the certificate from the certificate authority. According to another embodiment, the cryptographic handling module 625 may advantageously generate one or more new key pairs, and use the public keys corresponding thereto, to request the certificate from the certificate authority.

[0166] According to another embodiment, the trust engine 110 may advantageously include one or more certificate issuing modules capable of issuing one or more certificate types. According to this embodiment, the certificate issuing module may provide the

foregoing certificate. When the cryptographic handling module 625 acquires the certificate, the interoperability process 970 proceeds to step 976, and performs the cryptographic action or function using the public key, private key, or both corresponding to the acquired certificate.

[0167] When the user, in step 982, has not provided the trust engine 110 with enrollment authentication data meeting the requirements of chosen certificate authority, the cryptographic handling module 625 determines, in step 986 whether there are other certificate authorities that have different authentication requirements. For example, the cryptographic handling module 625 may look for certificate authorities having lower authentication requirements, but still issue the chosen certificates, or cross-certifications thereof.

[0168] When the foregoing certificate authority having lower requirements exists, the interoperability process 970 proceeds to step 980 and chooses that certificate authority. Alternatively, when no such certificate authority exists, in step 988, the trust engine 110 may request additional authentication tokens from the user. For example, the trust engine 110 may request new enrollment authentication data comprising, for example, biometric data. Also, the trust engine 110 may request the user appear before a trusted third party and provide appropriate authenticating credentials, such as, for example, appearing before a notary with a drivers license, social security card, bank card, birth certificate, military ID, or the like. When the trust engine 110 receives updated authentication data, the interoperability process 970

proceeds to step 984 and acquires the foregoing chosen certificate.

[0169] Through the foregoing interoperability process 970, the cryptographic handling module 625 advantageously provides seamless, transparent, translations and conversions between differing cryptographic systems. A skilled artisan will recognize from the disclosure herein, a wide number of advantages and implementations of the foregoing interoperable system. For example, the foregoing step 986 of the interoperability process 970 may advantageously include aspects of trust arbitrage, discussed in further detail below, where the certificate authority may under special circumstances accept lower levels of cross-certification. In addition, the interoperability process 970 may include ensuring interoperability between and employment of standard certificate revocations, such as employing certificate revocation lists (CRL), online certificate status protocols (OCSP), or the like.

[0170] FIGURE 10 illustrates a data flow of an authentication process 1000 according to aspects of an embodiment of the invention. According to one embodiment, the authentication process 1000 includes gathering current authentication data from a user and comparing that to the enrollment authentication data of the user. For example, the authentication process 1000 begins at step 1005 where a user desires to perform a transaction with, for example, a vendor. Such transactions may include, for example, selecting a purchase option, requesting access to a restricted area or device of the vendor system 120, or the like. At step 1010, a vendor provides the user with a

transaction ID and an authentication request. The transaction ID may advantageously include a 192 bit quantity having a 32 bit timestamp concatenated with a 128 bit random quantity, or a "nonce," concatenated with a 32 bit vendor specific constant. Such a transaction ID uniquely identifies the transaction such that copycat transactions can be refused by the trust engine 110.

[0171] The authentication request may advantageously include what level of authentication is needed for a particular transaction. For example, the vendor may specify a particular level of confidence that is required for the transaction at issue. If authentication cannot be made to this level of confidence, as will be discussed below, the transaction will not occur without either further authentication by the user to raise the level of confidence, or a change in the terms of the authentication between the vendor and the server. These issues are discussed more completely below.

[0172] According to one embodiment, the transaction ID and the authentication request may be advantageously generated by a vendor-side applet or other software program. In addition, the transmission of the transaction ID and authentication data may include one or more XML documents encrypted using conventional SSL technology, such as, for example, ½ SSL, or, in other words vendor-side authenticated SSL.

[0173] After the user system 105 receives the transaction ID and authentication request, the user system 105 gathers the current authentication data, potentially including current biometric information, from the user. The user system 105, at step 1015,

encrypts at least the current authentication data "B" and the transaction ID, with the public key of the authentication engine 215, and transfers that data to the trust engine 110. The transmission preferably comprises XML documents encrypted with at least conventional ½ SSL technology. In step 1020, the transaction engine 205 receives the transmission, preferably recognizes the data format or request in the URL or URI, and forwards the transmission to the authentication engine 215.

[0174] During steps 1015 and 1020, the vendor system 120, at step 1025, forwards the transaction ID and the authentication request to the trust engine 110, using the preferred FULL SSL technology. This communication may also include a vendor ID, although vendor identification may also be communicated through a non-random portion of the transaction ID. At steps 1030 and 1035, the transaction engine 205 receives the communication, creates a record in the audit trail, and generates a request for the user's enrollment authentication data to be assembled from the data storage facilities D1 through D4. At step 1040, the depository system 700 transfers the portions of the enrollment authentication data corresponding to the user to the authentication engine 215. At step 1045, the authentication engine 215 decrypts the transmission using its private key and compares the enrollment authentication data to the current authentication data provided by the user.

[0175] The comparison of step 1045 may advantageously apply heuristical context sensitive authentication, as referred to in the forgoing, and discussed in further detail below. For example, if the

biometric information received does not match perfectly, a lower confidence match results. In particular embodiments, the level of confidence of the authentication is balanced against the nature of the transaction and the desires of both the user and the vendor. Again, this is discussed in greater detail below.

[0176] At step 1050, the authentication engine 215 fills in the authentication request with the result of the comparison of step 1045. According to one embodiment of the invention, the authentication request is filled with a YES/NO or TRUE/FALSE result of the authentication process 1000. In step 1055 the filled-in authentication request is returned to the vendor for the vendor to act upon, for example, allowing the user to complete the transaction that initiated the authentication request. According to one embodiment, a confirmation message is passed to the user.

[0177] Based on the foregoing, the authentication process 1000 advantageously keeps sensitive data secure and produces results configured to maintain the integrity of the sensitive data. For example, the sensitive data is assembled only inside the authentication engine 215. For example, the enrollment authentication data is undecipherable until it is assembled in the authentication engine 215 by the data assembling module, and the current authentication data is undecipherable until it is unwrapped by the conventional SSL technology and the private key of the authentication engine 215. Moreover, the authentication result transmitted to the vendor does not include the sensitive data, and the user may not

even know whether he or she produced valid authentication data.

[0178] Although the authentication process 1000 is disclosed with reference to its preferred and alternative embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize from the disclosure herein, a wide number of alternatives for the authentication process 1000. For example, the vendor may advantageously be replaced by almost any requesting application, even those residing with the user system 105. For example, a client application, such as Microsoft Word, may use an application program interface (API) or a cryptographic API (CAPI) to request authentication before unlocking a document. Alternatively, a mail server, a network, a cellular phone, a personal or mobile computing device, a workstation, or the like, may all make authentication requests that can be filled by the authentication process 1000. In fact, after providing the foregoing trusted authentication process 1000, the requesting application or device may provide access to or use of a wide number of electronic or computer devices or systems.

[0179] Moreover, the authentication process 1000 may employ a wide number of alternative procedures in the event of authentication failure. For example, authentication failure may maintain the same transaction ID and request that the user reenter his or her current authentication data. As mentioned in the foregoing, use of the same transaction ID allows the comparator of the authentication engine 215 to monitor and limit the number of authentication attempts for a

particular transaction, thereby creating a more secure cryptographic system 100.

[0180] In addition, the authentication process 1000 may be advantageously be employed to develop elegant single sign-on solutions, such as, unlocking a sensitive data vault. For example, successful or positive authentication may provide the authenticated user the ability to automatically access any number of passwords for an almost limitless number of systems and applications. For example, authentication of a user may provide the user access to password, login, financial credentials, or the like, associated with multiple online vendors, a local area network, various personal computing devices, Internet service providers, auction providers, investment brokerages, or the like. By employing a sensitive data vault, users may choose truly large and random passwords because they no longer need to remember them through association. Rather, the authentication process 1000 provides access thereto. For example, a user may choose a random alphanumeric string that is twenty plus digits in length rather than something associated with a memorable data, name, etc.

[0181] According to one embodiment, a sensitive data vault associated with a given user may advantageously be stored in the data storage facilities of the depository 210, or split and stored in the depository system 700. According to this embodiment, after positive user authentication, the trust engine 110 serves the requested sensitive data, such as, for example, to the appropriate password to the requesting application. According to another embodiment, the trust engine 110 may include a separate system for storing the sensitive data vault. For example, the

trust engine 110 may include a stand-alone software engine implementing the data vault functionality and figuratively residing "behind" the foregoing front-end security system of the trust engine 110. According to this embodiment, the software engine serves the requested sensitive data after the software engine receives a signal indicating positive user authentication from the trust engine 110.

[0182] In yet another embodiment, the data vault may be implemented by a third-party system. Similar to the software engine embodiment, the third-party system may advantageously serve the requested sensitive data after the third-party system receives a signal indicating positive user authentication from the trust engine 110. According to yet another embodiment, the data vault may be implemented on the user system 105. A user-side software engine may advantageously serve the foregoing data after receiving a signal indicating positive user authentication from the trust engine 110.

[0183] Although the foregoing data vaults are disclosed with reference to alternative embodiments, a skilled artisan will recognize from the disclosure herein, a wide number of additional implementations thereof. For example, a particular data vault may include aspects from some or all of the foregoing embodiments. In addition, any of the foregoing data vaults may employ one or more authentication requests at varying times. For example, any of the data vaults may require authentication every one or more transactions, periodically, every one or more sessions, every access to one or more Webpages or Websites, at one or more other specified intervals, or the like.

[0184] FIGURE 11 illustrates a data flow of a signing process 1100 according to aspects of an embodiment of the invention. As shown in FIGURE 11, the signing process 1100 includes steps similar to those of the authentication process 1000 described in the foregoing with reference to FIGURE 10. According to one embodiment of the invention, the signing process 1100 first authenticates the user and then performs one or more of several digital signing functions as will be discussed in further detail below. According to another embodiment, the signing process 1100 may advantageously store data related thereto, such as hashes of messages or documents, or the like. This data may advantageously be used in an audit or any other event, such as for example, when a participating party attempts to repudiate a transaction.

[0185] As shown in FIGURE 11, during the authentication steps, the user and vendor may advantageously agree on a message, such as, for example, a contract. During signing, the signing process 1100 advantageously ensures that the contract signed by the user is identical to the contract supplied by the vendor. Therefore, according to one embodiment, during authentication, the vendor and the user include a hash of their respective copies of the message or contract, in the data transmitted to the authentication engine 215. By employing only a hash of a message or contract, the trust engine 110 may advantageously store a significantly reduced amount of data, providing for a more efficient and cost effective cryptographic system. In addition, the stored hash may be advantageously compared to a hash of a document in question to determine whether the document in question

matches one signed by any of the parties. The ability to determine whether the document is identical to one relating to a transaction provides for additional evidence that can be used against a claim for repudiation by a party to a transaction.

[0186] In step 1103, the authentication engine 215 assembles the enrollment authentication data and compares it to the current authentication data provided by the user. When the comparator of the authentication engine 215 indicates that the enrollment authentication data matches the current authentication data, the comparator of the authentication engine 215 also compares the hash of the message supplied by the vendor to the hash of the message supplied by the user. Thus, the authentication engine 215 advantageously ensures that the message agreed to by the user is identical to that agreed to by the vendor.

[0187] In step 1105, the authentication engine 215 transmits a digital signature request to the cryptographic engine 220. According to one embodiment of the invention, the request includes a hash of the message or contract. However, a skilled artisan will recognize from the disclosure herein that the cryptographic engine 220 may encrypt virtually any type of data, including, but not limited to, video, audio, biometrics, images or text to form the desired digital signature. Returning to step 1105, the digital signature request preferably comprises an XML document communicated through conventional SSL technologies.

[0188] In step 1110, the authentication engine 215 transmits a request to each of the data storage facilities D1 through D4, such that each of the data storage facilities D1 through D4 transmit their

respective portion of the cryptographic key or keys corresponding to a signing party. According to another embodiment, the cryptographic engine 220 employs some or all of the steps of the interoperability process 970 discussed in the foregoing, such that the cryptographic engine 220 first determines the appropriate key or keys to request from the depository 210 or the depository system 700 for the signing party, and takes actions to provide appropriate matching keys. According to still another embodiment, the authentication engine 215 or the cryptographic engine 220 may advantageously request one or more of the keys associated with the signing party and stored in the depository 210 or depository system 700.

[0189] According to one embodiment, the signing party includes one or both the user and the vendor. In such case, the authentication engine 215 advantageously requests the cryptographic keys corresponding to the user and/or the vendor. According to another embodiment, the signing party includes the trust engine 110. In this embodiment, the trust engine 110 is certifying that the authentication process 1000 properly authenticated the user, vendor, or both. Therefore, the authentication engine 215 requests the cryptographic key of the trust engine 110, such as, for example, the key belonging to the cryptographic engine 220, to perform the digital signature. According to another embodiment, the trust engine 110 performs a digital notary-like function. In this embodiment, the signing party includes the user, vendor, or both, along with the trust engine 110. Thus, the trust engine 110 provides the digital signature of the user and/or vendor, and then indicates with its own digital

signature that the user and/or vendor were properly authenticated. In this embodiment, the authentication engine 215 may advantageously request assembly of the cryptographic keys corresponding to the user, the vendor, or both. According to another embodiment, the authentication engine 215 may advantageously request assembly of the cryptographic keys corresponding to the trust engine 110.

[0190] According to another embodiment, the trust engine 110 performs power of attorney-like functions. For example, the trust engine 110 may digitally sign the message on behalf of a third party. In such case, the authentication engine 215 requests the cryptographic keys associated with the third party. According to this embodiment, the signing process 1100 may advantageously include authentication of the third party, before allowing power of attorney-like functions. In addition, the authentication process 1000 may include a check for third party constraints, such as, for example, business logic or the like dictating when and in what circumstances a particular third-party's signature may be used.

[0191] Based on the foregoing, in step 1110, the authentication engine requested the cryptographic keys from the data storage facilities D1 through D4 corresponding to the signing party. In step 1115, the data storage facilities D1 through D4 transmit their respective portions of the cryptographic key corresponding to the signing party to the cryptographic engine 220. According to one embodiment, the foregoing transmissions include SSL technologies. According to another embodiment, the foregoing transmissions may

advantageously be super-encrypted with the public key of the cryptographic engine 220.

[0192] In step 1120, the cryptographic engine 220 assembles the foregoing cryptographic keys of the signing party and encrypts the message therewith, thereby forming the digital signature(s). In step 1125 of the signing process 1100, the cryptographic engine 220 transmits the digital signature(s) to the authentication engine 215. In step 1130, the authentication engine 215 transmits the filled-in authentication request along with a copy of the hashed message and the digital signature(s) to the transaction engine 205. In step 1135, the transaction engine 205 transmits a receipt comprising the transaction ID, an indication of whether the authentication was successful, and the digital signature(s), to the vendor. According to one embodiment, the foregoing transmission may advantageously include the digital signature of the trust engine 110. For example, the trust engine 110 may encrypt the hash of the receipt with its private key, thereby forming a digital signature to be attached to the transmission to the vendor.

[0193] According to one embodiment, the transaction engine 205 also transmits a confirmation message to the user. Although the signing process 1100 is disclosed with reference to its preferred and alternative embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize from the disclosure herein, a wide number of alternatives for the signing process 1100. For example, the vendor may be replaced with a user application, such as an email application. For

example, the user may wish to digitally sign a particular email with his or her digital signature. In such an embodiment, the transmission throughout the signing process 1100 may advantageously include only one copy of a hash of the message. Moreover, a skilled artisan will recognize from the disclosure herein that a wide number of client applications may request digital signatures. For example, the client applications may comprise word processors, spreadsheets, emails, voicemail, access to restricted system areas, or the like.

[0194] In addition, a skilled artisan will recognize from the disclosure herein that steps 1105 through 1120 of the signing process 1100 may advantageously employ some or all of the steps of the interoperability process 970 of FIGURE 9B, thereby providing interoperability between differing cryptographic systems that may, for example, need to process the digital signature under differing signature types.

[0195] FIGURE 12 illustrates a data flow of an encryption/decryption process 1200 according to aspects of an embodiment of the invention. As shown in FIGURE 12, the decryption process 1200 begins by authenticating the user using the authentication process 1000. According to one embodiment, the authentication process 1000 includes in the authentication request, a synchronous session key. For example, in conventional PKI technologies, it is understood by skilled artisans that encrypting or decrypting data using public and private keys is mathematically intensive and may require significant system resources. However, in symmetric key cryptographic systems, or systems where the sender and

receiver of a message share a single common key that is used to encrypt and decrypt a message, the mathematical operations are significantly simpler and faster. Thus, in the conventional PKI technologies, the sender of a message will generate synchronous session key, and encrypt the message using the simpler, faster symmetric key system. Then, the sender will encrypt the session key with the public key of the receiver. The encrypted session key will be attached to the synchronously encrypted message and both data are sent to the receiver. The receiver uses his or her private key to decrypt the session key, and then uses the session key to decrypt the message. Based on the foregoing, the simpler and faster symmetric key system is used for the majority of the encryption/decryption processing. Thus, in the decryption process 1200, the decryption advantageously assumes that a synchronous key has been encrypted with the public key of the user. Thus, as mentioned in the foregoing, the encrypted session key is included in the authentication request.

[0196] Returning to the decryption process 1200, after the user has been authenticated in step 1205, the authentication engine 215 forwards the encrypted session key to the cryptographic engine 220. In step 1210, the authentication engine 215 forwards a request to each of the data storage facilities, D1 through D4, requesting the cryptographic key data of the user. In step 1215, each data storage facility, D1 through D4, transmits their respective portion of the cryptographic key to the cryptographic engine 220. According to one embodiment, the foregoing transmission is encrypted with the public key of the cryptographic engine 220.

[0197] In step 1220 of the decryption process 1200, the cryptographic engine 220 assembles the cryptographic key and decrypts the session key therewith. In step 1225, the cryptographic engine forwards the session key to the authentication engine 215. In step 1227, the authentication engine 215 fills in the authentication request including the decrypted session key, and transmits the filled-in authentication request to the transaction engine 205. In step 1230, the transaction engine 205 forwards the authentication request along with the session key to the requesting application or vendor. Then, according to one embodiment, the requesting application or vendor uses the session key to decrypt the encrypted message.

[0198] Although the decryption process 1200 is disclosed with reference to its preferred and alternative embodiments, a skilled artisan will recognize from the disclosure herein, a wide number of alternatives for the decryption process 1200. For example, the decryption process 1200 may forego synchronous key encryption and rely on full public-key technology. In such an embodiment, the requesting application may transmit the entire message to the cryptographic engine 220, or, may employ some type of compression or reversible hash in order to transmit the message to the cryptographic engine 220. A skilled artisan will also recognize from the disclosure herein that the foregoing communications may advantageously include XML documents wrapped in SSL technology.

[0199] The encryption/decryption process 1200 also provides for encryption of documents or other data. Thus, in step 1235, a requesting application or vendor may advantageously transmit to the transaction engine

205 of the trust engine 110, a request for the public key of the user. The requesting application or vendor makes this request because the requesting application or vendor uses the public key of the user, for example, to encrypt the session key that will be used to encrypt the document or message. As mentioned in the enrollment process 900, the transaction engine 205 stores a copy of the digital certificate of the user, for example, in the mass storage 225. Thus, in step 1240 of the encryption process 1200, the transaction engine 205 requests the digital certificate of the user from the mass storage 225. In step 1245, the mass storage 225 transmits the digital certificate corresponding to the user, to the transaction engine 205. In step 1250, the transaction engine 205 transmits the digital certificate to the requesting application or vendor. According to one embodiment, the encryption portion of the encryption process 1200 does not include the authentication of a user. This is because the requesting vendor needs only the public key of the user, and is not requesting any sensitive data.

[0200] A skilled artisan will recognize from the disclosure herein that if a particular user does not have a digital certificate, the trust engine 110 may employ some or all of the enrollment process 900 in order to generate a digital certificate for that particular user. Then, the trust engine 110 may initiate the encryption/decryption process 1200 and thereby provide the appropriate digital certificate. In addition, a skilled artisan will recognize from the disclosure herein that steps 1220 and 1235 through 1250 of the encryption/decryption process 1200 may advantageously employ some or all of the steps of the

interoperability process of FIGURE 9B, thereby providing interoperability between differing cryptographic systems that may, for example, need to process the encryption.

[0201] FIGURE 13 illustrates a simplified block diagram of a trust engine system 1300 according to aspects of yet another embodiment of the invention. As shown in FIGURE 13, the trust engine system 1300 comprises a plurality of distinct trust engines 1305, 1310, 1315, and 1320, respectively. To facilitate a more complete understanding of the invention, FIGURE 13 illustrates each trust engine, 1305, 1310, 1315, and 1320 as having a transaction engine, a depository, and an authentication engine. However, a skilled artisan will recognize that each transaction engine may advantageously comprise some, a combination, or all of the elements and communication channels disclosed with reference to FIGURES 1-8. For example, one embodiment may advantageously include trust engines having one or more transaction engines, depositories, and cryptographic servers or any combinations thereof.

[0202] According to one embodiment of the invention, each of the trust engines 1305, 1310, 1315 and 1320 are geographically separated, such that, for example, the trust engine 1305 may reside in a first location, the trust engine 1310 may reside in a second location, the trust engine 1315 may reside in a third location, and the trust engine 1320 may reside in a fourth location. The foregoing geographic separation advantageously decreases system response time while increasing the security of the overall trust engine system 1300.

[0203] For example, when a user logs onto the cryptographic system 100, the user may be nearest the

first location and may desire to be authenticated. As described with reference to FIGURE 10, to be authenticated, the user provides current authentication data, such as a biometric or the like, and the current authentication data is compared to that user's enrollment authentication data. Therefore, according to one example, the user advantageously provides current authentication data to the geographically nearest trust engine 1305. The transaction engine 1321 of the trust engine 1305 then forwards the current authentication data to the authentication engine 1322 also residing at the first location. According to another embodiment, the transaction engine 1321 forwards the current authentication data to one or more of the authentication engines of the trust engines 1310, 1315, or 1320.

[0204] The transaction engine 1321 also requests the assembly of the enrollment authentication data from the depositories of, for example, each of the trust engines, 1305 through 1320. According to this embodiment, each depository provides its portion of the enrollment authentication data to the authentication engine 1322 of the trust engine 1305. The authentication engine 1322 then employs the encrypted data portions from, for example, the first two depositories to respond, and assembles the enrollment authentication data into deciphered form. The authentication engine 1322 compares the enrollment authentication data with the current authentication data and returns an authentication result to the transaction engine 1321 of the trust engine 1305.

[0205] Based on the above, the trust engine system 1300 employs the nearest one of a plurality of

geographically separated trust engines, 1305 through 1320, to perform the authentication process. According to one embodiment of the invention, the routing of information to the nearest transaction engine may advantageously be performed at client-side applets executing on one or more of the user system 105, vendor system 120, or certificate authority 115. According to an alternative embodiment, a more sophisticated decision process may be employed to select from the trust engines 1305 through 1320. For example, the decision may be based on the availability, operability, speed of connections, load, performance, geographic proximity, or a combination thereof, of a given trust engine.

[0206] In this way, the trust engine system 1300 lowers its response time while maintaining the security advantages associated with geographically remote data storage facilities, such as those discussed with reference to FIGURE 7 where each data storage facility stores randomized portions of sensitive data. For example, a security compromise at, for example, the depository 1325 of the trust engine 1315 does not necessarily compromise the sensitive data of the trust engine system 1300. This is because the depository 1325 contains only non-decipherable randomized data that, without more, is entirely useless.

[0207] According to another embodiment, the trust engine system 1300 may advantageously include multiple cryptographic engines arranged similar to the authentication engines. The cryptographic engines may advantageously perform cryptographic functions such as those disclosed with reference to FIGURES 1-8. According to yet another embodiment, the trust engine

system 1300 may advantageously replace the multiple authentication engines with multiple cryptographic engines, thereby performing cryptographic functions such as those disclosed with reference to FIGURES 1-8. According to yet another embodiment of the invention, the trust engine system 1300 may replace each multiple authentication engine with an engine having some or all of the functionality of the authentication engines, cryptographic engines, or both, as disclosed in the foregoing,

[0208] Although the trust engine system 1300 is disclosed with reference to its preferred and alternative embodiments, a skilled artisan will recognize that the trust engine system 1300 may comprise portions of trust engines 1305 through 1320. For example, the trust engine system 1300 may include one or more transaction engines, one or more depositories, one or more authentication engines, or one or more cryptographic engines or combinations thereof.

[0209] FIGURE 14 illustrates a simplified block diagram of a trust engine System 1400 according to aspects of yet another embodiment of the invention. As shown in FIGURE 14, the trust engine system 1400 includes multiple trust engines 1405, 1410, 1415 and 1420. According to one embodiment, each of the trust engines 1405, 1410, 1415 and 1420, comprise some or all of the elements of trust engine 110 disclosed with reference to FIGURES 1-8. According to this embodiment, when the client side applets of the user system 105, the vendor system 120, or the certificate authority 115, communicate with the trust engine system 1400, those communications are sent to the IP address

of each of the trust engines 1405 through 1420. Further, each transaction engine of each of the trust engines, 1405, 1410, 1415, and 1420, behaves similar to the transaction engine 1321 of the trust engine 1305 disclosed with reference to FIGURE 13. For example, during an authentication process, each transaction engine of each of the trust engines 1405, 1410, 1415, and 1420 transmits the current authentication data to their respective authentication engines and transmits a request to assemble the randomized data stored in each of the depositories of each of the trust engines 1405 through 1420. FIGURE 14 does not illustrate all of these communications; as such illustration would become overly complex. Continuing with the authentication process, each of the depositories then communicates its portion of the randomized data to each of the authentication engines of the each of the trust engines 1405 through 1420. Each of the authentication engines of the each of the trust engines employs its comparator to determine whether the current authentication data matches the enrollment authentication data provided by the depositories of each of the trust engines 1405 through 1420. According to this embodiment, the result of the comparison by each of the authentication engines is then transmitted to a redundancy module of the other three trust engines. For example, the result of the authentication engine from the trust engine 1405 is transmitted to the redundancy modules of the trust engines 1410, 1415, and 1420. Thus, the redundancy module of the trust engine 1405 likewise receives the result of the authentication engines from the trust engines 1410, 1415, and 1420.

[0210] FIGURE 15 illustrates a block diagram of the redundancy module of FIGURE 14. The redundancy module comprises a comparator configured to receive the authentication result from three authentication engines and transmit that result to the transaction engine of the fourth trust engine. The comparator compares the authentication result form the three authentication engines, and if two of the results agree, the comparator concludes that the authentication result should match that of the two agreeing authentication engines. This result is then transmitted back to the transaction engine corresponding to the trust engine not associated with the three authentication engines.

[0211] Based on the foregoing, the redundancy module determines an authentication result from data received from authentication engines that are preferably geographically remote from the trust engine of that the redundancy module. By providing such redundancy functionality, the trust engine system 1400 ensures that a compromise of the authentication engine of one of the trust engines 1405 through 1420, is insufficient to compromise the authentication result of the redundancy module of that particular trust engine. A skilled artisan will recognize that redundancy module functionality of the trust engine system 1400 may also be applied to the cryptographic engine of each of the trust engines 1405 through 1420. However, such cryptographic engine communication was not shown in FIGURE 14 to avoid complexity. Moreover, a skilled artisan will recognize a wide number of alternative authentication result conflict resolution algorithms for the comparator of FIGURE 15 are suitable for use in the present invention.

[0212] According to yet another embodiment of the invention, the trust engine system 1400 may advantageously employ the redundancy module during cryptographic comparison steps. For example, some or all of the foregoing redundancy module disclosure with reference to FIGURES 14 and 15 may advantageously be implemented during a hash comparison of documents provided by one or more parties during a particular transaction.

[0213] Although the foregoing invention has been described in terms of certain preferred and alternative embodiments, other embodiments will be apparent to those of ordinary skill in the art from the disclosure herein. For example, the trust engine 110 may issue short-term certificates, where the private cryptographic key is released to the user for a predetermined period of time. For example, current certificate standards include a validity field that can be set to expire after a predetermined amount of time. Thus, the trust engine 110 may release a private key to a user where the private key would be valid for, for example, 24 hours. According to such an embodiment, the trust engine 110 may advantageously issue a new cryptographic key pair to be associated with a particular user and then release the private key of the new cryptographic key pair. Then, once the private cryptographic key is released, the trust engine 110 immediately expires any internal valid use of such private key, as it is no longer securable by the trust engine 110.

[0214] In addition, a skilled artisan will recognize that the cryptographic system 100 or the trust engine 110 may include the ability to recognize any type of

devices, such as, but not limited to, a laptop, a cell phone, a network, a biometric device or the like. According to one embodiment, such recognition may come from data supplied in the request for a particular service, such as, a request for authentication leading to access or use, a request for cryptographic functionality, or the like. According to one embodiment, the foregoing request may include a unique device identifier, such as, for example, a processor ID. Alternatively, the request may include data in a particular recognizable data format. For example, mobile and satellite phones often do not include the processing power for full X509.v3 heavy encryption certificates, and therefore do not request them. According to this embodiment, the trust engine 110 may recognize the type of data format presented, and respond only in kind.

[0215] In an additional aspect of the system described above, context sensitive authentication can be provided using various techniques as will be described below. Context sensitive authentication, for example as shown in FIGURE 16, provides the possibility of evaluating not only the actual data which is sent by the user when attempting to authenticate himself, but also the circumstances surrounding the generation and delivery of that data. Such techniques may also support transaction specific trust arbitrage between the user and trust engine 110 or between the vendor and trust engine 110, as will be described below.

[0216] As discussed above, authentication is the process of proving that a user is who he says he is. Generally, authentication requires demonstrating some fact to an authentication authority. The trust engine

110 of the present invention represents the authority to which a user must authenticate himself. The user must demonstrate to the trust engine 110 that he is who he says he is by either: knowing something that only the user should know (knowledge-based authentication), having something that only the user should have (token-based authentication), or by being something that only the user should be (biometric-based authentication).

[0217] Examples of knowledge-based authentication include without limitation a password, PIN number, or lock combination. Examples of token-based authentication include without limitation a house key, a physical credit card, a driver's license, or a particular phone number. Examples of biometric-based authentication include without limitation a fingerprint, handwriting analysis, facial scan, hand scan, ear scan, iris scan, vascular pattern, DNA, a voice analysis, or a retinal scan.

[0218] Each type of authentication has particular advantages and disadvantages, and each provides a different level of security. For example, it is generally harder to create a false fingerprint that matches someone else's than it is to overhear someone's password and repeat it. Each type of authentication also requires a different type of data to be known to the authenticating authority in order to verify someone using that form of authentication.

[0219] As used herein, "authentication" will refer broadly to the overall process of verifying someone's identity to be who he says he is. An "authentication technique" will refer to a particular type of authentication based upon a particular piece of

knowledge, physical token, or biometric reading. "Authentication data" refers to information which is sent to or otherwise demonstrated to an authentication authority in order to establish identity. "Enrollment data" will refer to the data which is initially submitted to an authentication authority in order to establish a baseline for comparison with authentication data. An "authentication instance" will refer to the data associated with an attempt to authenticate by an authentication technique.

[0220] The internal protocols and communications involved in the process of authenticating a user is described with reference to FIGURE 10 above. The part of this process within which the context sensitive authentication takes place occurs within the comparison step shown as step 1045 of FIGURE 10. This step takes place within the authentication engine 215 and involves assembling the enrollment data 410 retrieved from the depository 210 and comparing the authentication data provided by the user to it. One particular embodiment of this process is shown in FIGURE 16 and described below.

[0221] The current authentication data provided by the user and the enrollment data retrieved from the depository 210 are received by the authentication engine 215 in step 1600 of FIGURE 16. Both of these sets of data may contain data which is related to separate techniques of authentication. The authentication engine 215 separates the authentication data associated with each individual authentication instance in step 1605. This is necessary so that the authentication data is compared with the appropriate subset of the enrollment data for the user (e.g.

fingerprint authentication data should be compared with fingerprint enrollment data, rather than password enrollment data).

[0222] Generally, authenticating a user involves one or more individual authentication instances, depending on which authentication techniques are available to the user. These methods are limited by the enrollment data which were provided by the user during his enrollment process (if the user did not provide a retinal scan when enrolling, he will not be able to authenticate himself using a retinal scan), as well as the means which may be currently available to the user (e.g. if the user does not have a fingerprint reader at his current location, fingerprint authentication will not be practical). In some cases, a single authentication instance may be sufficient to authenticate a user; however, in certain circumstances a combination of multiple authentication instances may be used in order to more confidently authenticate a user for a particular transaction.

[0223] Each authentication instance consists of data related to a particular authentication technique (e.g. fingerprint, password, smart card, etc.) and the circumstances which surround the capture and delivery of the data for that particular technique. For example, a particular instance of attempting to authenticate via password will generate not only the data related to the password itself, but also circumstantial data, known as "metadata", related to that password attempt. This circumstantial data includes information such as: the time at which the particular authentication instance took place, the network address from which the authentication

information was delivered, as well as any other information as is known to those of skill in the art which may be determined about the origin of the authentication data (the type of connection, the processor serial number, etc.).

[0224] In many cases, only a small amount of circumstantial metadata will be available. For example, if the user is located on a network which uses proxies or network address translation or another technique which masks the address of the originating computer, only the address of the proxy or router may be determined. Similarly, in many cases information such as the processor serial number will not be available because of either limitations of the hardware or operating system being used, disabling of such features by the operator of the system, or other limitations of the connection between the user's system and the trust engine 110.

[0225] As shown in FIGURE 16, once the individual authentication instances represented within the authentication data are extracted and separated in step 1605, the authentication engine 215 evaluates each instance for its reliability in indicating that the user is who he claims to be. The reliability for a single authentication instance will generally be determined based on several factors. These may be grouped as factors relating to the reliability associated with the authentication technique, which are evaluated in step 1610, and factors relating to the reliability of the particular authentication data provided, which are evaluated in step 1815. The first group includes without limitation the inherent reliability of the authentication technique being used,

and the reliability of the enrollment data being used with that method. The second group includes without limitation the degree of match between the enrollment data and the data provided with the authentication instance, and the metadata associated with that authentication instance. Each of these factors may vary independently of the others.

[0226] The inherent reliability of an authentication technique is based on how hard it is for an imposter to provide someone else's correct data, as well as the overall error rates for the authentication technique. For passwords and knowledge based authentication methods, this reliability is often fairly low because there is nothing that prevents someone from revealing their password to another person and for that second person to use that password. Even a more complex knowledge based system may have only moderate reliability since knowledge may be transferred from person to person fairly easily. Token based authentication, such as having a proper smart card or using a particular terminal to perform the authentication, is similarly of low reliability used by itself, since there is no guarantee that the right person is in possession of the proper token.

[0227] However, biometric techniques are more inherently reliable because it is generally difficult to provide someone else with the ability to use your fingerprints in a convenient manner, even intentionally. Because subverting biometric authentication techniques is more difficult, the inherent reliability of biometric methods is generally higher than that of purely knowledge or token based authentication techniques. However, even biometric

techniques may have some occasions in which a false acceptance or false rejection is generated. These occurrences may be reflected by differing reliabilities for different implementations of the same biometric technique. For example, a fingerprint matching system provided by one company may provide a higher reliability than one provided by a different company because one uses higher quality optics or a better scanning resolution or some other improvement which reduces the occurrence of false acceptances or false rejections.

[0228] Note that this reliability may be expressed in different manners. The reliability is desirably expressed in some metric which can be used by the heuristics 530 and algorithms of the authentication engine 215 to calculate the confidence level of each authentication. One preferred mode of expressing these reliabilities is as a percentage or fraction. For instance, fingerprints might be assigned an inherent reliability of 97%, while passwords might only be assigned an inherent reliability of 50%. Those of skill in the art will recognize that these particular values are merely exemplary and may vary between specific implementations.

[0229] The second factor for which reliability must be assessed is the reliability of the enrollment. This is part of the "graded enrollment" process referred to above. This reliability factor reflects the reliability of the identification provided during the initial enrollment process. For instance, if the individual initially enrolls in a manner where they physically produce evidence of their identity to a notary or other public official, and enrollment data is

recorded at that time and notarized, the data will be more reliable than data which is provided over a network during enrollment and only vouched for by a digital signature or other information which is not truly tied to the individual.

[0230] Other enrollment techniques with varying levels of reliability include without limitation: enrollment at a physical office of the trust engine 110 operator; enrollment at a user's place of employment; enrollment at a post office or passport office; enrollment through an affiliated or trusted party to the trust engine 110 operator; anonymous or pseudonymous enrollment in which the enrolled identity is not yet identified with a particular real individual, as well as such other means as are known in the art.

[0231] These factors reflect the trust between the trust engine 110 and the source of identification provided during the enrollment process. For instance, if enrollment is performed in association with an employer during the initial process of providing evidence of identity, this information may be considered extremely reliable for purposes within the company, but may be trusted to a lesser degree by a government agency, or by a competitor. Therefore, trust engines operated by each of these other organizations may assign different levels of reliability to this enrollment.

[0232] Similarly, additional data which is submitted across a network, but which is authenticated by other trusted data provided during a previous enrollment with the same trust engine 110 may be considered as reliable as the original enrollment data was, even though the

latter data were submitted across an open network. In such circumstances, a subsequent notarization will effectively increase the level of reliability associated with the original enrollment data. In this way for example, an anonymous or pseudonymous enrollment may then be raised to a full enrollment by demonstrating to some enrollment official the identity of the individual matching the enrolled data.

[0233] The reliability factors discussed above are generally values which may be determined in advance of any particular authentication instance. This is because they are based upon the enrollment and the technique, rather than the actual authentication. In one embodiment, the step of generating reliability based upon these factors involves looking up previously determined values for this particular authentication technique and the enrollment data of the user. In a further aspect of an advantageous embodiment of the present invention, such reliabilities may be included with the enrollment data itself. In this way, these factors are automatically delivered to the authentication engine 215 along with the enrollment data sent from the depository 210.

[0234] While these factors may generally be determined in advance of any individual authentication instance, they still have an effect on each authentication instance which uses that particular technique of authentication for that user. Furthermore, although the values may change over time (e.g. if the user re-enrolls in a more reliable fashion), they are not dependent on the authentication data itself. By contrast, the reliability factors associated with a single specific instance's data may

vary on each occasion. These factors, as discussed below, must be evaluated for each new authentication in order to generate reliability scores in step 1815.

[0235] The reliability of the authentication data reflects the match between the data provided by the user in a particular authentication instance and the data provided during the authentication enrollment. This is the fundamental question of whether the authentication data matches the enrollment data for the individual the user is claiming to be. Normally, when the data do not match, the user is considered to not be successfully authenticated, and the authentication fails. The manner in which this is evaluated may change depending on the authentication technique used. The comparison of such data is performed by the comparator 515 function of the authentication engine 215 as shown in FIGURE 5.

[0236] For instance, matches of passwords are generally evaluated in a binary fashion. In other words, a password is either a perfect match, or a failed match. It is usually not desirable to accept as even a partial match a password which is close to the correct password if it is not exactly correct. Therefore, when evaluating a password authentication, the reliability of the authentication returned by the comparator 515 is typically either 100% (correct) or 0% (wrong), with no possibility of intermediate values.

[0237] Similar rules to those for passwords are generally applied to token based authentication methods, such as smart cards. This is because having a smart card which has a similar identifier or which is similar to the correct one, is still just as wrong as having any other incorrect token. Therefore tokens

tend also to be binary authenticators: a user either has the right token, or he doesn't.

[0238] However, certain types of authentication data, such as questionnaires and biometrics, are generally not binary authenticators. For example, a fingerprint may match a reference fingerprint to varying degrees. To some extent, this may be due to variations in the quality of the data captured either during the initial enrollment or in subsequent authentications. (A fingerprint may be smudged or a person may have a still healing scar or burn on a particular finger.) In other instances the data may match less than perfectly because the information itself is somewhat variable and based upon pattern matching. (A voice analysis may seem close but not quite right because of background noise, or the acoustics of the environment in which the voice is recorded, or because the person has a cold.) Finally, in situations where large amounts of data are being compared, it may simply be the case that much of the data matches well, but some doesn't. (A ten-question questionnaire may have resulted in eight correct answers to personal questions, but two incorrect answers.) For any of these reasons, the match between the enrollment data and the data for a particular authentication instance may be desirably assigned a partial match value by the comparator 515. In this way, the fingerprint might be said to be a 85% match, the voice print a 65% match, and the questionnaire an 80% match, for example.

[0239] This measure (degree of match) produced by the comparator 515 is the factor representing the basic issue of whether an authentication is correct or not.

However, as discussed above, this is only one of the factors which may be used in determining the reliability of a given authentication instance. Note also that even though a match to some partial degree may be determined, that ultimately, it may be desirable to provide a binary result based upon a partial match. In an alternate mode of operation, it is also possible to treat partial matches as binary, i.e. either perfect (100%) or failed (0%) matches, based upon whether or not the degree of match passes a particular threshold level of match. Such a process may be used to provide a simple pass/fail level of matching for systems which would otherwise produce partial matches.

[0240] Another factor to be considered in evaluating the reliability of a given authentication instance concerns the circumstances under which the authentication data for this particular instance are provided. As discussed above, the circumstances refer to the metadata associated with a particular authentication instance. This may include without limitation such information as: the network address of the authenticator, to the extent that it can be determined; the time of the authentication; the mode of transmission of the authentication data (phone line, cellular, network, etc.); and the serial number of the system of the authenticator.

[0241] These factors can be used to produce a profile of the type of authentication that is normally requested by the user. Then, this information can be used to assess reliability in at least two manners. One manner is to consider whether the user is requesting authentication in a manner which is consistent with the normal profile of authentication by

this user. If the user normally makes authentication requests from one network address during business days (when she is at work) and from a different network address during evenings or weekends (when she is at home), an authentication which occurs from the home address during the business day is less reliable because it is outside the normal authentication profile. Similarly, if the user normally authenticates using a fingerprint biometric and in the evenings, an authentication which originates during the day using only a password is less reliable.

[0242] An additional way in which the circumstantial metadata can be used to evaluate the reliability of an instance of authentication is to determine how much corroboration the circumstance provides that the authenticator is the individual he claims to be. For instance, if the authentication comes from a system with a serial number known to be associated with the user, this is a good circumstantial indicator that the user is who they claim to be. Conversely, if the authentication is coming from a network address which is known to be in Los Angeles when the user is known to reside in London, this is an indication that this authentication is less reliable based on its circumstances.

[0243] It is also possible that a cookie or other electronic data may be placed upon the system being used by a user when they interact with a vendor system or with the trust engine 110. This data is written to the storage of the system of the user and may contain an identification which may be read by a Web browser or other software on the user system. If this data is allowed to reside on the user system between sessions

(a "persistent cookie"), it may be sent with the authentication data as further evidence of the past use of this system during authentication of a particular user. In effect, the metadata of a given instance, particularly a persistent cookie, may form a sort of token based authenticator itself.

[0244] Once the appropriate reliability factors based on the technique and data of the authentication instance are generated as described above in steps 1610 and 1615 respectively, they are used to produce an overall reliability for the authentication instance provided in step 1620. One means of doing this is simply to express each reliability as a percentage and then to multiply them together.

[0245] For example, suppose the authentication data is being sent in from a network address known to be the user's home computer completely in accordance with the user's past authentication profile (100%), and the technique being used is fingerprint identification (97%), and the initial finger print data was roistered through the user's employer with the trust engine 110 (90%), and the match between the authentication data and the original fingerprint template in the enrollment data is very good (99%). The overall reliability of this authentication instance could then be calculated as the product of these reliabilities: 100% * 97% * 90% * 99% - 86.4% reliability.

[0246] This calculated reliability represents the reliability of one single instance of authentication. The overall reliability of a single authentication instance may also be calculated using techniques which treat the different reliability factors differently, for example by using formulas where different weights

are assigned to each reliability factor. Furthermore, those of skill in the art will recognize that the actual values used may represent values other than percentages and may use non-arithmetic systems. One embodiment may include a module used by an authentication requestor to set the weights for each factor and the algorithms used in establishing the overall reliability of the authentication instance.

[0247] The authentication engine 215 may use the above techniques and variations thereof to determine the reliability of a single authentication instance, indicated as step 1620. However, it may be useful in many authentication situations for multiple authentication instances to be provided at the same time. For example, while attempting to authenticate himself using the system of the present invention, a user may provide a user identification, fingerprint authentication data, a smart card, and a password. In such a case, three independent authentication instances are being provided to the trust engine 110 for evaluation. Proceeding to step 1625, if the authentication engine 215 determines that the data provided by the user includes more than one authentication instance, then each instance in turn will be selected as shown in step 1630 and evaluated as described above in steps 1610, 1615 and 1620.

[0248] Note that many of the reliability factors discussed may vary from one of these instances to another. For instance, the inherent reliability of these techniques is likely to be different, as well as the degree of match provided between the authentication data and the enrollment data. Furthermore, the user may have provided enrollment data at different times

and under different circumstances for each of these techniques, providing different enrollment reliabilities for each of these instances as well. Finally, even though the circumstances under which the data for each of these instances is being submitted is the same, the use of such techniques may each fit the profile of the user differently, and so may be assigned different circumstantial reliabilities. (For example, the user may normally use their password and fingerprint, but not their smart card.)

[0249] As a result, the final reliability for each of these authentication instances may be different from One another. However, by using multiple instances together, the overall confidence level for the authentication will tend to increase.

[0250] Once the authentication engine has performed steps 1610 through 1620 for all of the authentication instances provided in the authentication data, the reliability of each instance is used in step 1635 to evaluate the overall authentication confidence level. This process of combining the individual authentication instance reliabilities into the authentication confidence level may be modeled by various methods relating the individual reliabilities produced, and may also address the particular interaction between some of these authentication techniques. (For example, multiple knowledge-based systems such as passwords may produce less confidence than a single password and even a fairly weak biometric, such as a basic voice analysis.)

[0251] One means in which the authentication engine 215 may combine the reliabilities of multiple concurrent authentication instances to generate a final

confidence level is to multiply the unreliability of each instance to arrive at a total unreliability. The unreliability is generally the complementary percentage of the reliability. For example, a technique which is 84% reliable is 16% unreliable. The three authentication instances described above (fingerprint, smart card, password)which produce reliabilities of 86%, 75%, and 72% would have corresponding unreliabilities of (100- 86)%, (100- 75)% and (100- 72)%, or 14%, 25%, and 28%, respectively. By multiplying these unreliabilities, we get a cumulative unreliability of 14% * 25% * 28% - .98% unreliability, which corresponds to a reliability of 99.02%.

[0252] In an additional mode of operation, additional factors and heuristics 530 may be applied within the authentication engine 215 to account for the interdependence of various authentication techniques. For example, if someone has unauthorized access to a particular home computer, they probably have access to the phone line at that address as well. Therefore, authenticating based on an originating phone number as well as upon the serial number of the authenticating system does not add much to the overall confidence in the authentication. However, knowledge based authentication is largely independent of token based authentication (i.e. if someone steals your cellular phone or keys, they are no more likely to know your PIN or password than if they hadn't).

[0253] Furthermore, different vendors or other authentication requestors may wish to weigh different aspects of the authentication differently. This may include the use of separate weighing factors or algorithms used in calculating the reliability of

individual instances as well as the use of different means to evaluate authentication events with multiple instances.

[0254] For instance, vendors for certain types of transactions, for instance corporate email systems, may desire to authenticate primarily based upon heuristics and other circumstantial data by default. Therefore, they may apply high weights to factors related to the metadata and other profile related information associated with the circumstances surrounding authentication events. This arrangement could be used to ease the burden on users during normal operating hours, by not requiring more from the user than that he be logged on to the correct machine during business hours. However, another vendor may weigh authentications coming from a particular technique most heavily, for instance fingerprint matching, because of a policy decision that such a technique is most suited to authentication for the particular vendor's purposes.

[0255] Such varying weights may be defined by the authentication requestor in generating the authentication request and sent to the trust engine 110 with the authentication request in one mode of operation. Such options could also be set as preferences during an initial enrollment process for the authentication requestor and stored within the authentication engine in another mode of operation.

[0256] Once the authentication engine 215 produces an authentication confidence level for the authentication data provided, this confidence level is used to complete the authentication request in step 1640, and this information is forwarded from the authentication engine 215 to the transaction engine 205

for inclusion in a message to the authentication requestor.

[0257] The process described above is merely exemplary, and those of skill in the art will recognize that the steps need not be performed in the order shown or that only certain of the steps are desired to be performed, or that a variety of combinations of steps may be desired. Furthermore, certain steps, such as the evaluation of the reliability of each authentication instance provided, may be carried out in parallel with one another if circumstances permit.

[0258] In a further aspect of this invention, a method is provided to accommodate conditions when the authentication confidence level produced by the process described above fails to meet the required trust level of the vendor or other party requiring the authentication. In circumstances such as these where a gap exists between the level of confidence provided and the level of trust desired, the operator of the trust engine 110 is in a position to provide opportunities for one or both parties to provide alternate data or requirements in order to close this trust gap. This process will be referred to as "trust arbitrage" herein.

[0259] Trust arbitrage may take place within a framework of cryptographic authentication as described above with reference to FIGURES 10 and 11. As shown therein, a vendor or other party will request authentication of a particular user in association with a particular transaction. In one circumstance, the vendor simply requests an authentication, either positive or negative, and after receiving appropriate data from the user, the trust engine 110 will provide

such a binary authentication. In circumstances such as these, the degree of confidence required in order to secure a positive authentication is determined based upon preferences set within the trust engine 110.

[0260] However, it is also possible that the vendor may request a particular level of trust in order to complete a particular transaction. This required level may be included with the authentication request (e.g. authenticate this user to 98% confidence) or may be determined by the trust engine 110 based on other factors associated with the transaction (i.e. authenticate this user as appropriate for this transaction). One such factor might be the economic value of the transaction. For transactions which have greater economic value, a higher degree of trust may be required. Similarly, for transactions with high degrees of risk a high degree of trust may be required. Conversely, for transactions which are either of low risk or of low value, lower trust levels may be required by the vendor or other authentication requestor.

[0261] The process of trust arbitrage occurs between the steps of the trust engine 110 receiving the authentication data in step 1050 of FIGURE 10 and the return of an authentication result to the vendor in step 1055 of FIGURE 10. Between these steps, the process which leads to the evaluation of trust levels and the potential trust arbitrage occurs as shown in FIGURE 17. In circumstances where simple binary authentication is performed, the process shown in FIGURE 17 reduces to having the transaction engine 205 directly compare the authentication data provided with the enrollment data for the identified user as

discussed above with reference to FIGURE 10, flagging any difference as a negative authentication.

[0262] As shown in FIGURE 17, the first step after receiving the data in step 1050 is for the transaction engine 205 to determine the trust level which is required for a positive authentication for this particular transaction in step 1710. This step may be performed by one of several different methods. The required trust level may be specified to the trust engine 110 by the authentication requestor at the time when the authentication request is made. The authentication requestor may also set a preference in advance which is stored within the depository 210 or other storage which is accessible by the transaction engine 205. This preference may then be read and used each time an authentication request is made by this authentication requestor. The preference may also be associated with a particular user as a security measure such that a particular level of trust is always required in order to authenticate that user, the user preference being stored in the depository 210 or other storage media accessible by the transaction engine 205. The required level may also be derived by the transaction engine 205 or authentication engine 215 based upon information provided in the authentication request, such as the value and risk level of the transaction to be authenticated.

[0263] In one mode of operation, a policy management module or other software which is used when generating the authentication request is used to specify the required degree of trust for the authentication of the transaction. This may be used to provide a series of rules to follow when assigning the required level of

trust based upon the policies which are specified within the policy management module. One advantageous mode of operation is for such a module to be incorporated with the web server of a vendor in order to appropriately determine required level of trust for transactions initiated with the vendor's web server. In this way, transaction requests from users may be assigned a required trust level in accordance with the policies of the vendor and such information may be forwarded to the trust engine 110 along with the authentication request.

[0264] This required trust level correlates with the degree of certainty that the vendor wants to have that the individual authenticating is in fact who he identifies himself as. For example, if the transaction is one where the vendor wants a fair degree of certainty because goods are changing hands, the vendor may require a trust level of 85%. For situation where the vendor is merely authenticating the user to allow him to view members only content or exercise privileges on a chat room, the downside risk may be small enough that the vendor requires only a 60% trust level. However, to enter into a production contract with a value of tens of thousands of dollars, the vendor may require a trust level of 99% or more.

[0265] This required trust level represents a metric to which the user must authenticate himself in order to complete the transaction. If the required trust level is 85% for example, the user must provide authentication to the trust engine 110 sufficient for the trust engine 110 to say with 85% confidence that the user is who they say they are. It is the balance between this required trust level and the

authentication confidence level which produces either a positive authentication (to the satisfaction of the vendor) or a possibility of trust arbitrage.

[0266] As shown in FIGURE 17, after the transaction engine 205 receives the required trust level, it compares in step 1720 the required trust level to the authentication confidence level which the authentication engine 215 calculated for the current authentication (as discussed with reference to FIGURE 16). If the authentication confidence level is higher than the required trust level for the transaction in step 1730, then the process moves to step 1740 where a positive authentication for this transaction is produced by the transaction engine 205. A message to this effect will then be inserted into the authentication results and returned to the vendor by the transaction engine 205 as shown in step 1055 (see FIGURE 10).

[0267] However, if the authentication confidence level does not fulfill the required trust level in step 1730, then a confidence gap exists for the current authentication, and trust arbitrage is conducted in step 1750. Trust arbitrage is described more completely with reference to FIGURE 18 below. This process as described below takes place within the transaction engine 205 of the trust engine 110. Because no authentication or other cryptographic operations are needed to execute trust arbitrage (other than those required for the SSL communication between the transaction engine 205 and other components), the process may be performed outside the authentication engine 215. However, as will be discussed below, any reevaluation of authentication data or other

cryptographic or authentication events will require the transaction engine 205 to resubmit the appropriate data to the authentication engine 215. Those of skill in the art will recognize that the trust arbitrage process could alternately be structured to take place partially or entirely within the authentication engine 215 itself.

[0268] As mentioned above, trust arbitrage is a process where the trust engine 110 mediates a negotiation between the vendor and user in an attempt to secure a positive authentication where appropriate. As shown in step 1805, the transaction engine 205 first determines whether or not the current situation is appropriate for trust arbitrage. This may be determined based upon the circumstances of the authentication, e.g. whether this authentication has already been through multiple cycles of arbitrage, as well as upon the preferences of either the vendor or user, as will be discussed further below.

[0269] In such circumstances where arbitrage is not possible, the process proceeds to step 1810 where the transaction engine 205 generates a negative authentication and then inserts it into the authentication results which are sent to the vendor in step 1055 (see FIGURE 10). One limit which may be advantageously used to prevent authentications from pending indefinitely is to set a time-out period from the initial authentication request. In this way, any transaction which is not positively authenticated within the time limit is denied further arbitrage and negatively authenticated. Those of skill in the art will recognize that such a time limit may vary depending upon the circumstances of the transaction and

the desires of the user and vendor. Limitations may also be placed upon the number of attempts that may be made at providing a successful authentication. Such limitations may be handled by an attempt limiter 535 as shown in FIGURE 5.

[0270] If arbitrage is not prohibited in step 1805, the transaction engine 205 will then engage in negotiation with one or both of the transacting parties. The transaction engine 205 may send a message to the user requesting some form of additional authentication in order to boost the authentication confidence level produced as shown in step 1820. In the simplest form, this may simply indicates that authentication was insufficient. A request to produce one or more additional authentication instances to improve the overall confidence level of the authentication may also be sent.

[0271] If the user provides some additional authentication instances in step 1825, then the transaction engine 205 adds these authentication instances to the authentication data for the transaction and forwards it to the authentication engine 215 as shown in step 1015 (see FIGURE 10), and the authentication is reevaluated based upon both the pre-existing authentication instances for this transaction and the newly provided authentication instances.

[0272] An additional type of authentication may be a request from the trust engine 110 to make some form of person-to-person contact between the trust engine 110 operator (or a trusted associate) and the user, for example, by phone call. This phone call or other non-computer authentication can be used to provide

personal contact with the individual and also to conduct some form of questionnaire based authentication. This also may give the opportunity to verify an originating telephone number and potentially a voice analysis of the user when he calls in. Even if no additional authentication data can be provided, the additional context associated with the user's phone number may improve the reliability of the authentication context. Any revised data or circumstances based upon this phone call are fed into the trust engine 110 for use in consideration of the authentication request.

[0273] Additionally, in step 1820 the trust engine 110 may provide an opportunity for the user to purchase insurance, effectively buying a more confident authentication. The operator of the trust engine 110 may, at times, only want to make such an option available if the confidence level of the authentication is above a certain threshold to begin with. In effect, this user side insurance is a way for the trust engine 110 to vouch for the user when the authentication meets the normal required trust level of the trust engine 110 for authentication, but does not meet the required trust level of the vendor for this transaction. In this way, the user may still successfully authenticate to a very high level as may be required by the vendor, even though he only has authentication instances which produce confidence sufficient for the trust engine 110.

[0274] This function of the trust engine 110 allows the trust engine 110 to vouch for someone who is authenticated to the satisfaction of the trust engine 110, but not of the vendor. This is analogous to the function performed by a notary in adding his signature

to a document in order to indicate to someone reading the document at a later time that the person whose signature appears on the document is in fact the person who signed it. The signature of the notary testifies to the act of signing by the user. In the same way, the trust engine is providing an indication that the person transacting is who they say they are.

[0275] However, because the trust engine 110 is artificially boosting the level of confidence provided by the user, there is a greater risk to the trust engine 110 operator, since the user is not actually meeting the required trust level of the vendor. The cost of the insurance is designed to offset the risk of a false positive authentication to the trust engine 110 (who may be effectively notarizing the authentications of the user). The user pays the trust engine 110 operator to take the risk of authenticating to a higher level of confidence than has actually been provided.

[0276] Because such an insurance system allows someone to effectively buy a higher confidence rating from the trust engine 110, both vendors and users may wish to prevent the use of user side insurance in certain transactions. Vendors may wish to limit positive authentications to circumstances where they know that actual authentication data supports the degree of confidence which they require and so may indicate to the trust engine 110 that user side insurance is not to be allowed. Similarly, to protect his online identity, a user may wish to prevent the use of user side insurance on his account, or may wish to limit its use to situations where the authentication confidence level without the insurance is higher than a certain limit. This may be used as a security measure

to prevent someone from overhearing a password or stealing a smart card and using them to falsely authenticate to a low level of confidence, and then purchasing insurance to produce a very high level of (false) confidence. These factors may be evaluated in determining whether user side insurance is allowed.

[0277] If user purchases insurance in step 1840, then the authentication confidence level is adjusted based upon the insurance purchased in step 1845, and the authentication confidence level and required trust level are again compared in step 1730 (see FIGURE 17). The process continues from there, and may lead to either a positive authentication in step 1740 (see FIGURE 17), or back into the trust arbitrage process in step 1750 for either further arbitrage (if allowed) or a negative authentication in step 1810 if further arbitrage is prohibited.

[0278] In addition to sending a message to the user in step 1820, the transaction engine 205 may also send a message to the vendor in step 1830 which indicates that a pending authentication is currently below the required trust level. The message may also offer various options on how to proceed to the vendor. One of these Options is to simply inform the vendor of what the current authentication confidence level is and ask if the vendor wishes to maintain their current unfulfilled required trust level. This may be beneficial because in some cases, the vendor may have independent means for authenticating the transaction or may have been using a default set of requirements which generally result in a higher required level being initially specified than is actually needed for the particular transaction at hand.

[0279] For instance, it may be standard practice that all incoming purchase order transactions with the vendor are expected to meet a 98% trust level. However, if an order was recently discussed by phone between the vendor and a long-standing customer, and immediately thereafter the transaction is authenticated, but only to a 93% confidence level, the vendor may wish to simply lower the acceptance threshold for this transaction, because the phone call effectively provides additional authentication to the vendor. In certain circumstances, the vendor may be willing to lower their required trust level, but not all the way to the level of the current authentication confidence. For instance, the vendor in the above example might consider that the phone call prior to the order might merit a 4% reduction in the degree of trust needed; however, this is still greater than the 93% confidence produced by the user.

[0280] If the vendor does adjust their required trust level in step 1835, then the authentication confidence level produced by the authentication and the required trust level are compared in step 1730 (see FIGURE 17). If the confidence level now exceeds the required trust level, a positive authentication may be generated in the transaction engine 205 in step 1740 (see FIGURE 17). If not, further arbitrage may be attempted as discussed above if it is permitted.

[0281] In addition to requesting an adjustment to the required trust level, the transaction engine 205 may also offer vendor side insurance to the vendor requesting the authentication. This insurance serves a similar purpose to that described above for the user side insurance. Here, however, rather than the cost

corresponding to the risk being taken by the trust engine 110 in authenticating above the actual authentication confidence level produced, the cost of the insurance corresponds to the risk being taken by the vendor in accepting a lower trust level in the authentication.

[0282] Instead of just lowering their actual required trust level, the vendor has the option of purchasing insurance to protect itself from the additional risk associated with a lower level of trust in the authentication of the user. As described above, it may be advantageous for the vendor to only consider purchasing such insurance to cover the trust gap in conditions where the existing authentication is already above a certain threshold.

[0283] The availability of such vendor side insurance allows the vendor the option to either: lower his trust requirement directly at no additional cost to himself, bearing the risk of a false authentication himself (based on the lower trust level required); or, buying insurance for the trust gap between the authentication confidence level and his requirement, with the trust engine 110 operator bearing the risk of the lower confidence level which has been provided. By purchasing the insurance, the vendor effectively keeps his high trust level requirement; because the risk of a false authentication is shifted to the trust engine 110 operator.

[0284] If the vendor purchases insurance in step 1840, the authentication confidence level and required trust levels are compared in step 1730 (see FIGURE 17), and the process continues as described above.

[0285] Note that it is also possible that both the user and the vendor respond to messages from the trust engine 110. Those of skill in the art will recognize that there are multiple ways in which such situations can be handled. One advantageous mode of handling the possibility of multiple responses is simply to treat the responses in a first-come, first-served manner. For example, if the vendor responds with a lowered required trust level and immediately thereafter the user also purchases insurance to raise his authentication level, the authentication is first reevaluated based upon the lowered trust requirement from the vendor. If the authentication is now positive, the user's insurance purchase is ignored. In another advantageous mode of operation, the user might only be charged for the level of insurance required to meet the new, lowered trust requirement of the vendor (if a trust gap remained even with the lowered vendor trust requirement).

[0286] If no response from either party is received during the trust arbitrage process at step 1850 within the time limit set for the authentication, the arbitrage is reevaluated in step 1805. This effectively begins the arbitrage process again. If the time limit was final or other circumstances prevent further arbitrage in step 1805, a negative authentication is generated by the transaction engine 205 in step 1810 and returned to the vendor in step 1055 (see FIGURE 10). If not, new messages may be sent to the user and vendor, and the process may be repeated as desired.

[0287] Note that for certain types of transactions, for instance, digitally signing documents which are not

part of a transaction, there may not necessarily be a vendor or other third party; therefore the transaction is primarily between the user and the trust engine 110. In circumstances such as these, the trust engine 110 will have its own required trust level which must be satisfied in order to generate a positive authentication. However, in such circumstances, it will often not be desirable for the trust engine 110 to offer insurance to the user in order for him to raise the confidence of his own signature.

[0288] The process described above and shown in FIGURES 16-18 may be carried out using various communications modes as described above with reference to the trust engine 110. For instance, the messages may be web-based and sent using SSL connections between the trust engine 110 and applets downloaded in real time to browsers running on the user or vendor systems. In an alternate mode of operation, certain dedicated applications may be in use by the user and vendor which facilitate such arbitrage and insurance transactions. In another alternate mode of operation, secure email operations may be used to mediate the arbitrage described above, thereby allowing deferred evaluations and batch processing of authentications. Those of skill in the art will recognize that different communications modes may be used as are appropriate for the circumstances and authentication requirements of the vendor.

[0289] The following description with reference to FIGURE 19 describes a sample transaction which integrates the various aspects of the present invention as described above. This example illustrates the overall process between a user and a vendor as mediates

by the trust engine 110. Although the various steps and components as described in detail above may be used to carry out the following transaction, the process illustrated focuses on the interaction between the trust engine 110, user and vendor.

[0290] The transaction begins when the user, while viewing web pages online, fills out an order form on the web site of the vendor in step 1900. The user wishes to submit this order form to the vendor, signed with his digital signature. In order to do this, the user submits the order form with his request for a signature to the trust engine 110 in step 1905. The user will also provide authentication data which will be used as described above to authenticate his identity.

[0291] In step 1910 the authentication data is compared to the enrollment data by the trust engine 110 as discussed above, and if a positive authentication is produced, the hash of the order form, signed with the private key of the user, is forwarded to the vendor along with the order form itself.

[0292] The vendor receives the signed form in step 1915, and then the vendor will generate an invoice or other contract related to the purchase to be made in step 1920. This contract is sent back to the user with a request for a signature in step 1925. The vendor also sends an authentication request for this contract transaction to the trust engine 110 in step 1930 including a hash of the contract which will be signed by both parties. To allow the contract to be digitally signed by both parties, the vendor also includes authentication data for itself so that the vendor's

signature upon the contract can later be verified if necessary.

[0293] As discussed above, the trust engine 110 then verifies the authentication data provided by the vendor to confirm the vendor's identity, and if the data produces a positive authentication in step 1935, continues with step 1955 when the data is received from the user. If the vendor's authentication data does not match the enrollment data of the vendor to the desired degree, a message is returned to the vendor requesting further authentication. Trust arbitrage may be performed here if necessary, as described above, in order for the vendor to successfully authenticate itself to the trust engine 110.

[0294] When the user receives the contract in step 1940, he reviews it, generates authentication data to sign it if it is acceptable in step 1945, and then sends a hash of the contract and his authentication data to the trust engine 110 in step 1950. The trust engine 110 verifies the authentication data in step 1955 and if the authentication is good, proceeds to process the contract as described below. As discussed above with reference to FIGURES 17 and 18, trust arbitrage may be performed as appropriate to close any trust gap which exists between the authentication confidence level and the required authentication level for the transaction.

[0295] The trust engine 110 signs the hash of the contract with the user's private key, and sends this signed hash to the vendor in step 1960, signing the complete message on its own behalf, i.e., including a hash of the complete message (including the user's signature) encrypted with the private key 510 of the

trust engine 110. This message is received by the vendor in step 1965. The message represents a signed contract (hash of contract encrypted using user's private key) and a receipt from the trust engine 110 (the hash of the message including the signed contract, encrypted using the trust engine 110's private key).

[0296] The trust engine 110 similarly prepares a hash of the contract with the vendor's private key in step 1970, and forwards this to the user, signed by the trust engine 110. In this way, the user also receives a copy of the contract, signed by the vendor, as well as a receipt, signed by the trust engine 110, for delivery of the signed contract in step 1975.

[0297] In addition to the foregoing, an additional aspect of the invention provides a cryptographic Service Provider Module (SPM) which may be available to a client side application as a means to access functions provided by the trust engine 110 described above. One advantageous way to provide such a service is for the cryptographic SPM is to mediate communications between a third party Application Programming Interface (API) and a trust engine 110 which is accessible via a network or other remote connection. A sample cryptographic SPM is described below with reference to FIGURE 20.

[0298] For example, on a typical system, a number of API's are available to programmers. Each API provides a set of function calls which may be made by an application 2000 running upon the system. Examples of API's which provide programming interfaces suitable for cryptographic functions, authentication functions, and other security function include the Cryptographic API (CAPI) 2010 provided by Microsoft with its Windows

operating systems, and the Common Data Security Architecture (CDSA), sponsored by IBM, Intel and other members of the Open Group. CAPI will be used as an exemplary security API in the discussion that follows. However, the cryptographic SPM described could be used with CDSA or other security API's as are known in the art.

[0299] This API is used by a user system 105 or vendor system 120 when a call is made for a cryptographic function. Included among these functions may be requests associated with performing various cryptographic operations, such as encrypting a document with a particular key, signing a document, requesting a digital certificate, verifying a signature upon a signed document, and such other cryptographic functions as are described herein or known to those of skill in the art.

[0300] Such cryptographic functions are normally performed locally to the system upon which CAPI 2010 is located. This is because generally the functions called require the use of either resources of the local user system 105, such as a fingerprint reader, or software functions which are programmed using libraries which are executed on the local machine. Access to these local resources is normally provided by one or more Service Provider Modules (SPM's) 2015, 2020 as referred to above which provide resources with which the cryptographic functions are carried out. Such SPM's may include software libraries 2015 to perform encrypting or decrypting operations, or drivers and applications 2020 which are capable of accessing specialized hardware 2025, such as biometric scanning devices. In much the way that CAPI 2010 provides

functions which may be used by an application 2000 of the system 105, the SPM's 2015, 2020 provide CAPI with access to the lower level functions and resources associated with the available services upon the system.

[0301] In accordance with the invention, it is possible to provide a cryptographic SPM 2030 which is capable of accessing the cryptographic functions provided by the trust engine 110 and making these functions available to an application 2000 through CAPI 2010. Unlike embodiments where CAPI 2010 is only able to access resources which are locally available through SPM's 2015, 2020, a cryptographic SPM 2030 as described herein would be able to submit requests for cryptographic operations to a remotely-located, network-accessible trust engine 110 in order to perform the operations desired.

[0302] For instance, if an application 2000 has a need for a cryptographic operation, such as signing a document, the application 2000 makes a function call to the appropriate CAPI 2010 function. CAPI 2010 in turn will execute this function, making use of the resources which are made available to it by the SPM's 2015, 2020 and the cryptographic SPM 2030. In the case of a digital signature function, the cryptographic SPM 2030 will generate an appropriate request which will be sent to the trust engine 110 across the communication link 125.

[0303] The operations which occur between the cryptographic SPM 2030 and the trust engine 110 are the same operations that would be possible between any other system and the trust engine 110. However, these functions are effectively made available to a user system 105 through CAPI 2010 such that they appear to

be locally available upon the user system 105 itself. However, unlike ordinary SPM's 2015, 2020, the functions are being carried out on the remote trust engine 110 and the results relayed to the cryptographic SPM 2030 in response to appropriate requests across the communication link 125.

[0304] This cryptographic SPM 2030 makes a number of operations available to the user system 105 or a vendor system 120 which might not otherwise be available. These functions include without limitation: encryption and decryption of documents; issuance of digital certificates; digital signing of documents; verification of digital signatures; and such other operations as will be apparent to those of skill in the art.

[0305] In a separate embodiment, the present invention comprises a complete system for performing the data securing methods of the present invention on any data set. The computer system of this embodiment comprises a data splitting module that comprises the functionality shown in FIGURE 8 and described herein. In one embodiment of the present invention, the data splitting module, sometimes referred to herein as a secure data parser, comprises a parser program or software suite which comprises data splitting, encryption and decryption, reconstitution or reassembly functionality. This embodiment may further comprise a data storage facility or multiple data storage facilities, as well. The data splitting module, or secure data parser, comprises a cross-platform software module suite which integrates within an electronic infrastructure, or as an add-on to any application which requires the ultimate security of its data

elements. This parsing process operates on any type of data set, and on any and all file types, or in a database on any row, column or cell of data in that database.

[0306] The parsing process of the present invention may, in one embodiment, be designed in a modular tiered fashion, and any encryption process is suitable for use in the process of the present invention. The modular tiers of the parsing and splitting process of the present invention may include, but are not limited to, 1) cryptographic split, dispersed and securely stored in multiple locations; 2) encrypt, cryptographically split, dispersed and securely stored in multiple locations; 3) encrypt, cryptographically split, encrypt each share, then dispersed and securely stored in multiple locations; and 4) encrypt, cryptographically split, encrypt each share with a different type of encryption than was used in the first step, then dispersed and securely stored in multiple locations.

[0307] The process comprises, in one embodiment, splitting of the data according to the contents of a generated random number, or key and performing the same cryptographic splitting of the key used in the encryption of splitting of the data to be secured into two or more portions, or shares, of parsed and split data, and in one embodiment, preferably into four or more portions of parsed and split data, encrypting all of the portions, then scattering and storing these portions back into the database, or relocating them to any named device, fixed or removable, depending on the requestor's need for privacy and security. Alternatively, in another embodiment, encryption may occur prior to the splitting of the data set by the

splitting module or secure data parser. The original data processed as described in this embodiment is encrypted and obfuscated and is secured. The dispersion of the encrypted elements, if desired, can be virtually anywhere, including, but not limited to, a single server or data storage device, or among separate data storage facilities or devices. Encryption key management in one embodiment may be included within the software suite, or in another embodiment may be integrated into an existing infrastructure or any other desired location.

[0308] A cryptographic split (cryptosplit) partitions the data into N number of shares. The partitioning can be on any size unit of data, including an individual bit, bits, bytes, kilobytes, megabytes, or larger units, as well as any pattern or combination of data unit sizes whether predetermined or randomly generated. The units can also be of different sized, based on either a random or predetermined set of values. This means the data can be viewed as a sequence of these units. In this manner the size of the data units themselves may render the data more secure, for example by using one or more predetermined or randomly generated pattern, sequence or combination of data unit sizes. The units are then distributed (either randomly or by a predetermined set of values) into the N shares. This distribution could also involve a shuffling of the order of the units in the shares. It is readily apparent to those of ordinary skill in the art that the distribution of the data units into the shares may be performed according to a wide variety of possible selections, including but not limited to size-fixed, predetermined sizes, or one or

more combination, pattern or sequence of data unit sizes that are predetermined or randomly generated.

[0309] One example of this cryptographic split process, or cryptosplit, would be to consider the data to be 23 bytes in size, with the data unit size chosen to be one byte, and with the number of shares selected to be 4. Each byte would be distributed into one of the 4 shares. Assuming a random distribution, a key would be obtained to create a sequence of 23 random numbers (r1, r2, r3 through r23), each with a value between 1 and 4 corresponding to the four shares. Each of the units of data (in this example 23 individual bytes of data) is associated with one of the 23 random numbers corresponding to one of the four shares. The distribution of the bytes of data into the four shares would occur by placing the first byte of the data into share number r1, byte two into share r2, byte three into share r3, through the 23rd byte of data into share r23. It is readily apparent to those of ordinary skill in the art that a wide variety of other possible steps or combination or sequence of steps, including the size of the data units, may be used in the cryptosplit process of the present invention, and the above example is a non-limiting description of one process for cryptosplitting data. To recreate the original data, the reverse operation would be performed.

[0310] In another embodiment of the cryptosplit process of the present invention, an option for the cryptosplitting process is to provide sufficient redundancy in the shares such that only a subset of the shares are needed to reassemble or restore the data to its original or useable form. As a non-limiting example, the cryptosplit may be done as a "3 of 4"

cryptosplit such that only three of the four shares are necessary to reassemble or restore the data to its original or useable form. This is also referred to as a "M of N cryptosplit" wherein N is the total number of shares, and M is at least one less than N. It is readily apparent to those of ordinary skill in the art that there are many possibilities for creating this redundancy in the cryptosplitting process of the present invention.

[0311] In one embodiment of the cryptosplitting process of the present invention, each unit of data is stored in two shares, the primary share and the backup share. Using the "3 of 4" cryptosplitting process described above, any one share can be missing, and this is sufficient to reassemble or restore the original data with no missing data units since only three of the total four shares are required. As described herein, a random number is generated that corresponds to one of the shares. The random number is associated with a data unit, and stored in the corresponding share, based on a key. One key is used, in this embodiment, to generate the primary and backup share random number. As described herein for the cryptosplitting process of the present invention, a set of random numbers (also referred to as primary share numbers) from 0 to 3 are generated equal to the number of data units. Then another set of random numbers is generated (also referred to as backup share numbers) from 1 to 3 equal to the number of data units. Each unit of data is then associated with a primary share number and a backup share number. Alternatively, a set of random numbers may be generated that is fewer than the number of data units, and repeating the random number set, but this

may reduce the security of the sensitive data. The primary share number is used to determine into which share the data unit is stored. The backup share number is combined with the primary share number to create a third share number between 0 and 3, and this number is used to determine into which share the data unit is stored. In this example, the equation to determine the third share number is:

(primary share number + backup share number) MOD 4 = third share number.

[0312] In the embodiment described above where the primary share number is between 0 and 3, and the backup share number is between 1 and 3 ensures that the third share number is different from the primary share number. This results in the data unit being stored in two different shares. It is readily apparent to those of ordinary skill in the art that there are many ways of performing redundant cryptosplitting and non-redundant cryptosplitting in addition to the embodiments disclosed herein. For example, the data units in each share could be shuffled utilizing a different algorithm. This data unit shuffling may be performed as the original data is split into the data units, or after the data units are placed into the shares, or after the share is full, for example.

[0313] The various cryptosplitting processes and data shuffling processes described herein, and all other embodiments of the cryptosplitting and data shuffling methods of the present invention may be performed on data units of any size, including but not limited to, as small as an individual bit, bits, bytes, kilobytes, megabytes or larger.

[0314] An example of one embodiment of source code that would perform the cryptosplitting process described herein is:

```
DATA [1:24] - array of bytes with the data to be split
SHARES[0:3; 1:24] - 2-dimensionalarray with each row representing one of the shares
RANDOM[1:24] - array random numbers in the range of 0..3
S1 = 1;
S2 = 1;
S3 = 1;
S4 = 1;

For J = 1 to 24 do
     Begin
     IF RANDOM[J[ ==0 then
          Begin
          SHARES[1,S1] = DATA [J];
          S1 = S1 + 1;
          End
     ELSE IF RANDOM[J[ ==1 then
          Begin
          SHARES[2,S2] = DATA [J];
          S2 = S2 + 1;
          END
     ELSE IF RANDOM[J[ ==2 then
          Begin
          Shares[3,S3] = data [J];
          S3 = S3 + 1;
          End
     Else   begin
          Shares[4,S4] = data [J];
          S4 = S4 + 1;
```

```
        End;
    END;
```

[0315] An example of one embodiment of source code that would perform the cryptosplitting RAID process described herein is:

[0316] Generate two sets of numbers, PrimaryShare is 0 to 3, BackupShare is 1 to 3. Then put each data unit into share[primaryshare[1]] and share[(primaryshare[1]+backupshare[1]) mod 4, with the same process as in cryptosplitting described above. This method will be scalable to any size N, where only N-1 shares are necessary to restore the data.

[0317] The retrieval, recombining, reassembly or reconstituting of the encrypted data elements may utilize any number of authentication techniques, including, but not limited to, biometrics, such as fingerprint recognition, facial scan, hand scan, iris scan, retinal scan, ear scan, vascular pattern recognition or DNA analysis. The data splitting and/or parser modules of the present invention may be integrated into a wide variety of infrastructure products or applications as desired.

[0318] Traditional encryption technologies known in the art rely on one or more key used to encrypt the data and render it unusable without the key. The data, however, remains whole and intact and subject to attack. The secure data parser of the present invention, in one embodiment, addresses this problem by performing a cryptographic parsing and splitting of the encrypted file into two or more portions or shares, and in another embodiment, preferably four or more shares, adding another layer of encryption to each share of the data, then storing the shares in different physical

and/or logical locations. When one or more data shares are physically removed from the system, either by using a removable device, such as a data storage device, or by placing the share under another party's control, any possibility of compromise of secured data is effectively removed.

[0319] An example of one embodiment of the secure data parser of the present invention and an example of how it may be utilized is shown in FIGURE 21 and described below. However, it is readily apparent to those of ordinary skill in the art that the secure data parser of the present invention may be utilized in a wide variety of ways in addition to the non-limiting example below. As a deployment option, and in one embodiment, the secure data parser may be implemented with external session key management or secure internal storage of session keys. Upon implementation, a Parser Master Key will be generated which will be used for securing the application and for encryption purposes. It should be also noted that the incorporation of the Parser Master key in the resulting secured data allows for a flexibility of sharing of secured data by individuals within a workgroup, enterprise or extended audience.

[0320] As shown in Figure 21, this embodiment of the present invention shows the steps of the process performed by the secure data parser on data to store the session master key with the parsed data:

[0321] 1. Generating a session master key and encrypt the data using RS1 stream cipher.

[0322] 2. Separating the resulting encrypted data into four shares or portions of parsed data according to the pattern of the session master key.

[0323] 3. In this embodiment of the method, the session master key will be stored along with the secured data shares in a data depository. Separating the session master key according to the pattern of the Parser Master Key and append the key data to the encrypted parsed data.

[0324] 4. The resulting four shares of data will contain encrypted portions of the original data and portions of the session master key. Generate a stream cipher key for each of the four data shares.

[0325] 5. Encrypting each share, then store the encryption keys in different locations from the encrypted data portions or shares: Share 1 gets Key 4, Share 2 gets Key 1, Share 3 gets Key 2, Share 4 gets Key 3.

[0326] To restore the original data format, the steps are reversed.

[0327] It is readily apparent to those of ordinary skill in the art that certain steps of the methods described herein may be performed in different order, or repeated multiple times, as desired. It is also readily apparent to those skilled in the art that the portions of the data may be handled differently from one another. For example, multiple parsing steps may be performed on only one portion of the parsed data. Each portion of parsed data may be uniquely secured in any desirable way provided only that the data may be reassembled, reconstituted, reformed, decrypted or restored to its original or other usable form.

[0328] As shown in FIGURE 22 and described herein, another embodiment of the present invention comprises the steps of the process performed by the secure data

parser on data to store the session master key data in one or more separate key management table:

[0329] 1. Generating a session master key and encrypt the data using RS1 stream cipher.

[0330] 2. Separating the resulting encrypted data into four shares or portions of parsed data according to the pattern of the session master key.

[0331] 3. In this embodiment of the method of the present invention, the session master key will be stored in a separate key management table in a data depository. Generating a unique transaction ID for this transaction. Storing the transaction ID and session master key in a separate key management table. Separating the transaction ID according to the pattern of the Parser Master Key and append the data to the encrypted parsed or separated data.

[0332] 4. The resulting four shares of data will contain encrypted portions of the original data and portions of the transaction ID.

[0333] 5. Generating a stream cipher key for each of the four data shares.

[0334] 6. Encrypting each share, then store the encryption keys in different locations from the encrypted data portions or shares: Share 1 gets Key 4, Share 2 gets Key 1, Share 3 gets Key 2, Share 4 gets Key 3.

[0335] To restore the original data format, the steps are reversed.

[0336] It is readily apparent to those of ordinary skill in the art that certain steps of the method described herein may be performed in different order, or repeated multiple times, as desired. It is also readily apparent to those skilled in the art that the

portions of the data may be handled differently from one another. For example, multiple separating or parsing steps may be performed on only one portion of the parsed data. Each portion of parsed data may be uniquely secured in any desirable way provided only that the data may be reassembled, reconstituted, reformed, decrypted or restored to its original or other usable form.

[0337] As shown in Figure 23, this embodiment of the present invention shows the steps of the process performed by the secure data parser on data to store the session master key with the parsed data:

[0338] 1. Accessing the parser master key associated with the authenticated user

[0339] 2. Generating a unique Session Master key

[0340] 3. Derive an Intermediary Key from an exclusive OR function of the Parser Master Key and Session Master key

[0341] 4. Optional encryption of the data using an existing or new encryption algorithm keyed with the Intermediary Key.

[0342] 5. Separating the resulting optionally encrypted data into four shares or portions of parsed data according to the pattern of the Intermediary key.

[0343] 6. In this embodiment of the method, the session master key will be stored along with the secured data shares in a data depository. Separating the session master key according to the pattern of the Parser Master Key and append the key data to the optionally encrypted parsed data shares.

[0344] 7. The resulting multiple shares of data will contain optionally encrypted portions of the original data and portions of the session master key.

[0345] 8. Optionally generate an encryption key for each of the four data shares.

[0346] 9. Optionally encrypting each share with an existing or new encryption algorithm, then store the encryption keys in different locations from the encrypted data portions or shares: for example, Share 1 gets Key 4, Share 2 gets Key 1, Share 3 gets Key 2, Share 4 gets Key 3.

[0347] To restore the original data format, the steps are reversed.

[0348] It is readily apparent to those of ordinary skill in the art that certain steps of the methods described herein may be performed in different order, or repeated multiple times, as desired. It is also readily apparent to those skilled in the art that the portions of the data may be handled differently from one another. For example, multiple parsing steps may be performed on only one portion of the parsed data. Each portion of parsed data may be uniquely secured in any desirable way provided only that the data may be reassembled, reconstituted, reformed, decrypted or restored to its original or other usable form.

[0349] As shown in FIGURE 24 and described herein, another embodiment of the present invention comprises the steps of the process performed by the secure data parser on data to store the session master key data in one or more separate key management table:

[0350] 1. Accessing the Parser Master Key associated with the authenticated user

[0351] 2. Generating a unique Session Master Key

[0352] 3. Derive an Intermediary Key from an exclusive OR function of the Parser Master Key and Session Master key

[0353] 4. Optionally encrypt the data using an existing or new encryption algorithm keyed with the Intermediary Key.

[0354] 5. Separating the resulting optionally encrypted data into four shares or portions of parsed data according to the pattern of the Intermediary Key.

[0355] 6. In this embodiment of the method of the present invention, the session master key will be stored in a separate key management table in a data depository. Generating a unique transaction ID for this transaction. Storing the transaction ID and session master key in a separate key management table or passing the Session Master Key and transaction ID back to the calling program for external management. Separating the transaction ID according to the pattern of the Parser Master Key and append the data to the optionally encrypted parsed or separated data.

[0356] 7. The resulting four shares of data will contain optionally encrypted portions of the original data and portions of the transaction ID.

[0357] 8. Optionally generate an encryption key for each of the four data shares.

[0358] 9. Optionally encrypting each share, then store the encryption keys in different locations from the encrypted data portions or shares. For example: Share 1 gets Key 4, Share 2 gets Key 1, Share 3 gets Key 2, Share 4 gets Key 3.

[0359] To restore the original data format, the steps are reversed.

[0360] It is readily apparent to those of ordinary skill in the art that certain steps of the method described herein may be performed in different order, or repeated multiple times, as desired. It is also

readily apparent to those skilled in the art that the portions of the data may be handled differently from one another. For example, multiple separating or parsing steps may be performed on only one portion of the parsed data. Each portion of parsed data may be uniquely secured in any desirable way provided only that the data may be reassembled, reconstituted, reformed, decrypted or restored to its original or other usable form.

[0361] A wide variety of encryption methodologies are suitable for use in the methods of the present invention, as is readily apparent to those skilled in the art. The One Time Pad algorithm, is often considered one of the most secure encryption methods, and is suitable for use in the method of the present invention. Using the One Time Pad algorithm requires that a key be generated which is as long as the data to be secured. The use of this method may be less desirable in certain circumstances such as those resulting in the generation and management of very long keys because of the size of the data set to be secured. In the One-Time Pad (OTP) algorithm, the simple exclusive-or function, XOR, is used. For two binary streams x and y of the same length, x XOR y means the bitwise exclusive-or of x and y.

[0362] At the bit level is generated:

0 XOR 0 = 0

0 XOR 1 = 1

1 XOR 0 = 1

1 XOR 1 = 0

[0363] An example of this process is described herein for an n-byte secret, s, (or data set) to be

split. The process will generate an n-byte random value, a, and then set:

b = a XOR s.

[0364] Note that one can derive "s" via the equation:

s = a XOR b.

[0365] The values a and b are referred to as shares or portions and are placed in separate depositories. Once the secret s is split into two or more shares, it is discarded in a secure manner.

[0366] The secure data parser of the present invention may utilize this function, performing multiple XOR functions incorporating multiple distinct secret key values: K1, K2, K3, Kn, K5. At the beginning of the operation, the data to be secured is passed through the first encryption operation, secure data = data XOR secret key 5:

S = D XOR K5

[0367] In order to securely store the resulting encrypted data in, for example, four shares, S1, S2, S3, Sn, the data is parsed and split into "n" segments, or shares, according to the value of K5. This operation results in "n" pseudorandom shares of the original encrypted data. Subsequent XOR functions may then be performed on each share with the remaining secret key values, for example: Secure data segment 1 = encrypted data share 1 XOR secret key 1:

SD1 = S1 XOR K1

SD2 = S2 XOR K2

SD3 = S3 XOR K3

SDn = Sn XOR Kn.

[0368] In one embodiment, it may not be desired to have any one depository contain enough information to

decrypt the information held there, so the key required to decrypt the share is stored in a different data depository:

Depository 1: SD1, Kn
Depository 2: SD2, K1
Depository 3: SD3, K2
Depository n: SDn, K3.

[0369] Additionally, appended to each share may be the information required to retrieve the original session encryption key, K5. Therefore, in the key management example described herein, the original session master key is referenced by a transaction ID split into "n" shares according to the contents of the installation dependant Parser Master Key (TID1, TID2, TID3, TIDn):

Depository 1: SD1, Kn, TID1
Depository 2: SD2, K1, TID2
Depository 3: SD3, K2, TID3
Depository n: SDn, K3, TIDn.

[0370] In the incorporated session key example described herein, the session master key is split into "n" shares according to the contents of the installation dependant Parser Master Key (SK1, SK2, SK3, SKn):

Depository 1: SD1, Kn, SK1
Depository 2: SD2, K1, SK2
Depository 3: SD3, K2, SK3
Depository n: SDn, K3, SKn.

[0371] Unless all four shares are retrieved, the data cannot be reassembled according to this example. Even if all four shares are captured, there is no possibility of reassembling or restoring the original

information without access to the session master key and the Parser Master Key.

[0372] This example has described an embodiment of the method of the present invention, and also describes, in another embodiment, the algorithm used to place shares into depositories so that shares from all depositories can be combined to form the secret authentication material. The computations needed are very simple and fast. However, with the One Time Pad (OTP) algorithm there may be circumstances that cause it to be less desirable, such as a large data set to be secured, because the key size is the same size as the data to be stored. Therefore, there would be a need to store and transmit about twice the amount of the original data which may be less desirable under certain circumstances.

Stream Cipher RS1

[0373] The stream cipher RS1 splitting technique is very similar to the OTP splitting technique described herein. Instead of an n-byte random value, an n' = min(n, 16)-byte random value is generated and used to key the RS1 Stream Cipher algorithm. The advantage of the RS1 Stream Cipher algorithm is that a pseudorandom key is generated from a much smaller seed number. The speed of execution of the RS1 Stream Cipher encryption is also rated at approximately 10 times the speed of the well known in the art Triple DES encryption without compromising security. The RS1 Stream Cipher algorithm is well known in the art, and may be used to generate the keys used in the XOR function. The RS1 Stream Cipher algorithm is interoperable with other commercially available stream cipher algorithms, such

as the RC4™ stream cipher algorithm of RSA Security, Inc and is suitable for use in the methods of the present invention.

[0374] Using the key notation above, K1 thru K5 are now an n' byte random values and we set:

SD1 = S1 XOR E(K1)

SD2 = S2 XOR E(K2)

SD3 = S3 XOR E(K3)

SDn = Sn XOR E(Kn)

where E(K1) thru E(Kn) are the first n' bytes of output from the RS1 Stream Cipher algorithm keyed by K1 thru Kn. The shares are now placed into data depositories as described herein.

[0375] In this stream cipher RS1 algorithm, the required computations needed are nearly as simple and fast as the OTP algorithm. The benefit in this example using the RS1 Stream Cipher is that the system needs to store and transmit on average only about 16 bytes more than the size of the original data to be secured per share. When the size of the original data is more than 16 bytes, this RS1 algorithm is more efficient than the OTP algorithm because it is simply shorter. It is readily apparent to those of ordinary skill in the art that a wide variety of encryption methods or algorithms are suitable for use in the present invention, including, but not limited to RS1, OTP, RC4™, Triple DES and AES.

[0376] There are major advantages provided by the data security methods and computer systems of the present invention over traditional encryption methods. One advantage is the security gained from moving shares of the data to different locations on one or more data depositories or storage devices, that may be in

different logical, physical or geographical locations. When the shares of data are split physically and under the control of different personnel, for example, the possibility of compromising the data is greatly reduced.

[0377] Another advantage provided by the methods and system of the present invention is the combination of the steps of the method of the present invention for securing data to provide a comprehensive process of maintaining security of sensitive data. The data is encrypted with a secure key and split into one or more shares, and in one embodiment, four shares, according to the secure key. The secure key is stored safely with a reference pointer which is secured into four shares according to a secure key. The data shares are then encrypted individually and the keys are stored safely with different encrypted shares. When combined, the entire process for securing data according to the methods disclosed herein becomes a comprehensive package for data security.

[0378] The data secured according to the methods of the present invention is readily retrievable and restored, reconstituted, reassembled, decrypted, or otherwise returned into its original or other suitable form for use. In order to restore the original data, the following items may be utilized:

[0379] 1. All shares or portions of the data set.

[0380] 2. Knowledge of and ability to reproduce the process flow of the method used to secure the data.

[0381] 3. Access to the session master key.

[0382] 4. Access to the Parser Master Key.

[0383] Therefore, it may be desirable to plan a secure installation wherein at least one of the above

elements may be physically separated from the remaining components of the system (under the control of a different system administrator for example).

[0384] Protection against a rogue application invoking the data securing methods application may be enforced by use of the Parser Master Key. A mutual authentication handshake between the secure data parser and the application may be required in this embodiment of the present invention prior to any action taken.

[0385] The security of the system dictates that there be no "backdoor" method for recreation of the original data. For installations where data recovery issues may arise, the secure data parser can be enhanced to provide a mirror of the four shares and session master key depository. Hardware options such as RAID (redundant array of inexpensive disks, used to spread information over several disks) and software options such as replication can assist as well in the data recovery planning.

Key Management

[0386] In one embodiment of the present invention, the data securing method uses three sets of keys for an encryption operation. Each set of keys may have individual key storage, retrieval, security and recovery options, based on the installation. The keys that may be used, include, but are not limited to:

The Parser Master Key

[0387] This key is an individual key associated with the installation of the secure data parser. It is installed on the server on which the secure data parser has been deployed. There are a variety of options suitable for securing this key including, but not

limited to, a smart card, separate hardware key store, standard key stores, custom key stores or within a secured database table, for example.

The Session Master Key

[0388] A Session Master Key may be generated each time data is secured. The Session Master Key is used to encrypt the data prior to the parsing and splitting operations. It may also be incorporated (if the Session Master Key is not integrated into the parsed data) as a means of parsing the encrypted data. The Session Master Key may be secured in a variety of manners, including, but not limited to, a standard key store, custom key store, separate database table, or secured within the encrypted shares, for example.

The Share Encryption Keys

[0389] For each share or portions of a data set that is created, an individual Share Encryption Key may be generated to further encrypt the shares. The Share Encryption Keys may be stored in different shares than the share that was encrypted.

[0390] It is readily apparent to those of ordinary skill in the art that the data securing methods and computer system of the present invention are widely applicable to any type of data in any setting or environment. In addition to commercial applications conducted over the Internet or between customers and vendors, the data securing methods and computer systems of the present invention are highly applicable to non-commercial or private settings or environments. Any data set that is desired to be kept secure from any unauthorized user may be secured using the methods and systems described herein. For example, access to a particular database within a company or organization

may be advantageously restricted to only selected users by employing the methods and systems of the present invention for securing data. Another example is the generation, modification or access to documents wherein it is desired to restrict access or prevent unauthorized or accidental access or disclosure outside a group of selected individuals, computers or workstations. These and other examples of the ways in which the methods and systems of data securing of the present invention are applicable to any non-commercial or commercial environment or setting for any setting, including, but not limited to any organization, government agency or corporation.

[0391] In another embodiment of the present invention, the data securing method uses three sets of keys for an encryption operation. Each set of keys may have individual key storage, retrieval, security and recovery options, based on the installation. The keys that may be used, include, but are not limited to:

1. The Parser Master Key

[0392] This key is an individual key associated with the installation of the secure data parser. It is installed on the server on which the secure data parser has been deployed. There are a variety of options suitable for securing this key including, but not limited to, a smart card, separate hardware key store, standard key stores, custom key stores or within a secured database table, for example.

2. The Session Master Key

[0393] A Session Master Key may be generated each time data is secured. The Session Master Key is used in conjunction with the Parser Master key to derive the Intermediary Key. The Session Master Key may be

secured in a variety of manners, including, but not limited to, a standard key store, custom key store, separate database table, or secured within the encrypted shares, for example.

3. The Intermediary Key

[0394] An Intermediary Key may be generated each time data is secured. The Intermediary Key is used to encrypt the data prior to the parsing and splitting operation. It may also be incorporated as a means of parsing the encrypted data.

4. The Share Encryption Keys

[0395] For each share or portions of a data set that is created, an individual Share Encryption Key may be generated to further encrypt the shares. The Share Encryption Keys may be stored in different shares than the share that was encrypted.

[0396] It is readily apparent to those of ordinary skill in the art that the data securing methods and computer system of the present invention are widely applicable to any type of data in any setting or environment. In addition to commercial applications conducted over the Internet or between customers and vendors, the data securing methods and computer systems of the present invention are highly applicable to non-commercial or private settings or environments. Any data set that is desired to be kept secure from any unauthorized user may be secured using the methods and systems described herein. For example, access to a particular database within a company or organization may be advantageously restricted to only selected users by employing the methods and systems of the present invention for securing data. Another example is the generation, modification or access to documents wherein

it is desired to restrict access or prevent unauthorized or accidental access or disclosure outside a group of selected individuals, computers or workstations. These and other examples of the ways in which the methods and systems of data securing of the present invention are applicable to any non-commercial or commercial environment or setting for any setting, including, but not limited to any organization, government agency or corporation.

Workgroup, Project, Individual PC/Laptop or Cross Platform Data Security

[0397] The data securing methods and computer systems of the present invention are also useful in securing data by workgroup, project, individual PC/Laptop and any other platform that is in use in, for example, businesses, offices, government agencies, or any setting in which sensitive data is created, handled or stored. The present invention provides methods and computer systems to secure data that is known to be sought after by organizations, such as the U.S. Government, for implementation across the entire government organization or between governments at a state or federal level.

[0398] The data securing methods and computer systems of the present invention provide the ability to not only parse and split flat files but also data fields, sets and or table of any type. Additionally, all forms of data are capable of being secured under this process, including, but not limited to, text, video, images, biometrics and voice data. Scalability, speed and data throughput of the methods of securing data of the present invention are only limited to the hardware the user has at their disposal.

[0399] In one embodiment of the present invention, the data securing methods are utilized as described below in a workgroup environment. In one embodiment, as shown in FIGURE 23 and described below, the Workgroup Scale data securing method of the present invention uses the private key management functionality of the TrustEngine to store the user/group relationships and the associated private keys (Parser Group Master Keys) necessary for a group of users to share secure data. The method of the present invention has the capability to secure data for an enterprise, workgroup, or individual user, depending on how the Parser Master Key was deployed.

[0400] In one embodiment, additional key management and user/group management programs may be provided, enabling wide scale workgroup implementation with a single point of administration and key management. Key generation, management and revocation are handled by the single maintenance program, which all become especially important as the number of users increase. In another embodiment, key management may also be set up across one or several different system administrators, which may not allow any one person or group to control data as needed. This allows for the management of secured data to be obtained by roles, responsibilities, membership, rights, etc., as defined by an organization, and the access to secured data can be limited to just those who are permitted or required to have access only to the portion they are working on, while others, such as managers or executives, may have access to all of the secured data. This embodiment allows for the sharing of secured data among different groups within a company or organization while at the

same time only allowing certain selected individuals, such as those with the authorized and predetermined roles and responsibilities, to observe the data as a whole. In addition, this embodiment of the methods and systems of the present invention also allows for the sharing of data among, for example, separate companies, or separate departments or divisions of companies, or any separate organization departments, groups, agencies, or offices, or the like, of any government or organization or any kind, where some sharing is required, but not any one party may be permitted to have access to all the data. Particularly apparent examples of the need and utility for such a method and system of the present invention are to allow sharing, but maintain security, in between government areas, agencies and offices, and between different divisions, departments or offices of a large company, or any other organization, for example.

[0401] An example of the applicability of the methods of the present invention on a smaller scale is as follows. A Parser Master key is used as a serialization or branding of the secure data parser to an organization. As the scale of use of the Parser Master key is reduced from the whole enterprise to a smaller workgroup, the data securing methods described herein are used to share files within groups of users.

[0402] In the example shown in FIGURE 25 and described below, there are six users defined along with their title or role within the organization. The side bar represents five possible groups that the users can belong to according to their role. The arrow represents membership by the user in one or more of the groups.

[0403] When configuring the secure data parser for use in this example, the system administrator accesses the user and group information from the operating system by a maintenance program. This maintenance program generates and assigns Parser Group Master Keys to users based on their membership in groups.

[0404] In this example, there are three members in the Senior Staff group. For this group, the actions would be:

[0405] 1. Access Parser Group Master Key for the Senior Staff group (generate a key if not available);

[0406] 2. Generate a digital certificate associating CEO with the Senior Staff group;

[0407] 3. Generate a digital certificate associating CFO with the Senior Staff group;

[0408] 4. Generate a digital certificate associating Vice President, Marketing with the Senior Staff group.

[0409] The same set of actions would be done for each group, and each member within each group. When the maintenance program is complete, the Parser Group Master Key becomes a shared credential for each member of the group. Revocation of the assigned digital certificate may be done automatically when a user is removed from a group through the maintenance program without affecting the remaining members of the group.

[0410] Once the shared credentials have been defined, the parsing and splitting process remains the same. When a file, document or data element is to be secured, the user is prompted for the target group to be used when securing the data. The resulting secured data is only accessible by other members of the target group. This functionality of the methods and systems

of the present invention may be used with any other computer system or software platform, any may be, for example, integrated into existing application programs or used standalone for file security.

[0411] It is readily apparent to those of ordinary skill in the art that any one or combination of encryption algorithms are suitable for use in the methods and systems of the present invention. For example, the encryption steps may, in one embodiment, be repeated to produce a multi-layered encryption scheme. In addition, a different encryption algorithm, or combination of encryption algorithms, may be used in repeat encryption steps such that different encryption algorithms are applied to the different layers of the multi-layered encryption scheme. As such, the encryption scheme itself may become a component of the methods of the present invention for securing sensitive data from unauthorized use or access.

[0412] The secure data parser may include as an internal component, as an external component, or as both an error-checking component. For example, in one suitable approach, as portions of data are created using the secure data parser in accordance with the present invention, to assure the integrity of the data within a portion, a hash value is taken at preset intervals within the portion and is appended to the end of the interval. The hash value is a predictable and reproducible numeric representation of the data. If any bit within the data changes, the hash value would be different. A scanning module (either as a stand-alone component external to the secure data parser or as an internal component) may then scan the portions of data generated by the secure data parser. Each portion

of data (or alternatively, less than all portions of data according to some interval or by a random or pseudo-random sampling) is compared to the appended hash value or values and an action may be taken. This action may include a report of values that match and do not match, an alert for values that do not match, or invoking of some external or internal program to trigger a recovery of the data. For example, recovery of the data could be performed by invoking a recovery module based on the concept that fewer than all portions may be needed to generate original data in accordance with the present invention.

[0413] Any other suitable integrity checking may be implemented using any suitable integrity information appended anywhere in all or a subset of data portions. Integrity information may include any suitable information that can be used to determine the integrity of data portions. Examples of integrity information may include hash values computed based on any suitable parameter (e.g., based on respective data portions), digital signature information, message authentication code (MAC) information, any other suitable information, or any combination thereof.

[0414] The secure data parser of the present invention may be used in any suitable application. Namely, the secure data parser described herein has a variety of applications in different areas of computing and technology. Several such areas are discussed below. It will be understood that these are merely illustrative in nature and that any other suitable applications may make use of the secure data parser. It will further be understood that the examples described are merely illustrative embodiments that may

be modified in any suitable way in order to satisfy any suitable desires. For example, parsing and splitting may be based on any suitable units, such as by bits, by bytes, by kilobytes, by megabytes, by any combination thereof, or by any other suitable unit.

[0415] The secure data parser of the present invention may be used to implement secure physical tokens, whereby data stored in a physical token may be required in order to access additional data stored in another storage area. In one suitable approach, a physical token, such as a compact USB flash drive, a floppy disk, an optical disk, a smart card, or any other suitable physical token, may be used to store one of at least two portions of parsed data in accordance with the present invention. In order to access the original data, the USB flash drive would need to be accessed. Thus, a personal computer holding one portion of parsed data would need to have the USB flash drive, having the other portion of parsed data, attached before the original data can be accessed. FIGURE 26 illustrates this application. Storage area 2500 includes a portion of parsed data 2502. Physical token 2504, having a portion of parsed data 2506 would need to be coupled to storage area 2500 using any suitable communications interface 2508 (e.g., USB, serial, parallel, Bluetooth, IR, IEEE 1394, Ethernet, or any other suitable communications interface) in order to access the original data. This is useful in a situation where, for example, sensitive data on a computer is left alone and subject to unauthorized access attempts. By removing the physical token (e.g., the USB flash drive), the sensitive data is inaccessible. It will be understood that any other

suitable approach for using physical tokens may be used.

[0416] The secure data parser of the present invention may be used to implement a secure authentication system whereby user enrollment data (e.g., passwords, private encryption keys, fingerprint templates, biometric data or any other suitable user enrollment data) is parsed and split using the secure data parser. The user enrollment data may be parsed and split whereby one or more portions are stored on a smart card, a government Common Access Card, any suitable physical storage device (e.g., magnetic or optical disk, USB key drive, etc.), or any other suitable device. One or more other portions of the parsed user enrollment data may be stored in the system performing the authentication. This provides an added level of security to the authentication process (e.g., in addition to the biometric authentication information obtained from the biometric source, the user enrollment data must also be obtained via the appropriate parsed and split data portion).

[0417] The secure data parser of the present invention may be integrated into any suitable existing system in order to provide the use of its functionality in each system's respective environment. FIGURE 27 shows a block diagram of an illustrative system 2600, which may include software, hardware, or both for implementing any suitable application. System 2600 may be an existing system in which secure data parser 2602 may be retrofitted as an integrated component. Alternatively, secure data parser 2602 may be integrated into any suitable system 2600 from, for example, its earliest design stage. Secure data parser

2600 may be integrated at any suitable level of system 2600. For example, secure data parser 2602 may be integrated into system 2600 at a sufficiently back-end level such that the presence of secure data parser 2602 may be substantially transparent to an end user of system 2600. Secure data parser 2602 may be used for parsing and splitting data among one or more storage devices 2604 in accordance with the present invention. Some illustrative examples of systems having the secure data parser integrated therein are discussed below.

[0418] The secure data parser of the present invention may be integrated into an operating system kernel (e.g., Linux, Unix, or any other suitable commercial or proprietary operating system). This integration may be used to protect data at the device level whereby, for example, data that would ordinarily be stored in one or more devices is separated into a certain number of portions by the secure data parser integrated into the operating system and stored among the one or more devices. When original data is attempted to be accessed, the appropriate software, also integrated into the operating system, may recombine the parsed data portions into the original data in a way that may be transparent to the end user.

[0419] The secure data parser of the present invention may be integrated into a volume manager or any other suitable component of a storage system to protect local and networked data storage across any or all supported platforms. For example, with the secure data parser integrated, a storage system may make use of the redundancy offered by the secure data parser (i.e., which is used to implement the feature of needing fewer than all separated portions of data in

order to reconstruct the original data) to protect against data loss. The secure data parser also allows all data written to storage devices, whether using redundancy or not, to be in the form of multiple portions that are generated according to the parsing of the present invention. When original data is attempted to be accessed, the appropriate software, also integrated into the volume manager or other suitable component of the storage system, may recombine the parsed data portions into the original data in a way that may be transparent to the end user.

[0420] In one suitable approach, the secure data parser of the present invention may be integrated into a RAID controller (as either hardware or software). This allows for the secure storage of data to multiple drives while maintaining fault tolerance in case of drive failure.

[0421] The secure data parser of the present invention may be integrated into a database in order to, for example, protect sensitive table information. For example, in one suitable approach, data associated with particular cells of a database table (e.g., individual cells, one or more particular columns, one or more particular rows, any combination thereof, or an entire database table) may be parsed and separated according to the present invention (e.g., where the different portions are stored on one or more storage devices at one or more locations or on a single storage device). Access to recombine the portions in order to view the original data may be granted by traditional authentication methods (e.g., username and password query).

[0422] The secure parser of the present invention may be integrated in any suitable system that involves data in motion (i.e., transfer of data from one location to another). Such systems include, for example, email, streaming data broadcasts, and wireless (e.g., WiFi) communications. With respect to email, in one suitable approach, the secure parser may be used to parse outgoing messages (i.e., containing text, binary data, or both (e.g., files attached to an email message)) and sending the different portions of the parsed data along different paths thus creating multiple streams of data. If any one of these streams of data is compromised, the original message remains secure because the system may require that more than one of the portions be combined, in accordance with the present invention, in order to generate the original data. In another suitable approach, the different portions of data may be communicated along one path sequentially so that if one portion is obtained, it may not be sufficient to generate the original data. The different portions arrive at the intended recipient's location and may be combined to generate the original data in accordance with the present invention.

[0423] FIGURES 28 and 29 are illustrative block diagrams of such email systems. FIGURE 28 shows a sender system 2700, which may include any suitable hardware, such as a computer terminal, personal computer, handheld device (e.g., PDA, Blackberry), cellular telephone, computer network, any other suitable hardware, or any combination thereof. Sender system 2700 is used to generate and/or store a message 2704, which may be, for example, an email message, a binary data file (e.g., graphics, voice, video, etc.),

or both. Message 2704 is parsed and split by secure data parser 2702 in accordance with the present invention. The resultant data portions may be communicated across one or more separate communications paths 2706 over network 2708 (e.g., the Internet, an intranet, a LAN, WiFi, Bluetooth, any other suitable hard-wired or wireless communications means, or any combination thereof) to recipient system 2710. The data portions may be communicated parallel in time or alternatively, according to any suitable time delay between the communication of the different data portions. Recipient system 2710 may be any suitable hardware as described above with respect to sender system 2700. The separate data portions carried along communications paths 2706 are recombined at recipient system 2710 to generate the original message or data in accordance with the present invention.

[0424] FIGURE 29 shows a sender system 2800, which may include any suitable hardware, such as a computer terminal, personal computer, handheld device (e.g., PDA), cellular telephone, computer network, any other suitable hardware, or any combination thereof. Sender system 2800 is used to generate and/or store a message 2804, which may be, for example, an email message, a binary data file (e.g., graphics, voice, video, etc.), or both. Message 2804 is parsed and split by secure data parser 2802 in accordance with the present invention. The resultant data portions may be communicated across a single communications paths 2806 over network 2808 (e.g., the Internet, an intranet, a LAN, WiFi, Bluetooth, any other suitable communications means, or any combination thereof) to recipient system 2810. The data portions may be communicated serially

across communications path 2806 with respect to one another. Recipient system 2810 may be any suitable hardware as described above with respect to sender system 2800. The separate data portions carried along communications path 2806 are recombined at recipient system 2810 to generate the original message or data in accordance with the present invention.

[0425] It will be understood that the arrangement of FIGURES 28 and 29 are merely illustrative. Any other suitable arrangement may be used. For example, in another suitable approach, the features of the systems of FIGURES 28 and 29 may be combined whereby the multi-path approach of FIGURE 28 is used and in which one or more of communications paths 2706 are used to carry more than one portion of data as communications path 2806 does in the context of FIGURE 29.

[0426] The secure data parser may be integrated at any suitable level of a data-in motion system. For example, in the context of an email system, the secure parser may be integrated at the user-interface level (e.g., into Microsoft® Outlook), in which case the user may have control over the use of the secure data parser features when using email. Alternatively, the secure parser may be implemented in a back-end component such as at the exchange server, in which case messages may be automatically parsed, split, and communicated along different paths in accordance with the present invention without any user intervention.

[0427] Similarly, in the case of streaming broadcasts of data (e.g., audio, video), the outgoing data may be parsed and separated into multiple streams each containing a portion of the parsed data. The multiple streams may be transmitted along one or more

paths and recombined at the recipient's location in accordance with the present invention. One of the benefits of this approach is that it avoids the relatively large overhead associated with traditional encryption of data followed by transmission of the encrypted data over a single communications channel. The secure parser of the present invention allows data in motion to be sent in multiple parallel streams, increasing speed and efficiency.

[0428] It will be understand that the secure data parser may be integrated for protection of and fault tolerance of any type of data in motion through any transport medium, including, for example, wired, wireless, or physical. For example, voice over Internet protocol (VoIP) applications may make use of the secure data parser of the present invention. Wireless or wired data transport from or to any suitable personal digital assistant (PDA) devices such as Blackberries and SmartPhones may be secured using the secure data parser of the present invention. Communications using wireless 802.11 protocols for peer to peer and hub based wireless networks, satellite communications, point to point wireless communications, Internet client/server communications, or any other suitable communications may involve the data in motion capabilities of the secure data parser in accordance with the present invention. Data communication between computer peripheral device (e.g., printer, scanner, monitor, keyboard, network router, biometric authentication device (e.g., fingerprint scanner), or any other suitable peripheral device) between a computer and a computer peripheral device, between a computer peripheral device and any other suitable

device, or any combination thereof may make use of the data in motion features of the present invention.

[0429] The data in motion features of the present invention may also apply to physical transportation of secure shares using for example, separate routes, vehicles, methods, any other suitable physical transportation, or any combination thereof. For example, physical transportation of data may take place on digital/magnetic tapes, floppy disks, optical disks, physical tokens, USB drives, removable hard drives, consumer electronic devices with flash memory (e.g., Apple IPODs or other MP3 players), flash memory, any other suitable medium used for transporting data, or any combination thereof.

[0430] The secure data parser of the present invention may provide security with the ability for disaster recovery. According to the present invention, fewer than all portions of the separated data generated by the secure data parser may be necessary in order to retrieve the original data. That is, out of m portions stored, n may be the minimum number of these m portions necessary to retrieve the original data, where n <= m. For example, if each of four portions is stored in a different physical location relative to the other three portions, then, if n=2 in this example, two of the locations may be compromised whereby data is destroyed or inaccessible, and the original data may still be retrieved from the portions in the other two locations. Any suitable value for n or m may be used.

[0431] In addition, the n of m feature of the present invention may be used to create a "two man rule" whereby in order to avoid entrusting a single individual or any other entity with full access to what

may be sensitive data, two or more distinct entities, each with a portion of the separated data parsed by the secure parser of the present invention may need to agree to put their portions together in order to retrieve the original data.

[0432] The secure data parser of the present invention may be used to provide a group of entities with a group-wide key that allows the group members to access particular information authorized to be accessed by that particular group. The group key may be one of the data portions generated by the secure parser in accordance with the present invention that may be required to be combined with another portion centrally stored, for example in order to retrieve the information sought. This feature allows for, for example, secure collaboration among a group. It may be applied in for example, dedicated networks, virtual private networks, intranets, or any other suitable network.

[0433] Specific applications of this use of the secure parser include, for example, coalition information sharing in which, for example, multi-national friendly government forces are given the capability to communicate operational and otherwise sensitive data on a security level authorized to each respective country over a single network or a dual network (i.e., as compared to the many networks involving relatively substantial manual processes currently used). This capability is also applicable for companies or other organizations in which information needed to be known by one or more specific individuals (within the organization or without) may be communicated over a single network without the need to

worry about unauthorized individuals viewing the information.

[0434] Another specific application includes a multi-level security hierarchy for government systems. That is, the secure parser of the present invention may provide for the ability to operate a government system at different levels of classified information (e.g., unclassified, classified, secret, top secret) using a single network. If desired, more networks may be used (e.g., a separate network for top secret), but the present invention allows for substantially fewer than current arrangement in which a separate network is used for each level of classification.

[0435] It will be understood that any combination of the above described applications of the secure parser of the present invention may be used. For example, the group key application can be used together with the data in motion security application (i.e., whereby data that is communicated over a network can only be accessed by a member of the respective group and where, while the data is in motion, it is split among multiple paths (or sent in sequential portions) in accordance with the present invention).

[0436] The secure data parser of the present invention may be integrated into any middleware application to enable applications to securely store data to different database products or to different devices without modification to either the applications or the database. Middleware is a general term for any product that allows two separate and already existing programs to communicate. For example, in one suitable approach, middleware having the secure data parser integrated, may be used to allow programs written for a

particular database to communicate with other databases without custom coding.

[0437] The secure data parser of the present invention may be implemented having any combination of any suitable capabilities, such as those discussed herein. In some embodiments of the present invention, for example, the secure data parser may be implemented having only certain capabilities whereas other capabilities may be obtained through the use of external software, hardware, or both interfaced either directly or indirectly with the secure data parser.

[0438] FIGURE 30, for example, shows an illustrative implementation of the secure data parser as secure data parser 3000. Secure data parser 3000 may be implemented with very few built-in capabilities. As illustrated, secure data parser 3000 may include built-in capabilities for parsing and splitting data into portions (also referred to herein as shares) of data using module 3002 in accordance with the present invention. Secure data parser 3000 may also include built in capabilities for performing redundancy in order to be able to implement, for example, the m of n feature described above (i.e., recreating the original data using fewer than all shares of parsed and split data) using module 3004. Secure data parser 3000 may also include share distribution capabilities using module 3006 for placing the shares of data into buffers from which they are sent for communication to a remote location, for storage, etc. in accordance with the present invention. It will be understood that any other suitable capabilities may be built into secure data parser 3000.

[0439] Assembled data buffer 3008 may be any suitable memory used to store the original data (although not necessarily in its original form) that will be parsed and split by secure data parser 3000. In a splitting operation, assembled data buffer 3008 provides input to secure data parser 3008. In a restore operation, assembled data buffer 3008 may be used to store the output of secure data parser 3000.

[0440] Split shares buffers 3010 may be one or more memory modules that may be used to store the multiple shares of data that resulted from the parsing and splitting of original data. In a splitting operation, split shares buffers 3010 hold the output of the secure data parser. In a restore operation, split shares buffers hold the input to secure data parser 3000.

[0441] It will be understood that any other suitable arrangement of capabilities may be built-in for secure data parser 3000. Any additional features may be built-in and any of the features illustrated may be removed, made more robust, made less robust, or may otherwise be modified in any suitable way. Buffers 3008 and 3010 are likewise merely illustrative and may be modified, removed, or added to in any suitable way.

[0442] Any suitable modules implemented in software, hardware or both may be called by or may call to secure data parser 3000. If desired, even capabilities that are built into secure data parser 3000 may be replaced by one or more external modules. As illustrated, some external modules include random number generator 3012, cipher feedback key generator 3014, hash algorithm 3016, any one or more types of encryption 3018, and key management 3020. It will be understood that these are merely illustrative external modules. Any other

suitable modules may be used in addition to or in place of those illustrated.

[0443] Cipher feedback key generator 3014 may, externally to secure data parser 3000, generate for each secure data parser operation, a unique key, or random number (using, for example, random number generator 3012), to be used as a seed value for an operation that extends an original session key size (e.g., a value of 128, 256, 512, or 1024 bits) into a value equal to the length of the data to be parsed and split. Any suitable algorithm may be used for the cipher feedback key generation, including, for example, the AES cipher feedback key generation algorithm.

[0444] In order to facilitate integration of secure data parser 3000 and its external modules (i.e., secure data parser layer 3026) into an application layer 3024 (e.g., email application, database application, etc.), a wrapping layer that may make use of, for example, API function calls may be used. Any other suitable arrangement for facilitating integration of secure data parser layer 3026 into application layer 3024 may be used.

[0445] FIGURE 31 illustratively shows how the arrangement of FIGURE 30 may be used when a write (e.g., to a storage device), insert (e.g., in a database field), or transmit (e.g., across a network) command is issued in application layer 3024. At step 3100 data to be secured is identified and a call is made to the secure data parser. The call is passed through wrapper layer 3022 where at step 3102, wrapper layer 3022 streams the input data identified at step 3100 into assembled data buffer 3008. Also at step 3102, any suitable share information, filenames, any

other suitable information, or any combination thereof may be stored (e.g., as information 3106 at wrapper layer 3022). Secure data processor 3000 then parses and splits the data it takes as input from assembled data buffer 3008 in accordance with the present invention. It outputs the data shares into split shares buffers 3010. At step 3104, wrapper layer 3022 obtains from stored information 3106 any suitable share information (i.e., stored by wrapper 3022 at step 3102) and share location(s) (e.g., from one or more configuration files). Wrapper layer 3022 then writes the output shares (obtained from split shares buffers 3010) appropriately (e.g., written to one or more storage devices, communicated onto a network, etc.).

[0446] FIGURE 32 illustratively shows how the arrangement of FIGURE 30 may be used when a read (e.g., from a storage device), select (e.g., from a database field), or receive (e.g., from a network) occurs. At step 3200, data to be restored is identified and a call to secure data parser 3000 is made from application layer 3024. At step 3202, from wrapper layer 3022, any suitable share information is obtained and share location is determined. Wrapper layer 3022 loads the portions of data identified at step 3200 into split shares buffers 3010. Secure data parser 3000 then processes these shares in accordance with the present invention (e.g., if only three of four shares are available, then the redundancy capabilities of secure data parser 3000 may be used to restore the original data using only the three shares). The restored data is then stored in assembled data buffer 3008. At step 3204, application layer 3022 converts the data stored in assembled data buffer 3008 into its original data

format (if necessary) and provides the original data in its original format to application layer 3024.

[0447] It will be understood that the parsing and splitting of original data illustrated in FIGURE 31 and the restoring of portions of data into original data illustrated in FIGURE 32 is merely illustrative. Any other suitable processes, components, or both may be used in addition to or in place of those illustrated.

[0448] FIGURE 33 is a block diagram of an illustrative process flow for parsing and splitting original data into two or more portions of data in accordance with one embodiment of the present invention. As illustrated, the original data desired to be parsed and split is plain text 3306 (i.e., the word "SUMMIT" is used as an example). It will be understood that any other type of data may be parsed and split in accordance with the present invention. A session key 3300 is generated. If the length of session key 3300 is not compatible with the length of original data 3306, then cipher feedback session key 3304 may be generated.

[0449] In one suitable approach, original data 3306 may be encrypted prior to parsing, splitting, or both. For example, as FIGURE 33 illustrates, original data 3306 may be XORed with any suitable value (e.g., with cipher feedback session key 3304, or with any other suitable value). It will be understood that any other suitable encryption technique may be used in place of or in addition to the XOR technique illustrate. It will further be understood that although FIGURE 33 is illustrated in terms of byte by byte operations, the operation may take place at the bit level or at any other suitable level. It will further be understood

that, if desired, there need not be any encryption whatsoever of original data 3306.

[0450] The resultant encrypted data (or original data if no encryption took place) is then hashed to determine how to split the encrypted (or original) data among the output buckets (e.g., of which there are four in the illustrated example). In the illustrated example, the hashing takes place by bytes and is a function of cipher feedback session key 3304. It will be understood that this is merely illustrative. The hashing may be performed at the bit level, if desired. The hashing may be a function of any other suitable value besides cipher feedback session key 3304. In another suitable approach, hashing need not be used. Rather, any other suitable technique for splitting data may be employed.

[0451] FIGURE 34 is a block diagram of an illustrative process flow for restoring original data 3306 from two or more parsed and split portions of original data 3306 in accordance with one embodiment of the present invention. The process involves hashing the portions in reverse (i.e., to the process of FIGURE 33) as a function of cipher feedback session key 3304 to restore the encrypted original data (or original data if there was no encryption prior to the parsing and splitting). The encryption key may then be used to restore the original data (i.e., in the illustrated example, cipher feedback session key 3304 is used to decrypt the XOR encryption by XORing it with the encrypted data). This the restores original data 3306.

[0452] FIGURE 35 shows how bit-splitting may be implemented in the example of FIGURES 33 and 34. A hash may be used (e.g., as a function of the cipher

feedback session key, as a function of any other suitable value) to determine a bit value at which to split each byte of data. It will be understood that this is merely one illustrative way in which to implement splitting at the bit level. Any other suitable technique may be used.

[0453] It will be understood that any reference to hash functionality made herein may be made with respect to any suitable hash algorithm. These include for example, MD5 and SHA-1. Different hash algorithms may be used at different times and by different components of the present invention.

[0454] After a split point has been determined in accordance with the above illustrative procedure or through any other procedure or algorithm, a determination may be made with regard to which data portions to append each of the left and right segments. Any suitable algorithm may be used for making this determination. For example, in one suitable approach, a table of all possible distributions (e.g., in the form of pairings of destinations for the left segment and for the right segment) may be created, whereby a destination share value for each of the left and right segment may be determined by using any suitable hash function on corresponding data in the session key, cipher feedback session key, or any other suitable random or pseudo-random value, which may be generated and extended to the size of the original data. For example, a hash function of a corresponding byte in the random or pseudo-random value may be made. The output of the hash function is used to determine which pairing of destinations (i.e., one for the left segment and one for the right segment) to select from the table of all

the destination combinations. Based on this result, each segment of the split data unit is appended to the respective two shares indicated by the table value selected as a result of the hash function.

[0455] Redundancy information may be appended to the data portions in accordance with the present invention to allow for the restoration of the original data using fewer than all the data portions. For example, if two out of four portions are desired to be sufficient for restoration of data, then additional data from the shares may be accordingly appended to each share in, for example, a round-robin manner (e.g., where the size of the original data is 4MB, then share 1 gets its own shares as well as those of shares 2 and 3; share 2 gets its own share as well as those of shares 3 and 4; share 3 gets its own share as well as those of shares 4 and 1; and share 4 gets its own shares as well as those of shares 1 and 2). Any such suitable redundancy may be used in accordance with the present invention.

[0456] It will be understood that any other suitable parsing and splitting approach may be used to generate portions of data from an original data set in accordance with the present invention. For example, parsing and splitting may be randomly or pseudo-randomly processed on a bit by bit basis. A random or pseudo-random value may be used (e.g., session key, cipher feedback session key, etc.) whereby for each bit in the original data, the result of a hash function on corresponding data in the random or pseudo-random value may indicate to which share to append the respective bit. In one suitable approach the random or pseudo-random value may be generated as, or extended to, 8 times the size of the original data so that the hash

function may be performed on a corresponding byte of the random or pseudo-random value with respect to each bit of the original data. Any other suitable algorithm for parsing and splitting data on a bit by bit level may be used in accordance with the present invention. It will further be appreciated that redundancy data may be appended to the data shares such as, for example, in the manner described immediately above in accordance with the present invention.

[0457] In one suitable approach, parsing and splitting need not be random or pseudo-random. Rather, any suitable deterministic algorithm for parsing and splitting data may be used. For example, breaking up the original data into sequential shares may be employed as a parsing and splitting algorithm. Another example is to parse and split the original data bit by bit, appending each respective bit to the data shares sequentially in a round-robin manner. It will further be appreciated that redundancy data may be appended to the data shares such as, for example, in the manner described above in accordance with the present invention.

[0458] In one embodiment of the present invention, after the secure data parser generates a number of portions of original data, in order to restore the original data, certain one or more of the generated portions may be mandatory. For example, if one of the portions is used as an authentication share (e.g., saved on a physical token device), and if the fault tolerance feature of the secure data parser is being used (i.e., where fewer than all portions are necessary to restore the original data), then even though the secure data parser may have access to a sufficient

number of portions of the original data in order to restore the original data, it may require the authentication share stored on the physical token device before it restores the original data. It will be understood that any number and types of particular shares may be required based on, for example, application, type of data, user, any other suitable factors, or any combination thereof.

[0459] In one suitable approach, the secure data parser or some external component to the secure data parser may encrypt one or more portions of the original data. The encrypted portions may be required to be provided and decrypted in order to restore the original data. The different encrypted portions may be encrypted with different encryption keys. For example, this feature may be used to implement a more secure "two man rule" whereby a first user would need to have a particular share encrypted using a first encryption and a second user would need to have a particular share encrypted using a second encryption key. In order to access the original data, both users would need to have their respective encryption keys and provide their respective portions of the original data. In one suitable approach, a public key may be used to encrypt one or more data portions that may be a mandatory share required to restore the original data. A private key may then be used to decrypt the share in order to be used to restore to the original data.

[0460] Any such suitable paradigm may be used that makes use of mandatory shares where fewer than all shares are needed to restore original data.

[0461] In one suitable embodiment of the present invention, distribution of data into a finite number of

shares of data may be processed randomly or pseudo-randomly such that from a statistical perspective, the probability that any particular share of data receives a particular unit of data is equal to the probability that any one of the remaining shares will receive the unit of data. As a result, each share of data will have an approximately equal amount of data bits.

[0462] According to another embodiment of the present invention, each of the finite number of shares of data need not have an equal probability of receiving units of data from the parsing and splitting of the original data. Rather certain one or more shares may have a higher or lower probability than the remaining shares. As a result, certain shares may be larger or smaller in terms of bit size relative to other shares. For example, in a two-share scenario, one share may have a 1% probability of receiving a unit of data whereas the second share has a 99% probability. It should follow, therefore that once the data units have been distributed by the secure data parser among the two share, the first share should have approximately 1% of the data and the second share 99%. Any suitable probabilities may be used in accordance with the present invention.

[0463] It will be understood that the secure data parser may be programmed to distribute data to shares according to an exact (or near exact) percentage as well. For example, the secure data parser may be programmed to distribute 80% of data to a first share and the remaining 20% of data to a second share.

[0464] According to another embodiment of the present invention, the secure data parser may generate data shares, one or more of which have predefined

sizes. For example, the secure data parser may split original data into data portions where one of the portions is exactly 256 bits. In one suitable approach, if it is not possible to generate a data portion having the requisite size, then the secure data parser may pad the portion to make it the correct size. Any suitable size may be used.

[0465] In one suitable approach, the size of a data portion may be the size of an encryption key, a splitting key, any other suitable key, or any other suitable data element.

[0466] As previously discussed, the secure data parser may use keys in the parsing and splitting of data. For purposes of clarity and brevity, these keys shall be referred to herein as "splitting keys." For example, the Session Master Key, previously introduced, is one type of splitting key. Also, as previously discussed, splitting keys may be secured within shares of data generated by the secure data parser. Any suitable algorithms for securing splitting keys may be used to secure them among the shares of data. For example, the Shamir algorithm may be used to secure the splitting keys whereby information that may be used to reconstruct a splitting key is generated and appended to the shares of data. Any other such suitable algorithm may be used in accordance with the present invention.

[0467] Similarly, any suitable encryption keys may be secured within one or more shares of data according to any suitable algorithm such as the Shamir algorithm. For example, encryption keys used to encrypt a data set prior to parsing and splitting, encryption keys used to encrypt a data portions after parsing and splitting, or

both may be secured using, for example, the Shamir algorithm or any other suitable algorithm.

[0468] According to one embodiment of the present invention, an All or Nothing Transform (AoNT), such as a Full Package Transform, may be used to further secure data by transforming splitting keys, encryption keys, any other suitable data elements, or any combination thereof. For example, an encryption key used to encrypt a data set prior to parsing and splitting in accordance with the present invention may be transformed by an AoNT algorithm. The transformed encryption key may then be distributed among the data shares according to, for example, the Shamir algorithm or any other suitable algorithm. In order to reconstruct the encryption key, the encrypted data set must be restored (e.g., not necessarily using all the data shares if redundancy was used in accordance with the present invention) in order to access the necessary information regarding the transformation in accordance with AoNTs as is well known by one skilled in the art. When the original encryption key is retrieved, it may be used to decrypt the encrypted data set to retrieve the original data set. It will be understood that the fault tolerance features of the present invention may be used in conjunction with the AoNT feature. Namely, redundancy data may be included in the data portions such that fewer than all data portions are necessary to restore the encrypted data set.

[0469] It will be understood that the AoNT may be applied to encryption keys used to encrypt the data portions following parsing and splitting either in place of or in addition to the encryption and AoNT of the respective encryption key corresponding to the data

set prior to parsing and splitting. Likewise, AoNT may be applied to splitting keys.

[0470] In one embodiment of the present invention, encryption keys, splitting keys, or both as used in accordance with the present invention may be further encrypted using, for example, a workgroup key in order to provide an extra level of security to a secured data set.

[0471] In one embodiment of the present invention, an audit module may be provided that tracks whenever the secure data parser is invoked to split data.

[0472] FIGURE 36 illustrates possible options 3600 for using the components of the secure data parser in accordance with the invention. Each combination of options is outlined below and labeled with the appropriate step numbers from FIGURE 36. The secure data parser may be modular in nature, allowing for any known algorithm to be used within each of the function blocks shown in FIGURE 36. For example, other key splitting (e.g., secret sharing) algorithms such as Blakely may be used in place of Shamir, or the AES encryption could be replaced by other known encryption algorithms such as Triple DES. The labels shown in the example of FIGURE 36 merely depict one possible combination of algorithms for use in one embodiment of the invention. It should be understood that any suitable algorithm or combination of algorithms may be used in place of the labeled algorithms.

[0473] 1) 3610, 3612, 3614, 3615, 3616, 3617, 3618, 3619

[0474] Using previously encrypted data at step 3610, the data may be eventually split into a predefined number of shares. If the split algorithm requires a

key, a split encryption key may be generated at step 3612 using a cryptographically secure pseudo-random number generator. The split encryption key may optionally be transformed using an All or Nothing Transform (AoNT) into a transform split key at step 3614 before being key split to the predefined number of shares with fault tolerance at step 3615. The data may then be split into the predefined number of shares at step 3616. A fault tolerant scheme may be used at step 3617 to allow for regeneration of the data from less than the total number of shares. Once the shares are created, authentication/integrity information may be embedded into the shares at step 3618. Each share may be optionally post-encrypted at step 3619.

[0475] 2) 3111, 3612, 3614, 3615, 3616, 3617, 3618, 3619

[0476] In some embodiments, the input data may be encrypted using an encryption key provided by a user or an external system. The external key is provided at step 3611. For example, the key may be provided from an external key store. If the split algorithm requires a key, the split encryption key may be generated using a cryptographically secure pseudo-random number generator at step 3612. The split key may optionally be transformed using an All or Nothing Transform (AoNT) into a transform split encryption key at step 3614 before being key split to the predefined number of shares with fault tolerance at step 3615. The data is then split to a predefined number of shares at step 3616. A fault tolerant scheme may be used at step 3617 to allow for regeneration of the data from less than the total number of shares. Once the shares are created, authentication/integrity information may be

embedded into the shares at step 3618. Each share may be optionally post-encrypted at step 3619.

[0477] 3) 3612, 3613, 3614, 3615, 3612, 3614, 3615, 3616, 3617, 3618, 3619

[0478] In some embodiments, an encryption key may be generated using a cryptographically secure pseudo-random number generator at step 3612 to transform the data. Encryption of the data using the generated encryption key may occur at step 3613. The encryption key may optionally be transformed using an All or Nothing Transform (AoNT) into a transform encryption key at step 3614. The transform encryption key and/or generated encryption key may then be split into the predefined number of shares with fault tolerance at step 3615. If the split algorithm requires a key, generation of the split encryption key using a cryptographically secure pseudo-random number generator may occur at step 3612. The split key may optionally be transformed using an All or Nothing Transform (AoNT) into a transform split encryption key at step 3614 before being key split to the predefined number of shares with fault tolerance at step 3615. The data may then be split into a predefined number of shares at step 3616. A fault tolerant scheme may be used at step 3617 to allow for regeneration of the data from less than the total number of shares. Once the shares are created, authentication/integrity information will be embedded into the shares at step 3618. Each share may then be optionally post-encrypted at step 3619.

[0479] 4) 3612, 3614, 3615, 3616, 3617, 3618, 3619

[0480] In some embodiments, the data may be split into a predefined number of shares. If the split algorithm requires a key, generation of the split

encryption key using a cryptographically secure pseudo-random number generator may occur at step 3612. The split key may optionally be transformed using an All or Nothing Transform (AoNT) into a transformed split key at step 3614 before being key split into the predefined number of shares with fault tolerance at step 3615. The data may then be split at step 3616. A fault tolerant scheme may be used at step 3617 to allow for regeneration of the data from less than the total number of shares. Once the shares are created, authentication/integrity information may be embedded into the shares at step 3618. Each share may be optionally post-encrypted at step 3619.

[0481] Although the above four combinations of options are preferably used in some embodiments of the invention, any other suitable combinations of features, steps, or options may be used with the secure data parser in other embodiments.

[0482] The secure data parser may offer flexible data protection by facilitating physical separation. Data may be first encrypted, then split into shares with "m of n" fault tolerance. This allows for regeneration of the original information when less than the total number of shares is available. For example, some shares may be lost or corrupted in transmission. The lost or corrupted shares may be recreated from fault tolerance or integrity information appended to the shares, as discussed in more detail below.

[0483] In order to create the shares, a number of keys are optionally utilized by the secure data parser. These keys may include one or more of the following:

[0484] Pre-encryption key: When pre-encryption of the shares is selected, an external key may be passed to the secure data parser. This key may be generated and stored externally in a key store (or other location) and may be used to optionally encrypt data prior to data splitting.

[0485] Split encryption key: This key may be generated internally and used by the secure data parser to encrypt the data prior to splitting. This key may then be stored securely within the shares using a key split algorithm.

[0486] Split session key: This key is not used with an encryption algorithm; rather, it may be used to key the data partitioning algorithms when random splitting is selected. When a random split is used, a split session key may be generated internally and used by the secure data parser to partition the data into shares. This key may be stored securely within the shares using a key splitting algorithm.

[0487] Post encryption key: When post encryption of the shares is selected, an external key may be passed to the secure data parser and used to post encrypt the individual shares. This key may be generated and stored externally in a key store or other suitable location.

[0488] In some embodiments, when data is secured using the secure data parser in this way, the information may only be reassembled provided that all of the required shares and external encryption keys are present.

[0489] FIGURE 37 shows illustrative overview process 3700 for using the secure data parser of the present invention in some embodiments. As described above, two

well-suited functions for secure data parser 3706 may include encryption 3702 and backup 3704. As such, secure data parser 3706 may be integrated with a RAID or backup system or a hardware or software encryption engine in some embodiments.

[0490] The primary key processes associated with secure data parser 3706 may include one or more of pre-encryption process 3708, encrypt/transform process 3710, key secure process 3712, parse/distribute process 3714, fault tolerance process 3716, share authentication process 3716, and post-encryption process 3720. These processes may be executed in several suitable orders or combinations, as detailed in FIGURE 36. The combination and order of processes used may depend on the particular application or use, the level of security desired, whether optional pre-encryption, post-encryption, or both, are desired, the redundancy desired, the capabilities or performance of an underlying or integrated system, or any other suitable factor or combination of factors.

[0491] The output of illustrative process 3700 may be two or more shares 3722. As described above, data may be distributed to each of these shares randomly (or pseudo-randomly) in some embodiments. In other embodiments, a deterministic algorithm (or some suitable combination of random, pseudo-random, and deterministic algorithms) may be used.

[0492] In addition to the individual protection of information assets, there is sometimes a requirement to share information among different groups of users or communities of interest. It may then be necessary to either control access to the individual shares within that group of users or to share credentials among those

users that would only allow members of the group to reassemble the shares. To this end, a workgroup key may be deployed to group members in some embodiments of the invention. The workgroup key should be protected and kept confidential, as compromise of the workgroup key may potentially allow those outside the group to access information. Some systems and methods for workgroup key deployment and protection are discussed below.

[0493] The workgroup key concept allows for enhanced protection of information assets by encrypting key information stored within the shares. Once this operation is performed, even if all required shares and external keys are discovered, an attacker has no hope of recreating the information without access to the workgroup key.

[0494] FIGURE 38 shows illustrative block diagram 3800 for storing key and data components within the shares. In the example of diagram 3800, the optional pre-encrypt and post-encrypt steps are omitted, although these steps may be included in other embodiments.

[0495] The simplified process to split the data includes encrypting the data using encryption key 3804 at encryption stage 3802. Portions of encryption key 3804 may then be split and stored within shares 3810 in accordance with the present invention. Portions of split encryption key 3806 may also be stored within shares 3810. Using the split encryption key, data 3808 is then split and stored in shares 3810.

[0496] In order to restore the data, split encryption key 3806 may be retrieved and restored in accordance with the present invention. The split

operation may then be reversed to restore the ciphertext. Encryption key 3804 may also be retrieved and restored, and the ciphertext may then be decrypted using the encryption key.

[0497] When a workgroup key is utilized, the above process may be changed slightly to protect the encryption key with the workgroup key. The encryption key may then be encrypted with the workgroup key prior to being stored within the shares. The modified steps are shown in illustrative block diagram 3900 of FIGURE 39.

[0498] The simplified process to split the data using a workgroup key includes first encrypting the data using the encryption key at stage 3902. The encryption key may then be encrypted with the workgroup key at stage 3904. The encryption key encrypted with the workgroup key may then be split into portions and stored with shares 3912. Split key 3908 may also be split and stored in shares 3912. Finally, portions of data 3910 are split and stored in shares 3912 using split key 3908.

[0499] In order to restore the data, the split key may be retrieved and restored in accordance with the present invention. The split operation may then be reversed to restore the ciphertext in accordance with the present invention. The encryption key (which was encrypted with the workgroup key) may be retrieved and restored. The encryption key may then be decrypted using the workgroup key. Finally, the ciphertext may be decrypted using the encryption key.

[0500] There are several secure methods for deploying and protecting workgroup keys. The selection of which method to use for a particular application depends on a

number of factors. These factors may include security level required, cost, convenience, and the number of users in the workgroup. Some commonly used techniques used in some embodiments are provided below:

[0501] Hardware-based Key Storage

Hardware-based solutions generally provide the strongest guarantees for the security of encryption/decryption keys in an encryption system. Examples of hardware-based storage solutions include tamper-resistant key token devices which store keys in a portable device (e.g., smartcard/dongle), or non-portable key storage peripherals. These devices are designed to prevent easy duplication of key material by unauthorized parties. Keys may be generated by a trusted authority and distributed to users, or generated within the hardware. Additionally, many key storage systems provide for multi-factor authentication, where use of the keys requires access both a physical object (token) and a passphrase or biometric.

[0502] Software-based Key Storage

While dedicated hardware-based storage may be desirable for high-security deployments or applications, other deployments may elect to store keys directly on local hardware (e.g., disks, RAM or non-volatile RAM stores such as USB drives). This provides a lower level of protection against insider attacks, or in instances where an attacker is able to directly access the encryption machine.

[0503] To secure keys on disk, software-based key management often protects keys by storing them in encrypted form under a key derived from a combination of other authentication metrics, including: passwords

and passphrases, presence of other keys (e.g., from a hardware-based solution), biometrics, or any suitable combination of the foregoing. The level of security provided by such techniques may range from the relatively weak key protection mechanisms provided by some operating systems (e.g., MS Windows and Linux), to more robust solutions implemented using multi-factor authentication.

[0504] The secure data parser of the present invention may be advantageously used in a number of applications and technologies. For example, email system, RAID systems, video broadcasting systems, database systems, tape backup systems, or any other suitable system may have the secure data parser integrated at any suitable level. As previously discussed, it will be understand that the secure data parser may also be integrated for protection and fault tolerance of any type of data in motion through any transport medium, including, for example, wired, wireless, or physical transport mediums. As one example, voice over Internet protocol (VoIP) applications may make use of the secure data parser of the present invention to solve problems relating to echoes and delays that are commonly found in VoIP. The need for network retry on dropped packets may be eliminated by using fault tolerance, which guarantees packet delivery even with the loss of a predetermined number of shares. Packets of data (e.g., network packets) may also be efficiently split and restored "on-the-fly" with minimal delay and buffering, resulting in a comprehensive solution for various types of data in motion. The secure data parser may act on network data packets, network voice packets, file

system data blocks, or any other suitable unit of information. In addition to being integrated with a VoIP application, the secure data parser may be integrated with a file-sharing application (e.g., a peer-to-peer file-sharing application), a video broadcasting application, an electronic voting or polling application (which may implement an electronic voting protocol and blind signatures, such as the Sensus protocol), an email application, or any other network application that may require or desire secure communication.

[0505] In some embodiments, support for network data in motion may be provided by the secure data parser of the present invention in two distinct phases -- a header generation phase and a data partitioning phase. Simplified header generation process 4000 and simplified data partitioning process 4010 are shown in FIGURES 40A and 40B, respectively. One or both of these processes may be performed on network packets, file system blocks, or any other suitable information.

[0506] In some embodiments, header generation process 4000 may be performed one time at the initiation of a network packet stream. At step 4002, a random (or pseudo-random) split encryption key, K, may be generated. The split encryption key, K, may then be optionally encrypted (e.g., using the workgroup key described above) at AES key wrap step 4004. Although an AES key wrap may be used in some embodiments, any suitable key encryption or key wrap algorithm may be used in other embodiments. AES key wrap step 4004 may operate on the entire split encryption key, K, or the split encryption key may be parsed into several blocks (e.g., 64-bit blocks). AES key wrap step 4004 may then

operate on blocks of the split encryption key, if desired.

[0507] At step 4006, a secret sharing algorithm (e.g., Shamir) may be used to split the split encryption key, K, into key shares. Each key share may then be embedded into one of the output shares (e.g., in the share headers). Finally, a share integrity block and (optionally) a post-authentication tag (e.g., MAC) may be appended to the header block of each share. Each header block may be designed to fit within a single data packet.

[0508] After header generation is complete (e.g., using simplified header generation process 4000), the secure data parser may enter the data partitioning phase using simplified data splitting process 4010. Each incoming data packet or data block in the stream is encrypted using the split encryption key, K, at step 4012. At step 4014, share integrity information (e.g., a hash H) may be computed on the resulting ciphertext from step 4012. For example, a SHA-256 hash may be computed. At step 4106, the data packet or data block may then be partitioned into two or more data shares using one of the data splitting algorithms described above in accordance with the present invention. In some embodiments, the data packet or data block may be split so that each data share contains a substantially random distribution of the encrypted data packet or data block. The integrity information (e.g., hash H) may then be appended to each data share. An optional post-authentication tag (e.g., MAC) may also be computed and appended to each data share in some embodiments.

[0509] Each data share may include metadata, which may be necessary to permit correct reconstruction of the data blocks or data packets. This information may be included in the share header. The metadata may include such information as cryptographic key shares, key identities, share nonces, signatures/MAC values, and integrity blocks. In order to maximize bandwidth efficiency, the metadata may be stored in a compact binary format.

[0510] For example, in some embodiments, the share header includes a cleartext header chunk, which is not encrypted and may include such elements as the Shamir key share, per-session nonce, per-share nonce, key identifiers (e.g., a workgroup key identifier and a post-authentication key identifier). The share header may also include an encrypted header chunk, which is encrypted with the split encryption key. An integrity header chunk, which may include integrity checks for any number of the previous blocks (e.g., the previous two blocks) may also be included in the header. Any other suitable values or information may also be included in the share header.

[0511] As shown in illustrative share format 4100 of FIGURE 41, header block 4102 may be associated with two or more output blocks 4104. Each header block, such as header block 4102, may be designed to fit within a single network data packet. In some embodiments, after header block 4102 is transmitted from a first location to a second location, the output blocks may then be transmitted. Alternatively, header block 4102 and output blocks 4104 may be transmitted at the same time in parallel. The transmission may occur over one or more similar or dissimilar communications paths.

[0512] Each output block may include data portion 4106 and integrity/authenticity portion 4108. As described above, each data share may be secured using a share integrity portion including share integrity information (e.g., a SHA-256 hash) of the encrypted, pre-partitioned data. To verify the integrity of the outputs blocks at recovery time, the secure data parser may compare the share integrity blocks of each share and then invert the split algorithm. The hash of the recovered data may then be verified against the share hash.

[0513] As previously mentioned, in some embodiments of the present invention, the secure date parser may be used in conjunction with a tape backup system. For example, an individual tape may be used as a node (*i.e.*, portion/share) in accordance with the present invention. Any other suitable arrangement may be used. For example, a tape library or subsystem, which is made up of two or more tapes, may be treated as a single node.

[0514] Redundancy may also be used with the tapes in accordance with the present invention. For example, if a data set is apportioned among four tapes (*i.e.*, portions/shares), then two of the four tapes may be necessary in order to restore the original data. It will be understood that any suitable number of nodes (*i.e.*, less than the total number of nodes) may be required to restore the original data in accordance with the redundancy features of the present invention. This substantially increases the probability for restoration when one or more tapes expire.

[0515] Each tape may also be digitally protected with a SHA-256, HMAC hash value, any other suitable

value, or any combination thereof to insure against tampering. Should any data on the tape or the hash value change, that tape would not be a candidate for restoration and any minimum required number of tapes of the remaining tapes would be used to restore the data.

[0516] In conventional tape backup systems, when a user calls for data to be written to or read from a tape, the tape management system (TMS) presents a number that corresponds to a physical tape mount. This tape mount points to a physical drive where the data will be mounted. The tape is loaded either by a human tape operator or by a tape robot in a tape silo.

[0517] Under the present invention, the physical tape mount may be considered a logical mount point that points to a number of physical tapes. This not only increases the data capacity but also improves the performance because of the parallelism.

[0518] For increased performance the tape nodes may be or may include a RAID array of disks used for storing tape images. This allows for high-speed restoration because the data may always be available in the protected RAID.

[0519] In any of the foregoing embodiments, the data to be secured may be distributed into a plurality of shares using deterministic, probabilistic, or both deterministic and probabilistic data distribution techniques. In order to prevent an attacker from beginning a crypto attack on any cipher block, the bits from cipher blocks may be deterministically distributed to the shares. For example, the distribution may be performed using the BitSegment routine, or the BlockSegment routine may be modified to allow for distribution of portions of blocks to multiple shares.

This strategy may defend against an attacker who has accumulated less than "M" shares.

[0520] In some embodiments, a keyed secret sharing routine may be employed using keyed information dispersal (*e.g.*, through the use of a keyed information dispersal algorithm or "IDA"). The key for the keyed IDA may also be protected by one or more external workgroup keys, one or more shared keys, or any combination of workgroup keys and shared keys. In this way, a multi-factor secret sharing scheme may be employed. To reconstruct the data, at least "M" shares plus the workgroup key(s) (and/or shared key(s)) may be required in some embodiments. The IDA (or the key for the IDA) may also be driven into the encryption process. For example, the transform may be driven into the clear text (*e.g.*, during the pre-processing layer before encrypting) and may further protect the clear text before it is encrypted.

[0521] For example, in some embodiments, keyed information dispersal is used to distribute unique portions of data from a data set into two or more shares. The keyed information dispersal may use a session key to first encrypt the data set, to distribute unique portions of encrypted data from the data set into two or more encrypted data set shares, or both encrypt the data set and distribute unique portions of encrypted data from the data set into the two or more encrypted data set shares. For example, to distribute unique portions of the data set or encrypted data set, secret sharing (or the methods described above, such as BitSegment or BlockSegment) may be used. The session key may then optionally be transformed (for example, using a full package transform or AoNT) and

shared using, for example, secret sharing (or the keyed information dispersal and session key).

[0522] In some embodiments, the session key may be encrypted using a shared key (e.g., a workgroup key) before unique portions of the key are distributed or shared into two or more session key shares. Two or more user shares may then be formed by combining at least one encrypted data set share and at least one session key share. In forming a user share, in some embodiments, the at least one session key share may be interleaved into an encrypted data set share. In other embodiments, the at least one session key share may be inserted into an encrypted data set share at a location based at least in part on the shared workgroup key. For example, keyed information dispersal may be used to distribute each session key share into a unique encrypted data set share to form a user share. Interleaving or inserting a session key share into an encrypted data set share at a location based at least in part on the shared workgroup may provide increased security in the face of cryptographic attacks. In other embodiments, one or more session key shares may be appended to the beginning or end of an encrypted data set share to form a user share. The collection of user shares may then be stored separately on at least one data depository. The data depository or depositories may be located in the same physical location (for example, on the same magnetic or tape storage device) or geographically separated (for example, on physically separated servers in different geographic locations). To reconstruct the original data set, an authorized set of user shares and the shared workgroup key may be required.

[0523] Keyed information dispersal may be secure even in the face of key-retrieval oracles. For example, take a blockcipher E and a key-retrieval oracle for E that takes a list $(X_1, Y_1), \ldots, (X_c, Y_c)$ of input/output pairs to the blockcipher, and returns a key K that is consistent with the input/output examples (*e.g.*, $Y_i = E_K(X_i)$ for all i). The oracle may return the distinguished value $\perp$ if there is no consistent key. This oracle may model a cryptanalytic attack that may recover a key from a list of input/output examples.

[0524] Standard blockcipher-based schemes may fail in the presence of a key-retrieval oracle. For example, CBC encryption or the CBC MAC may become completely insecure in the presence of a key-retrieval oracle.

[0525] If Π^{IDA} is an IDA scheme and Π^{Enc} is an encryption scheme given by a mode of operation of some blockcipher E, then (Π^{IDA}, Π^{Enc}) provides security in the face of a key-retrieval attack if the two schemes, when combined with an arbitrary perfect secret-sharing scheme (PSS) as per HK1 or HK2, achieve the robust computational secret sharing (RCSS) goal, but in the model in which the adversary has a key-retrieval oracle.

[0526] If there exists an IDA scheme Π^{IDA} and an encryption scheme Π^{Enc} such that the pair of schemes provides security in the face of key-retrieval attacks, then one way to achieve this pair may be to have a "clever" IDA and a "dumb" encryption scheme. Another way to achieve this pair of schemes may be to have a "dumb" IDA and a "clever" encryption scheme.

[0527] To illustrate the use of a clever IDA and a dumb encryption scheme, in some embodiments, the

encryption scheme may be CBC and the IDA may have a "weak privacy" property. The weak privacy property means, for example, that if the input to the IDA is a random sequence of blocks $M = M_1 \ldots M_l$ and the adversary obtains shares from a non-authorized collection, then there is some block index i such that it is infeasible for the adversary to compute M_i. Such a weakly-private IDA may be built by first applying to M an information-theoretic AoNT, such as Stinson's AoNT, and then applying a simple IDA such as BlockSegment, or a bit-efficient IDA like Rabin's scheme (*e.g.*, Reed-Solomon encoding).

[0528] To illustrate the use of a dumb IDA and a clever encryption scheme, in some embodiments, one may use a CBC mode with double encryption instead of single encryption. Now any IDA may be used, even replication. Having the key-retrieval oracle for the blockcipher would be useless to an adversary, as the adversary will be denied any singly-enciphered input/output example.

[0529] While a clever IDA has value, it may also be inessential in some contexts, in the sense that the "smarts" needed to provide security in the face of a key-retrieval attack could have been "pushed" elsewhere. For example, in some embodiments, no matter how smart the IDA, and for whatever goal is trying to be achieved with the IDA in the context of HK1/HK2, the smarts may be pushed out of the IDA and into the encryption scheme, being left with a fixed and dumb IDA.

[0530] Based on the above, in some embodiments, a "universally sound" clever IDA Π^{IDA} may be used. For example, an IDA is provided such that, for all

encryption schemes Π^{Enc}, the pair (Π^{IDA}, Π^{Enc}) universally provides security in the face of key-retrieval attacks.

[0531] In some embodiments, an encryption scheme is provided that is RCSS secure in the face of a key-retrieval oracle. The scheme may be integrated with HK1/HK2, with *any* IDA, to achieve security in the face of key-retrieval. Using the new scheme may be particularly useful, for example, for making symmetric encryption schemes more secure against key-retrieval attacks.

[0532] As mentioned above, classical secret-sharing notions are typically unkeyed. Thus, a secret is broken into shares, or reconstructed from them, in a way that requires neither the dealer nor the party reconstructing the secret to hold any kind of symmetric or asymmetric key. The secure data parser described herein, however, is optionally keyed. The dealer may provide a symmetric key that, if used for data sharing, may be required for data recovery. The secure data parser may use the symmetric key to disperse or distribute unique portions of the message to be secured into two or more shares.

[0533] The shared key may enable multi-factor or two-factor secret-sharing (2FSS). The adversary may then be required to navigate through two fundamentally different types of security in order to break the security mechanism. For example, to violate the secret-sharing goals, the adversary (1) may need to obtain the shares of an authorized set of players, *and* (2) may need to obtain a secret key that it should not be able to obtain (or break the cryptographic mechanism that is keyed by that key).

[0534] In some embodiments, a new set of additional requirements is added to the RCSS goal. The additional requirements may include the "second factor"—key possession. These additional requirements may be added without diminishing the original set of requirements. One set of requirements may relate to the adversary's inability to break the scheme if it knows the secret key but does not obtain enough shares (*e.g.*, the *classical* or *first-factor* requirements) while the other set of requirements may relate to the adversary's inability to break the scheme if it does have the secret key but manages to get hold of all of the shares (*e.g.*, the *new* or *second-factor* requirements).

[0535] In some embodiments, there may be two second-factor requirements: a privacy requirement and an authenticity requirement. In the privacy requirement, a game may be involved where a secret key K and a bit b are selected by the environment. The adversary now supplies a pair of equal-length messages in the domain of the secret-sharing scheme, M_1^0 and M_1^1. The environment computes the shares of M_1^b to get a vector of shares, $\boldsymbol{S}_1 = (\boldsymbol{S}_1[1], \ldots, \boldsymbol{S}_1[n])$, and it gives the shares $\boldsymbol{S}_1$ (all of them) to the adversary. The adversary may now choose another pair of messages (M_2^0, M_2^1) and everything proceeds as before, using the same key K and hidden bit b. The adversary's job is to output the bit b' that it believes to be b. The adversary privacy advantage is one less than twice the probability that $b = b'$. This games captures the notion that, even learning all the shares, the adversary still cannot learn anything about the shared secret if it lacks the secret key.

[0536] In the authenticity requirement, a game may

be involved where the environment chooses a secret key *K* and uses this in the subsequent calls to *Share* and *Recover*. *Share* and *Recover* may have their syntax modified, in some embodiments, to reflect the presence of this key. Then the adversary makes *Share* requests for whatever messages M_1, ... , M_q it chooses in the domain of the secret-sharing scheme. In response to each *Share* request it gets the corresponding *n*-vector of shares, $\boldsymbol{S}_1$, ... , $\boldsymbol{S}_q$. The adversary's aim is to *forge* a new plaintext; it wins if it outputs a vector of shares $\boldsymbol{S}'$ such that, when fed to the *Recover* algorithm, results in something *not* in $\{M_1, \ldots, M_q\}$. This is an "integrity of plaintext" notion.

[0537] There are two approaches to achieve multi-factor secret-sharing. The first is a generic approach -- generic in the sense of using an underlying (R)CSS scheme in a black-box way. An authenticated-encryption scheme is used to encrypt the message that is to be CSS-shared, and then the resulting ciphertext may be shared out, for example, using a secret sharing algorithm, such as Blakely or Shamir.

[0538] A potentially more efficient approach is to allow the shared key to be the workgroup key. Namely, (1) the randomly generated session key of the (R)CSS scheme may be encrypted using the shared key, and (2) the encryption scheme applied to the message (*e.g.*, the file) may be replaced by an authenticated-encryption scheme. This approach may entail only a minimal degradation in performance.

[0539] The secure data parser of the present invention may be used to implement a cloud computing security solution. Cloud computing is network-based computing, storage, or both where computing and storage

resources may be provided to computer systems and other devices over a network. Cloud computing resources are generally accessed over the Internet, but cloud computing may be performed over any suitable public or private network. Cloud computing may provide a level of abstraction between computing resources and their underlying hardware components (e.g., servers, storage devices, networks), enabling remote access to a pool of computing resources. These cloud computing resources may be collectively referred to as the "cloud." Cloud computing may be used to provide dynamically scalable and often virtualized resources as a service over the Internet or any other suitable network or combination of networks.

[0540] Security is an important concern with cloud computing because private data may be transferred over public networks and may be processed and stored within publicly accessible or shared systems. The secure data parser may be used to protect cloud computing resources and the data being communicated between the cloud and an end-user or device. For example, the secure data parser may be used to secure data storage in the cloud, data-in-motion in the cloud, network access in the cloud, data services in the cloud, access to high-performance computing resources in the cloud, and any other operations in the cloud.

[0541] FIGURE 42 is an illustrative block diagram of a cloud computing security solution. System 4200, including secure data parser 4210, is coupled to cloud 4250 including cloud resources 4260. System 4200 may include any suitable hardware, such as a computer terminal, personal computer, handheld device (e.g., PDA, Blackberry, smart phone, tablet device), cellular

telephone, computer network, any other suitable hardware, or any combination thereof. Secure data parser 4210 may be integrated at any suitable level of system 4200. For example, secure data parser 4210 may be integrated into the hardware and/or software of system 4200 at a sufficiently back-end level such that the presence of secure data parser 4210 may be substantially transparent to an end user of system 4200. The integration of the secure data parser within suitable systems is described in greater detail above with respect to, for example, FIGURES 27 and 28. Cloud 4250 includes multiple illustrative cloud resources 4260 including, data storage resources 4260a and 4260e, data service resources 4260b and 4260g, network access control resources 4260c and 4260h, and high performing computing resources 4260d and 4260f. Each of these resources will be described in greater detail below with respect to FIGURES 43-47. These cloud computing resources are merely illustrative. It should be understood that any suitable number and type of cloud computing resources may be accessible from system 4200.

[0542] One advantage of cloud computing is that the user of system 4200 may be able to access multiple cloud computing resources without having to invest in dedicated computer hardware. The user may have the ability to dynamically control the number and type of cloud computing resources accessible to system 4200. For example, system 4200 may be provided with on-demand storage resources in the cloud having capacities that are dynamically adjustable based on current needs. In some embodiments, one or more software applications executed on system 4200 may couple system 4200 to cloud

resources 4260. For example, an Internet web browser may be used to couple system 4200 to one or more cloud resources 4260 over the Internet. In some embodiments, hardware integrated with or connected to system 4200 may couple system 4200 to cloud resources 4260. In both embodiments, secure data parser 4210 may secure communications with cloud resources 4260 and/or the data stored within cloud resources 4260. The coupling of cloud resources 4260 to system 4200 may be transparent to system 4200 or the users of system 4200 such that cloud resources 4260 appear to system 4200 as local hardware resources. Furthermore shared cloud resources 4260 may appear to system 4200 as dedicated hardware resources.

[0543] Secure data parser 4210 may encrypt and split data such that no forensically discernable data will traverse or will be stored within the cloud. The underlying hardware components of the cloud (e.g., servers, storage devices, networks) may be geographically disbursed to ensure continuity of cloud resources in the event of a power grid failure, weather event or other man-made or natural event. As a result, even if some of the hardware components within the cloud suffer a catastrophic failure, the cloud resources may still be accessible. Cloud resources 4260 may be designed with redundancies to provide uninterrupted service in spite of one or more hardware failures.

[0544] FIGURE 43 is an illustrative block diagram of a cloud computing security solution for securing data in motion (i.e., during the transfer of data from one location to another) through the cloud. FIGURE 43 shows a sender system 4300, which may include any

suitable hardware, such as a computer terminal, personal computer, handheld device (e.g., PDA, Blackberry), cellular telephone, computer network, any other suitable hardware, or any combination thereof. Sender system 4300 is used to generate and/or store data, which may be, for example, an email message, a binary data file (e.g., graphics, voice, video, etc.), or both. The data is parsed and split by secure data parser 4310 in accordance with the present invention. The resultant data portions may be communicated over cloud 4350 to recipient system 4370. Recipient system 4370 may be any suitable hardware as described above with respect to sender system 4300. The separate data portions may be recombined at recipient system 4370 to generate the original data in accordance with the present invention. When traveling through cloud 4310 the data portions may be communicated across one or more communications paths including the Internet and/or one or more intranets, LANs, WiFi, Bluetooth, any other suitable hard-wired or wireless communications networks, or any combination thereof. As described above with respect to FIGURES 28 and 29, the original data is secured by the secure data parser even if some of the data portions are compromised.

[0545] FIGURE 44 is an illustrative block diagram of a cloud computing security solution for securing data services in the cloud. In this embodiment, a user 4400 may provide data services 4420 to an end user 4440 over cloud 4430. Secure parser 4410 may secure the data services in accordance with the disclosed embodiments. Data service 4420 may be any suitable application or software service that is accessible over cloud 4430. For example, data service 4420 may be a web-based

application implemented as part of a service-oriented architecture (SOA) system. Data service 4420 may be stored and executed on one or more systems within cloud 4430. The abstraction provided by this cloud computing implementation allows data service 4420 to appear as a virtualized resource to end user 4440 irrespective of the underlying hardware resources. Secure parser 4410 may secure data in motion between data service 4420 and end user 4440. Secure parser 4410 may also secure stored data associated with data service 4420. The stored data associated with data service 4420 may be secured within the system or systems implementing data service 4420 and/or within separate secure cloud data storage devices, which will be described in greater detail below. Although data service 4420 and other portions of FIGURE 44 are shown outside of cloud 4430, it should be understood that any of these elements may be incorporated within cloud 4430.

[0546] FIGURE 45 is an illustrative block diagram of a cloud computing security solution for securing data storage resources in the cloud. System 4500, including secure data parser 4510, is coupled to cloud 4550 which includes data storage resources 4560. Secure data parser 4510 may be used for parsing and splitting data among one or more data storage resources 4560. Each data storage resource 4560 may represent a one or more networked storage devices. These storage devices may be assigned to a single user/system of may be shared by multiple users/systems. The security provided by secure data parser 4510 may allow data from multiple users/systems to securely co-exist on the same storage devices. The abstraction provided by this cloud

computing implementation allows data storage resources 4560 to appear as a single virtualized storage resource to system 4500 irrespective of the number and location of the underlying data storage resources. When data is written to or read from data storage resources 4560, secure data parser 4510 may split and recombine the data in a way that may be transparent to the end user. In this manner, an end user may be able to access to dynamically scalable storage on demand.

[0547] Data storage in the cloud using secure data parser 4510 is secure, resilient, persistent, and private. Secure data parser 4510 secures the data by ensuring that no forensically discernable data traverses the cloud or is stored in a single storage device. The cloud storage system is resilient because of the redundancy offered by the secure data parser (i.e., fewer than all separated portions of data are needed to reconstruct the original data). Storing the separated portions within multiple storage devices and/or within multiple data storage resources 4560 ensures that the data may be reconstructed even if one or more of the storage devices fail or are inaccessible. The cloud storage system is persistent because loss of a storage device within data storage resources 4560 has no impact on the end user. If one storage device fails, the data portions that were stored within that storage device may be rebuilt at another storage device without having to expose the data. Furthermore, the storage resources 4560 (or even the multiple networked storage devices that make up a data storage resource 4560) may be geographically dispersed to limit the risk of multiple failures.

Finally, the data stored in the cloud may be kept private using one or more keys. As described above, data may be assigned to a user or a community of interest by unique keys such that only that user or community will have access to the data.

[0548] Data storage in the cloud using the secure data parser may also provide a performance boost over traditional local or networked storage. The throughput of the system may be improved by writing and reading separate portions of data to multiple storage devices in parallel. This increase in throughput may allow slower, less expensive storage devices to be used without substantially affecting the overall speed of the storage system.

[0549] FIGURE 46 is an illustrative block diagram for securing network access using a secure data parser in accordance with the disclosed embodiments. Secure data parser 4610 may be used with network access control block 4620 to control access to network resources. As illustrated in FIGURE 46, network access control block 4620 may be used to provide secure network communications between user 4600 and end user 4640. In some embodiments, network access control block 4620 may provide secure network access for one or more network resources in the cloud (e.g., cloud 4250, FIGURE 42). Authorized users (e.g., user 4600 and end user 4640) may be provided with group-wide keys that provide the users with the ability to securely communicate over a network and/or to access secure network resources. The secured network resources will not respond unless the proper credentials (e.g., group keys) are presented. This may prevent common networking attacks such as, for example, denial of

service attacks, port scanning attacks, man-in-the-middle attacks, and playback attacks.

[0550] In addition to providing security for data at rest stored within a communications network and security for data in motion through the communications network, network access control block 4620 may be used with secure data parser 4620 to share information among different groups of users or communities of interest. Collaboration groups may be set up to participate as secure communities of interest on secure virtual networks. A workgroup key may be deployed to group members to provide members of the group access to the network and networked resources. Systems and methods for workgroup key deployments have been discussed above.

[0551] FIGURE 47 is an illustrative block diagram for securing access to high performance computing resources using a secure data parser in accordance with the disclosed embodiments. Secure data parser 4710 may be used to provide secure access to high performance computing resources 4720. As illustrated in FIGURE 47 end user 4740 may access high performance computing resources 4720. In some embodiments, secure data parser 4710 may provide secure access to high performance resources in the cloud (e.g., cloud 4250, FIGURE 42). High performance computing resources may be large computer servers or server farms. These high performance computing resources may provide flexible, scalable, and configurable data services and data storage services to users.

[0552] In accordance with another embodiment, a secure data parser may be used to secure data access using virtual machines. A hypervisor, also referred to

as a virtual machine monitor (VMM) is a computer system that allows multiple virtual machines to run on a single host computer. FIGURE 48 shows an illustrative block diagram including hypervisor 4800 and a series of virtual machines 4810 running on hypervisor 4800. Hypervisor 4800 runs a fundamental operating system (e.g., Microsoft Windows® and Linux). Virtual machines 4810 may be firewalled off from the fundamental operating system such that attacks (e.g., viruses, worms, hacks, etc.) on the fundamental operating system do not affect virtual machines 4810. One or more secure data parsers may be integrated with hypervisor 4800 to secure virtual machines 4810. In particular, using the secure data parser, virtual machines 4810 may securely communicate with one or more servers or end users. In accordance with this embodiment, secure data access may be deployed to users by providing the users with secure virtual machine images. This embodiment may allow for on demand information sharing while assuring confidentiality and integrity of the data.

[0553] FIGURES 49 and 50 show alternative embodiments for integrating a secure data parser with a hypervisor. In FIGURE 49, secure data parser 4930 is implemented above hypervisor 4920. For example, secure data parser 4930 may be implemented as a software application or module operating on hypervisor 4920. In some embodiments, secure data parser 4930 may be implemented by a virtual machine running on hypervisor 4920. A virtual machine running on hypervisor 4920 may securely couple to server 4940 and end users 4950 using secure data parser 4930. In FIGURE 50, secure data parser 5030 is implemented below

hypervisor 5020. For example, secure data parser 5030 may be implemented within the hardware of hypervisor 5020. The virtual machine running on hypervisor 5020 may securely communicate with server 5040 and end users 5050 using secure data parser 5030.

[0554] In accordance with another embodiment, the secure data parser may be used to secure orthogonal division multiplexing (OFDM) communications channels. OFDM is a multiplexing scheme that is used for wideband digital communication. Broadband wireless standards (e.g., WiMAX and LTE) and broadband over power line (BPL) use OFDM. OFDM is unique because all adjacent channels are truly orthogonal. This eliminates crosstalk, cancellation, and induction of noise. Currently, in these OFDM standards, data is transmitted across a single OFDM communications channel. The secure data parser may secure OFDM communications by splitting data amongst multiple OFDM communications channels. As described above, splitting data amongst multiple data channels using the secure data parser secures the data because only a portion of the data is transmitted over each channel. As an additional benefit, the secure data parser may simultaneously transmit multiple data portions on multiple data channels. These simultaneous transmissions may increase the effective bandwidth of the data transmission. Additionally or alternatively, the secure data parser may transmit the same data portions on multiple data channels. This redundant transmission technique may increase transmission reliability. FIGURE 51 is an illustrative block diagram for securing an OFDM communications network. As illustrated in

FIGURE 51 end user 5110 may use secure data parser 5120 to send data over OFDM network 5140 to end user 5150. OFDM network 5140 may be a broadband over wireless network, a broadband over power line network, or any other suitable OFDM network.

[0555] In accordance with some other embodiments, the secure data parser may be used to protect critical infrastructure controls including, for example, the power grid. Internet Protocol version 6 (IPv6) is the next-generation Internet Protocol. IPv6 has a larger address space than the current Internet Protocol. When implemented, IPv6 will allow more devices to be directly accessed over the Internet. It is important that the controls of critical infrastructure be restricted to limit access to authorized individuals. As described above, the secure data parser may limit access to network resources to authorized users and groups. Critical systems may be protected using the "two man rule" whereby at least two users would need to provide their respective keys to access the critical systems. FIGURE 52 is an illustrative block diagram for securing the power grid. As illustrated in FIGURE 52 user 5210 may use secure data parser 5220 to provide secure access to power grid 5240 for end user end user 5250.

[0556] In some embodiments, power grid systems may be coupled to the Internet using broadband over power line networks to eliminate network cabling and associated equipment of typical communications networks. Coupling power grid systems to the Internet may enable smart grid technologies that allow for more efficient use of power by reporting usage in real time. As another benefit, high powered computing resources

and/or data storage facilities may be installed at Internet connected power monitoring facilities. These resources may provide reliable storage and processing nodes for protecting data in the cloud.

[0557] Although some applications of the secure data parser are described above, it should be clearly understood that the present invention may be integrated with any network application in order to increase security, fault-tolerance, anonymity, or any suitable combination of the foregoing.

[0558] Additionally, other combinations, additions, substitutions and modifications will be apparent to the skilled artisan in view of the disclosure herein.

What is Claimed is:

1. A method for securing data using a cloud computing storage network, the method comprising:

generating at a first system at least two portions of data from a data set;

communicating the at least two portions of data from the first system over a communications network to the cloud computing storage network, wherein the cloud computing storage network includes at least two respectively remote storage devices; and

storing the at least two portions of data within the cloud computing storage network on the at least two networked storage devices, whereby the data set is restorable from at least a subset of the at least two portions of data by recombining data from the subset of the at least two portions of data.

2. The method of claim 1, wherein generating the at least two portions of data and communicating the at least two portions of data to the cloud computing network is substantially transparent to the first system.

3. The method of claim 1, wherein the communications network comprises an unsecured public network.

4. The method of claim 1, wherein the data set is accessible at the first system.

5. The method of claim 1, wherein the data set is accessible by any system within a community of interest associated with the first system.

6. The method of claim 1, further comprising generating a virtual machine image, whereby the virtual machine image is operable to access the data set within the cloud computing storage network.

7. The method of claim 6, further comprising transmitting the virtual machine image to a second system, wherein the second system is operable to access the data set within the cloud computing storage network.

8. The method of claim 1, wherein the data set comprises a data service and wherein secure access to the data service is provided over the cloud computing storage network.

9. The method of claim 1, wherein the cloud computing storage network further comprises at least one resilient networked storage device.

10. The method of claim 9, wherein a data portion stored on the at least one resilient networked storage device is regenerated from other networked storage devices.

11. A method for securing data using an orthogonal frequency-division multiplexing (OFDM) network, the method comprising:

generating at a first location at least two portions of data from the data set; and

communicating from the first location to a second location the at least two portions of data over at least two OFDM channels, wherein different portions of data are communicated over different OFDM

channels, whereby the data set is restorable at the second location from at least a subset of the at least two portions of data by recombining data from the subset of the at least two portions of data received over the at least two OFDM channels.

12. The method of claim 11, wherein the OFDM network comprises a broadband wireless network.

13. The method of claim 11, wherein the OFDM network comprises a broadband powerline network.

14. An apparatus for securing data in a cloud computing storage network, the apparatus comprising:

a first system operable to:

generate at least two portions of data from a data set;

communicate the at least two portions of data over a communications network to the cloud computing storage network, wherein the cloud computing storage network includes at least two respectively remote storage devices; and

store the at least two portions of data within the cloud computing storage network on the at least two storage devices, whereby the data set is restorable from at least a subset of the at least two portions of data by recombining data from the subset of the at least two portions of data.

15. The apparatus of claim 14, wherein the first system is further operable to generate the at least two portions of data and to communicate the at least two portions of data to the cloud computing network in a substantially transparent manner.

16. The apparatus of claim 14, wherein the communications network comprises an unsecured public network.

17. The apparatus of claim 14, wherein the first system is further operable to access the data set within the cloud computing network.

18. The apparatus of claim 14, wherein the data set is accessible by any user system within a community of interest associated with the first system.

19. The apparatus of claim 14, wherein the first system is further operable to generate a virtual machine image, whereby the virtual machine image is operable to access the data set within the cloud computing storage network.

20. The apparatus of claim 19, wherein the first system is further operable to transmit the virtual machine image to a second system, wherein the second system is operable to access the data set within the cloud computing storage network.

21. The apparatus of claim 14, wherein the data set comprises a data service and wherein the first system is further operable to securely access to the data service over the cloud computing storage network.

22. The apparatus of claim 14, wherein the cloud computing storage network further comprises at least one resilient networked storage device.

23. The apparatus of claim 22, wherein the first system is further operable to regenerate a data

portion stored on the at least one resilient networked storage device from other networked storage devices.

24. An apparatus for securing data in an orthogonal frequency-division multiplexing (OFDM) network, the apparatus comprising:

a secure data parser operable to generate at a first location at least two portions of data from the data set; and

an OFDM transceiver operable to communicate from the first location to a second location the at least two portions of data over at least two OFDM channels, wherein different portions of data are communicated over different OFDM channels, whereby the data set is restorable at the second location from at least a subset of the at least two portions of data by recombining data from the subset of the at least two portions of data received over the at least two OFDM channels.

25. The apparatus of claim 24, wherein the OFDM network comprises a broadband wireless network.

26. The apparatus of claim 24, wherein the OFDM network comprises a broadband powerline network.

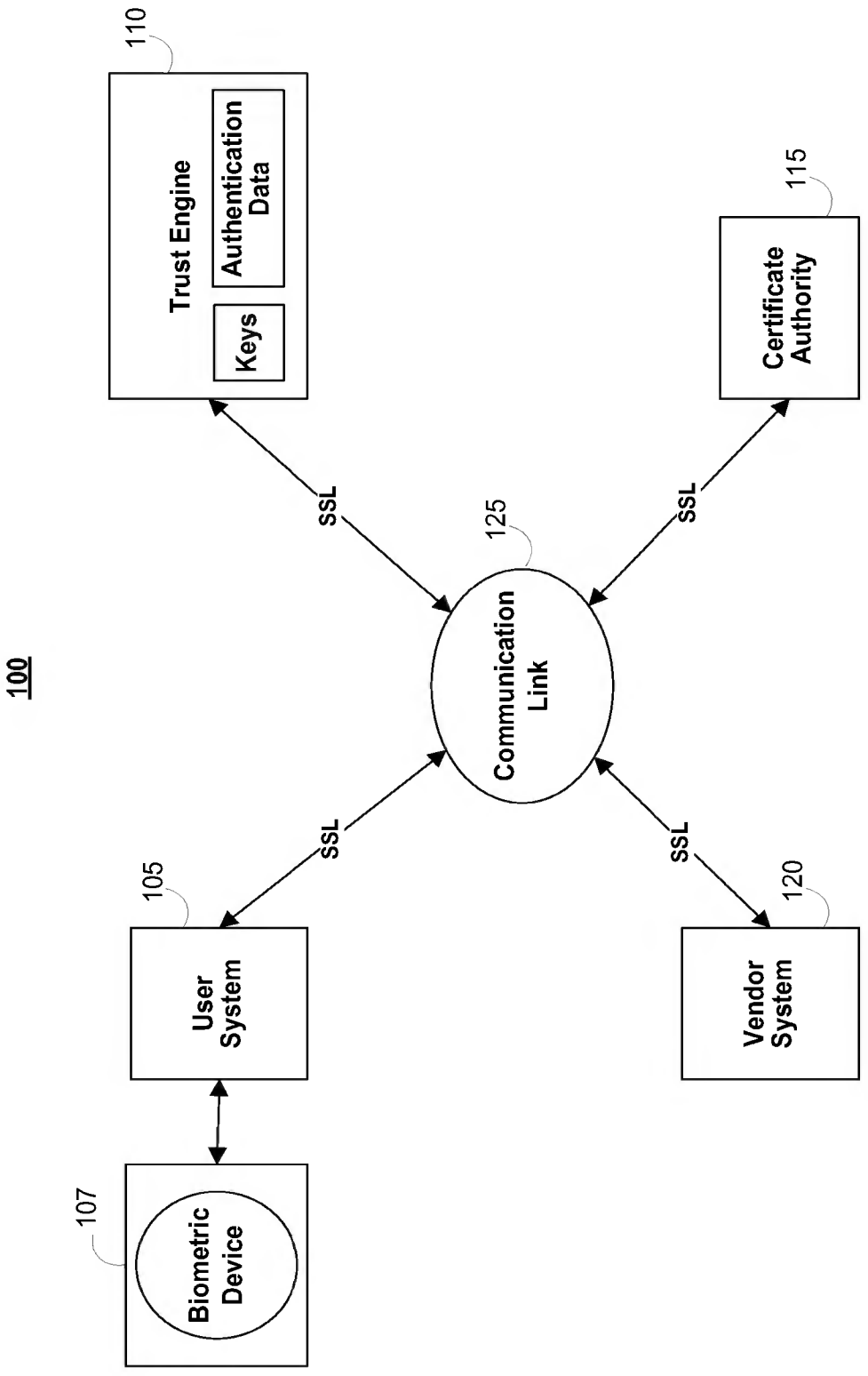

100
110
Trust Engine
Authentication Data
Keys
115
Certificate Authority
SSL
SSL
125
Communication Link
SSL
SSL
105
User System
120
Vendor System
107
Biometric Device

FIG. 1

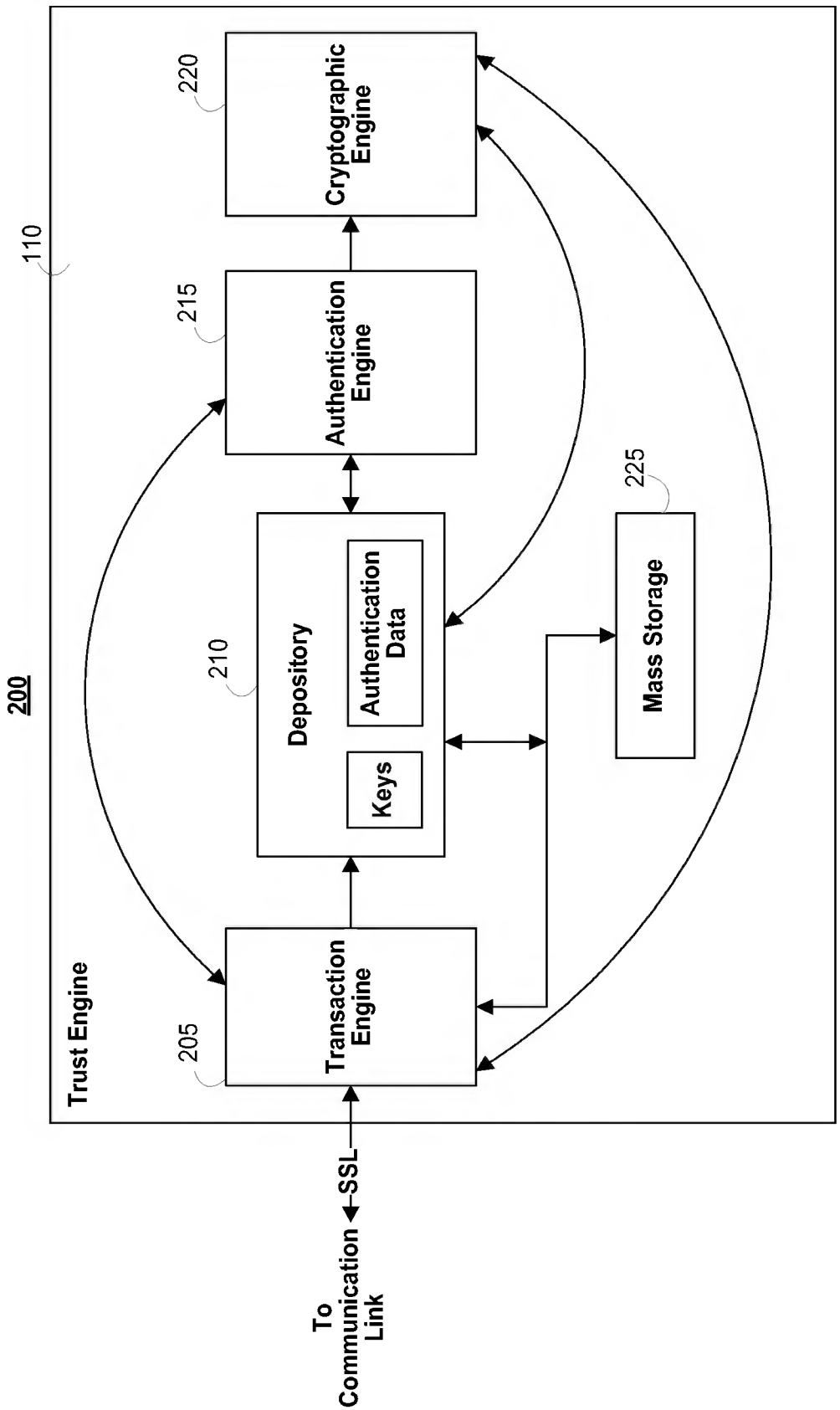

200
110
Trust Engine
205
Transaction Engine
210
Depository
Authentication Data
Keys
215
Authentication Engine
220
Cryptographic Engine
225
Mass Storage
SSL
To Communication Link

FIG. 2

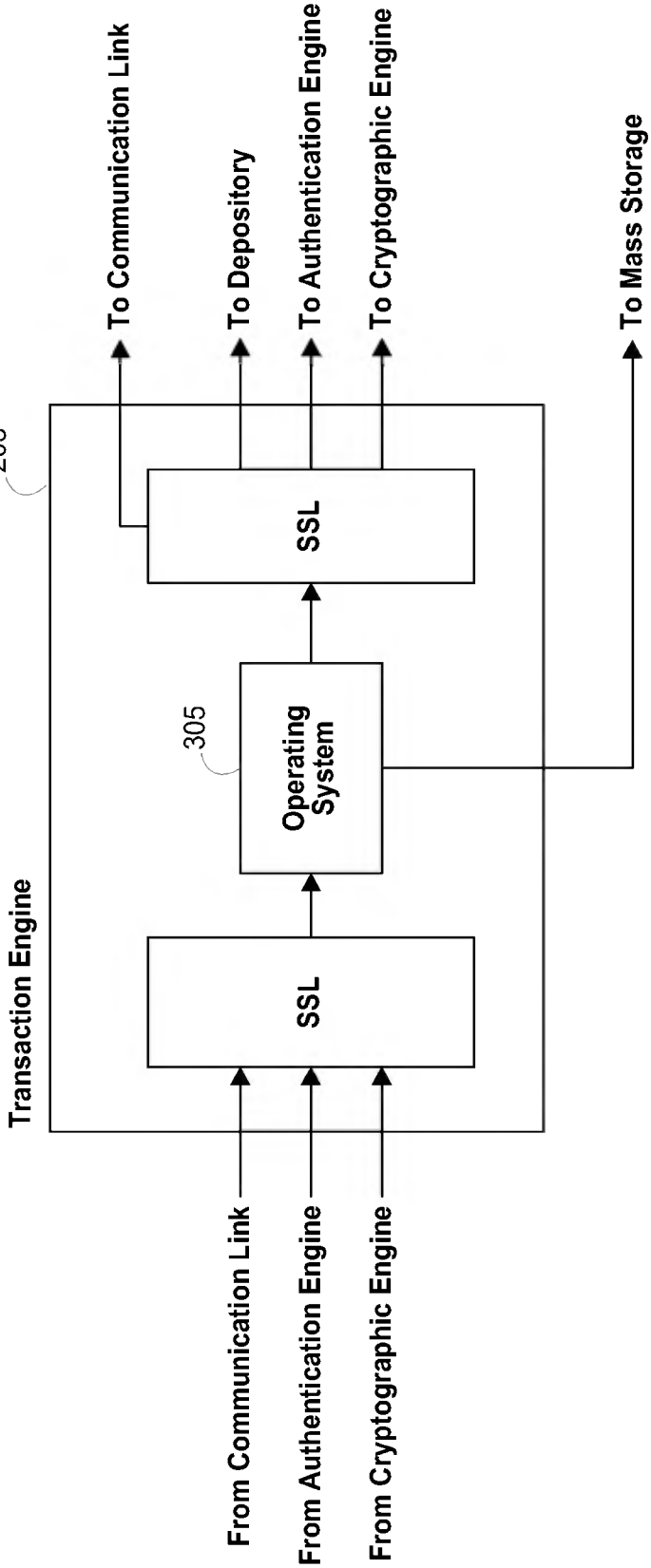

To Communication Link
To Depository
To Authentication Engine
To Cryptographic Engine
To Mass Storage
205
SSL
305
Operating System
SSL
Transaction Engine
From Communication Link
From Authentication Engine
From Cryptographic Engine

FIG. 3

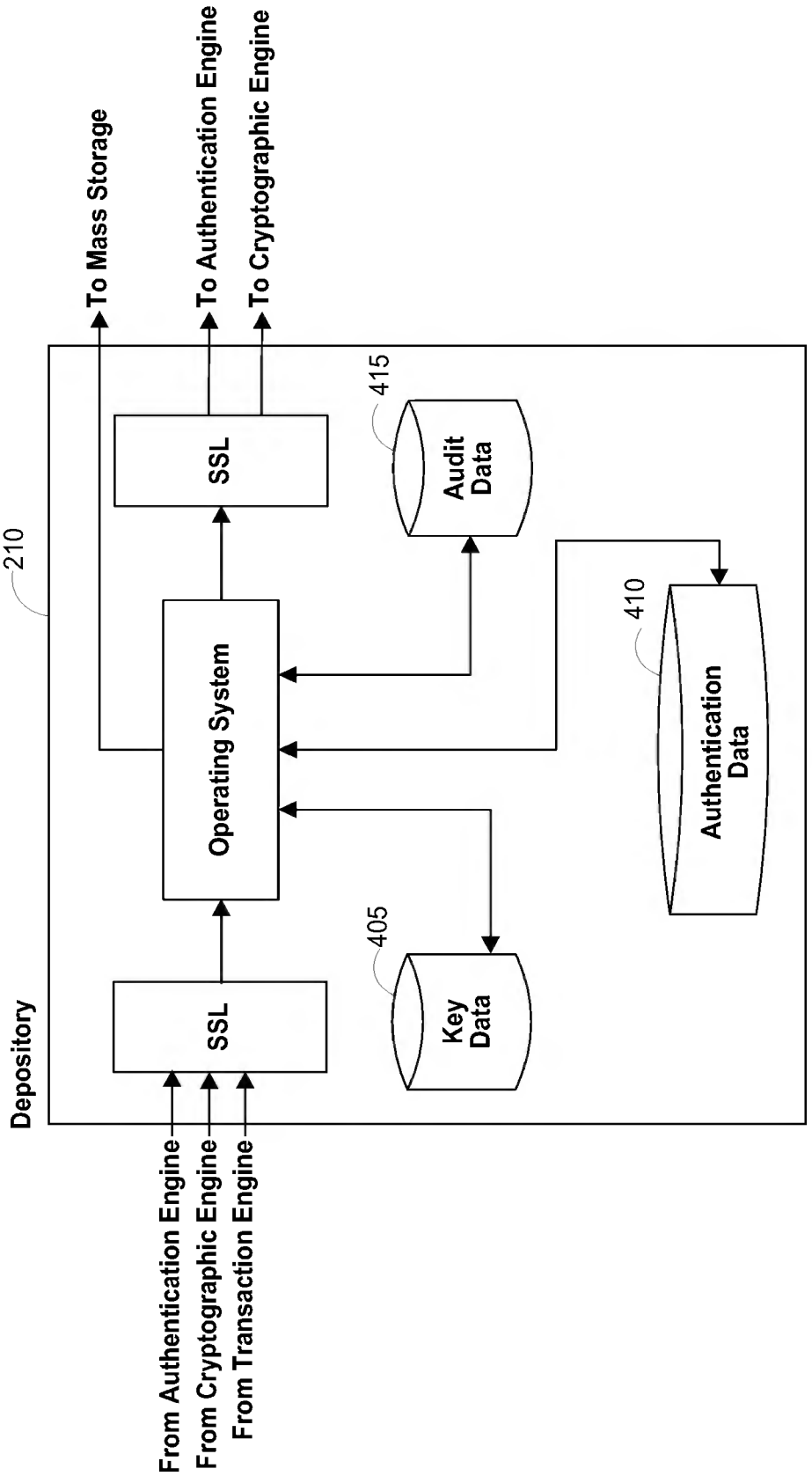

To Mass Storage
To Authentication Engine
To Cryptographic Engine
SSL
415
Audit Data
210
Operating System
410
Authentication Data
405
Key Data
SSL
Depository
From Authentication Engine
From Cryptographic Engine
From Transaction Engine

FIG. 4

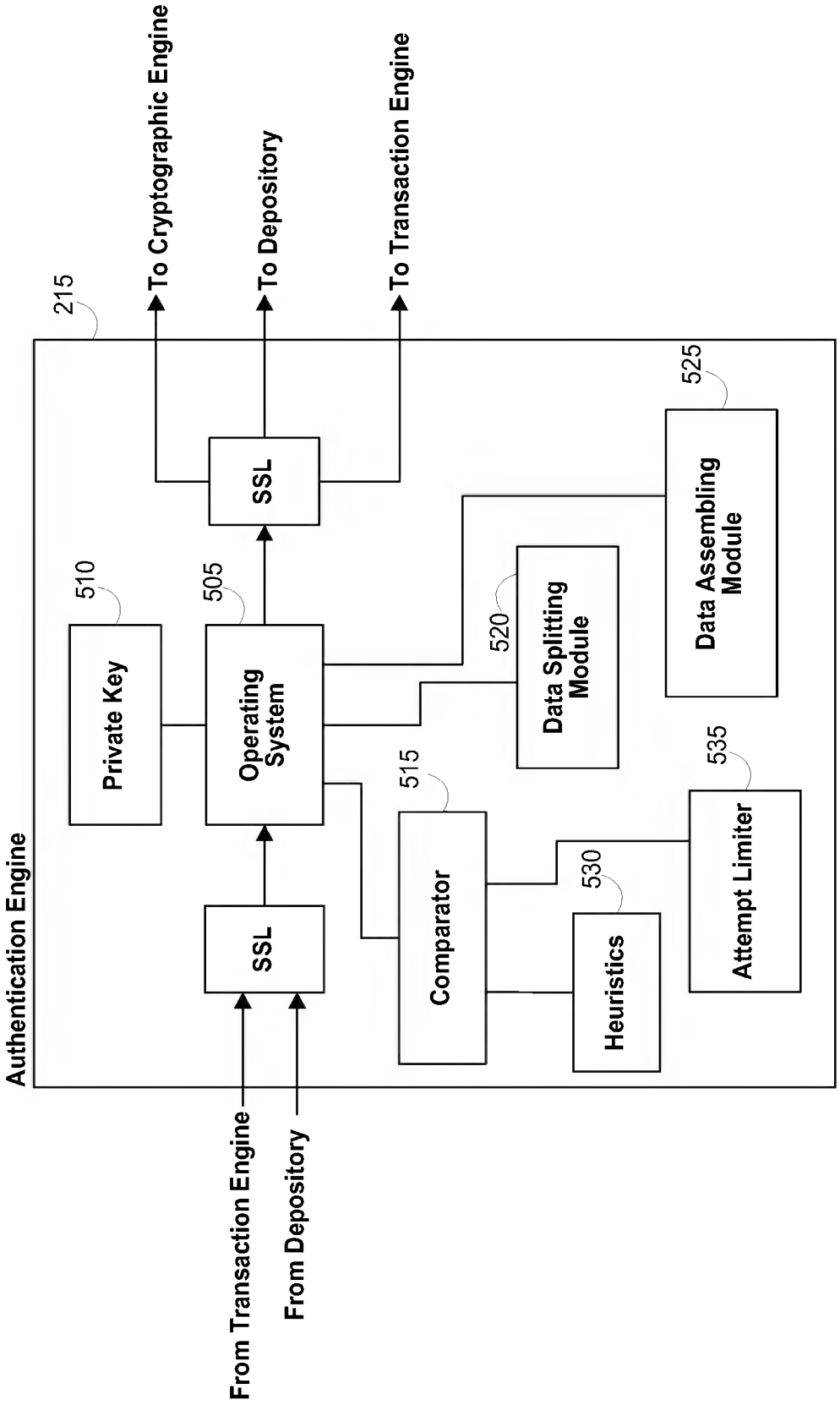

Authentication Engine
215
To Cryptographic Engine
To Depository
To Transaction Engine
From Transaction Engine
From Depository
SSL
SSL
Private Key
510
Operating System
505
Comparator
515
Heuristics
530
Attempt Limiter
535
Data Splitting Module
520
Data Assembling Module
525

FIG. 5

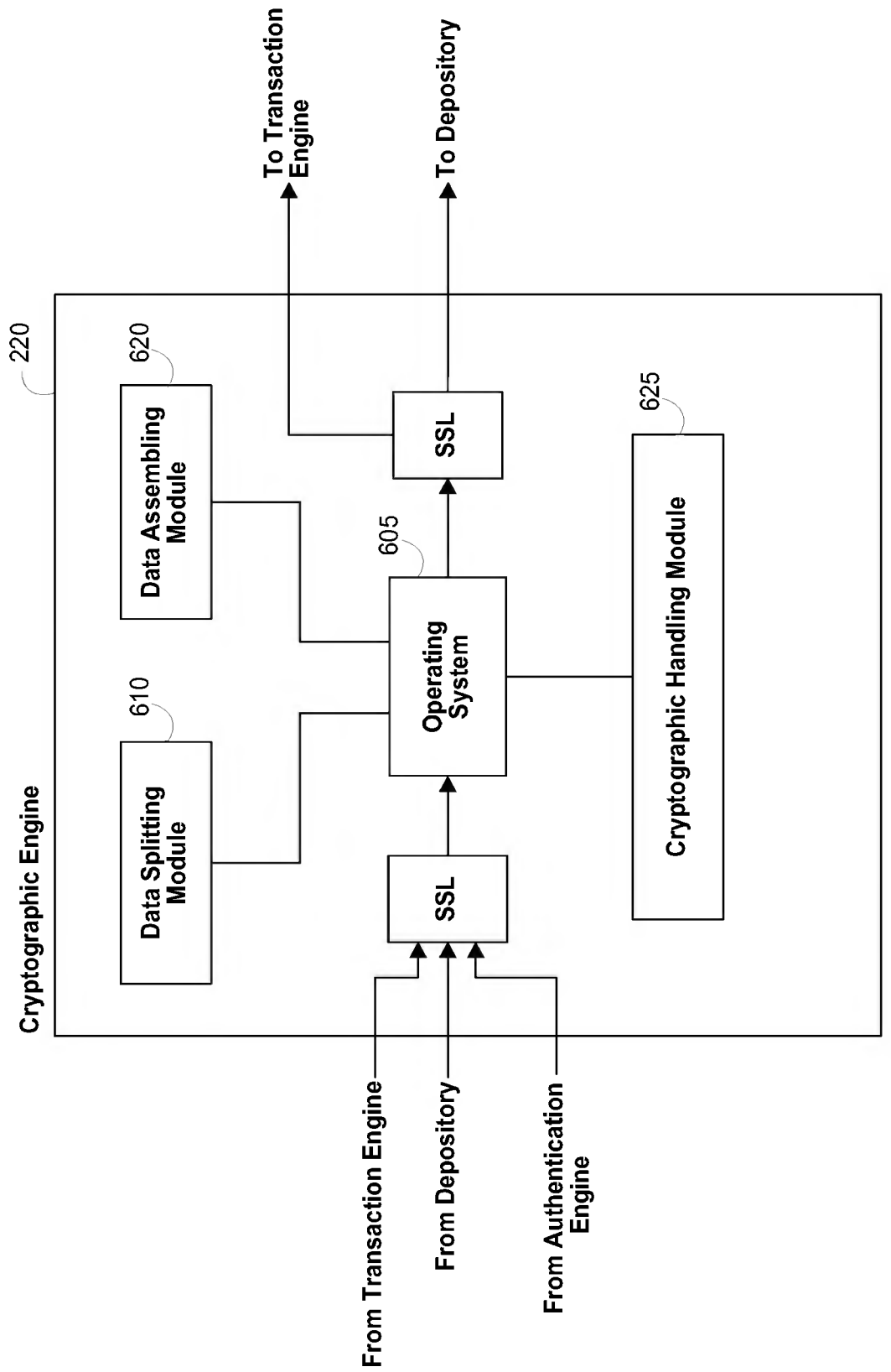

To Transaction Engine
To Depository
220
620
625
605
610
Data Assembling Module
SSL
Operating System
Cryptographic Handling Module
Data Splitting Module
SSL
Cryptographic Engine
From Transaction Engine
From Depository
From Authentication Engine

FIG. 6

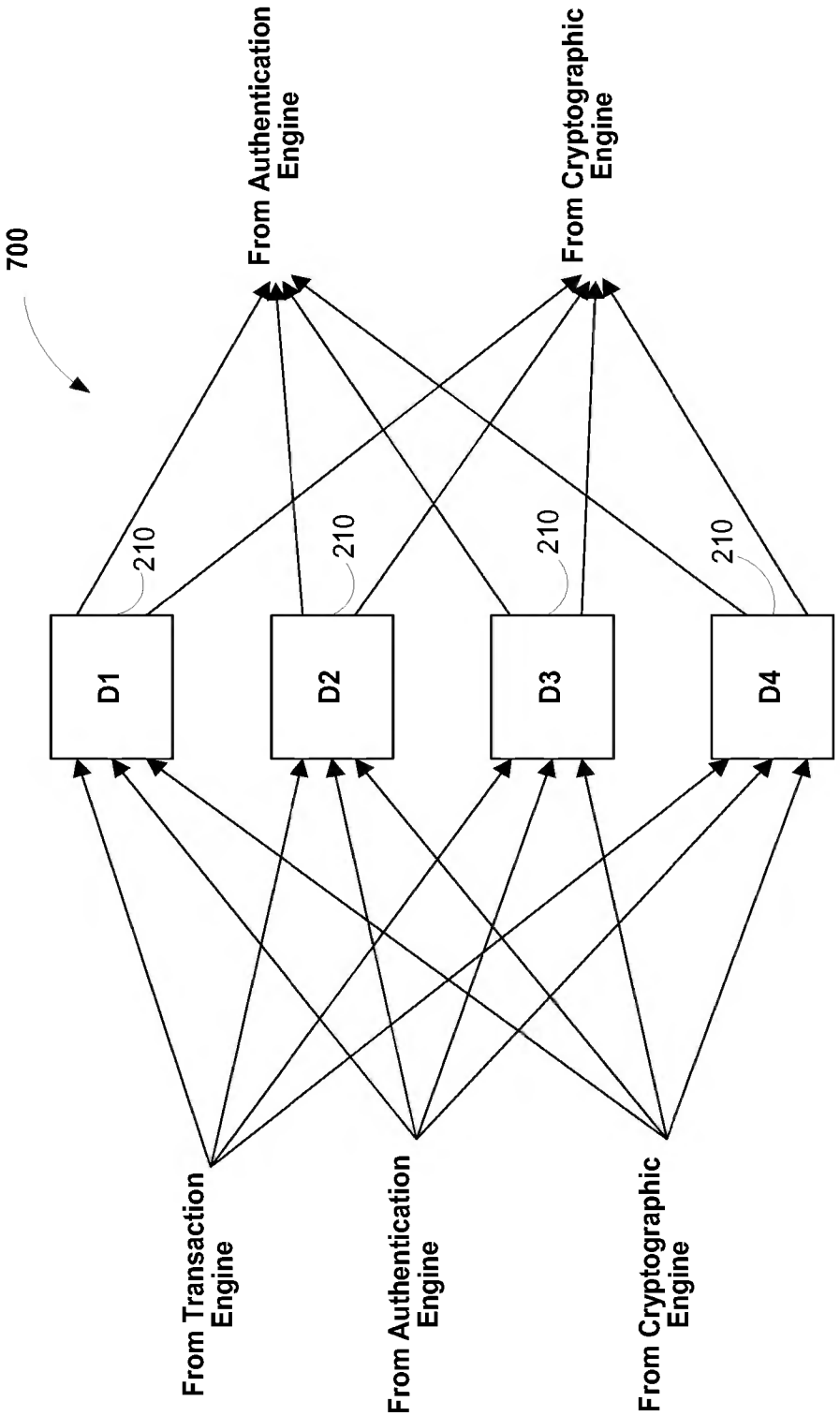

700
From Authentication Engine
From Cryptographic Engine
210
210
210
210
D1
D2
D3
D4
From Transaction Engine
From Authentication Engine
From Cryptographic Engine

FIG. 7

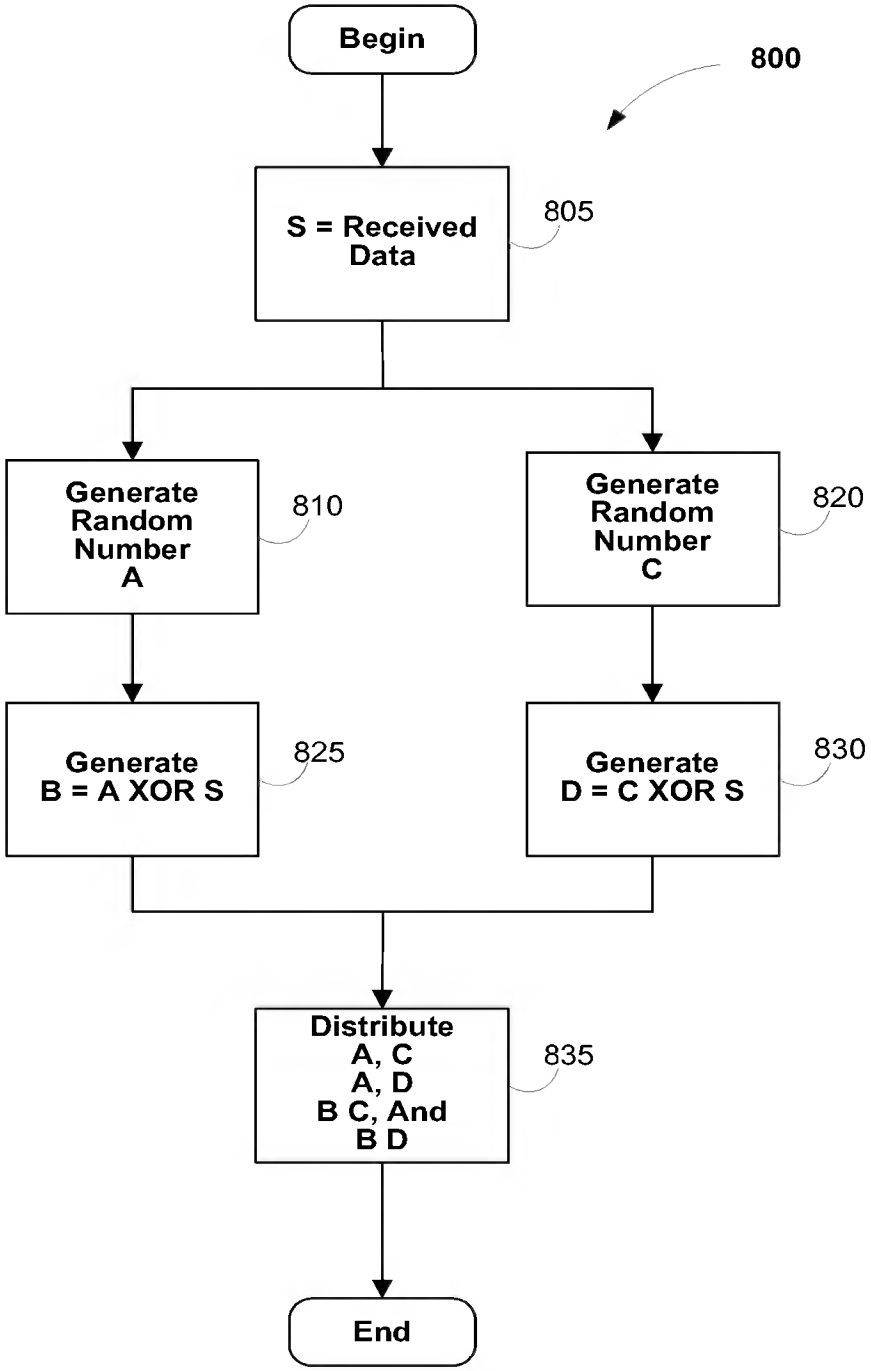
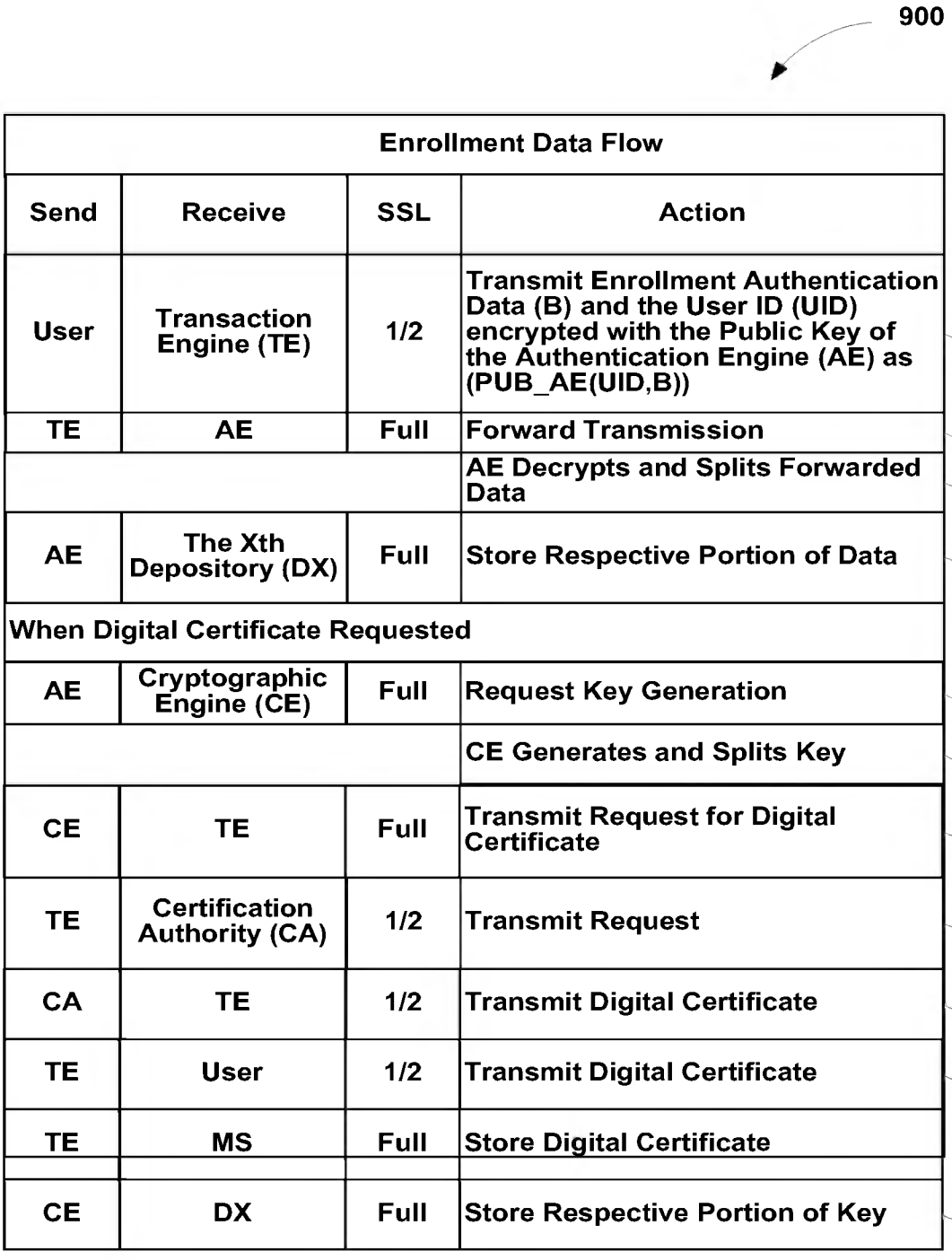

Begin
800
S = Received Data
805
Generate Random Number A
810
Generate Random Number C
820
Generate B = A XOR S
825
Generate D = C XOR S
830
Distribute A, C A, D B C, And B D
835
End

FIG. 8



Enrollment Data Flow			
Send	**Receive**	**SSL**	**Action**
User	Transaction Engine (TE)	1/2	Transmit Enrollment Authentication Data (B) and the User ID (UID) encrypted with the Public Key of the Authentication Engine (AE) as (PUB_AE(UID,B))
TE	AE	Full	Forward Transmission
			AE Decrypts and Splits Forwarded Data
AE	The Xth Depository (DX)	Full	Store Respective Portion of Data
When Digital Certificate Requested			
AE	Cryptographic Engine (CE)	Full	Request Key Generation
			CE Generates and Splits Key
CE	TE	Full	Transmit Request for Digital Certificate
TE	Certification Authority (CA)	1/2	Transmit Request
CA	TE	1/2	Transmit Digital Certificate
TE	User	1/2	Transmit Digital Certificate
TE	MS	Full	Store Digital Certificate
CE	DX	Full	Store Respective Portion of Key

FIG. 9, Panel A

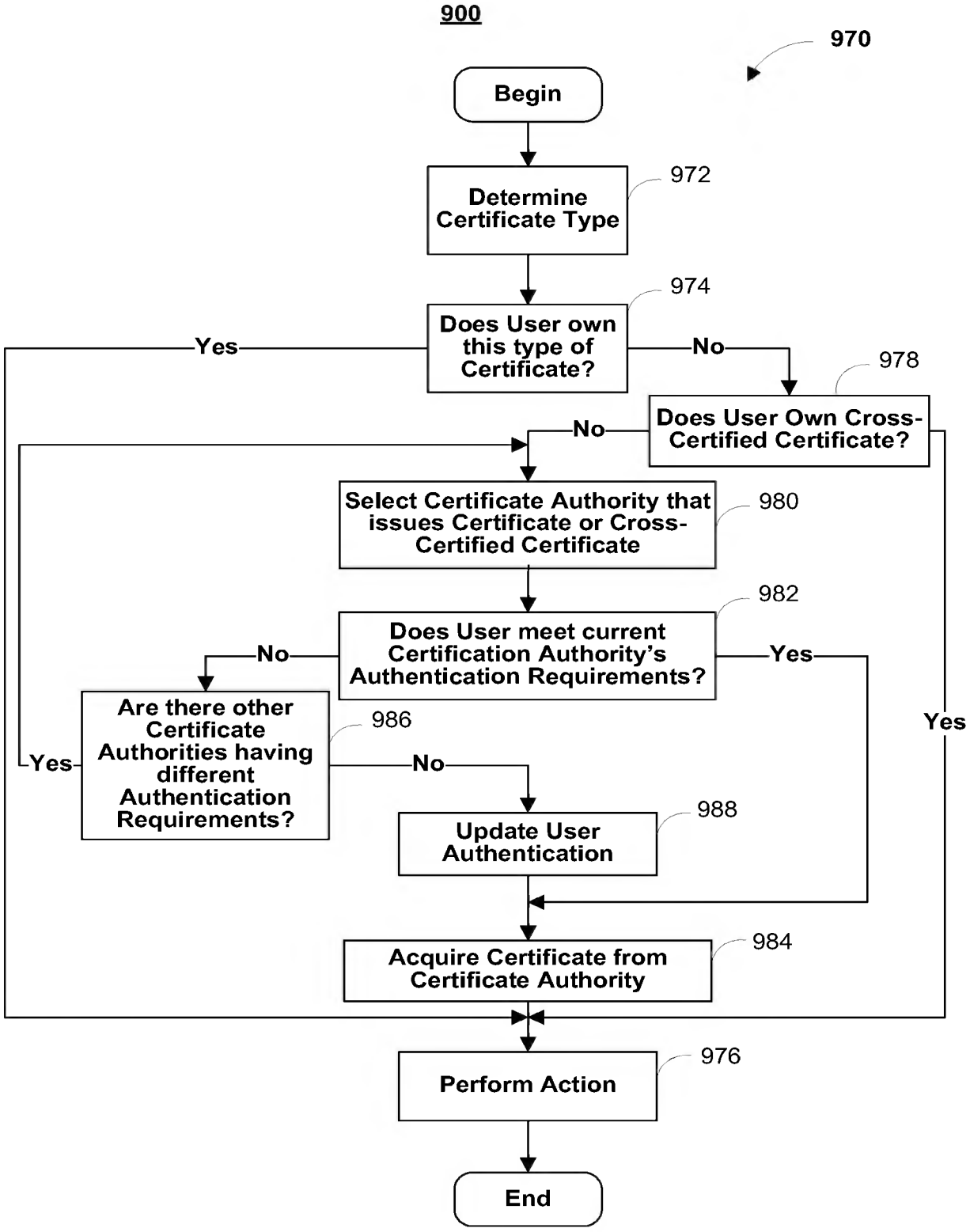

900
970
Begin
Determine Certificate Type
972
Does User own this type of Certificate?
974
Yes
No
Does User Own Cross-Certified Certificate?
978
No
Yes
Select Certificate Authority that issues Certificate or Cross-Certified Certificate
980
Does User meet current Certification Authority's Authentication Requirements?
982
No
Yes
Are there other Certificate Authorities having different Authentication Requirements?
986
Yes
No
Update User Authentication
988
Acquire Certificate from Certificate Authority
984
Perform Action
976
End

FIG. 9, Panel B



	Authentication Data Flow			
	SEND	RECEIVE	SSL	ACTION
1005	User	Vendor	1/2	Transaction occurs, such as selecting purchase
1010	Vendor	User	1/2	Transmit transaction ID (TID) and authentication request (AR)
				Authentication data (B') is gathered from User
1015	User	TE	1/2	Transmit TID and B' wrapped in the Public Key of the Authentication Engine (AE), as (PUB_AE(TID, B'))
1020	TE	AE	Full	Forward transmission
				Enrollment authentication data (B) is requested and gathered
1025	Vendor	Transaction Engine (TE)	Full	Transmits TID, AR
1030	TE	Mass Storage (MS)	Full	Create Record in database
1035	TE	The Xth Depository (DX)	Full	UID, TID
1040	DX	AE	Full	Transmit the TID and the portion of the authentication data stored at enrollment (BX) as (PUB_AE(TID, BX))
1045				AE assembles B and compares to B'
1050	AE	TE	Full	TID, the filled in AR
1055	TE	Vendor	Full	TID, Yes/No
	TE	User	1/2	TID, confirmation message

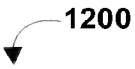
FIG. 10

1100

Signing Data Flow			
SEND	RECEIVE	SSL	ACTION
User	Vendor	1/2	Transaction occurs, such as agreeing on a deal
Vendor	User	1/2	Transmit transaction identification number (TID), authentication request (AR), and agreement or message (M)
			Current authentication data (B') and a hash of the message received by the User (h(M')) is is gathered from User
User	TE	1/2	Transmit TID, B', AR, and h(M') wrapped in the Public Key of the Authentication Engine (AE), as (PUB_AE(TID, B', h(M'))
TE	AE	Full	Forward transmission
			Gather enrollment authentication data
Vendor	Transaction Engine (TE)	Full	Transmits UID, TID, AR, and a hash of the message (h(M')).
TE	Mass Storage (MS)	Full	Create Record in database
TE	The Xth Depository (DX)	Full	UID, TID
DX	AE	Full	Transmit the TID and the portion of the authentication data stored at Enrollment (BX), as (PUB_AE(TID, BX))
			The original vendor message is transmitted to the AE
TE	AE	Full	Transmit h(M)
			AE assembles B, compares to B' and compares h(M) to h(M')
AE	Cryptographic Engine (CE)	Full	Request for digital signature and a message to be signed, for example, the hashed message
AE	DX	Full	TID, signing UID
DX	CE	Full	Transmit the portion of the Crypto-graphic Key corresponding to the signing party
			CE assembles key and signs
CE	AE	Full	Transmit the digital signature (S) of signing party
AE	TE	Full	TID, the filled in AR, h(M), and S
TE	Vendor	Full	TID, a receipt=(TID, Yes/No, and S), and the digital signature of the trust engine, for example, a hash of the receipt encrypted with the trust engine's Private Key (Priv_TE(h(receipt)))
TE	User	1/2	TID, confirmation message

1103 1105 1110 1115 1120 1125 1130 1135 1140

FIG. 11



Encryption/Decryption Data Flow				
Send	Receive	SSL	Action	
Decryption				
			Perform Authentication Data Process 1000, include the Session Key (sync) in the AR, where the sync has been encrypted with the Public Key of the User as PUB_USER(SYNC)	
			Authenticate the User	
AE	CE	Full	Forward PUB_USER(SYNC) to CE	1205
AE	DX	Full	UID, TID	1210
DX	CE	Full	Transmit the TID and the portion of the Private Key as (PUB_AE(TID, KEY_USER))	1215
			CE assembles the Cryptographic Key and decrypts the sync	1220
CE	AE	Full	TID, the filled in AR including decrypted sync	1225
AE	TE	Full	Forward to TE	1230
TE	Requesting APP/Vendor	1/2	TID, Yes/No, Sync	
Encryption				
Requesting APP/ Vendor	TE	1/2	Request for Public Key of User	1235
TE	MS	Full	Request Digital Certificate	1240
MS	TE	Full	Transmit Digital Certificate	1245
TE	Requesting APP/Vendor	1/2	Transmit Digital Certificate	1250

FIG. 12

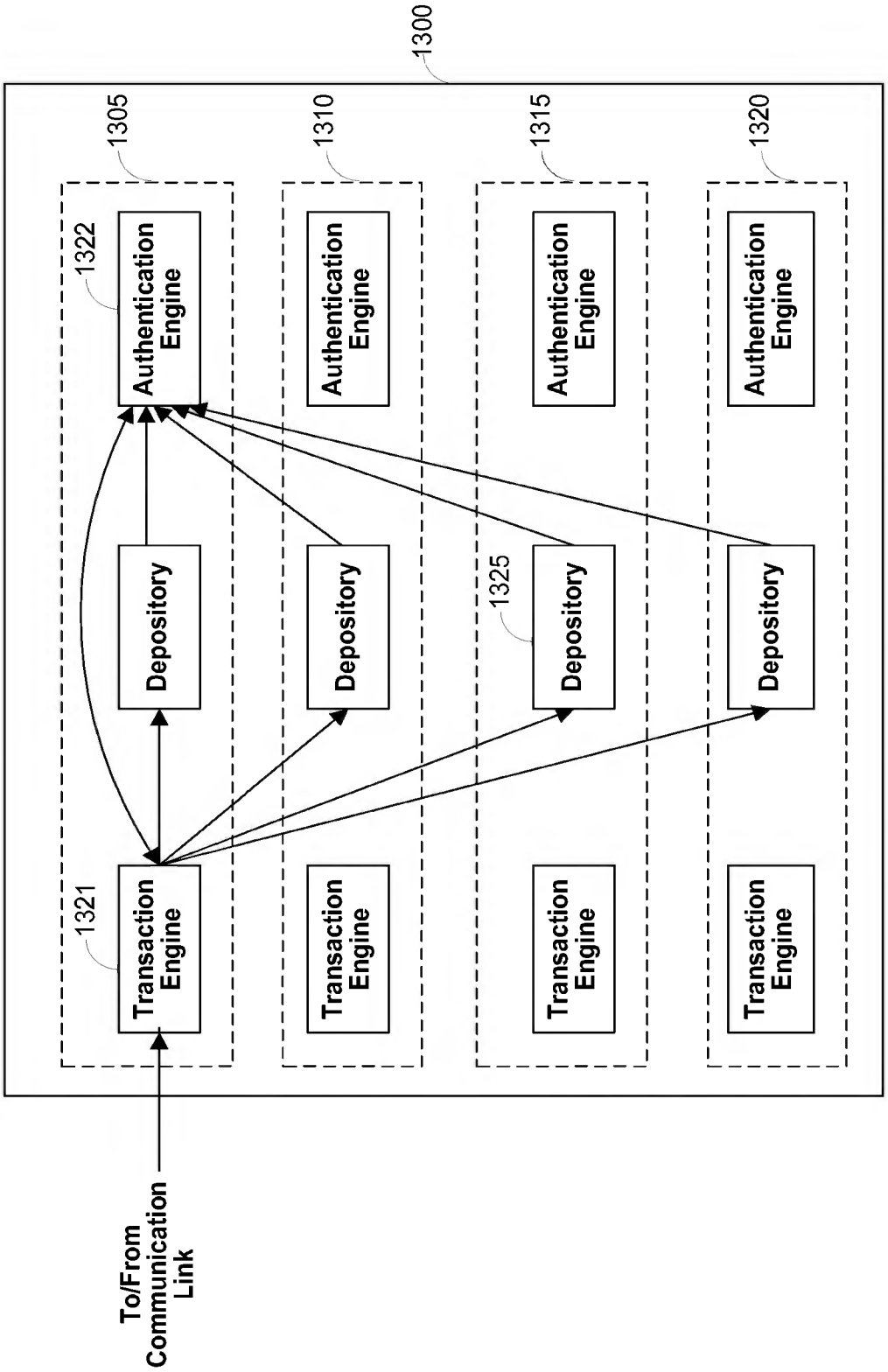

1300
1305
1310
1315
1320
1322
Authentication Engine
Authentication Engine
Authentication Engine
Authentication Engine
Depository
Depository
1325
Depository
Depository
1321
Transaction Engine
Transaction Engine
Transaction Engine
Transaction Engine
To/From Communication Link

FIG. 13

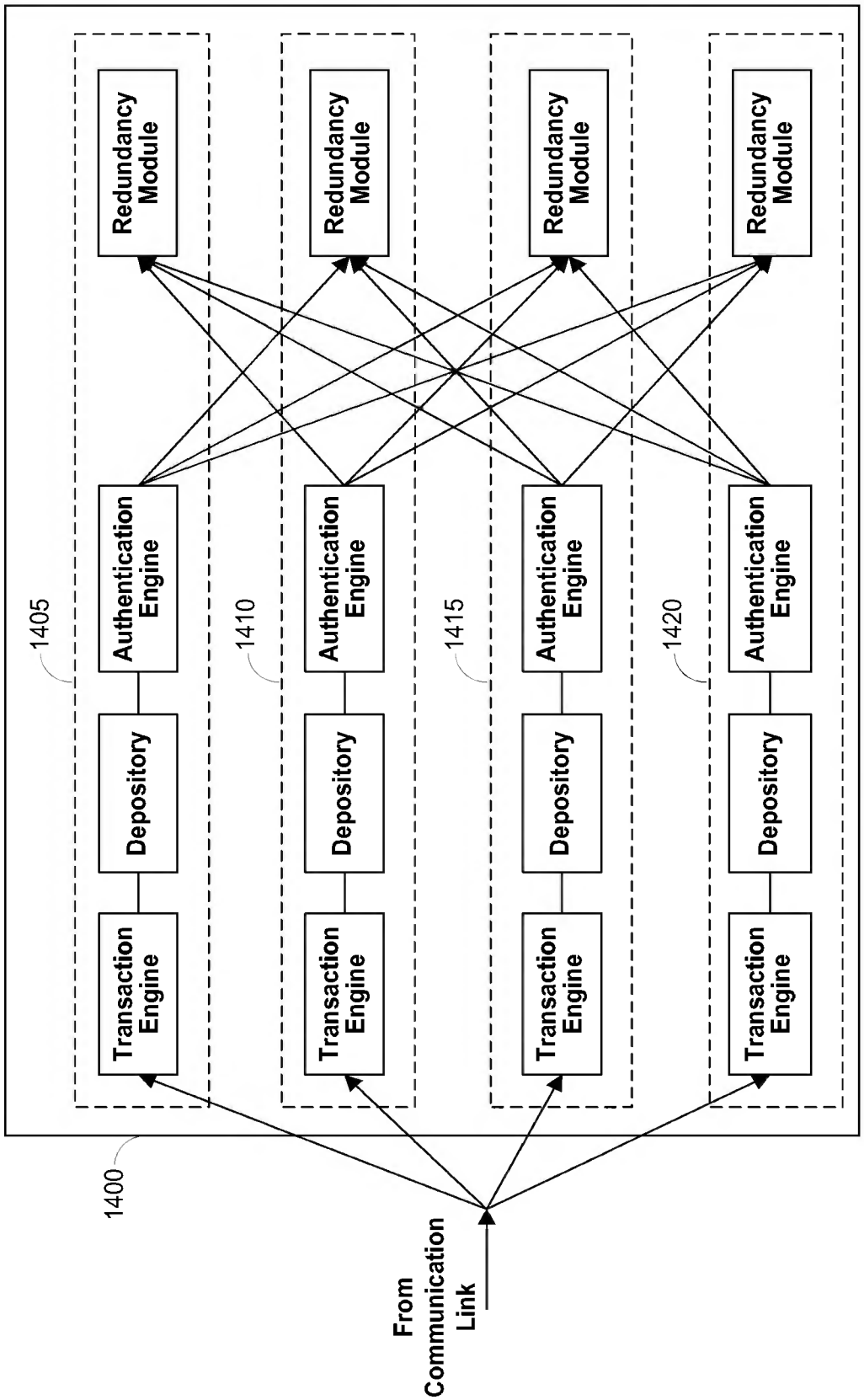

Redundancy Module
Redundancy Module
Redundancy Module
Redundancy Module
Authentication Engine
Authentication Engine
Authentication Engine
Authentication Engine
Depository
Depository
Depository
Depository
Transaction Engine
Transaction Engine
Transaction Engine
Transaction Engine
1405
1410
1415
1420
1400
From Communication Link

FIG. 14

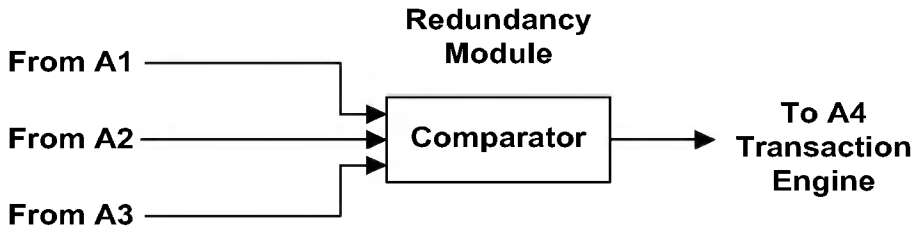

Redundancy
Module
From A1
From A2
From A3
Comparator
To A4
Transaction
Engine

FIG. 15

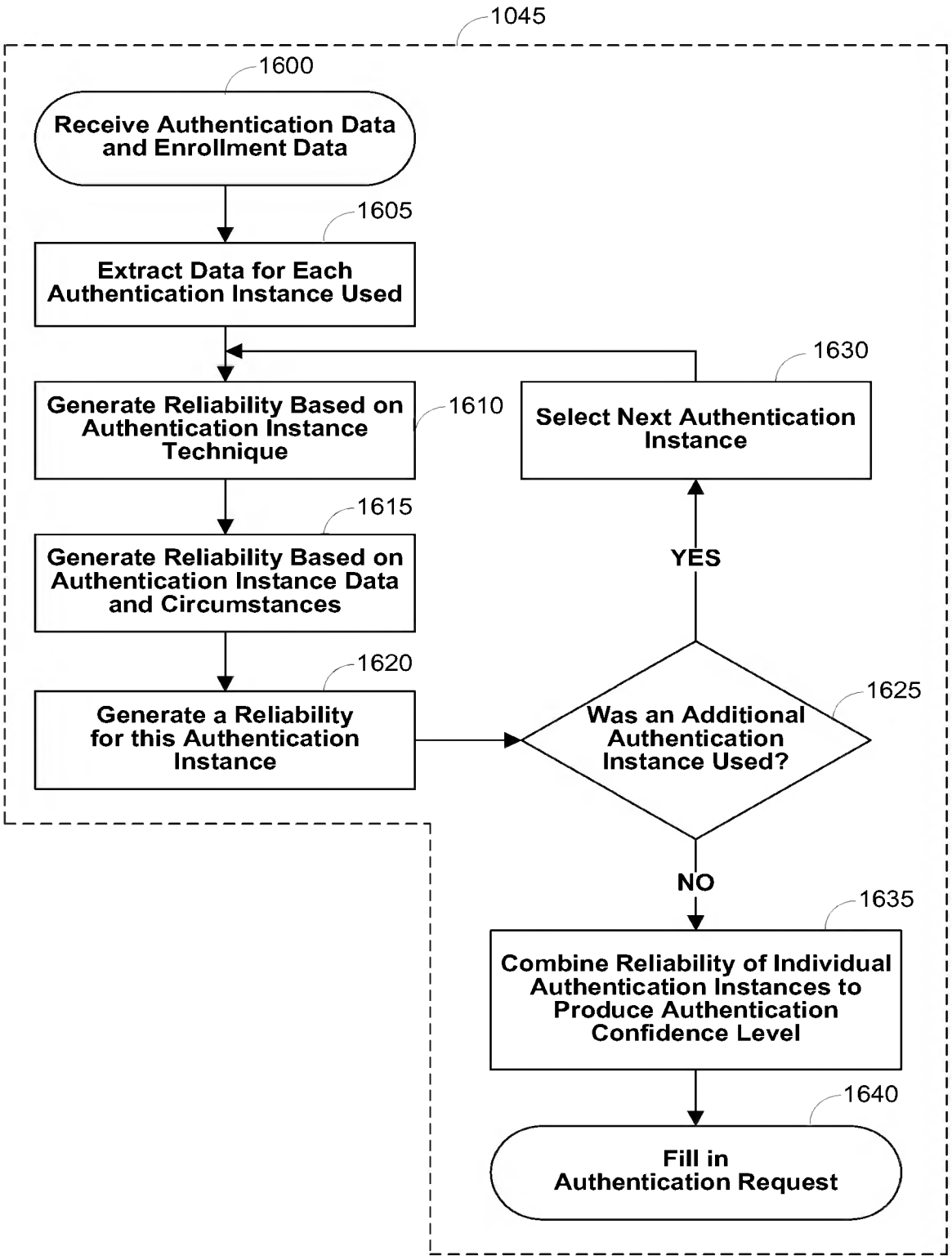

1045
1600
Receive Authentication Data and Enrollment Data
1605
Extract Data for Each Authentication Instance Used
1630
Generate Reliability Based on Authentication Instance Technique
1610
Select Next Authentication Instance
1615
Generate Reliability Based on Authentication Instance Data and Circumstances
YES
1620
Generate a Reliability for this Authentication Instance
1625
Was an Additional Authentication Instance Used?
NO
1635
Combine Reliability of Individual Authentication Instances to Produce Authentication Confidence Level
1640
Fill in Authentication Request

FIG. 16

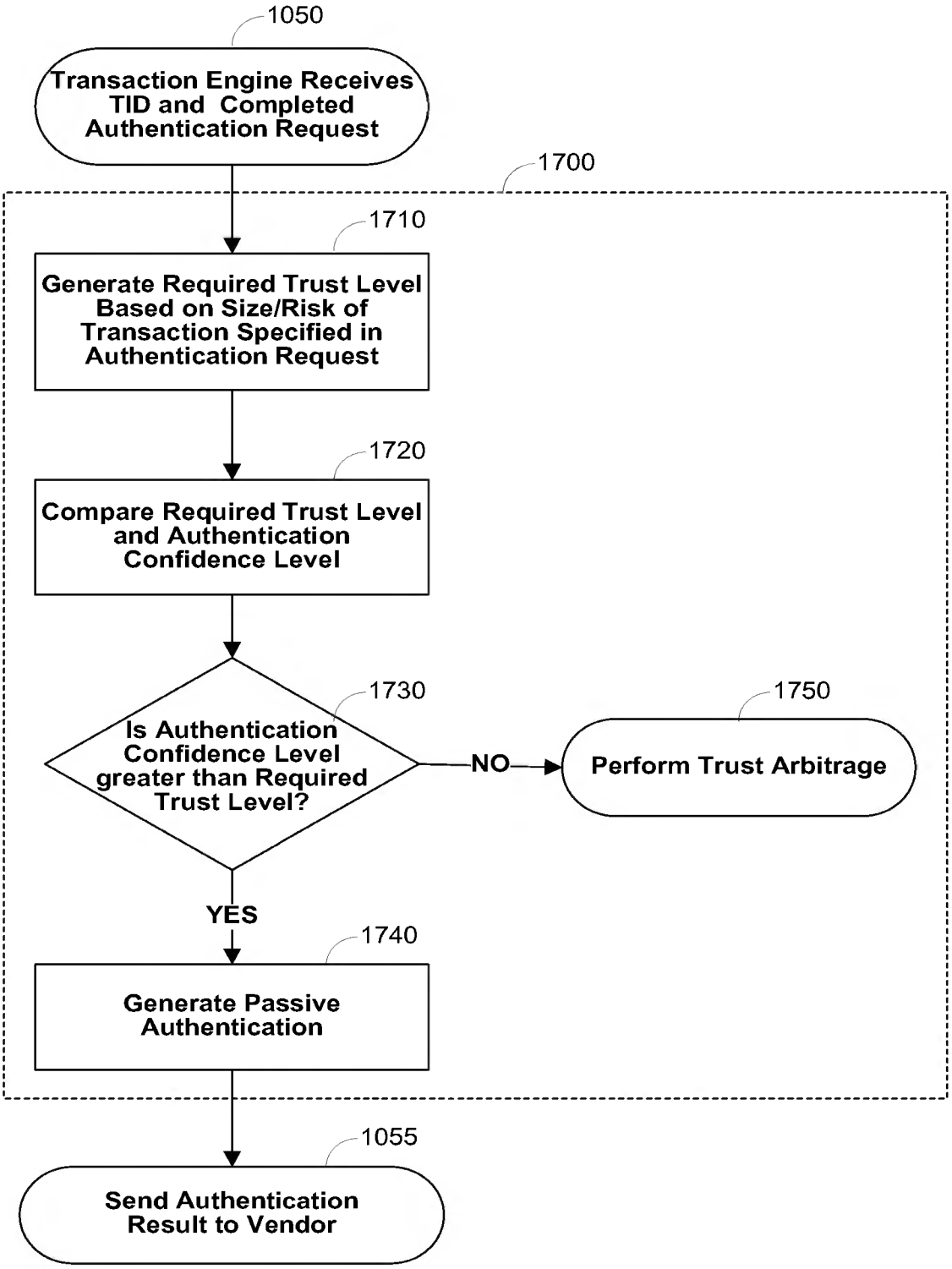

1050
Transaction Engine Receives TID and Completed Authentication Request
1700
1710
Generate Required Trust Level Based on Size/Risk of Transaction Specified in Authentication Request
1720
Compare Required Trust Level and Authentication Confidence Level
1730
Is Authentication Confidence Level greater than Required Trust Level?
NO
1750
Perform Trust Arbitrage
YES
1740
Generate Passive Authentication
1055
Send Authentication Result to Vendor

FIG. 17

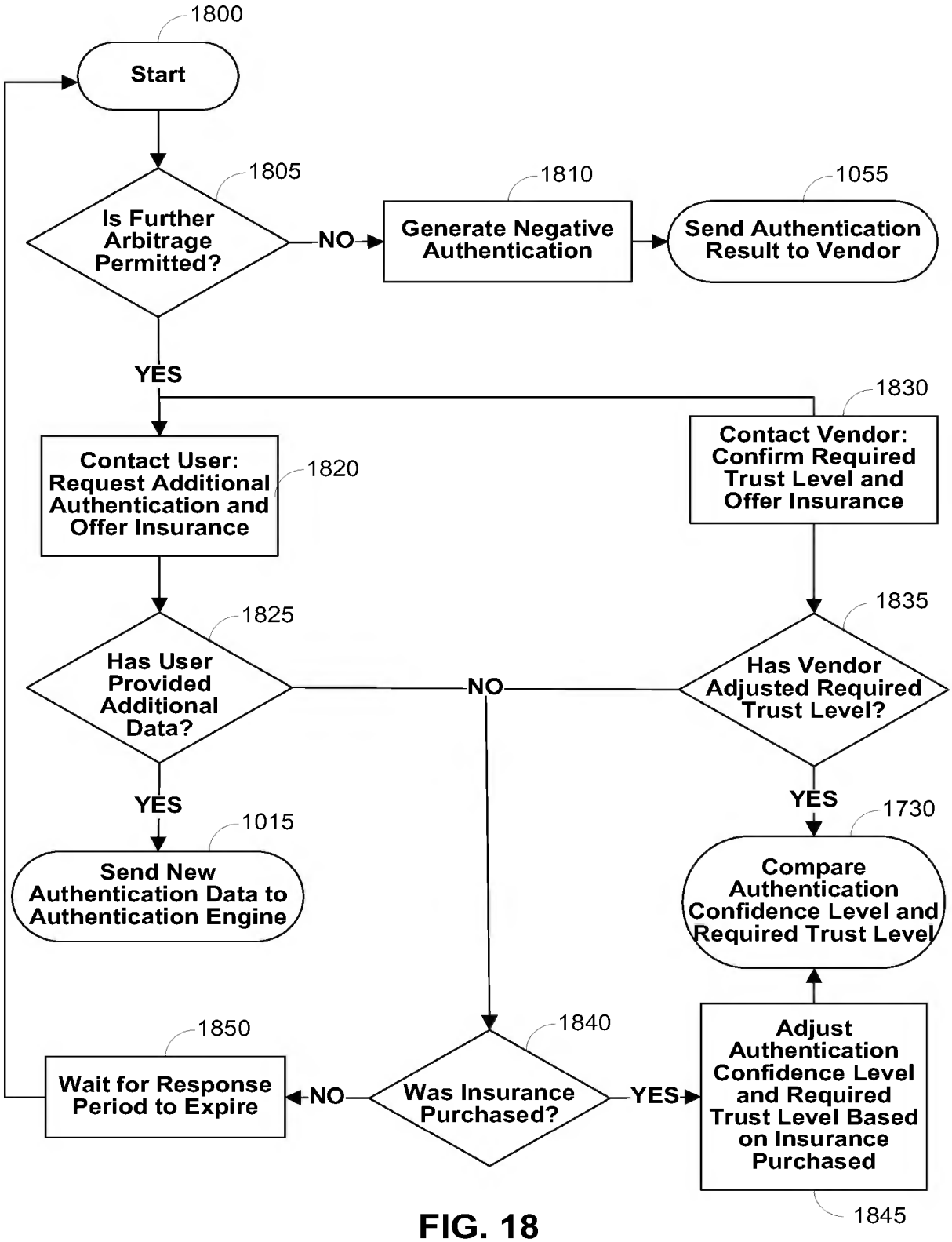

1800
Start
1805
Is Further Arbitrage Permitted?
NO
1810
Generate Negative Authentication
1055
Send Authentication Result to Vendor
YES
1830
Contact Vendor: Confirm Required Trust Level and Offer Insurance
Contact User: Request Additional Authentication and Offer Insurance
1820
1825
Has User Provided Additional Data?
NO
1835
Has Vendor Adjusted Required Trust Level?
YES
1015
Send New Authentication Data to Authentication Engine
YES
1730
Compare Authentication Confidence Level and Required Trust Level
1850
Wait for Response Period to Expire
NO
1840
Was Insurance Purchased?
YES
Adjust Authentication Confidence Level and Required Trust Level Based on Insurance Purchased
1845

FIG. 18

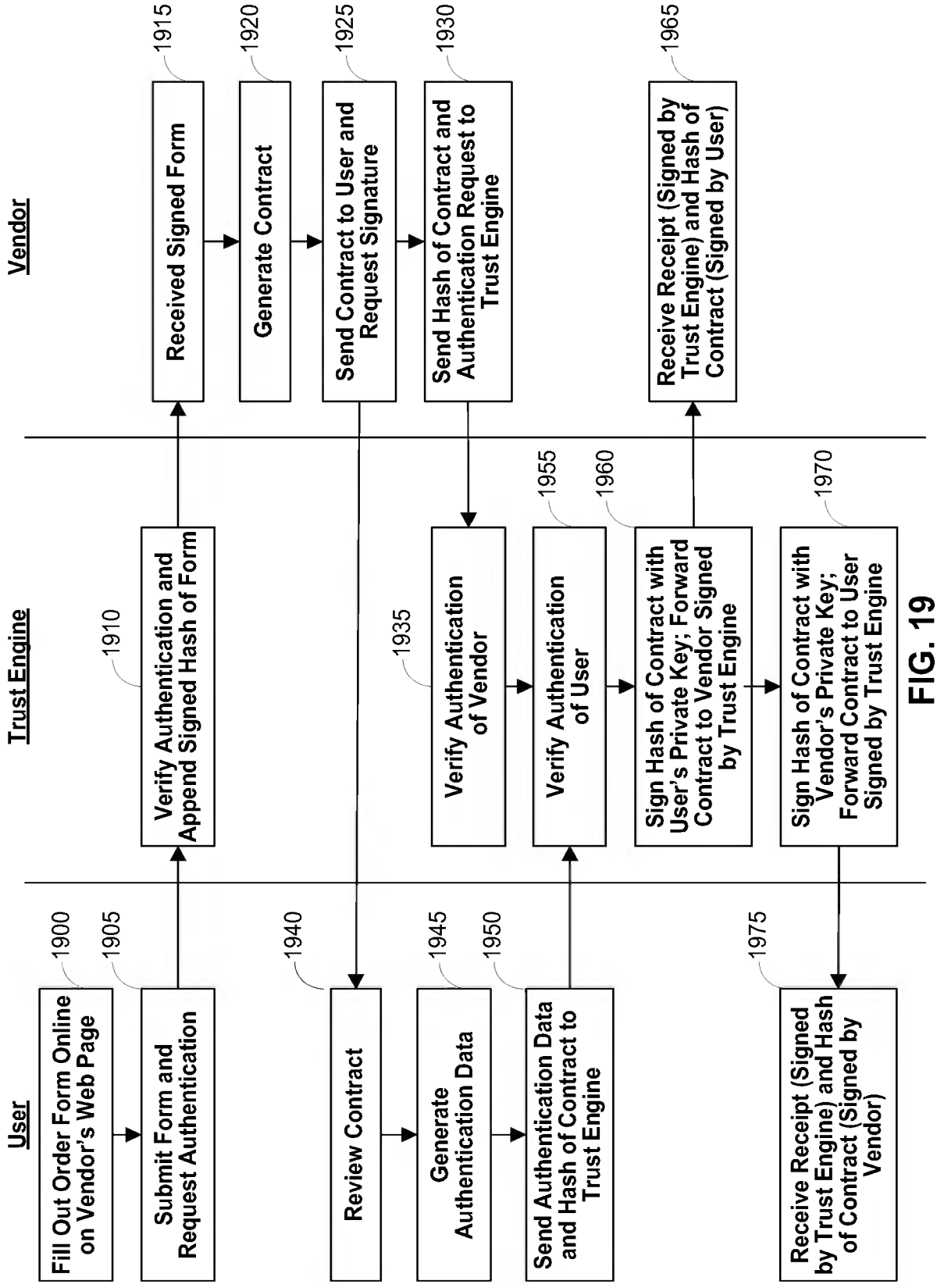

User
Trust Engine
Vendor
Fill Out Order Form Online on Vendor's Web Page
1900
Submit Form and Request Authentication
1905
Verify Authentication and Append Signed Hash of Form
1910
Received Signed Form
1915
Generate Contract
1920
Send Contract to User and Request Signature
1925
Send Hash of Contract and Authentication Request to Trust Engine
1930
Review Contract
1940
Generate Authentication Data
1945
Send Authentication Data and Hash of Contract to Trust Engine
1950
Verify Authentication of Vendor
1935
Verify Authentication of User
1955
Sign Hash of Contract with User's Private Key; Forward Contract to Vendor Signed by Trust Engine
1960
Receive Receipt (Signed by Trust Engine) and Hash of Contract (Signed by User)
1965
Sign Hash of Contract with Vendor's Private Key; Forward Contract to User Signed by Trust Engine
1970
Receive Receipt (Signed by Trust Engine) and Hash of Contract (Signed by Vendor)
1975

FIG. 19

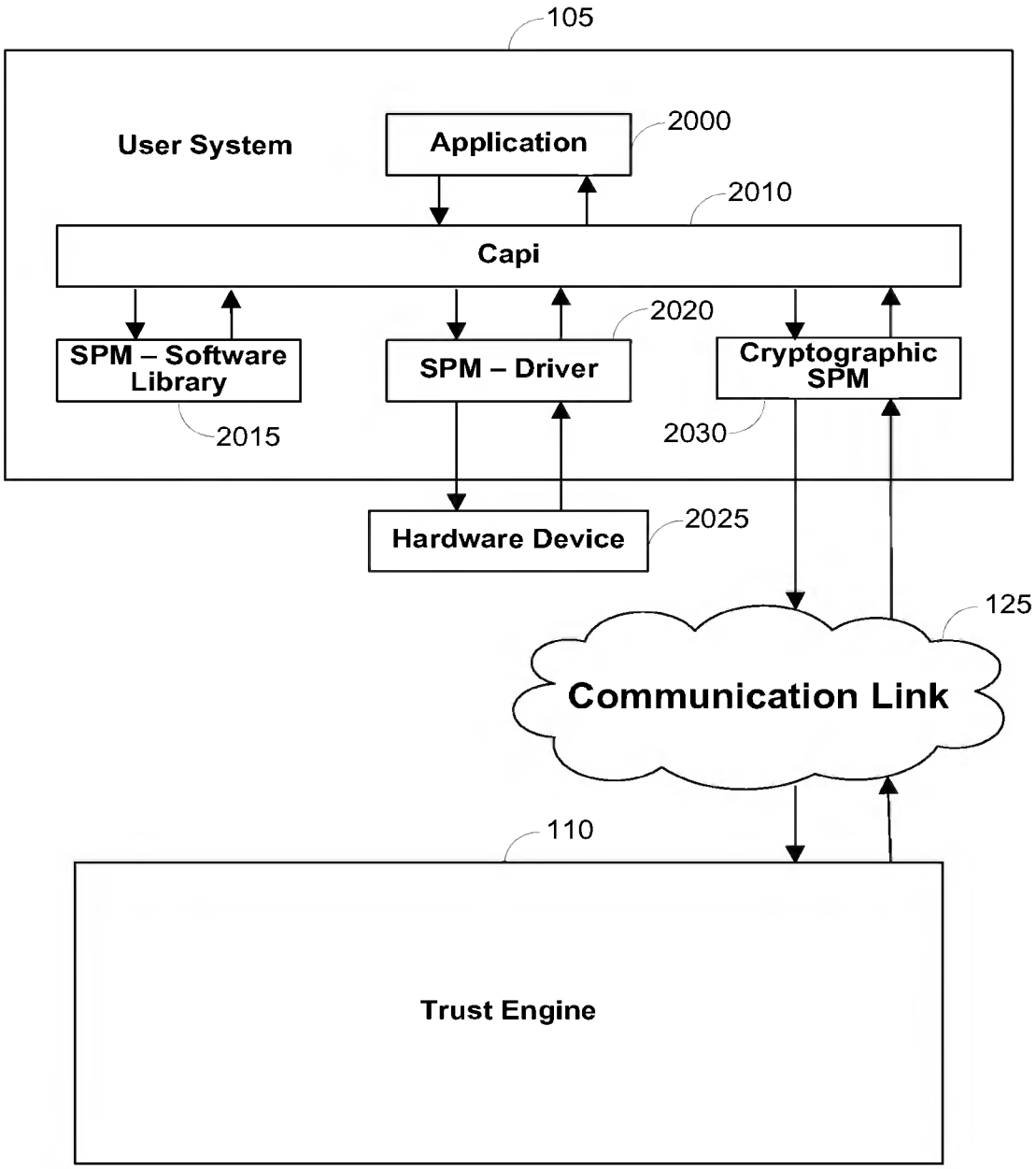

105
User System
Application
2000
2010
Capi
2020
SPM – Software Library
SPM – Driver
Cryptographic SPM
2015
2030
Hardware Device
2025
125
Communication Link
110
Trust Engine

FIG. 20

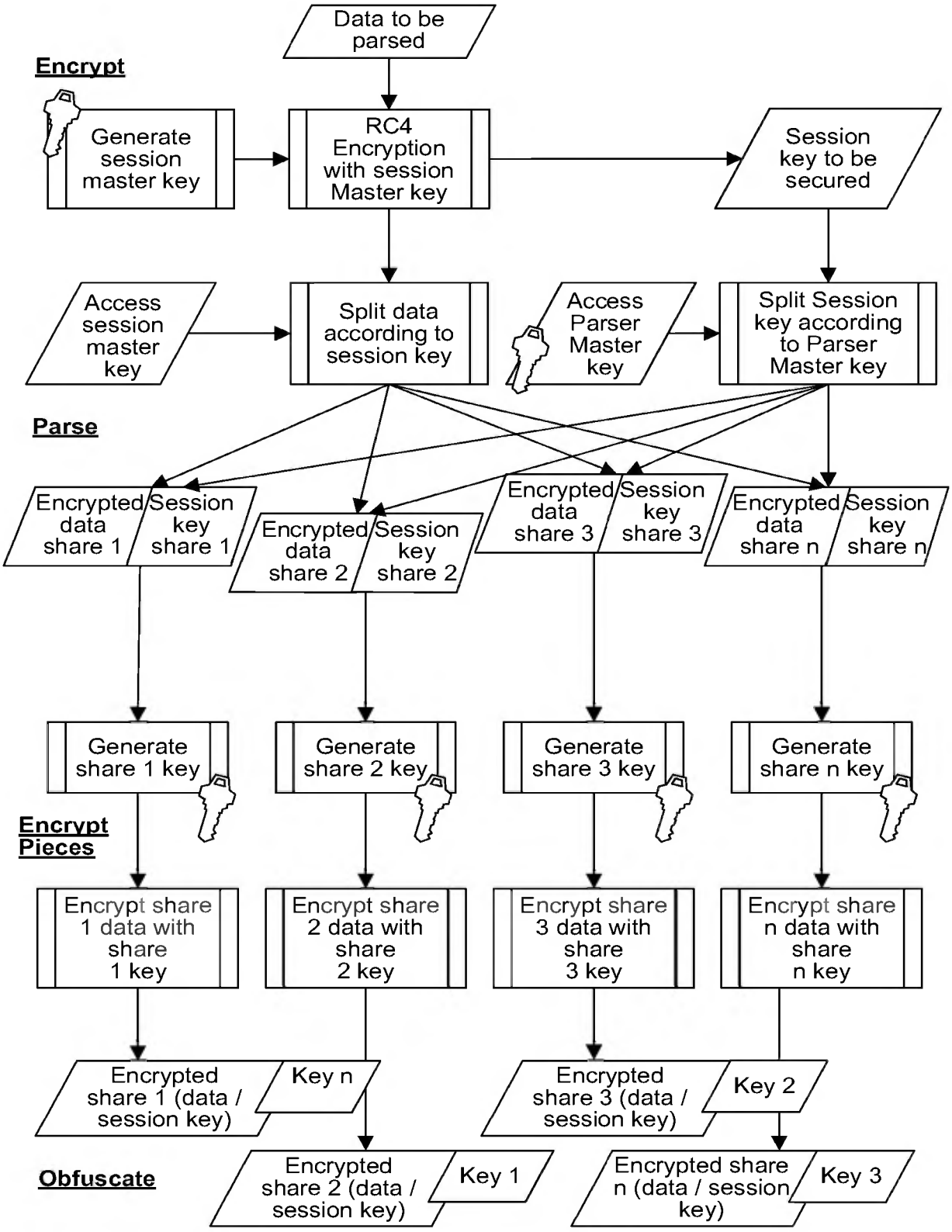

Data to be parsed
Encrypt
Generate session master key
RC4 Encryption with session Master key
Session key to be secured
Access session master key
Split data according to session key
Access Parser Master key
Split Session key according to Parser Master key
Parse
Encrypted data share 1
Session key share 1
Encrypted data share 2
Session key share 2
Encrypted data share 3
Session key share 3
Encrypted data share n
Session key share n
Generate share 1 key
Generate share 2 key
Generate share 3 key
Generate share n key
Encrypt Pieces
Encrypt share 1 data with share 1 key
Encrypt share 2 data with share 2 key
Encrypt share 3 data with share 3 key
Encrypt share n data with share n key
Encrypted share 1 (data / session key)
Key n
Encrypted share 3 (data / session key)
Key 2
Obfuscate
Encrypted share 2 (data / session key)
Key 1
Encrypted share n (data / session key)
Key 3

FIG. 21

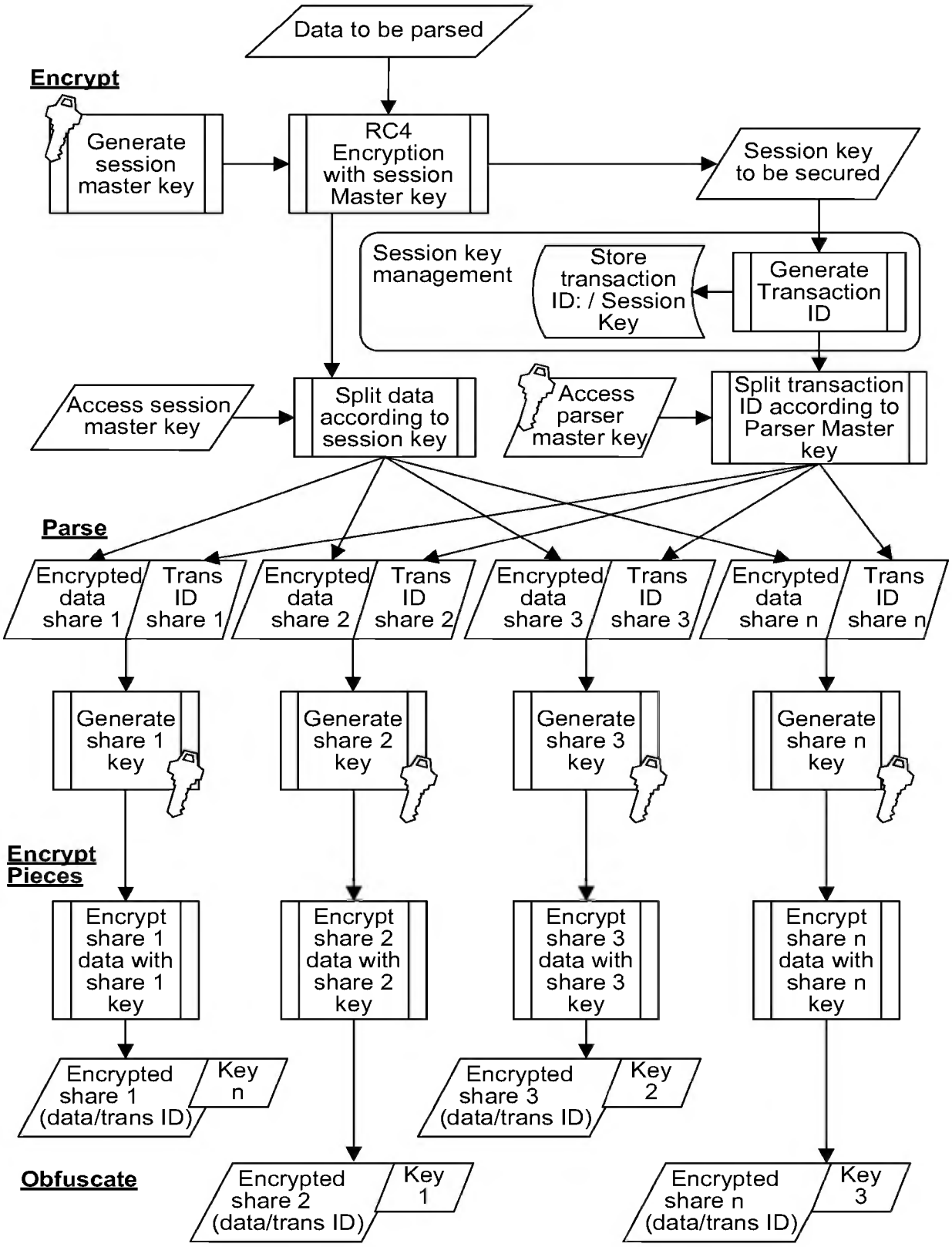

Data to be parsed
Encrypt
Generate session master key
RC4 Encryption with session Master key
Session key to be secured
Session key management
Store transaction ID: / Session Key
Generate Transaction ID
Access session master key
Split data according to session key
Access parser master key
Split transaction ID according to Parser Master key
Parse
Encrypted data share 1
Trans ID share 1
Encrypted data share 2
Trans ID share 2
Encrypted data share 3
Trans ID share 3
Encrypted data share n
Trans ID share n
Generate share 1 key
Generate share 2 key
Generate share 3 key
Generate share n key
Encrypt Pieces
Encrypt share 1 data with share 1 key
Encrypt share 2 data with share 2 key
Encrypt share 3 data with share 3 key
Encrypt share n data with share n key
Encrypted share 1 (data/trans ID)
Key n
Encrypted share 3 (data/trans ID)
Key 2
Obfuscate
Encrypted share 2 (data/trans ID)
Key 1
Encrypted share n (data/trans ID)
Key 3

FIG. 22

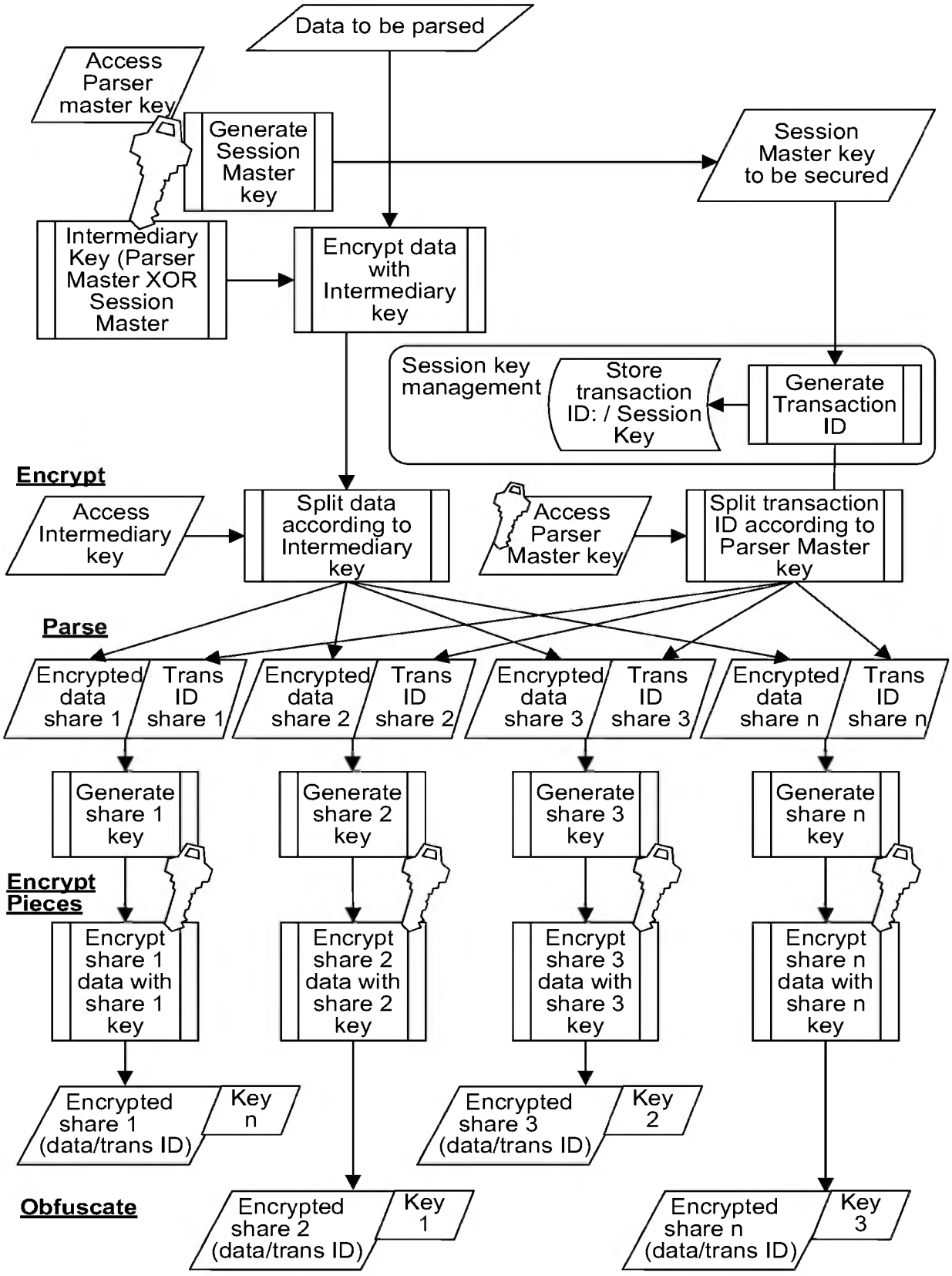

Data to be parsed
Access Parser master key
Generate Session Master key
Session Master key to be secured
Intermediary Key (Parser Master XOR Session Master
Encrypt data with Intermediary key
Session key management
Store transaction ID: / Session Key
Generate Transaction ID
Encrypt
Access Intermediary key
Split data according to Intermediary key
Access Parser Master key
Split transaction ID according to Parser Master key
Parse
Encrypted data share 1
Trans ID share 1
Encrypted data share 2
Trans ID share 2
Encrypted data share 3
Trans ID share 3
Encrypted data share n
Trans ID share n
Generate share 1 key
Generate share 2 key
Generate share 3 key
Generate share n key
Encrypt Pieces
Encrypt share 1 data with share 1 key
Encrypt share 2 data with share 2 key
Encrypt share 3 data with share 3 key
Encrypt share n data with share n key
Encrypted share 1 (data/trans ID)
Key n
Encrypted share 3 (data/trans ID)
Key 2
Obfuscate
Encrypted share 2 (data/trans ID)
Key 1
Encrypted share n (data/trans ID)
Key 3

FIG. 23

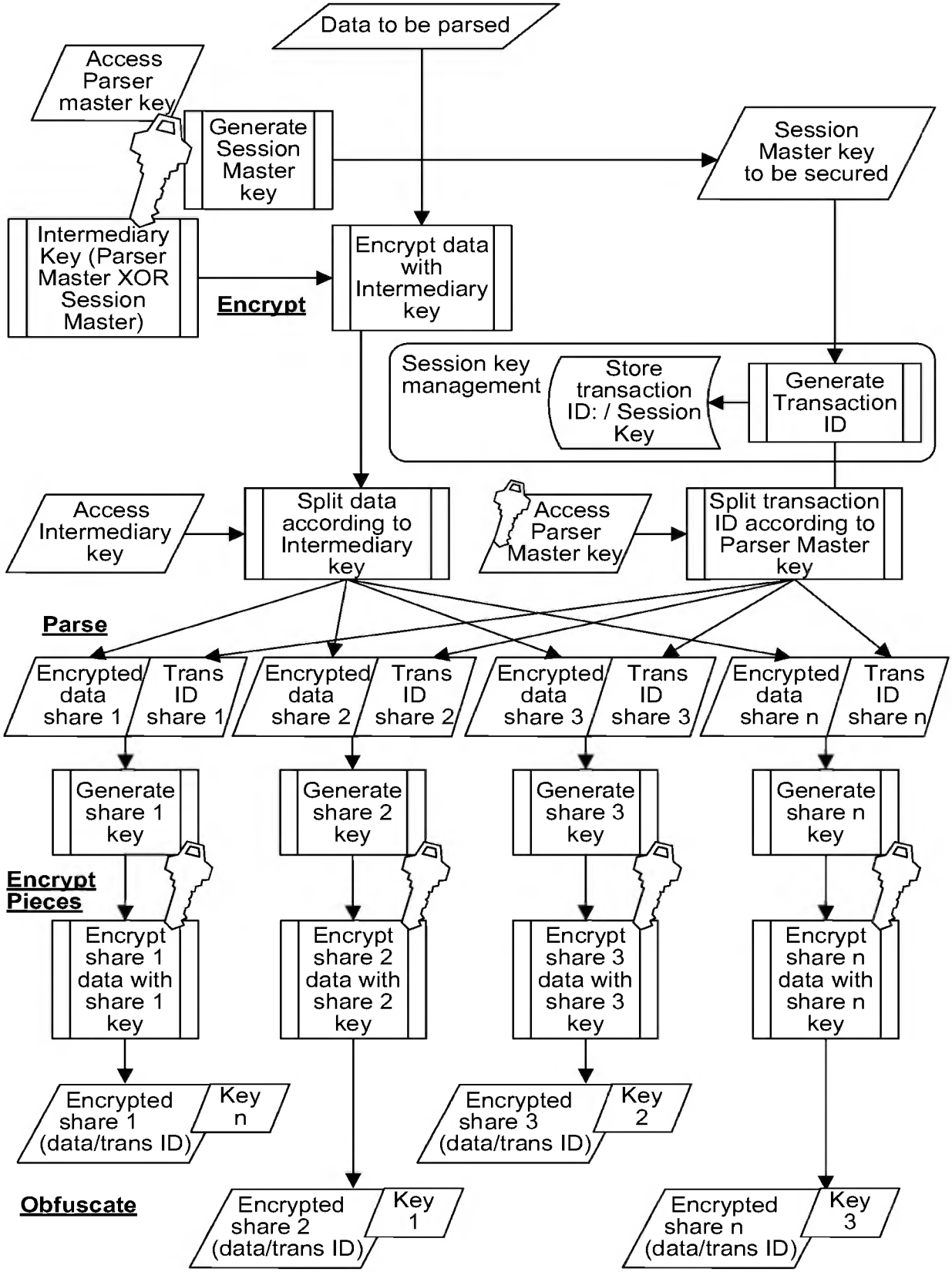

Data to be parsed
Access Parser master key
Generate Session Master key
Session Master key to be secured
Intermediary Key (Parser Master XOR Session Master)
Encrypt
Encrypt data with Intermediary key
Session key management
Store transaction ID: / Session Key
Generate Transaction ID
Access Intermediary key
Split data according to Intermediary key
Access Parser Master key
Split transaction ID according to Parser Master key
Parse
Encrypted data share 1
Trans ID share 1
Encrypted data share 2
Trans ID share 2
Encrypted data share 3
Trans ID share 3
Encrypted data share n
Trans ID share n
Generate share 1 key
Generate share 2 key
Generate share 3 key
Generate share n key
Encrypt Pieces
Encrypt share 1 data with share 1 key
Encrypt share 2 data with share 2 key
Encrypt share 3 data with share 3 key
Encrypt share n data with share n key
Encrypted share 1 (data/trans ID)
Key n
Encrypted share 3 (data/trans ID)
Key 2
Obfuscate
Encrypted share 2 (data/trans ID)
Key 1
Encrypted share n (data/trans ID)
Key 3

FIG. 24

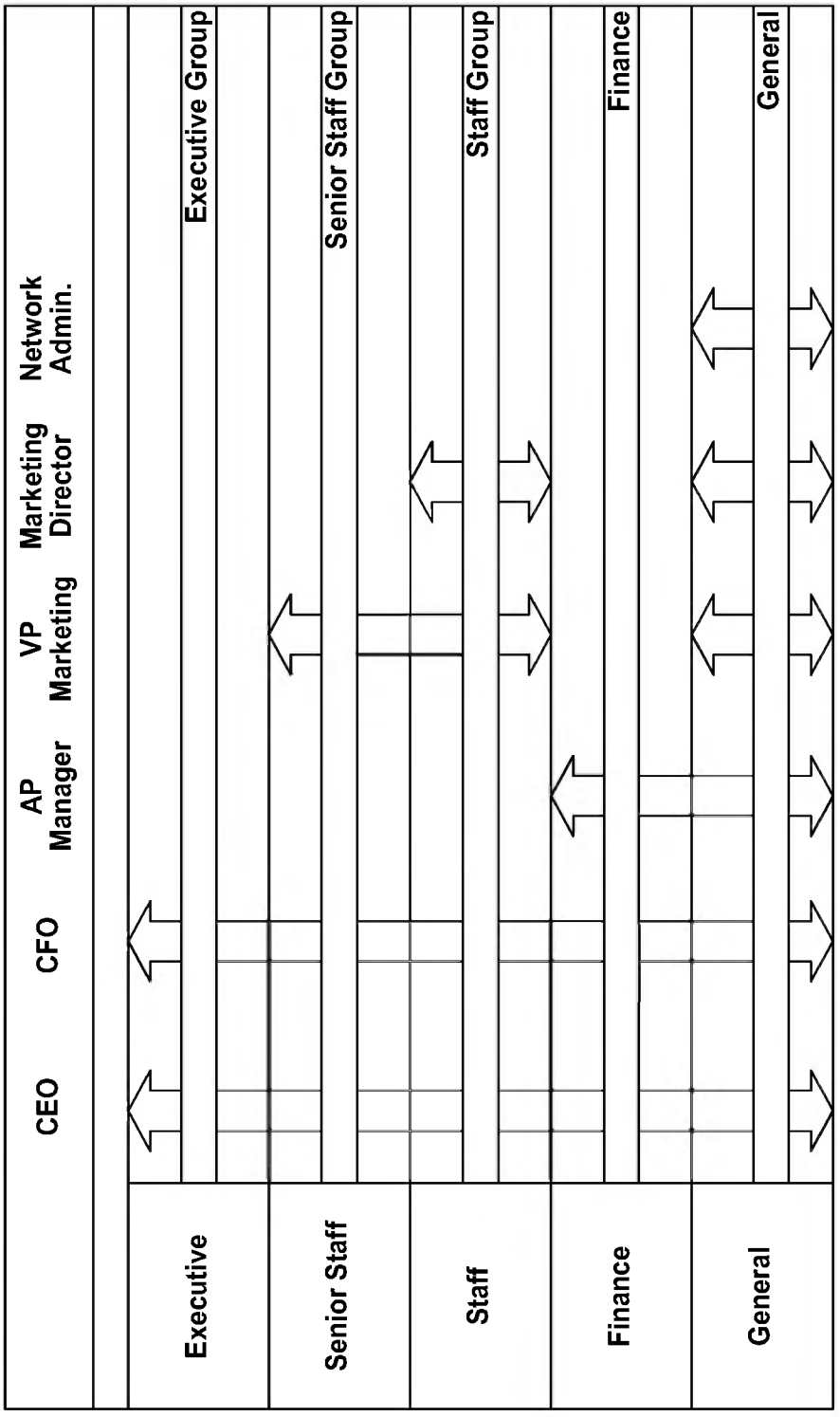

CEO
CFO
AP Manager
VP Marketing
Marketing Director
Network Admin.
Executive Group
Senior Staff Group
Staff Group
Finance
General
Executive
Senior Staff
Staff
Finance
General

FIG. 25

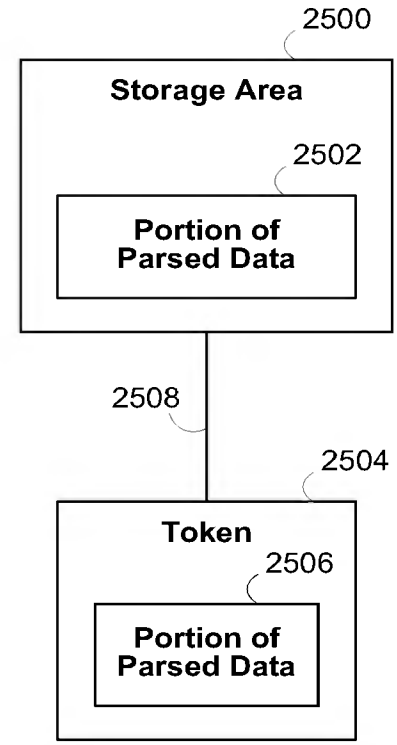

2500
Storage Area
2502
Portion of Parsed Data
2508
2504
Token
2506
Portion of Parsed Data

FIG. 26

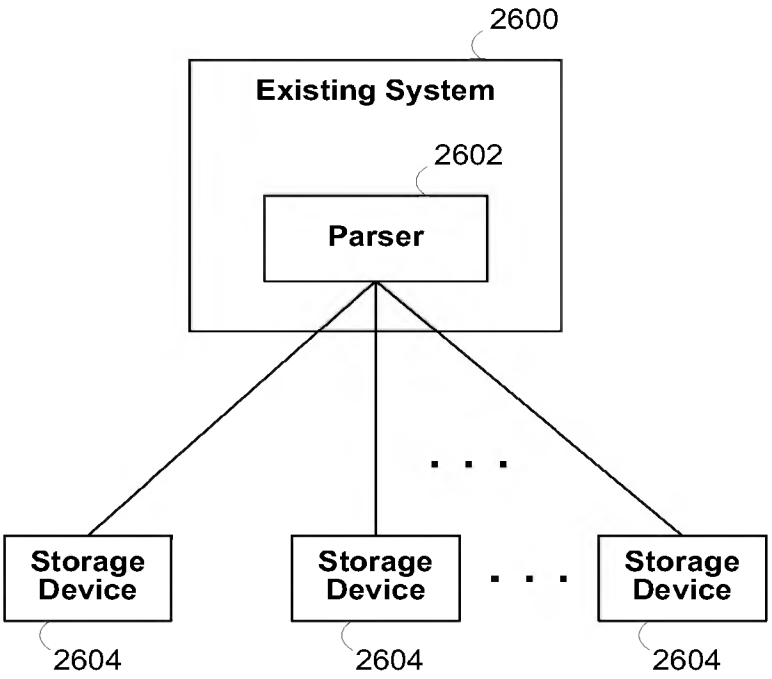

2600
Existing System
2602
Parser
Storage Device
2604
Storage Device
2604
Storage Device
2604

FIG. 27

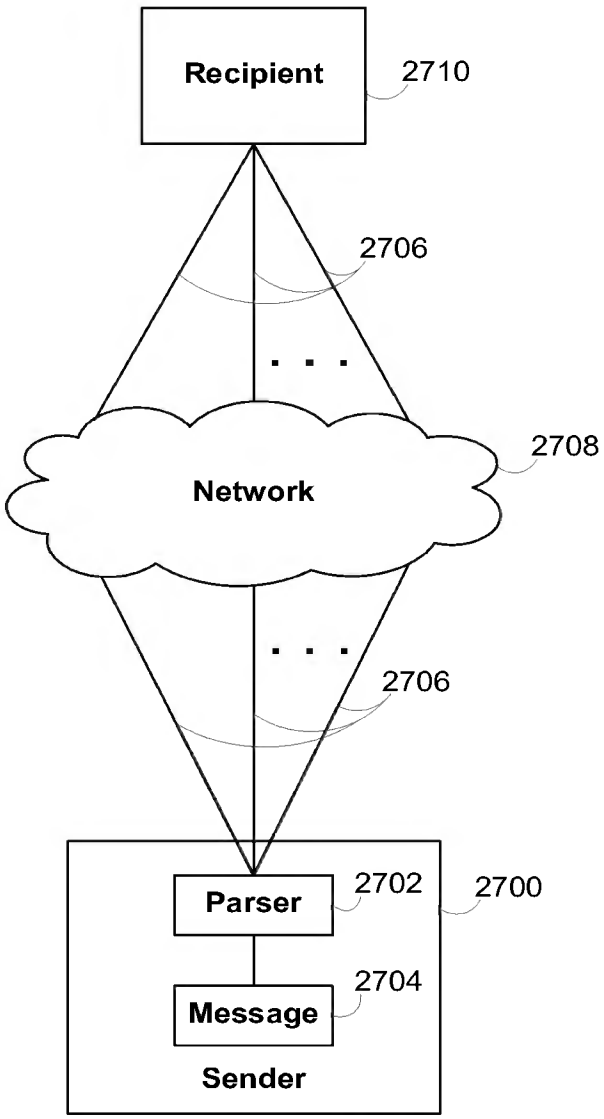

Recipient
2710
2706
2708
Network
2706
Parser
2702
2700
2704
Message
Sender

FIG. 28

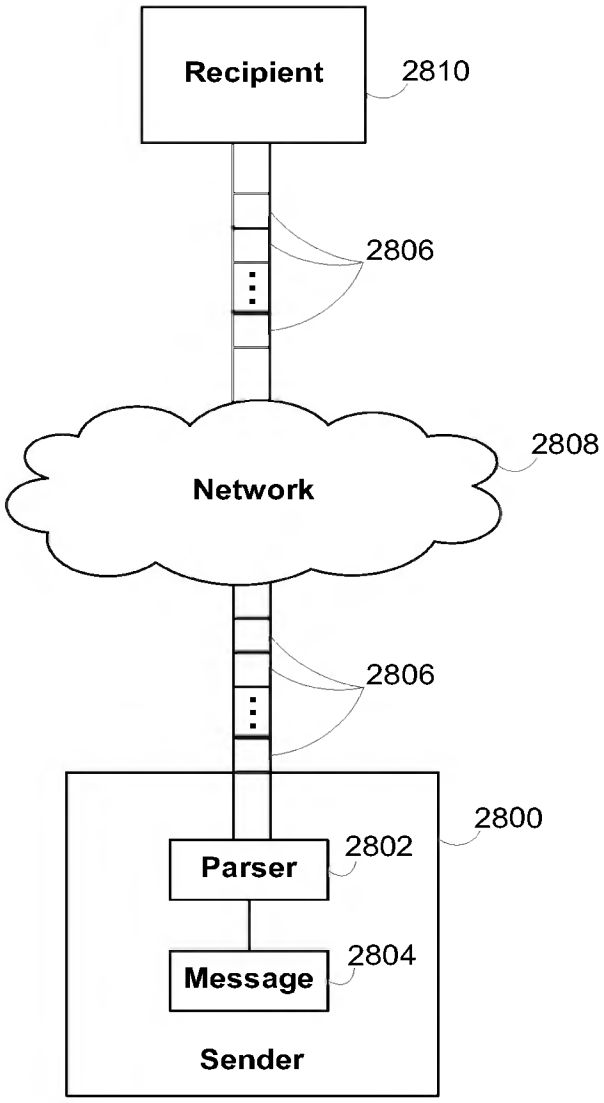

Recipient
2810
2806
Network
2808
2806
2800
Parser
2802
Message
2804
Sender

FIG. 29

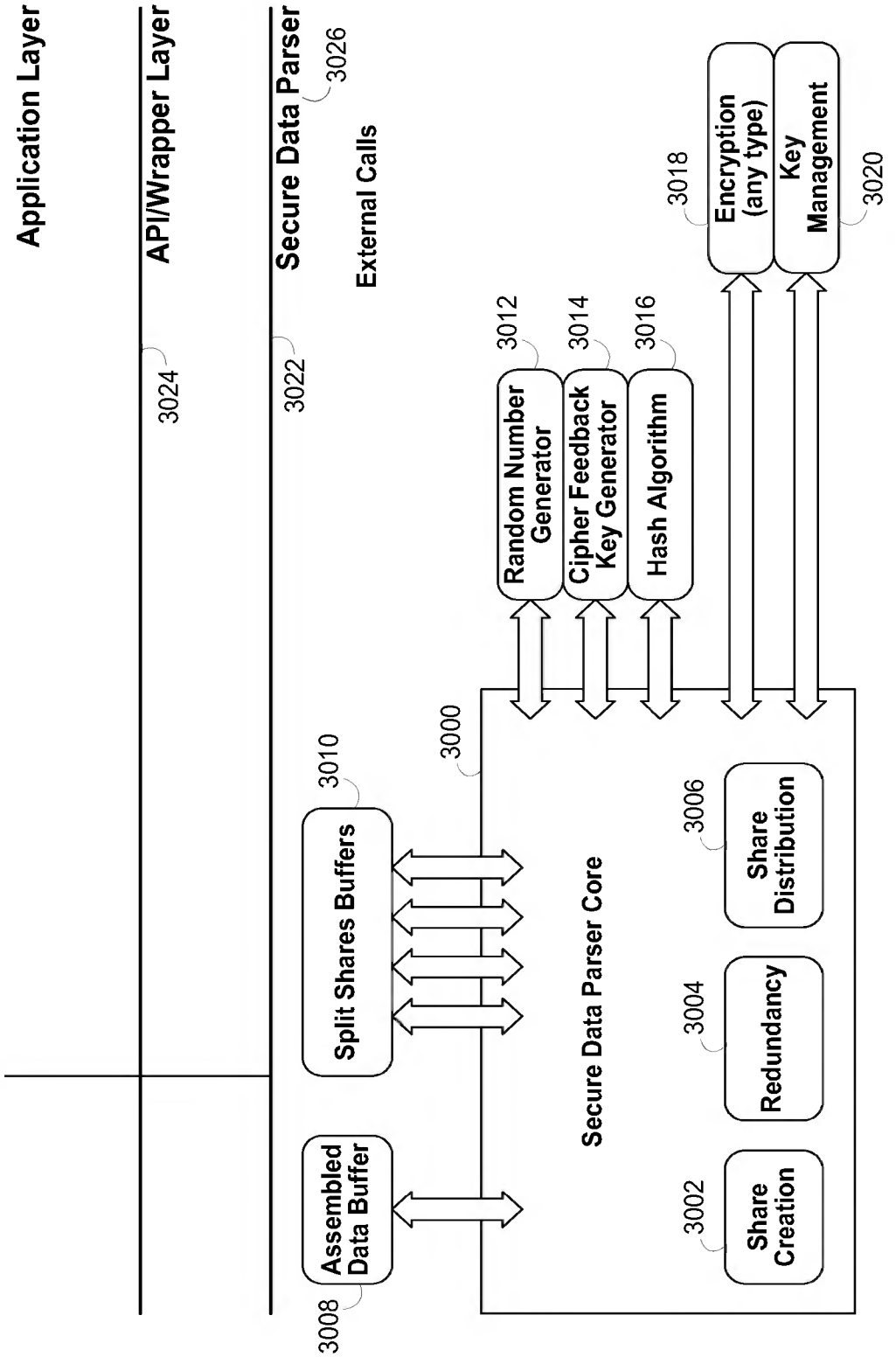

Application Layer
API/Wrapper Layer
Secure Data Parser
3026
External Calls
3024
3022
3012
3014
3016
3018
Random Number Generator
Cipher Feedback Key Generator
Hash Algorithm
Encryption (any type)
Key Management
3020
3000
3010
Split Shares Buffers
Secure Data Parser Core
3006
Share Distribution
3004
Redundancy
3002
Share Creation
Assembled Data Buffer
3008

FIG. 30

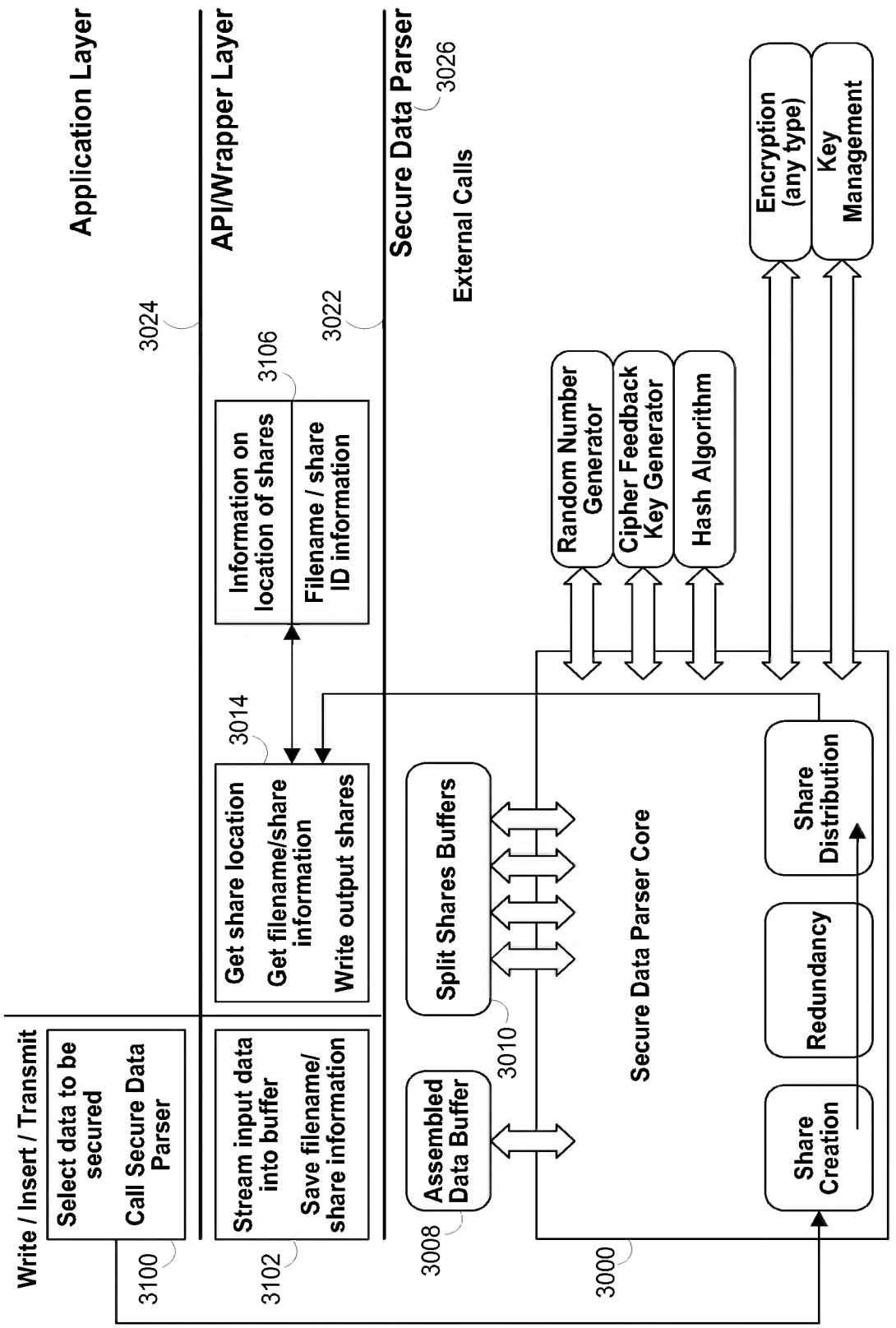

Application Layer
API/Wrapper Layer
Secure Data Parser
3026
External Calls
3024
3022
3106
Information on location of shares
Filename / share ID information
3014
Get share location
Get filename/share information
Write output shares
Write / Insert / Transmit
Select data to be secured
Call Secure Data Parser
3100
Stream input data into buffer
Save filename/ share information
3102
Split Shares Buffers
3010
Assembled Data Buffer
3008
Secure Data Parser Core
3000
Random Number Generator
Cipher Feedback Key Generator
Hash Algorithm
Encryption (any type)
Key Management
Share Distribution
Redundancy
Share Creation

FIG. 31

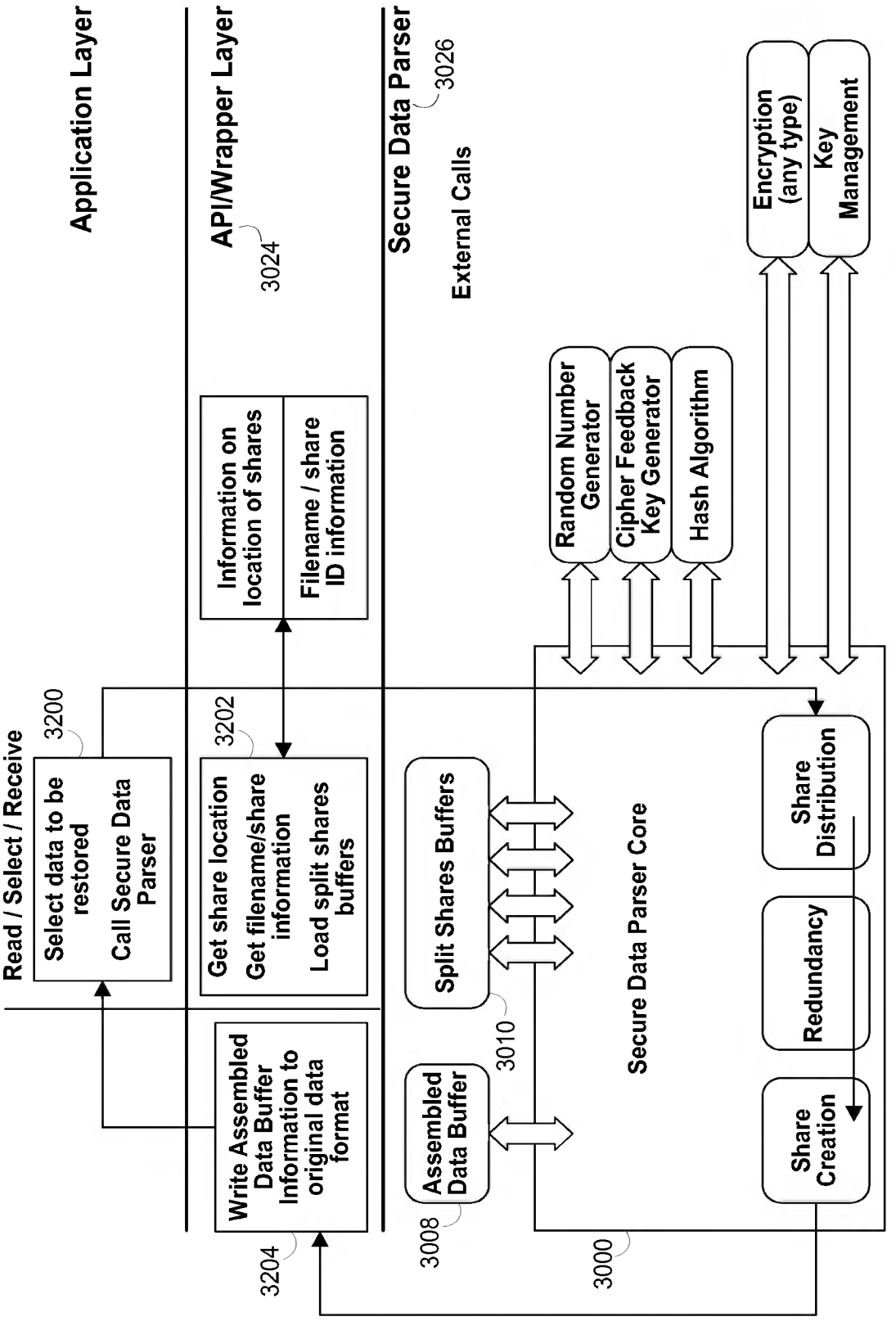

Application Layer
API/Wrapper Layer
3024
Secure Data Parser
3026
External Calls
Read / Select / Receive
3200
Select data to be restored
Call Secure Data Parser
3202
Get share location
Get filename/share information
Load split shares buffers
Information on location of shares
Filename / share ID information
Write Assembled Data Buffer Information to original data format
3204
Split Shares Buffers
3010
Assembled Data Buffer
3008
Secure Data Parser Core
3000
Random Number Generator
Cipher Feedback Key Generator
Hash Algorithm
Encryption (any type)
Key Management
Share Distribution
Redundancy
Share Creation

FIG. 32

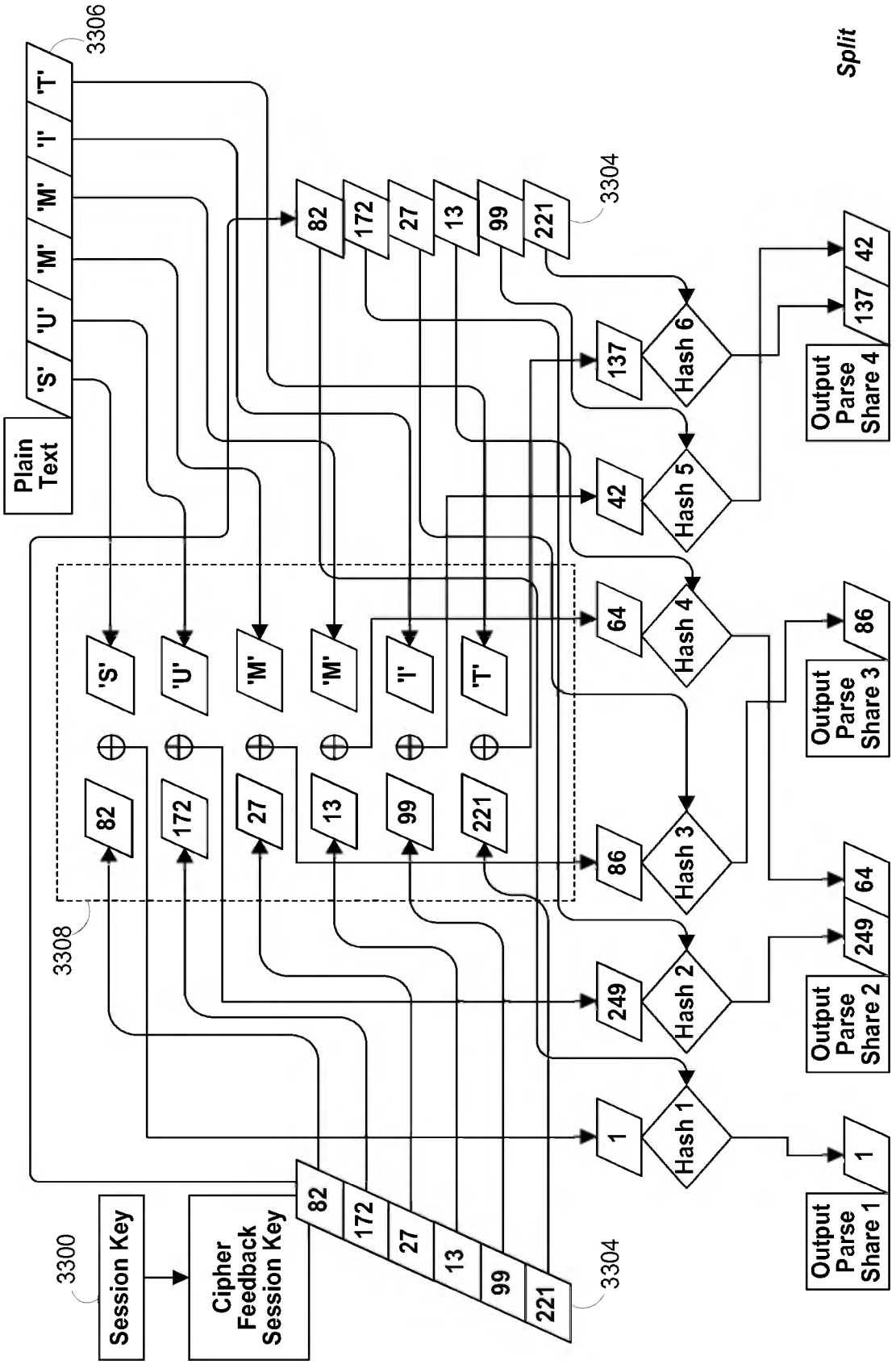

3306
Plain Text
'S' 'U' 'M' 'M' 'I' 'T'
3304
82 172 27 13 99 221
3308
Split
Hash 1
Hash 2
Hash 3
Hash 4
Hash 5
Hash 6
Output Parse Share 1
Output Parse Share 2
Output Parse Share 3
Output Parse Share 4
3300
Session Key
Cipher Feedback Session Key

FIG. 33

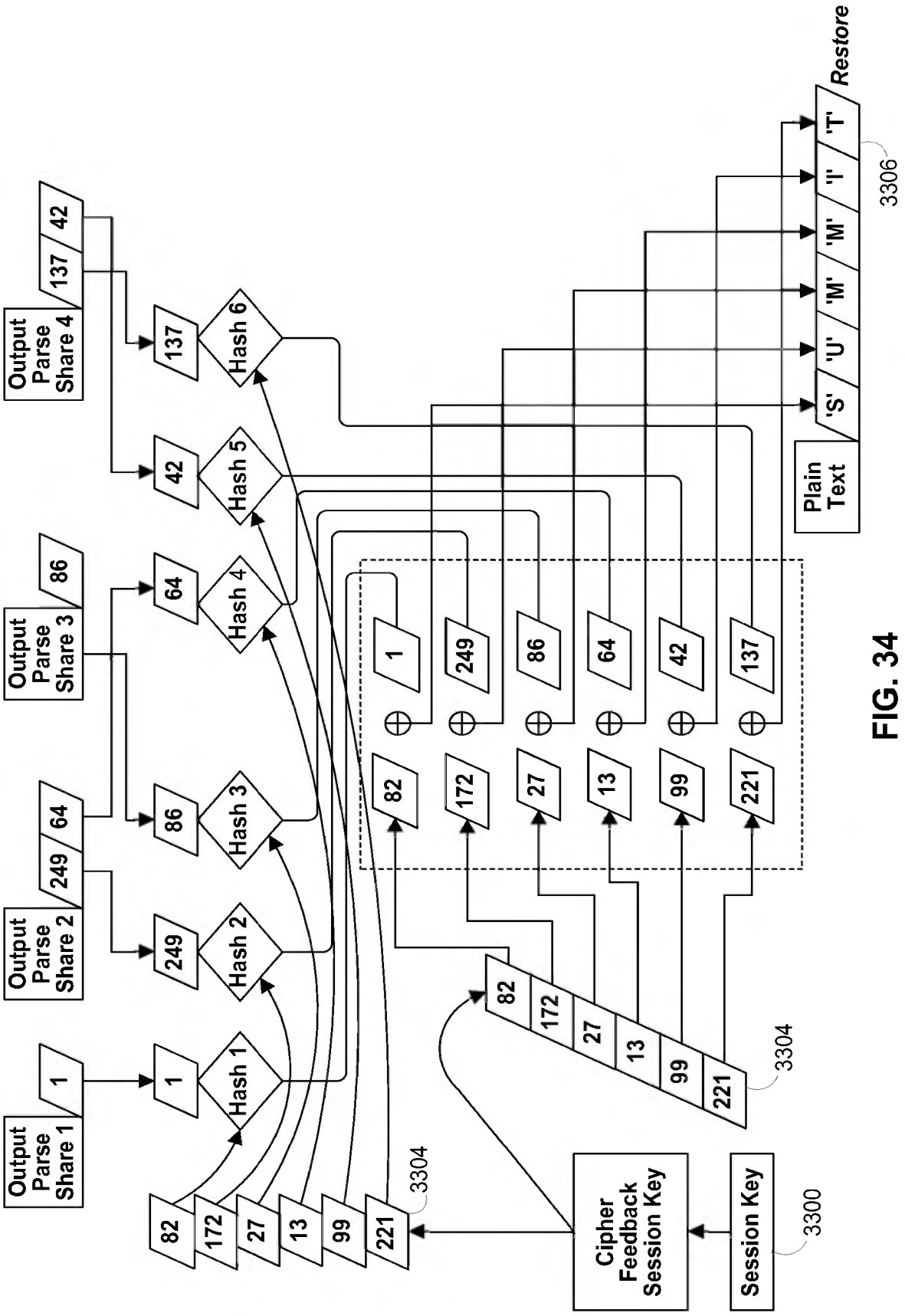

Output Parse Share 1
Output Parse Share 2
Output Parse Share 3
Output Parse Share 4
Hash 1
Hash 2
Hash 3
Hash 4
Hash 5
Hash 6
Cipher Feedback Session Key
Session Key
3300
3304
3304
3306
Plain Text
Restore
'S' 'U' 'M' 'M' 'I' 'T'

FIG. 34

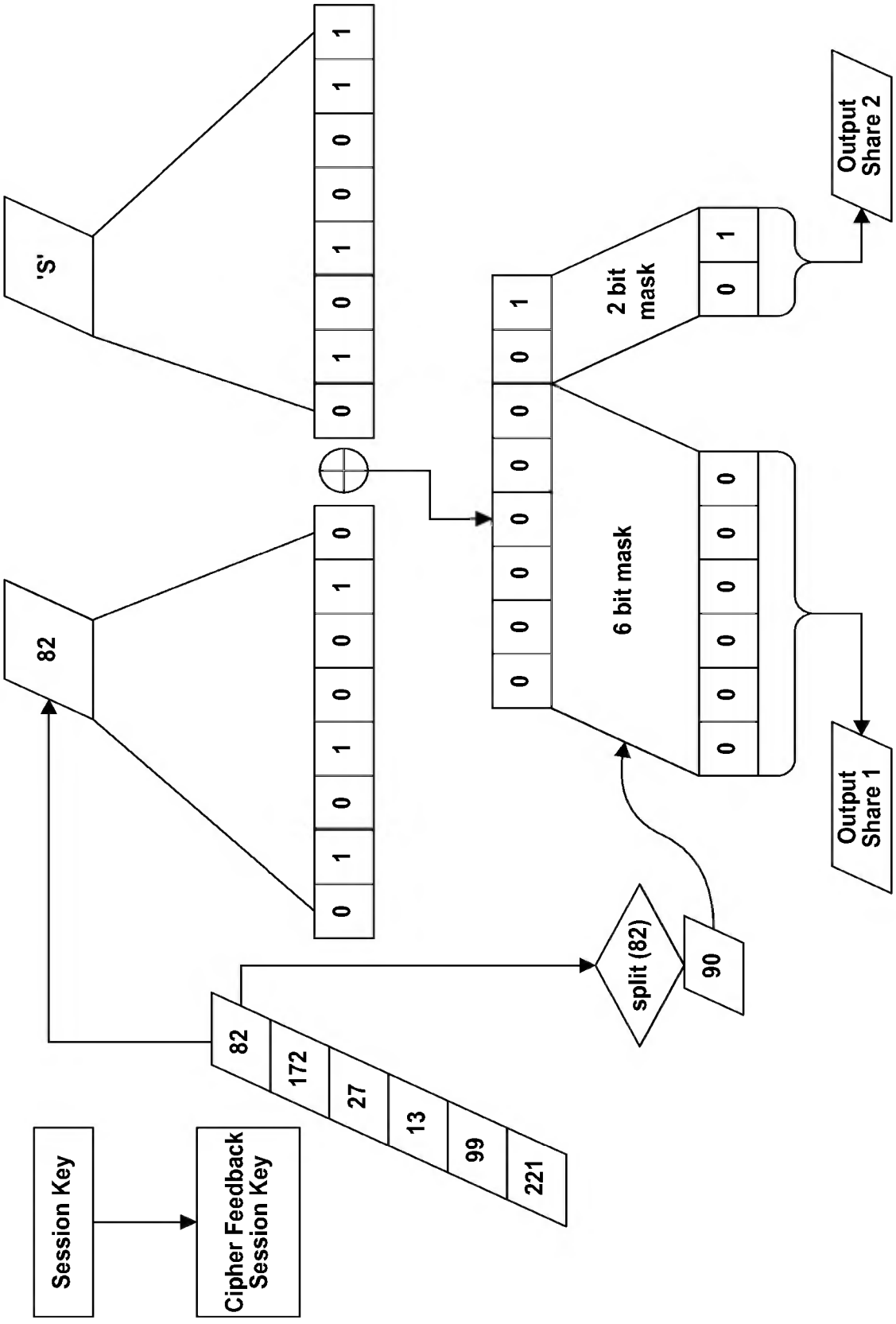

Session Key
Cipher Feedback Session Key
82
172
27
13
99
221
split (82)
90
'S'
82
6 bit mask
2 bit mask
Output Share 1
Output Share 2

FIG. 35

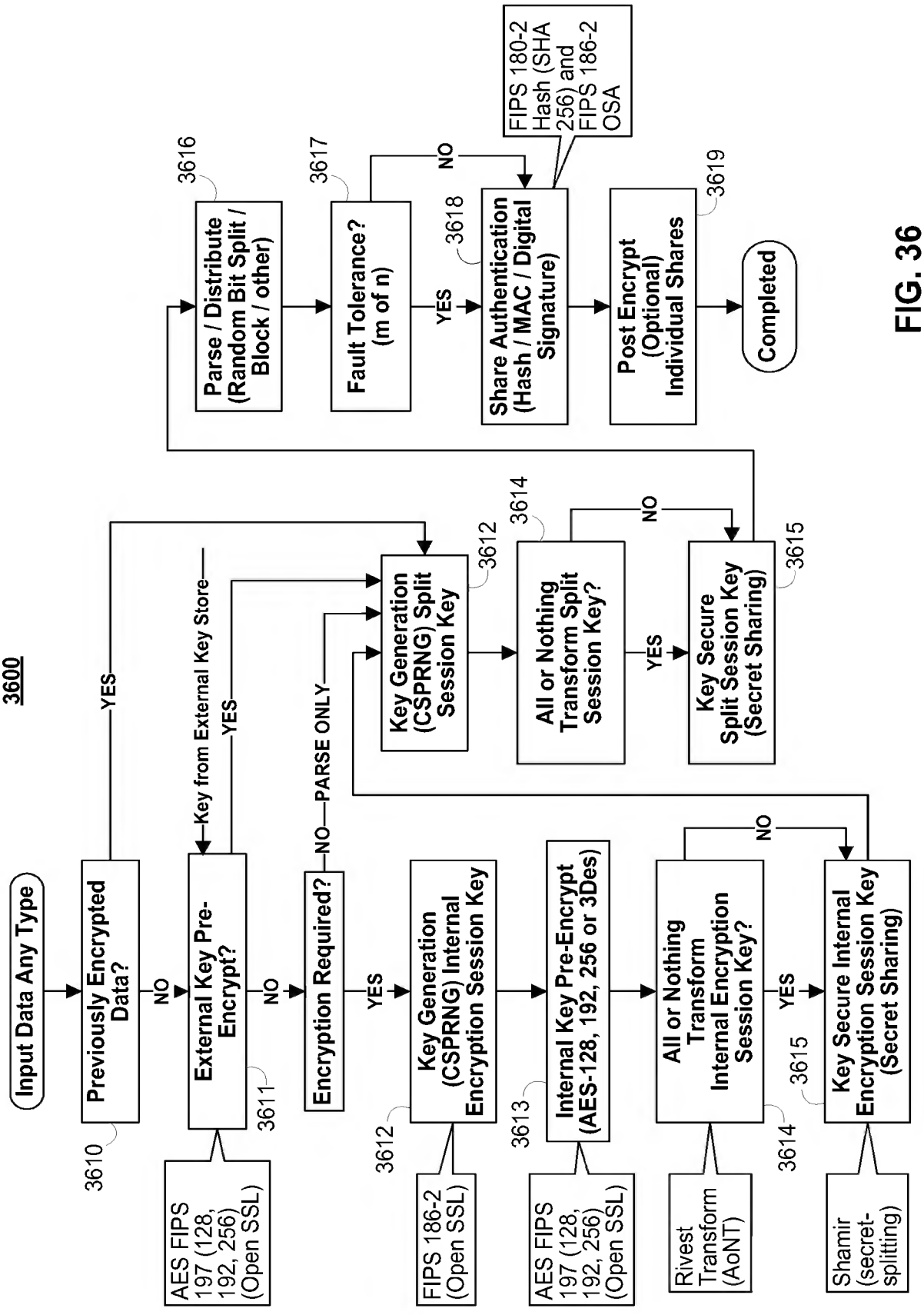

3600
Input Data Any Type
Previously Encrypted Data?
YES
NO
3610
External Key Pre-Encrypt?
Key from External Key Store
YES
NO
3611
AES FIPS 197 (128, 192, 256) (Open SSL)
Encryption Required?
NO—PARSE ONLY
YES
Key Generation (CSPRNG) Internal Encryption Session Key
3612
FIPS 186-2 (Open SSL)
Internal Key Pre-Encrypt (AES-128, 192, 256 or 3Des)
3613
AES FIPS 197 (128, 192, 256) (Open SSL)
All or Nothing Transform Internal Encryption Session Key?
NO
YES
3614
Rivest Transform (AoNT)
Key Secure Internal Encryption Session Key (Secret Sharing)
3615
Shamir (secret-splitting)
Key Generation (CSPRNG) Split Session Key
3612
All or Nothing Transform Split Session Key?
3614
NO
YES
Key Secure Split Session Key (Secret Sharing)
3615
Parse / Distribute (Random Bit Split / Block / other)
3616
Fault Tolerance? (m of n)
3617
NO
YES
3618
FIPS 180-2 Hash (SHA 256) and FIPS 186-2 OSA
Share Authentication (Hash / MAC / Digital Signature)
Post Encrypt (Optional) Individual Shares
3619
Completed

FIG. 36

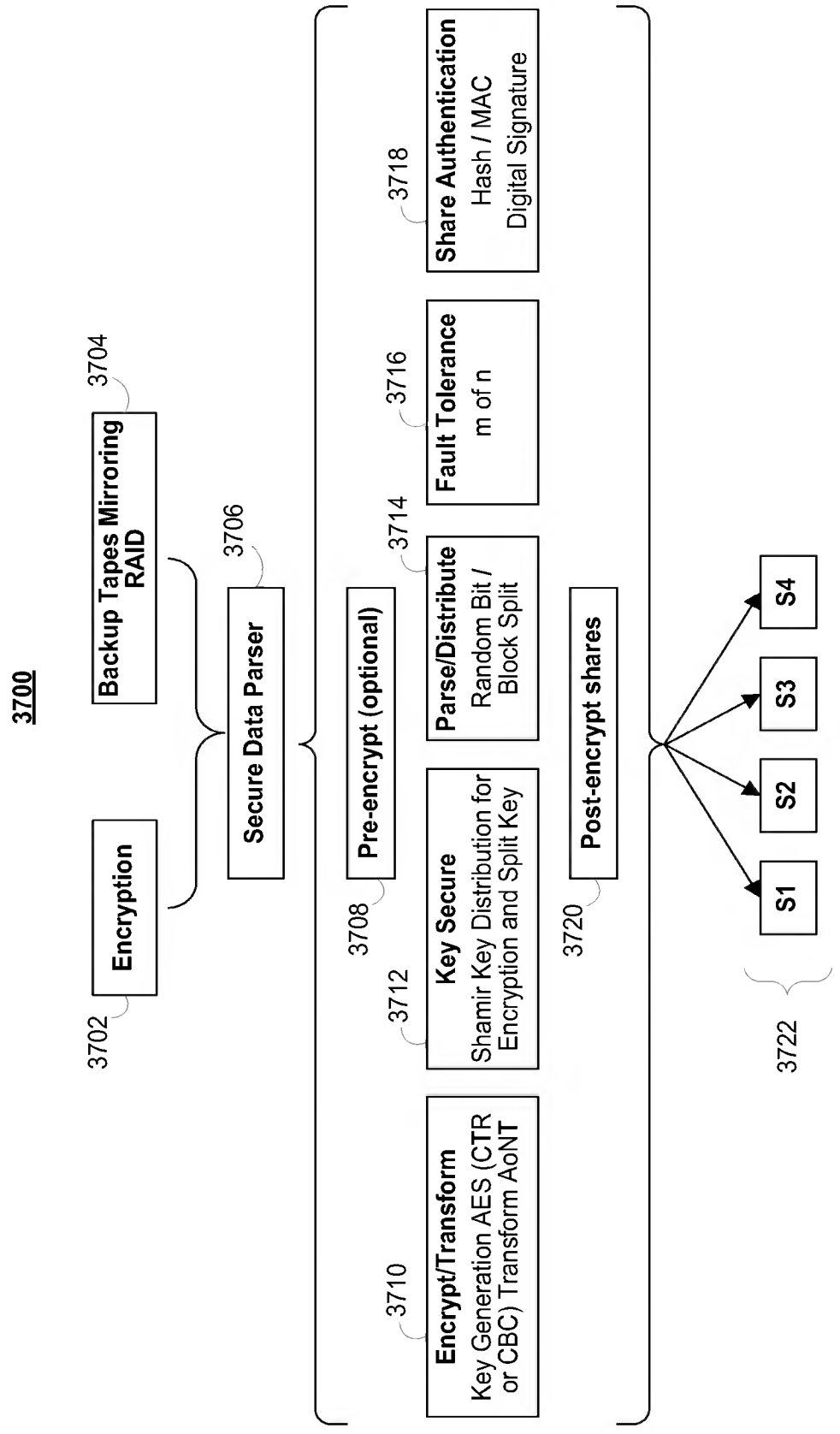

3700
Backup Tapes Mirroring RAID
3704
Encryption
3702
Secure Data Parser
3706
Pre-encrypt (optional)
3708
Key Secure
Shamir Key Distribution for Encryption and Split Key
3712
Encrypt/Transform
Key Generation AES (CTR or CBC) Transform AoNT
3710
Parse/Distribute
Random Bit / Block Split
3714
Fault Tolerance
m of n
3716
Share Authentication
Hash / MAC
Digital Signature
3718
Post-encrypt shares
3720
S1
S2
S3
S4
3722

FIG. 37

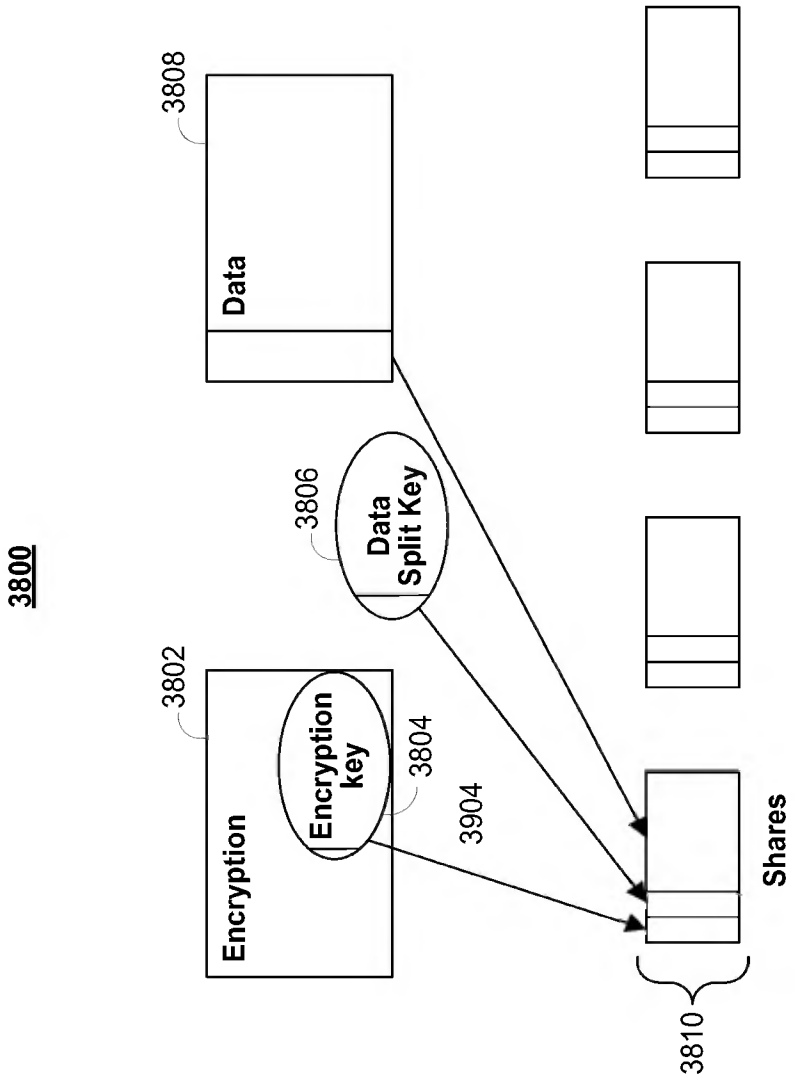

3800
3808
Data
3806
Data Split Key
3802
Encryption
Encryption key
3804
3904
Shares
3810

FIG. 38

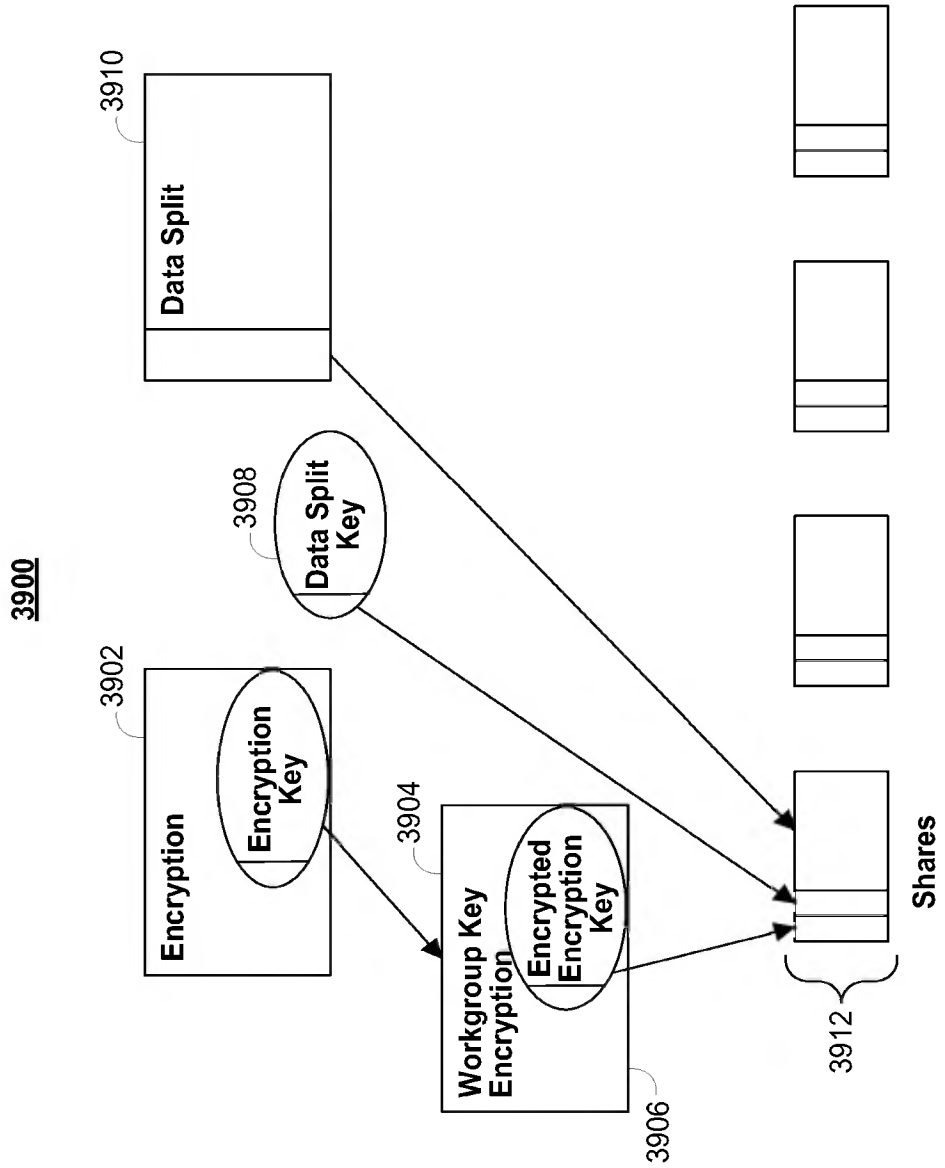

3900
3910
Data Split
3908
Data Split Key
3902
Encryption
Encryption Key
3904
Workgroup Key Encryption
Encrypted Encryption Key
3906
3912
Shares

FIG. 39

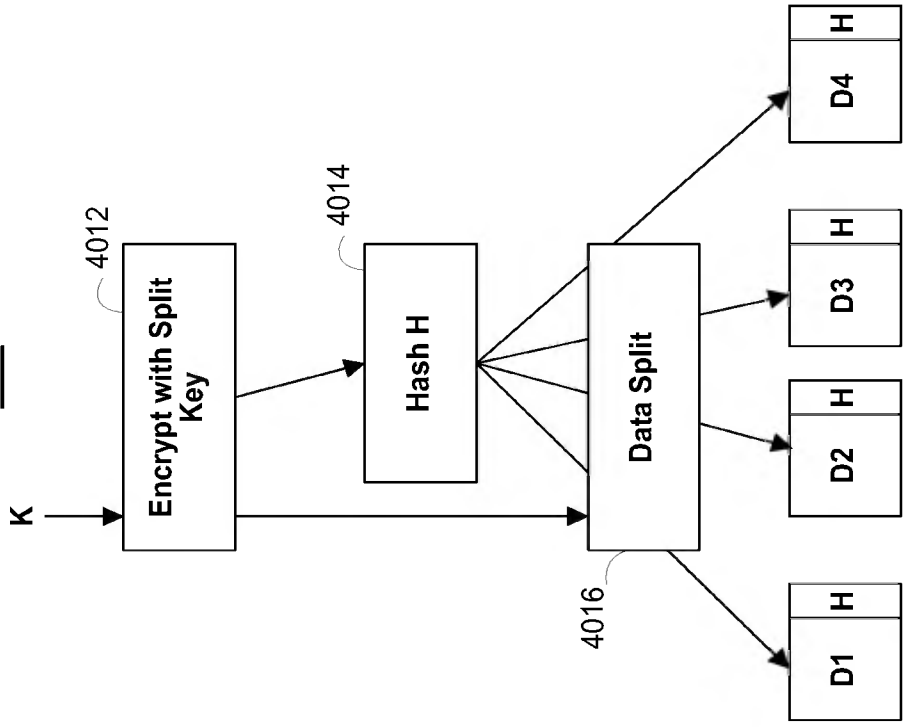


FIG. 40B

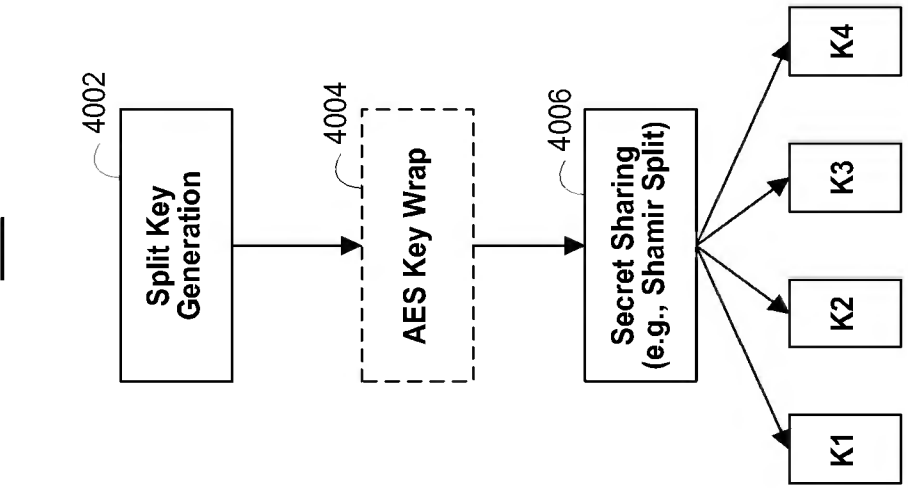


FIG. 40A

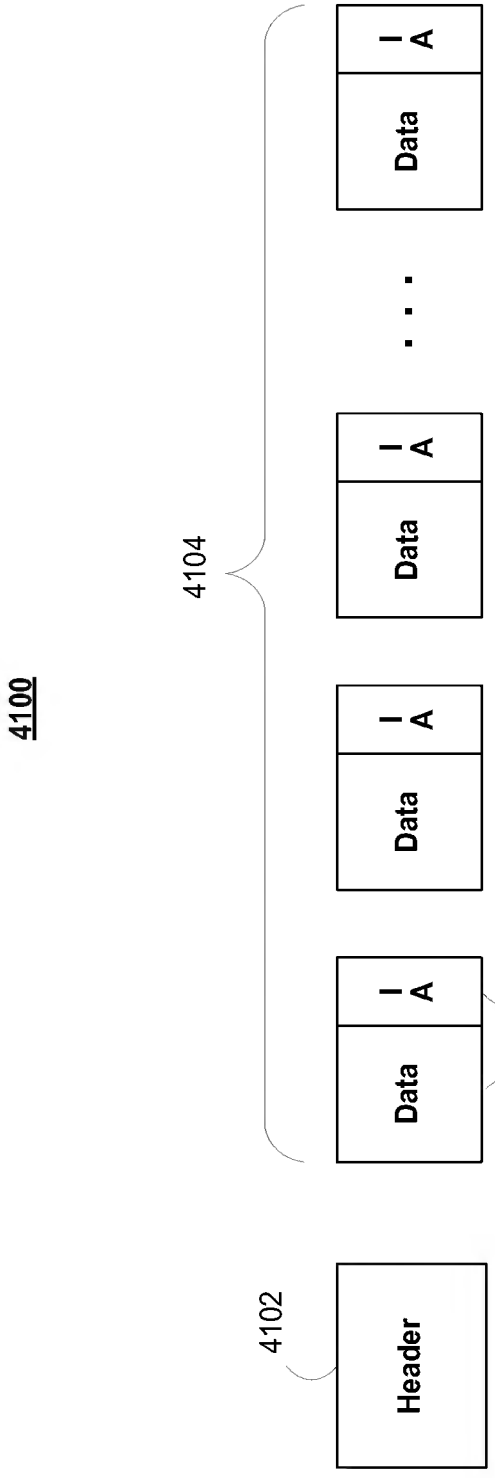


FIG. 41

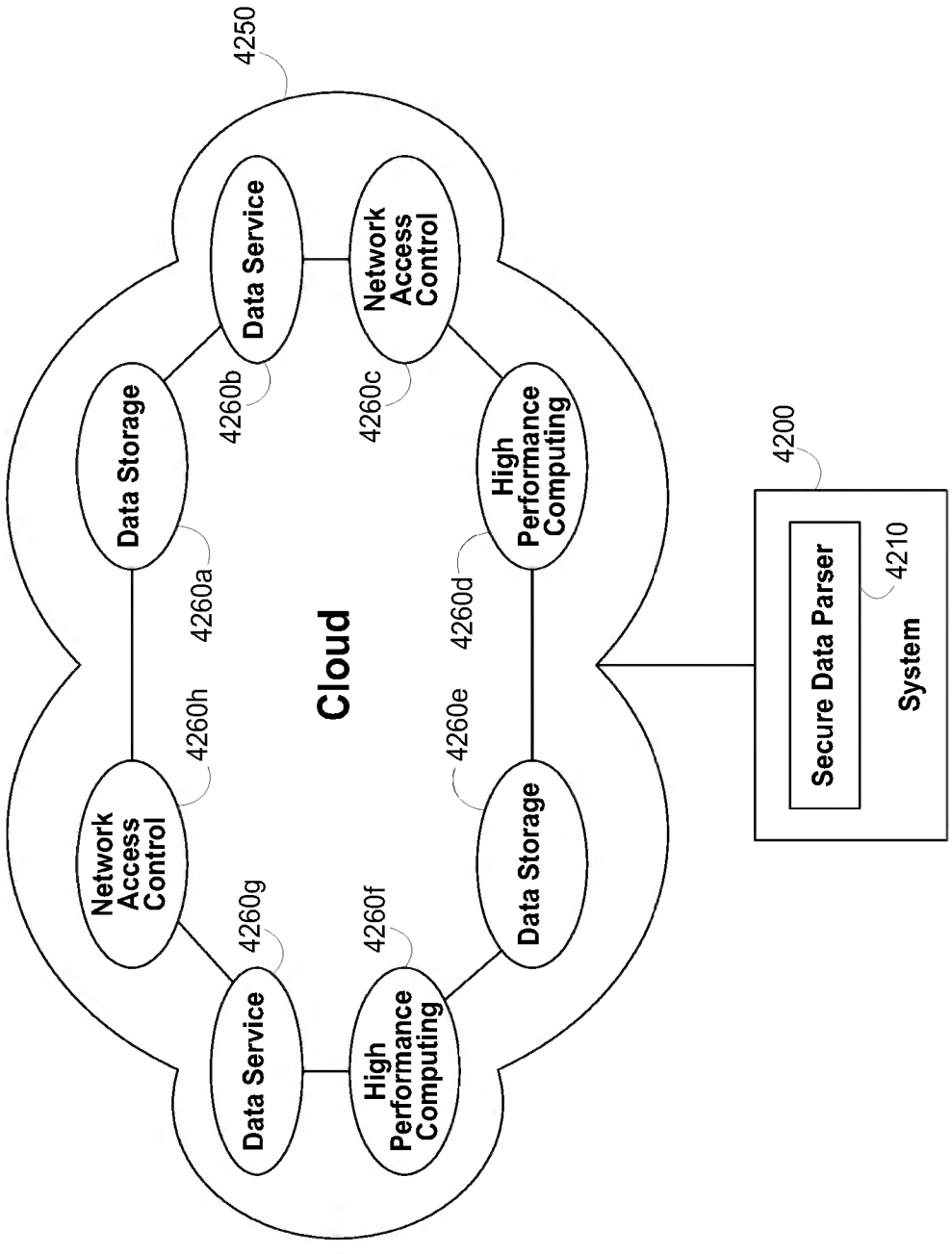

4250
Data Service
Network Access Control
Data Storage
High Performance Computing
4260b
4260c
4260a
4260d
Cloud
4260h
4260e
Network Access Control
Data Storage
4260g
4260f
Data Service
High Performance Computing
4200
Secure Data Parser
4210
System

FIG. 42

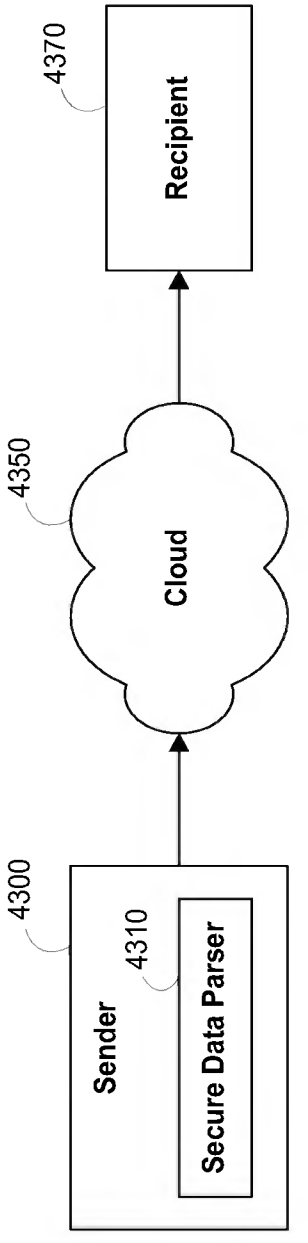

4370
Recipient
4350
Cloud
4300
Sender
4310
Secure Data Parser

FIG. 43

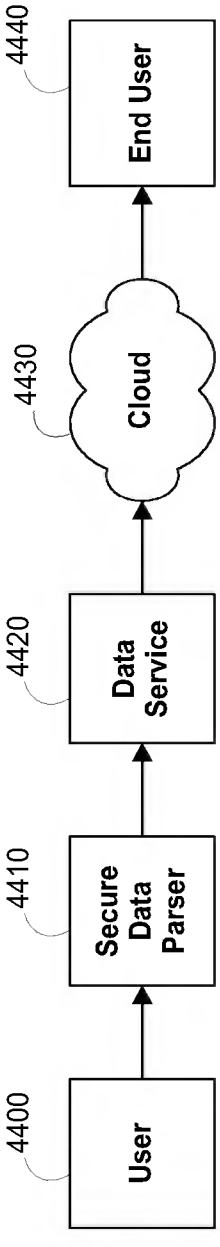

4440
End User
4430
Cloud
4420
Data Service
4410
Secure Data Parser
4400
User

FIG. 44

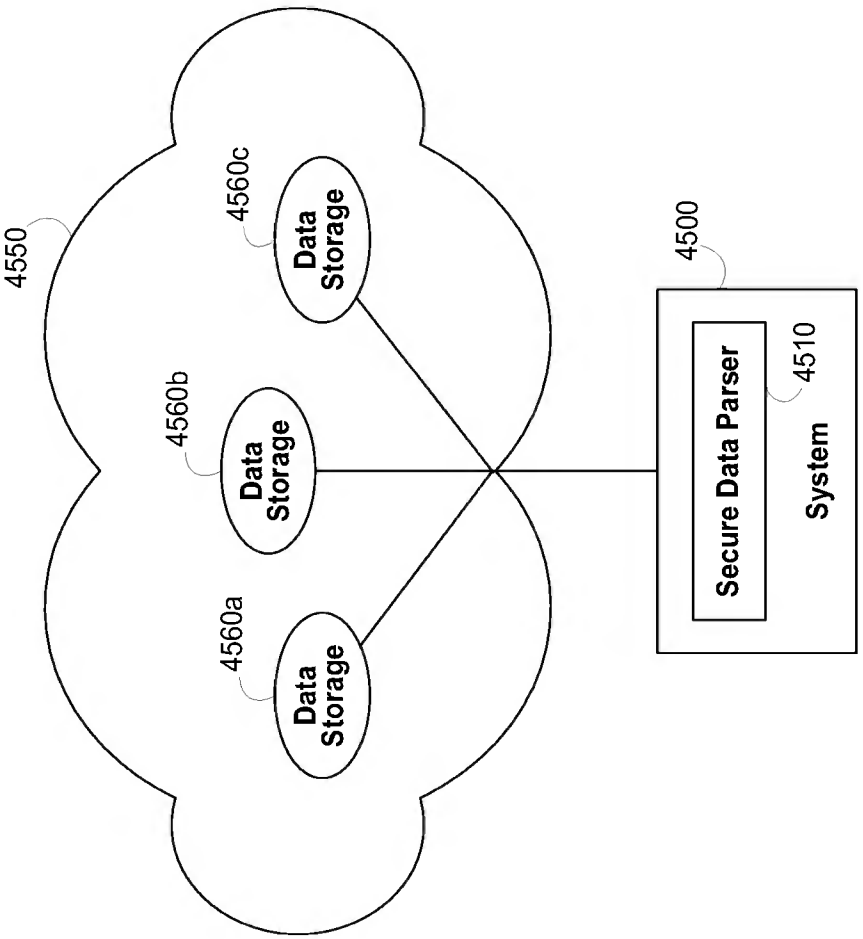

4550
4560c
Data Storage
4560b
Data Storage
4560a
Data Storage
4500
Secure Data Parser
4510
System

FIG. 45

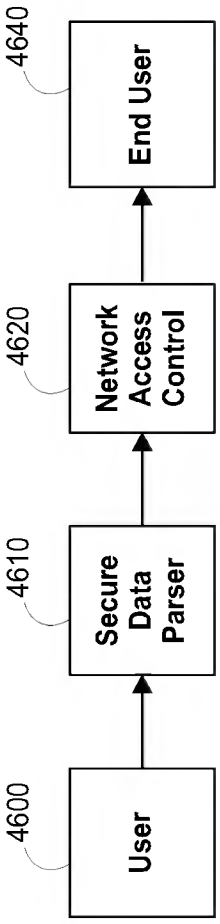

4640
End User
4620
Network Access Control
4610
Secure Data Parser
4600
User

FIG. 46

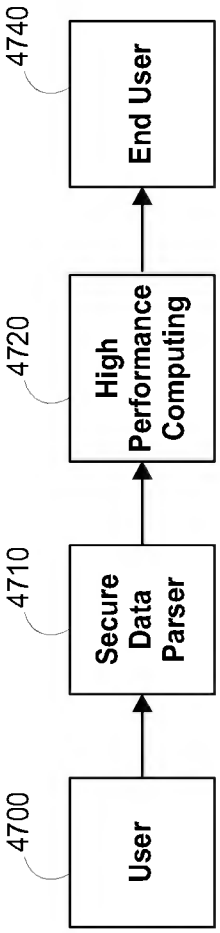

4740
End User
4720
High Performance Computing
4710
Secure Data Parser
4700
User

FIG. 47

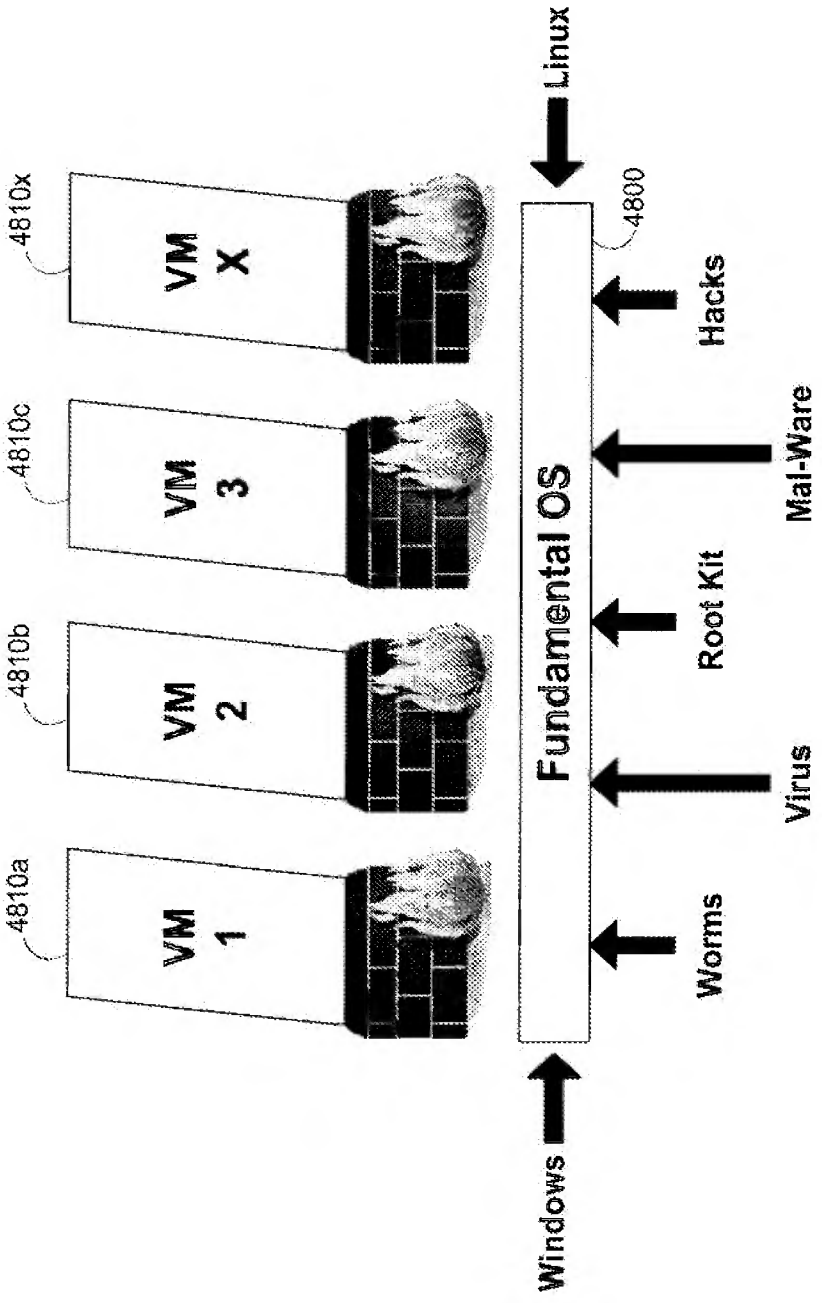

4810a
VM 1
4810b
VM 2
4810c
VM 3
4810x
VM X
Fundamental OS
4800
Windows
Linux
Worms
Virus
Root Kit
Mal-Ware
Hacks

FIG. 48

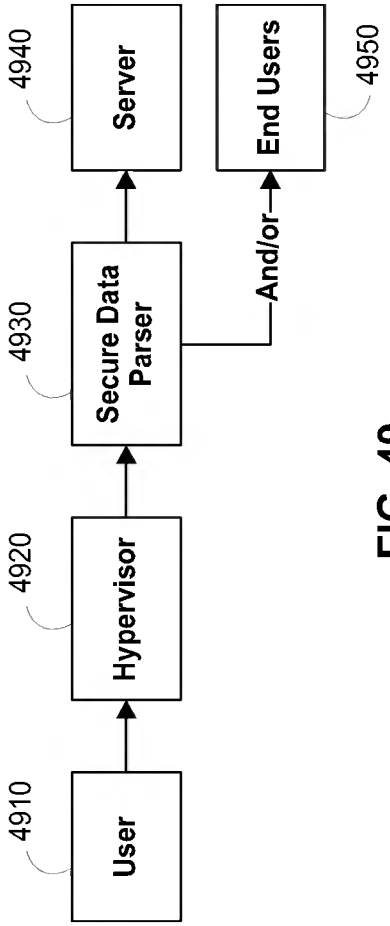

4910
User
4920
Hypervisor
4930
Secure Data Parser
4940
Server
And/or
4950
End Users

FIG. 49

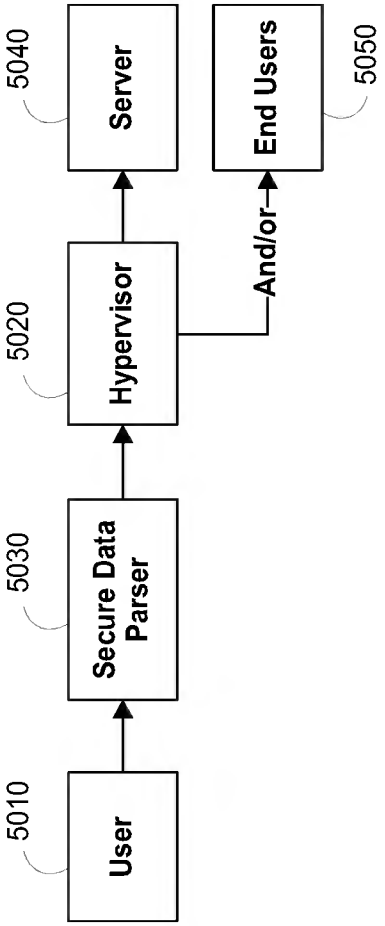

5010
User
5030
Secure Data Parser
5020
Hypervisor
5040
Server
And/or
5050
End Users

FIG. 50

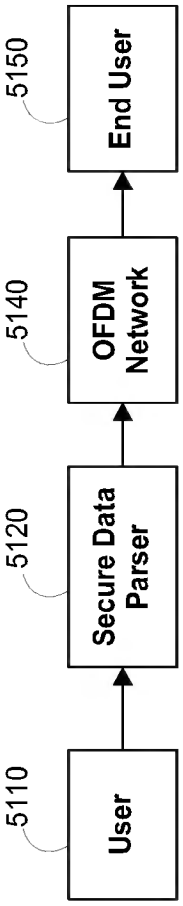

5110
User
5120
Secure Data Parser
5140
OFDM Network
5150
End User

FIG. 51

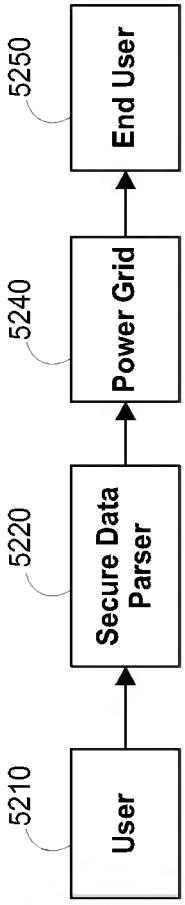

5250
End User
5240
Power Grid
5220
Secure Data Parser
5210
User

FIG. 52

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
25 November 2010 (25.11.2010)

PCT

(10) International Publication Number
WO 2010/135412 A3

(51) International Patent Classification:
H04L 29/06 (2006.01) *H04L 29/08* (2006.01)

(21) International Application Number:
PCT/US2010/035377

(22) International Filing Date:
19 May 2010 (19.05.2010)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
61/179,481 19 May 2009 (19.05.2009) US

(71) Applicant *(for all designated States except US)*: **SECURITY FIRST CORP.** [US/US]; 22362 Gilberto #130, Rancho Santa Margarita, California 92688 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **ORSINI, Rick, L.** [US/US]; 2100 Kings Forest Lane, Flower Mound, Texas 75028 (US). **O'HARE, Mark, S.** [US/US]; 8 Kennedy Court, Coto De Caza, California 92679 (US). **MARTIN, Don** [US/US]; 11006 Sterling Cover Drive, Chesterfield, Virginia 23838 (US).

(74) Agents: **INGERMAN, Jeffrey, H.** et al.; Ropes & Gray LLP, 1211 Avenue of the Americas, New York, New York 10036-8704 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

[Continued on next page]

(54) **Title**: SYSTEMS AND METHODS FOR SECURING DATA IN THE CLOUD

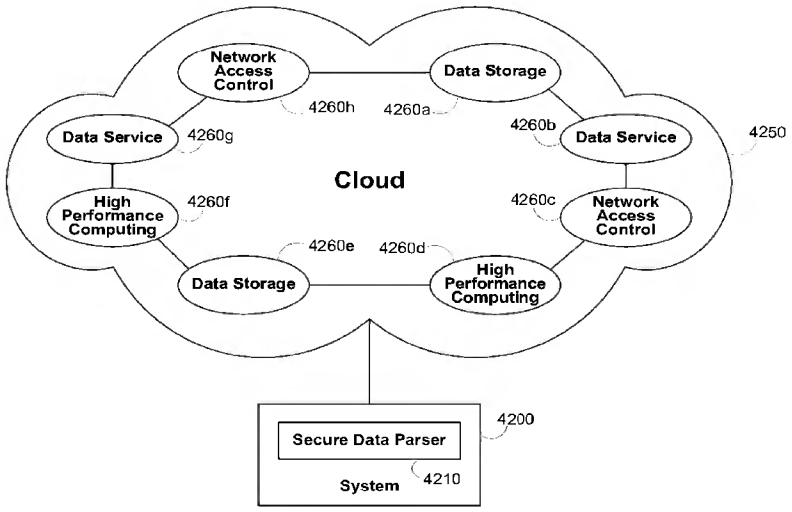


FIG. 42

(57) **Abstract**: A secure data parser is provided that may be integrated into any suitable system for securely storing data in and communicating data with cloud computing resources. The secure data parser parses data and then splits the data into multiple portions that are stored or communicated distinctly. Encryption of the original data, the portions of data, or both may be employed for additional security.

WO 2010/135412 A3

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))*

(88) Date of publication of the international search report:
3 February 2011

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2010/035377

A. CLASSIFICATION OF SUBJECT MATTER
INV. H04L29/06 H04L29/08

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)
EPO-Internal, COMPENDEX, INSPEC, PAJ, IBM-TDB, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2006/177061 A1 (ORSINI RICK L [US] ET AL) 10 August 2006 (2006-08-10) abstract paragraph [0118] - paragraph [0128] figures 7, 8	1-10, 14-23
X	US 2007/160198 A1 (ORSINI RICK [US] ET AL) 12 July 2007 (2007-07-12) abstract paragraph [0124] - paragraph [0134] figures 7, 8	1-10, 14-23

☐ Further documents are listed in the continuation of Box C.

☒ See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
22 September 2010	14/12/2010

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL – 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Horn, Marc-Philipp

Form PCT/ISA/210 (second sheet) (April 2005)

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2010/035377

Box No. II Observations where certain claims were found unsearchable (Continuation of item 2 of first sheet)

This international search report has not been established in respect of certain claims under Article 17(2)(a) for the following reasons:

1. ☐ Claims Nos.:
because they relate to subject matter not required to be searched by this Authority, namely:

2. ☐ Claims Nos.:
because they relate to parts of the international application that do not comply with the prescribed requirements to such an extent that no meaningful international search can be carried out, specifically:

3. ☐ Claims Nos.:
because they are dependent claims and are not drafted in accordance with the second and third sentences of Rule 6.4(a).

Box No. III Observations where unity of invention is lacking (Continuation of item 3 of first sheet)

This International Searching Authority found multiple inventions in this international application, as follows:

see additional sheet

1. ☐ As all required additional search fees were timely paid by the applicant, this international search report covers all searchable claims.

2. ☐ As all searchable claims could be searched without effort justifying an additional fees, this Authority did not invite payment of additional fees.

3. ☐ As only some of the required additional search fees were timely paid by the applicant, this international search report covers only those claims for which fees were paid, specifically claims Nos.:

4. ☒ No required additional search fees were timely paid by the applicant. Consequently, this international search report is restricted to the invention first mentioned in the claims; it is covered by claims Nos.:

1-10, 14-23

Remark on Protest

☐ The additional search fees were accompanied by the applicant's protest and, where applicable, the payment of a protest fee.

☐ The additional search fees were accompanied by the applicant's protest but the applicable protest fee was not paid within the time limit specified in the invitation.

☐ No protest accompanied the payment of additional search fees.

FURTHER INFORMATION CONTINUED FROM PCT/ISA/ 210

This International Searching Authority found multiple (groups of) inventions in this international application, as follows:

1. claims: 1-10, 14-23

Splitting data and storing the data on a pluraltiy of remote storage devices in a cloud computing storage network

---

2. claims: 11-13, 24-26

Sending split data from a first location to a second location over a plurality of orthogonal frequency-division multiplexing channels.

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/US2010/035377

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2006177061 A1	10-08-2006	NONE	
US 2007160198 A1	12-07-2007	AU 2006350252 A1	08-05-2008
		CA 2629015 A1	08-05-2008
		CN 101401341 A	01-04-2009
		EP 1952575 A2	06-08-2008
		WO 2008054406 A2	08-05-2008

Form PCT/ISA/210 (patent family annex) (April 2005)

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
6 October 2011 (06.10.2011)

PCT

(10) International Publication Number
WO 2011/123692 A2

(51) International Patent Classification:
H04L 29/08 (2006.01) *G06F 17/30* (2006.01)
G06F 21/00 (2006.01) *G06F 11/10* (2006.01)
H04L 9/08 (2006.01) *G06F 11/20* (2006.01)

(21) International Application Number:
PCT/US2011/030801

(22) International Filing Date:
31 March 2011 (31.03.2011)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
61/319,658 31 March 2010 (31.03.2010) US
61/320,242 1 April 2010 (01.04.2010) US

(72) Inventors; and
(71) Applicants : **ORSINI, Rick, L.** [US/US]; 2100 Kings Forest Lane, Flower Mound, TX 75028 (US). **O'HARE, Mark, S.** [US/US]; 8 Kennedy Court, Coto De Caza, CA 92679 (US).

(74) Agents: **INGERMAN, Jeffrey, H.** et al.; Ropes & Gray LLP, 1211 Avenue Of The Americas, New York, NY 10036-8704 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *without international search report and to be republished upon receipt of that report (Rule 48.2(g))*

(54) **Title**: SYSTEMS AND METHODS FOR SECURING DATA IN MOTION

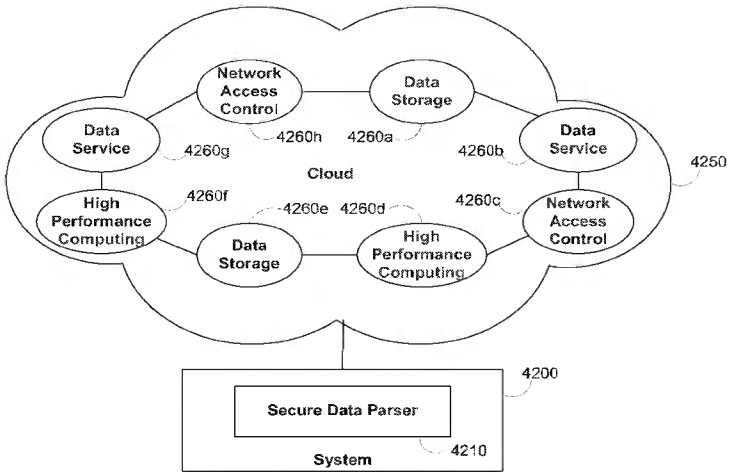


FIG. 42

(57) **Abstract**: The systems and methods of the present invention provide a solution that makes data provably secure and accessible - addressing data security at the bit level - thereby eliminating the need for multiple perimeter hardware and software technologies. Data security is incorporated or weaved directly into the data at the bit level. The systems and methods of the present invention enable enterprise communities of interest to leverage a common enterprise infrastructure. Because security is already woven into the data, this common infrastructure can be used without compromising data security and access control. In some applications, data is authenticated, encrypted, and parsed or split into multiple shares prior to being sent to multiple locations, e.g., a private or public cloud. The data is hidden while in transit to the storage location, and is inaccessible to users who do not have the correct credentials for access.

WO 2011/123692 A2

SYSTEMS AND METHODS FOR SECURING DATA IN MOTION

Cross-Reference to Related Applications

[0001] This application claims the benefit of U.S. Provisional Patent Application Serial No. 61/319,658, filed March 31, 2010, and U.S. Provisional Patent Application Serial No. 61/320,242, filed April 1, 2010. The contents of each of these provisional patent applications are hereby incorporated by reference herein in their entireties.

Field of the Invention

[0002] The present invention relates in general to systems and methods for securing data in motion. The systems and methods described herein may be used in conjunction with other systems and methods described in commonly-owned U.S. Patent No. 7,391,865 and commonly-owned U.S. Patent Application Nos. 11/258,839, filed October 25, 2005, 11/602,667, filed November 20, 2006, 11/983,355, filed November 7, 2007, 11/999,575, filed December 5, 2007, 12/148,365, filed April 18, 2008, 12/209,703, filed September 12, 2008, 12/349,897, filed January 7, 2009, 12/391,028, filed February 23, 2009, 12/783,276, filed May 19, 2010, 12/953,877, filed November 24, 2010, and U.S. Provisional Patent Application Nos. 61/436,991, filed January 27, 2011, 61/264,464, filed November 25, 2009, 61/319,658, filed March 31, 2010, 61/320,242, filed April 1, 2010, 61/349,560, filed May 28, 2010, 61/373,187, filed August 12, 2010, 61/374,950, filed August 18, 2010, and 61/384,583, filed September 20, 2010. The disclosures of each of the aforementioned, earlier-filed applications are hereby incorporated by reference herein in their entireties.

Summary

[0003] The need to collaborate requires an enterprise to share its data. This requirement to share is complicated by legacy, stove pipe architectures that are expensive to maintain and do not scale. These complex infrastructures are made more restrictive by risk mitigation and disaster recovery requirements and policies. Furthermore, these restrictions lead directly to poor resource

utilization, costly point products, and inconsistent information sharing. The key driver of the legacy stove pipe environment was the need to protect the confidentiality, availability and integrity of the data. As this environment evolved over time, information sharing and collaboration have been limited because of the growth in security concerns and vulnerabilities. Over time, these legacy environments have required numerous ad-hoc security fixes that further restrict information sharing and collaboration. These fixes, however, have not addressed the root cause of the trade off between data availability and data security.

[0004] Existing Information Assurance (IA) solutions are complex, hard to scale and reactive to security vulnerabilities. By design, these solutions cannot provide *both* data security and availability. Disaster recovery plans based on such IA solutions are often inferior, are rarely implemented effectively, are expensive to maintain, and are difficult to scale as the amount of data that is shared by the enterprise grows.

[0005] Certain data security solutions, e.g., VPN and token-based infrastructures are expensive and involve significant challenges to both deploy and maintain. Furthermore, some products merely address targeted security issues, but they are inefficient, costly, cumbersome and complex to manage. Furthermore, these solutions do not provide an end-to-end solution to the fundamental problem: secure connectivity and data transfer for remotely located end users that are increasingly accessing data in the cloud using one or more mobile devices.

[0006] Expanding numbers of remote users who have migrated to using cloud storage ("cloud") have also created an escalating problem for data security – both in the cloud and when transporting data to or from the cloud. In particular, such cloud storage may be public, private, secure, or any combination thereof. Furthermore, the cloud storage may be provided by more than one storage provider. When new, sophisticated data security threats are encountered, these threats are significant to individual users and the enterprise alike. Remote users need the flexibility to collaborate with others using the cloud as a medium for storage, but need to do so without exposing the data to security risks.

[0007] Therefore, there is a need to both secure enterprise data and simultaneously provide access to it. Additionally, there is a need to provide this secure access without service disruption and without regard to a user's location. In fact, there is a need for an end-to-end solution that is easy to deploy, requires no user intervention, eliminates the need for additional hardware, is highly secure, and does not compromise productivity. In fact, a need exists to provide a

cryptographic system whose security is user-location independent while still supporting the security of data while it is in motion or transit from one location to another.

[0008] According, one aspect of the present invention is to provide a server-based (e.g., Bitfiler from Security First Corp.) secure data solution that makes data provably secure and accessible, while eliminating the need for multiple perimeter hardware and software technologies. The server-based solution addresses security at the bit level. In other words, data security is incorporated or weaved directly into the data at the bit level. In some embodiments, the server-based solution may be a software application that runs on either a Windows or a Linux platform. In some embodiments, by operating at the kernel level, large improvements in performance and ease of use are achieved. In some embodiments, the server-based solution enables enterprise Communities of Interest (COI) to be established that can leverage a common enterprise infrastructure, both in terms of hardware and software. Because security is already woven into the data, this common infrastructure can be used without compromising data security and access control. Multiple COIs can co-exist within the same infrastructure and within a single secure storage system. With the server-based solution, no forensically discernible data is stored on any device or media. The server-based solution may integrate with existing enterprise access control systems, allowing simplified deployment without modification of current, established access solutions.

[0009] In another aspect, the server-based solution of the present invention is hardware and software independent. The server-based solution applies to existing enterprise network, storage and security solutions. The server-based solution also applies to any collaboration, CRM and ERP applications. The built-in security provided by the server-based solution enables the use of emerging, cost effective technologies and services such as the infrastructure for cloud-based storage, could-based computing and cloud-based applications.

[0010] The server-based solution of the present invention may leverage Security First Corp.'s SecureParser Eχtended™ (SPχ) core technology. In some embodiments, the SecureParser SPχ utilizes a multi-factor secret sharing algorithm to deliver defense-level security. Data is authenticated, encrypted (FIPS 140-2 certified, Suite B compliant), split, redundant bits added, integrity checked and encrypted again, before being sent to multiple locations (locally and/or geographically dispersed, e.g., in a private or public cloud). Data may be split using any suitable

information dispersal algorithm (IDA). The data is hidden while in transit to the storage location, and is inaccessible to users who do not have the correct credentials for access.

[0011] Another aspect of the invention includes a method of rebuilding a first subset of a set of data shares of data to be secured and stored in a storage network. The method comprises retrieving a second subset of data shares from the secure storage network. This second subset of data shares is sufficient to reconstruct the data. The method further comprises authenticating the second subset of data shares, and rebuilding the encrypted data corresponding to the set of data shares using the first subset of data shares. The method further includes regenerating the set of data shares by splitting the encrypted data and reauthenticating the regenerated data shares. The method further includes storing at least the first subset of regenerated data shares in the storage network.

[0012] In some embodiments, splitting includes the use of an information dispersal algorithm. In some embodiments, authenticating includes the use of an authentication key. In some embodiments, the regenerating of the set of data shares includes the use of a splitting key. In some embodiments, the storage network includes one of a private cloud, a public cloud, a hybrid cloud, a removable storage device, and a mass storage device. In some embodiments, the headers correspond to less than all of the headers in the shares.

[0013] In some embodiments, the reauthenticating includes the use of a second authentication key that is different from a first authentication key used for the authenticating. In some embodiments, this process comprises retrieving headers associated with the second subset of data shares from the secure storage network, extracting a key encryption key from the retrieved headers, encrypting the second authentication key with the key encryption key, and, storing the encrypted second authentication key within the headers of the regenerated data shares in the storage network.

[0014] In some embodiments, the regenerating includes the use of a second split key different from a first split key that was used to generated the set of data shares. In some embodiments, this process comprises, retrieving headers associated with the second subset of data shares from the secure storage network, extracting a key encryption key from the retrieved headers, encrypting the second split key with the key encryption key, and storing the encrypted second split key within the headers of the regenerated data shares in the storage network.

[0015] In another aspect, the invention relates to a system for securing data. The system includes a memory for storing data, a main processor coupled to the memory configured to perform at least one of data splitting and data encrypting, and a coprocessor coupled to the main processor and the memory. The coprocessor is configured to perform dedicated secure parsing functions including at least one of encrypting the parsed data, or decrypting encrypted data. In some embodiments, the system includes a field programmable gate array coupled to the coprocessor. The FPGA performs at least one of encrypting the parsed data, or decrypting encrypted data. In some embodiments, the coprocessor is coupled to the main processor via a PCIe bus. In some embodiments, the coprocessor is coupled to the main processor via an HT bus.

[0016] In another aspect, the invention relates to a method for securing data using a portable device. The method includes the steps of generating at least two portions of data from a set of data based at least in part on a cryptographic key and storing the key on the portable device. The two portions of data and the key are sufficient to reconstruct the set of data. In some embodiments, the portable device is a removable storage device. In some embodiments, the removable storage device couples to an end user device via a Universal Serial Bus (USB) interface. In some embodiments, the method further includes storing the at least two portions of data on the portable device.

[0017] In another aspect, the invention relates to a method for securing data using a portable device. The method includes the steps of generating at least two portions of data from a set of data based at least in part on a cryptographic key and storing at least a portion of the generated data portions on the portable device. The two portions of data and the key are sufficient to reconstruct the set of data. In some embodiments, the portable device is a removable storage device. In some embodiments, the removable storage device couples to an end user device via a Universal Serial Bus (USB) interface. In some embodiments, the method further includes storing the key on the portable device.

[0018] In some embodiments, one or more cryptographic keys may be stored on a user device such as a USB memory device. These cryptographic keys may be used to encrypt or decrypt data stored on the end user device itself, or elsewhere, e.g., in a public or private cloud storage. For example, a user may store a cryptographic key on a USB memory device and use this key to decrypt encrypted shares of data stored remotely in a public cloud provided by Dropbox.

[0019] In some embodiments, to allow for data viewing and/or reconstruction at each of a plurality of distinct end-user devices, one or more cryptographic keys and/or one or more data shares may be stored on a portable user device such as a USB memory device. In addition, one or more of the data shares may also be stored on a cloud storage device. Thus, a user in possession of the portable user device may access the portable user device from a different end user device to view and/or rebuild the data from the shares dispersed across the portable user device and if required, the cloud storage device.

[0020] In another aspect, the invention relates to a method for rebuilding a set of data shares that were generated from an encrypted data set by an information dispersal algorithm using a first split key. The method includes receiving at least a minimum number of data shares necessary for rebuilding the set of data shares and rebuilding the set of data shares from the minimum number of data shares without decrypting the minimum number of data shares. In some embodiments, the rebuilding is performed in response to a determination that the set of data shares has been compromised. In some embodiments, the minimum number of data shares are associated with a first authentication key, and the rebuilding comprises associating the rebuilt set of data shares with a second authentication key.

[0021] In some embodiments, the method further includes the steps of retrieving headers associated with the minimum number of data shares, extracting a key encryption key from the retrieved headers, encrypting the second authentication key with the key encryption key, and restoring the encrypted second authentication key within the headers of the rebuilt data shares. In some embodiments, the minimum number of data shares are rebuilt using a second split key that is different from the first split key. In some embodiments, the method further includes retrieving headers associated with the minimum number of data shares, extracting a key encryption key from the retrieved headers, encrypting the second split key with the key encryption key, and restoring the encrypted second split key within the headers of the rebuilt data shares. In some embodiments, the method includes the step of storing at least one of the rebuilt data shares on a storage network. In some embodiments, the storage network includes one of a private cloud, a public cloud, a hybrid cloud, a removable storage device, and a mass storage device.

[0022] In another aspect, the invention relates to a method for associating stubs with a set of data shares on the file system of a storage network. The method includes generating the set of

data shares from an encrypted data set by an information dispersal algorithm, and generating a set of stubs associated with the generated data shares. Each stub corresponds to a respective data share, and each stub includes information associated with the respective data share. The set of stubs are stored in a location on the storage network. The information includes one of the name of the respective data share, a date the respective data share was created, a time at which the respective data share was last modified, a pointer to the location of the respective data share within the file system. The storage network includes one or more storage devices associated with one of a private cloud, a public cloud, a hybrid cloud, a removable storage device, and a mass storage device. In some embodiments, the method further includes the steps of receiving a command to view the information associated with the generated data shares, retrieving the stubs from the location on the storage network, extracting the information from the stubs to create a file system of data shares, and displaying the file system of data shares. In some embodiments, the stubs are stored within the headers of the generated data shares, and the retrieving includes retrieving the headers of the generated data shares. In some embodiments, less than all of the headers are retrieved. In some embodiments, the stubs are stored in a stub directory, and retrieving includes retrieving the stubs from the stub directory. In some embodiments, the method further includes receiving an indication of a virtual or physical directory in which to store the stubs. In some embodiments, the indication is received from a user.

Brief Description of the Drawings

[0023] The present invention is described in more detail below in connection with the attached drawings, which are meant to illustrate and not to limit the invention, and in which:

[0024] FIGURE 1 illustrates a block diagram of a cryptographic system, according to aspects of an embodiment of the invention;

[0025] FIGURE 2 illustrates a block diagram of the trust engine of FIGURE 1, according to aspects of an embodiment of the invention;

[0026] FIGURE 3 illustrates a block diagram of the transaction engine of FIGURE 2, according to aspects of an embodiment of the invention;

[0027] FIGURE 4 illustrates a block diagram of the depository of FIGURE 2, according to aspects of an embodiment of the invention;

[0028] FIGURE 5 illustrates a block diagram of the authentication engine of FIGURE 2, according to aspects of an embodiment of the invention;

[0029] FIGURE 6 illustrates a block diagram of the cryptographic engine of FIGURE 2, according to aspects of an embodiment of the invention;

[0030] FIGURE 7 illustrates a block diagram of a depository system, according to aspects of another embodiment of the invention;

[0031] FIGURE 8 illustrates a flow chart of a data splitting process according to aspects of an embodiment of the invention;

[0032] FIGURE 9, Panel A illustrates a data flow of an enrollment process according to aspects of an embodiment of the invention;

[0033] FIGURE 9, Panel B illustrates a flow chart of an interoperability process according to aspects of an embodiment of the invention;

[0034] FIGURE 10 illustrates a data flow of an authentication process according to aspects of an embodiment of the invention;

[0035] FIGURE 11 illustrates a data flow of a signing process according to aspects of an embodiment of the invention;

[0036] FIGURE 12 illustrates a data flow and an encryption/decryption process according to aspects and yet another embodiment of the invention;

[0037] FIGURE 13 illustrates a simplified block diagram of a trust engine system according to aspects of another embodiment of the invention;

[0038] FIGURE 14 illustrates a simplified block diagram of a trust engine system according to aspects of another embodiment of the invention;

[0039] FIGURE 15 illustrates a block diagram of the redundancy module of FIGURE 14, according to aspects of an embodiment of the invention;

[0040] FIGURE 16 illustrates a process for evaluating authentications according to one aspect of the invention;

[0041] FIGURE 17 illustrates a process for assigning a value to an authentication according to one aspect as shown in FIGURE 16 of the invention;

[0042] FIGURE 18 illustrates a process for performing trust arbitrage in an aspect of the invention as shown in FIGURE 17; and

[0043] FIGURE 19 illustrates a sample transaction between a user and a vendor according to aspects of an embodiment of the invention where an initial web based contact leads to a sales contract signed by both parties.

[0044] FIGURE 20 illustrates a sample user system with a cryptographic service provider module which provides security functions to a user system.

[0045] FIGURE 21 illustrates a process for parsing, splitting and/or separating data with encryption and storage of the encryption master key with the data.

[0046] FIGURE 22 illustrates a process for parsing, splitting and/or separating data with encryption and storing the encryption master key separately from the data.

[0047] FIGURE 23 illustrates the intermediary key process for parsing, splitting and/or separating data with encryption and storage of the encryption master key with the data.

[0048] FIGURE 24 illustrates the intermediary key process for parsing, splitting and/or separating data with encryption and storing the encryption master key separately from the data.

[0049] FIGURE 25 illustrates utilization of the cryptographic methods and systems of the present invention with a small working group.

[0050] FIGURE 26 is a block diagram of an illustrative physical token security system employing the secure data parser in accordance with one embodiment of the present invention.

[0051] FIGURE 27 is a block diagram of an illustrative arrangement in which the secure data parser is integrated into a system in accordance with one embodiment of the present invention.

[0052] FIGURE 28 is a block diagram of an illustrative data in motion system in accordance with one embodiment of the present invention.

[0053] FIGURE 29 is a block diagram of another illustrative data in motion system in accordance with one embodiment of the present invention.

[0054] FIGURE 30-32 are block diagrams of an illustrative system having the secure data parser integrated in accordance with one embodiment of the present invention.

[0055] FIGURE 33 is a process flow diagram of an illustrative process for parsing and splitting data in accordance with one embodiment of the present invention.

[0056] FIGURE 34 is a process flow diagram of an illustrative process for restoring portions of data into original data in accordance with one embodiment of the present invention.

[0057] FIGURE 35 is a process flow diagram of an illustrative process for splitting data at the bit level in accordance with one embodiment of the present invention.

[0058] FIGURE 36 is a process flow diagram of illustrative steps and features, that may be used in any suitable combination, with any suitable additions, deletions, or modifications in accordance with one embodiment of the present invention.

[0059] FIGURE 37 is a process flow diagram of illustrative steps and features that may be used in any suitable combination, with any suitable additions, deletions, or modifications in accordance with one embodiment of the present invention.

[0060] FIGURE 38 is a simplified block diagram of the storage of key and data components within shares, that may be used in any suitable combination, with any suitable additions, deletions, or modifications in accordance with one embodiment of the present invention.

[0061] FIGURE 39 is a simplified block diagram of the storage of key and data components within shares using a workgroup key, that may be used in any suitable combination, with any suitable additions, deletions, or modifications in accordance with one embodiment of the present invention.

[0062] FIGURES 40A and 40B are simplified and illustrative process flow diagrams for header generation and data splitting for data in motion, that may be used in any suitable combination, with any suitable additions, deletions, or modifications in accordance with one embodiment of the present invention.

[0063] FIGURE 41 is a simplified block diagram of an illustrative share format, that may be used in any suitable combination, with any suitable additions, deletions, or modifications in accordance with one embodiment of the present invention.

[0064] FIGURE 42 is a block diagram of an illustrative arrangement in which the secure data parser is integrated into a system connected to cloud computing resources in accordance with one embodiment of the present invention.

[0065] FIGURE 43 is a block diagram of an illustrative arrangement in which the secure data parser is integrated into a system for sending data through the cloud in accordance with one embodiment of the present invention.

[0066] FIGURE 44 is a block diagram of an illustrative arrangement in which the secure data parser is used to secure data services in the cloud in accordance with one embodiment of the present invention.

[0067] FIGURE 45 is a block diagram of an illustrative arrangement in which the secure data parser is used to secure data storage in the cloud in accordance with one embodiment of the present invention.

[0068] FIGURE 46 is a block diagram of an illustrative arrangement in which the secure data parser is used to secure network access control in accordance with one embodiment of the present invention.

[0069] FIGURE 47 is a block diagram of an illustrative arrangement in which the secure data parser is used to secure high performance computing resources in accordance with one embodiment of the present invention.

[0070] FIGURE 48 is a schematic of an illustrative arrangement in which the secure data parser is used to secure data storage in a plurality of storage devices in a cloud in accordance with one embodiment of the present invention.

[0071] FIGURE 49 is a schematic of an illustrative arrangement in which the secure data parser is used to secure data storage in a plurality of private and public clouds in accordance with one embodiment of the present invention.

[0072] FIGURE 50 is a schematic of an illustrative arrangement in which the secure data parser is used to secure data storage in a plurality of private and public clouds via a public Internet in accordance with one embodiment of the present invention.

[0073] FIGURE 51 is a schematic of an illustrative arrangement in which the secure data parser is used to secure data storage in a user's removable storage device in accordance with one embodiment of the present invention.

[0074] FIGURE 52 is a schematic of an illustrative arrangement in which the secure data parser is used to secure data storage in a plurality of user storage devices in accordance with one embodiment of the present invention.

[0075] FIGURE 53 is a schematic of an illustrative arrangement in which the secure data parser is used to secure data storage in a plurality of public and private clouds and at least one user storage device in accordance with one embodiment of the present invention.

[0076] FIGURE 54 is a schematic of a co-processor acceleration device for the secure data parser in accordance with one embodiment of the present invention.

[0077] FIGURE 55 is a first process flow diagram of an illustrative acceleration process using the co-processor acceleration device of FIGURE 54 for the secure data parser in accordance with one embodiment of the present invention.

[0078] FIGURE 56 is a second process flow diagram of an illustrative acceleration process using the co-processor acceleration device of FIGURE 54 for the secure data parser in accordance with one embodiment of the present invention.

[0079] FIGURE 57 illustrates a process by which data is split into N shares and stored, according to an illustrative embodiment of the present invention.

[0080] FIGURE 58 illustrates a process by which shares of data are rebuilt and/or re-keyed, according to an illustrative embodiment of the present invention.

Detailed Description of the Illustrative Embodiments

[0081] One aspect of the present invention is to provide a cryptographic system where one or more secure servers, or a trust engine, stores cryptographic keys and user authentication data. The system may store data across one or more storage devices in a cloud. The cloud may include private storage devices (accessible only to a particular set of users) or public storage devices (accessible to any set of users that subscribes to the storage provider).

[0082] Users access the functionality of conventional cryptographic systems through network access to the trust engine, however, the trust engine does not release actual keys and other authentication data and therefore, the keys and data remain secure. This server-centric storage of keys and authentication data provides for user-independent security, portability, availability, and straightforwardness.

[0083] Because users can be confident in, or trust, the cryptographic system to perform user and document authentication and other cryptographic functions, a wide variety of functionality may be incorporated into the system. For example, the trust engine provider can ensure against agreement repudiation by, for example, authenticating the agreement participants, digitally signing the agreement on behalf of or for the participants, and storing a record of the agreement digitally signed by each participant. In addition, the cryptographic system may monitor agreements and determine to apply varying degrees of authentication, based on, for example, price, user, vendor, geographic location, place of use, or the like.

[0084] To facilitate a complete understanding of the invention, the remainder of the detailed description describes the invention with reference to the figures, wherein like elements are referenced with like numerals throughout.

[0085] FIGURE 1 illustrates a block diagram of a cryptographic system 100, according to aspects of an embodiment of the invention. As shown in FIGURE 1, the cryptographic system 100 includes a user system 105, a trust engine 110, a certificate authority 115, and a vendor system 120, communicating through a communication link 125.

[0086] According to one embodiment of the invention, the user system 105 comprises a conventional general-purpose computer having one or more microprocessors, such as, for example, an Intel-based processor. Moreover, the user system 105 includes an appropriate operating system, such as, for example, an operating system capable of including graphics or windows, such as Windows, Unix, Linux, or the like. As shown in FIGURE 1, the user system 105 may include a biometric device 107. The biometric device 107 may advantageously capture a user's biometric and transfer the captured biometric to the trust engine 110. According to one embodiment of the invention, the biometric device may advantageously comprise a device having attributes and features similar to those disclosed in U.S. Patent Application No. 08/926,277, filed on September 5, 1997, entitled *"RELIEF OBJECT IMAGE GENERATOR,"* U.S. Patent Application No. 09/558,634, filed on April 26, 2000, entitled *"IMAGING DEVICE FOR A RELIEF OBJECT AND SYSTEM AND METHOD OF USING THE IMAGE DEVICE,"* U.S. Patent Application No. 09/435,011, filed on November 5, 1999, entitled *"RELIEF OBJECT SENSOR ADAPTOR,"* and U.S. Patent Application No. 09/477,943, filed on January 5, 2000, entitled *"PLANAR OPTICAL IMAGE SENSOR AND SYSTEM FOR GENERATING AN ELECTRONIC IMAGE OF A RELIEF OBJECT FOR FINGERPRINT READING,"* all of which are owned by the instant assignee, and all of which are hereby incorporated by reference herein.

[0087] In addition, the user system 105 may connect to the communication link 125 through a conventional service provider, such as, for example, a dial up, digital subscriber line (DSL), cable modem, fiber connection, or the like. According to another embodiment, the user system 105 connects the communication link 125 through network connectivity such as, for example, a local or wide area network. According to one embodiment, the operating system includes a TCP/IP stack that handles all incoming and outgoing message traffic passed over the communication link 125.

[0088] Although the user system 105 is disclosed with reference to the foregoing embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize from the disclosure herein, a wide number of alternatives embodiments of the user system 105, including almost any computing device capable of sending or receiving information from another computer system. For example, the user system 105 may include, but is not limited to, a computer workstation, an interactive television, an interactive kiosk, a personal mobile computing device, such as a digital assistant, mobile phone, laptop, or the like, a wireless communications device, a smartcard, an embedded computing device, or the like, which can interact with the communication link 125. In such alternative systems, the operating systems will likely differ and be adapted for the particular device. However, according to one embodiment, the operating systems advantageously continue to provide the appropriate communications protocols needed to establish communication with the communication link 125.

[0089] FIGURE 1 illustrates the trust engine 110. According to one embodiment, the trust engine 110 comprises one or more secure servers for accessing and storing sensitive information, which may be any type or form of data, such as, but not limited to text, audio, video, user authentication data and public and private cryptographic keys. According to one embodiment, the authentication data includes data designed to uniquely identify a user of the cryptographic system 100. For example, the authentication data may include a user identification number, one or more biometrics, and a series of questions and answers generated by the trust engine 110 or the user, but answered initially by the user at enrollment. The foregoing questions may include demographic data, such as place of birth, address, anniversary, or the like, personal data, such as mother's maiden name, favorite ice cream, or the like, or other data designed to uniquely identify the user. The trust engine 110 compares a user's authentication data associated with a current transaction, to the authentication data provided at an earlier time, such as, for example, during enrollment. The trust engine 110 may advantageously require the user to produce the authentication data at the time of each transaction, or, the trust engine 110 may advantageously allow the user to periodically produce authentication data, such as at the beginning of a string of transactions or the logging onto a particular vendor website.

[0090] According to the embodiment where the user produces biometric data, the user provides a physical characteristic, such as, but not limited to, facial scan, hand scan, ear scan, iris scan, retinal scan, vascular pattern, DNA, a fingerprint, writing or speech, to the biometric device 107.

The biometric device advantageously produces an electronic pattern, or biometric, of the physical characteristic. The electronic pattern is transferred through the user system 105 to the trust engine 110 for either enrollment or authentication purposes.

[0091] Once the user produces the appropriate authentication data and the trust engine 110 determines a positive match between that authentication data (current authentication data) and the authentication data provided at the time of enrollment (enrollment authentication data), the trust engine 110 provides the user with complete cryptographic functionality. For example, the properly authenticated user may advantageously employ the trust engine 110 to perform hashing, digitally signing, encrypting and decrypting (often together referred to only as encrypting), creating or distributing digital certificates, and the like. However, the private cryptographic keys used in the cryptographic functions will not be available outside the trust engine 110, thereby ensuring the integrity of the cryptographic keys.

[0092] According to one embodiment, the trust engine 110 generates and stores cryptographic keys. According to another embodiment, at least one cryptographic key is associated with each user. Moreover, when the cryptographic keys include public-key technology, each private key associated with a user is generated within, and not released from, the trust engine 110. Thus, so long as the user has access to the trust engine 110, the user may perform cryptographic functions using his or her private or public key. Such remote access advantageously allows users to remain completely mobile and access cryptographic functionality through practically any Internet connection, such as cellular and satellite phones, kiosks, laptops, hotel rooms and the like.

[0093] According to another embodiment, the trust engine 110 performs the cryptographic functionality using a key pair generated for the trust engine 110. According to this embodiment, the trust engine 110 first authenticates the user, and after the user has properly produced authentication data matching the enrollment authentication data, the trust engine 110 uses its own cryptographic key pair to perform cryptographic functions on behalf of the authenticated user.

[0094] A skilled artisan will recognize from the disclosure herein that the cryptographic keys may advantageously include some or all of symmetric keys, public keys, and private keys. In addition, a skilled artisan will recognize from the disclosure herein that the foregoing keys may be implemented with a wide number of algorithms available from commercial technologies, such as, for example, RSA, ELGAMAL, or the like.

[0095] FIGURE 1 also illustrates the certificate authority 115. According to one embodiment, the certificate authority 115 may advantageously comprise a trusted third-party organization or company that issues digital certificates, such as, for example, VeriSign, Baltimore, Entrust, or the like. The trust engine 110 may advantageously transmit requests for digital certificates, through one or more conventional digital certificate protocols, such as, for example, PKCS10, to the certificate authority 115. In response, the certificate authority 115 will issue a digital certificate in one or more of a number of differing protocols, such as, for example, PKCS7. According to one embodiment of the invention, the trust engine 110 requests digital certificates from several or all of the prominent certificate authorities 115 such that the trust engine 110 has access to a digital certificate corresponding to the certificate standard of any requesting party.

[0096] According to another embodiment, the trust engine 110 internally performs certificate issuances. In this embodiment, the trust engine 110 may access a certificate system for generating certificates and/or may internally generate certificates when they are requested, such as, for example, at the time of key generation or in the certificate standard requested at the time of the request. The trust engine 110 will be disclosed in greater detail below.

[0097] FIGURE 1 also illustrates the vendor system 120. According to one embodiment, the vendor system 120 advantageously comprises a Web server. Typical Web servers may serve content over the Internet using one of several internet markup languages or document format standards, such as the Hyper-Text Markup Language (HTML) or the Extensible Markup Language (XML). The Web server accepts requests from browsers like Netscape and Internet Explorer and then returns the appropriate electronic documents. A number of server or client-side technologies can be used to increase the power of the Web server beyond its ability to deliver standard electronic documents. For example, these technologies include Common Gateway Interface (CGI) scripts, Secure Sockets Layer (SSL) security, and Active Server Pages (ASPs). The vendor system 120 may advantageously provide electronic content relating to commercial, personal, educational, or other transactions.

[0098] Although the vendor system 120 is disclosed with reference to the foregoing embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize from the disclosure herein that the vendor system 120 may advantageously comprise any of the devices described with reference to the user system 105 or combination thereof.

[0099] FIGURE 1 also illustrates the communication link 125 connecting the user system 105, the trust engine 110, the certificate authority 115, and the vendor system 120. According to one embodiment, the communication link 125 preferably comprises the Internet. The Internet, as used throughout this disclosure is a global network of computers. The structure of the Internet, which is well known to those of ordinary skill in the art, includes a network backbone with networks branching from the backbone. These branches, in turn, have networks branching from them, and so on. Routers move information packets between network levels, and then from network to network, until the packet reaches the neighborhood of its destination. From the destination, the destination network's host directs the information packet to the appropriate terminal, or node. In one advantageous embodiment, the Internet routing hubs comprise domain name system (DNS) servers using Transmission Control Protocol/Internet Protocol (TCP/IP) as is well known in the art. The routing hubs connect to one or more other routing hubs via high-speed communication links.

[0100] One popular part of the Internet is the World Wide Web. The World Wide Web contains different computers, which store documents capable of displaying graphical and textual information. The computers that provide information on the World Wide Web are typically called "websites." A website is defined by an Internet address that has an associated electronic page. The electronic page can be identified by a Uniform Resource Locator (URL). Generally, an electronic page is a document that organizes the presentation of text, graphical images, audio, video, and so forth.

[0101] Although the communication link 125 is disclosed in terms of its preferred embodiment, one of ordinary skill in the art will recognize from the disclosure herein that the communication link 125 may include a wide range of interactive communications links. For example, the communication link 125 may include interactive television networks, telephone networks, wireless data transmission systems, two-way cable systems, customized private or public computer networks, interactive kiosk networks, automatic teller machine networks, direct links, satellite or cellular networks, and the like.

[0102] FIGURE 2 illustrates a block diagram of the trust engine 110 of FIGURE 1 according to aspects of an embodiment of the invention. As shown in FIGURE 2, the trust engine 110 includes a transaction engine 205, a depository 210, an authentication engine 215, and a cryptographic engine 220. According to one embodiment of the invention, the trust engine 110

also includes mass storage 225. As further shown in FIGURE 2, the transaction engine 205 communicates with the depository 210, the authentication engine 215, and the cryptographic engine 220, along with the mass storage 225. In addition, the depository 210 communicates with the authentication engine 215, the cryptographic engine 220, and the mass storage 225. Moreover, the authentication engine 215 communicates with the cryptographic engine 220. According to one embodiment of the invention, some or all of the foregoing communications may advantageously comprise the transmission of XML documents to IP addresses that correspond to the receiving device. As mentioned in the foregoing, XML documents advantageously allow designers to create their own customized document tags, enabling the definition, transmission, validation, and interpretation of data between applications and between organizations. Moreover, some or all of the foregoing communications may include conventional SSL technologies.

[0103] According to one embodiment, the transaction engine 205 comprises a data routing device, such as a conventional Web server available from Netscape, Microsoft, Apache, or the like. For example, the Web server may advantageously receive incoming data from the communication link 125. According to one embodiment of the invention, the incoming data is addressed to a front-end security system for the trust engine 110. For example, the front-end security system may advantageously include a firewall, an intrusion detection system searching for known attack profiles, and/or a virus scanner. After clearing the front-end security system, the data is received by the transaction engine 205 and routed to one of the depository 210, the authentication engine 215, the cryptographic engine 220, and the mass storage 225. In addition, the transaction engine 205 monitors incoming data from the authentication engine 215 and cryptographic engine 220, and routes the data to particular systems through the communication link 125. For example, the transaction engine 205 may advantageously route data to the user system 105, the certificate authority 115, or the vendor system 120.

[0104] According to one embodiment, the data is routed using conventional HTTP routing techniques, such as, for example, employing URLs or Uniform Resource Indicators (URIs). URIs are similar to URLs, however, URIs typically indicate the source of files or actions, such as, for example, executables, scripts, and the like. Therefore, according to the one embodiment, the user system 105, the certificate authority 115, the vendor system 120, and the components of

the trust engine 210, advantageously include sufficient data within communication URLs or URIs for the transaction engine 205 to properly route data throughout the cryptographic system.

[0105] Although the data routing is disclosed with reference to its preferred embodiment, a skilled artisan will recognize a wide number of possible data routing solutions or strategies. For example, XML or other data packets may advantageously be unpacked and recognized by their format, content, or the like, such that the transaction engine 205 may properly route data throughout the trust engine 110. Moreover, a skilled artisan will recognize that the data routing may advantageously be adapted to the data transfer protocols conforming to particular network systems, such as, for example, when the communication link 125 comprises a local network.

[0106] According to yet another embodiment of the invention, the transaction engine 205 includes conventional SSL encryption technologies, such that the foregoing systems may authenticate themselves, and vice-versa, with transaction engine 205, during particular communications. As will be used throughout this disclosure, the term "½ SSL" refers to communications where a server but not necessarily the client, is SSL authenticated, and the term "FULL SSL" refers to communications where the client and the server are SSL authenticated. When the instant disclosure uses the term "SSL", the communication may comprise ½ or FULL SSL.

[0107] As the transaction engine 205 routes data to the various components of the cryptographic system 100, the transaction engine 205 may advantageously create an audit trail. According to one embodiment, the audit trail includes a record of at least the type and format of data routed by the transaction engine 205 throughout the cryptographic system 100. Such audit data may advantageously be stored in the mass storage 225.

[0108] FIGURE 2 also illustrates the depository 210. According to one embodiment, the depository 210 comprises one or more data storage facilities, such as, for example, a directory server, a database server, or the like. As shown in FIGURE 2, the depository 210 stores cryptographic keys and enrollment authentication data. The cryptographic keys may advantageously correspond to the trust engine 110 or to users of the cryptographic system 100, such as the user or vendor. The enrollment authentication data may advantageously include data designed to uniquely identify a user, such as, user ID, passwords, answers to questions, biometric data, or the like. This enrollment authentication data may advantageously be acquired at

enrollment of a user or another alternative later time. For example, the trust engine 110 may include periodic or other renewal or reissue of enrollment authentication data.

[0109] According to one embodiment, the communication from the transaction engine 205 to and from the authentication engine 215 and the cryptographic engine 220 comprises secure communication, such as, for example conventional SSL technology. In addition, as mentioned in the foregoing, the data of the communications to and from the depository 210 may be transferred using URLs, URIs, HTTP or XML documents, with any of the foregoing advantageously having data requests and formats embedded therein.

[0110] As mentioned above, the depository 210 may advantageously comprises a plurality of secure data storage facilities. In such an embodiment, the secure data storage facilities may be configured such that a compromise of the security in one individual data storage facility will not compromise the cryptographic keys or the authentication data stored therein. For example, according to this embodiment, the cryptographic keys and the authentication data are mathematically operated on so as to statistically and substantially randomize the data stored in each data storage facility. According to one embodiment, the randomization of the data of an individual data storage facility renders that data undecipherable. Thus, compromise of an individual data storage facility produces only a randomized undecipherable number and does not compromise the security of any cryptographic keys or the authentication data as a whole.

[0111] FIGURE 2 also illustrates the trust engine 110 including the authentication engine 215. According to one embodiment, the authentication engine 215 comprises a data comparator configured to compare data from the transaction engine 205 with data from the depository 210. For example, during authentication, a user supplies current authentication data to the trust engine 110 such that the transaction engine 205 receives the current authentication data. As mentioned in the foregoing, the transaction engine 205 recognizes the data requests, preferably in the URL or URI, and routes the authentication data to the authentication engine 215. Moreover, upon request, the depository 210 forwards enrollment authentication data corresponding to the user to the authentication engine 215. Thus, the authentication engine 215 has both the current authentication data and the enrollment authentication data for comparison.

[0112] According to one embodiment, the communications to the authentication engine comprise secure communications, such as, for example, SSL technology. Additionally, security can be provided within the trust engine 110 components, such as, for example, super-encryption

using public key technologies. For example, according to one embodiment, the user encrypts the current authentication data with the public key of the authentication engine 215. In addition, the depository 210 also encrypts the enrollment authentication data with the public key of the authentication engine 215. In this way, only the authentication engine's private key can be used to decrypt the transmissions.

[0113] As shown in FIGURE 2, the trust engine 110 also includes the cryptographic engine 220. According to one embodiment, the cryptographic engine comprises a cryptographic handling module, configured to advantageously provide conventional cryptographic functions, such as, for example, public-key infrastructure (PKI) functionality. For example, the cryptographic engine 220 may advantageously issue public and private keys for users of the cryptographic system 100. In this manner, the cryptographic keys are generated at the cryptographic engine 220 and forwarded to the depository 210 such that at least the private cryptographic keys are not available outside of the trust engine 110. According to another embodiment, the cryptographic engine 220 randomizes and splits at least the private cryptographic key data, thereby storing only the randomized split data. Similar to the splitting of the enrollment authentication data, the splitting process ensures the stored keys are not available outside the cryptographic engine 220. According to another embodiment, the functions of the cryptographic engine can be combined with and performed by the authentication engine 215.

[0114] According to one embodiment, communications to and from the cryptographic engine include secure communications, such as SSL technology. In addition, XML documents may advantageously be employed to transfer data and/or make cryptographic function requests.

[0115] FIGURE 2 also illustrates the trust engine 110 having the mass storage 225. As mentioned in the foregoing, the transaction engine 205 keeps data corresponding to an audit trail and stores such data in the mass storage 225. Similarly, according to one embodiment of the invention, the depository 210 keeps data corresponding to an audit trail and stores such data in the mass storage device 225. The depository audit trail data is similar to that of the transaction engine 205 in that the audit trail data comprises a record of the requests received by the depository 210 and the response thereof. In addition, the mass storage 225 may be used to store digital certificates having the public key of a user contained therein.

[0116] Although the trust engine 110 is disclosed with reference to its preferred and alternative embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will

recognize in the disclosure herein, a wide number of alternatives for the trust engine 110. For example, the trust engine 110, may advantageously perform only authentication, or alternatively, only some or all of the cryptographic functions, such as data encryption and decryption. According to such embodiments, one of the authentication engine 215 and the cryptographic engine 220 may advantageously be removed, thereby creating a more straightforward design for the trust engine 110. In addition, the cryptographic engine 220 may also communicate with a certificate authority such that the certificate authority is embodied within the trust engine 110. According to yet another embodiment, the trust engine 110 may advantageously perform authentication and one or more cryptographic functions, such as, for example, digital signing.

[0117] FIGURE 3 illustrates a block diagram of the transaction engine 205 of FIGURE 2, according to aspects of an embodiment of the invention. According to this embodiment, the transaction engine 205 comprises an operating system 305 having a handling thread and a listening thread. The operating system 305 may advantageously be similar to those found in conventional high volume servers, such as, for example, Web servers available from Apache. The listening thread monitors the incoming communication from one of the communication link 125, the authentication engine 215, and the cryptographic engine 220 for incoming data flow. The handling thread recognizes particular data structures of the incoming data flow, such as, for example, the foregoing data structures, thereby routing the incoming data to one of the communication link 125, the depository 210, the authentication engine 215, the cryptographic engine 220, or the mass storage 225. As shown in FIGURE 3, the incoming and outgoing data may advantageously be secured through, for example, SSL technology.

[0118] FIGURE 4 illustrates a block diagram of the depository 210 of FIGURE 2 according to aspects of an embodiment of the invention. According to this embodiment, the depository 210 comprises one or more lightweight directory access protocol (LDAP) servers. LDAP directory servers are available from a wide variety of manufacturers such as Netscape, ISO, and others. FIGURE 4 also shows that the directory server preferably stores data 405 corresponding to the cryptographic keys and data 410 corresponding to the enrollment authentication data. According to one embodiment, the depository 210 comprises a single logical memory structure indexing authentication data and cryptographic key data to a unique user ID. The single logical memory structure preferably includes mechanisms to ensure a high degree of trust, or security, in the data stored therein. For example, the physical location of the depository 210 may advantageously

include a wide number of conventional security measures, such as limited employee access, modern surveillance systems, and the like. In addition to, or in lieu of, the physical securities, the computer system or server may advantageously include software solutions to protect the stored data. For example, the depository 210 may advantageously create and store data 415 corresponding to an audit trail of actions taken. In addition, the incoming and outgoing communications may advantageously be encrypted with public key encryption coupled with conventional SSL technologies.

[0119] According to another embodiment, the depository 210 may comprise distinct and physically separated data storage facilities, as disclosed further with reference to FIGURE 7.

[0120] FIGURE 5 illustrates a block diagram of the authentication engine 215 of FIGURE 2 according to aspects of an embodiment of the invention. Similar to the transaction engine 205 of FIGURE 3, the authentication engine 215 comprises an operating system 505 having at least a listening and a handling thread of a modified version of a conventional Web server, such as, for example, Web servers available from Apache. As shown in FIGURE 5, the authentication engine 215 includes access to at least one private key 510. The private key 510 may advantageously be used for example, to decrypt data from the transaction engine 205 or the depository 210, which was encrypted with a corresponding public key of the authentication engine 215.

[0121] FIGURE 5 also illustrates the authentication engine 215 comprising a comparator 515, a data splitting module 520, and a data assembling module 525. According to the preferred embodiment of the invention, the comparator 515 includes technology capable of comparing potentially complex patterns related to the foregoing biometric authentication data. The technology may include hardware, software, or combined solutions for pattern comparisons, such as, for example, those representing finger print patterns or voice patterns. In addition, according to one embodiment, the comparator 515 of the authentication engine 215 may advantageously compare conventional hashes of documents in order to render a comparison result. According to one embodiment of the invention, the comparator 515 includes the application of heuristics 530 to the comparison. The heuristics 530 may advantageously address circumstances surrounding an authentication attempt, such as, for example, the time of day, IP address or subnet mask, purchasing profile, email address, processor serial number or ID, or the like.

[0122] Moreover, the nature of biometric data comparisons may result in varying degrees of confidence being produced from the matching of current biometric authentication data to enrollment data. For example, unlike a traditional password which may only return a positive or negative match, a fingerprint may be determined to be a partial match, e.g. a 90% match, a 75% match, or a 10% match, rather than simply being correct or incorrect. Other biometric identifiers such as voice print analysis or face recognition may share this property of probabilistic authentication, rather than absolute authentication.

[0123] When working with such probabilistic authentication or in other cases where an authentication is considered less than absolutely reliable, it is desirable to apply the heuristics 530 to determine whether the level of confidence in the authentication provided is sufficiently high to authenticate the transaction which is being made.

[0124] It will sometimes be the case that the transaction at issue is a relatively low value transaction where it is acceptable to be authenticated to a lower level of confidence. This could include a transaction which has a low dollar value associated with it (e.g., a $10 purchase) or a transaction with low risk (e.g., admission to a members-only web site).

[0125] Conversely, for authenticating other transactions, it may be desirable to require a high degree of confidence in the authentication before allowing the transaction to proceed. Such transactions may include transactions of large dollar value (e.g., signing a multi-million dollar supply contract) or transaction with a high risk if an improper authentication occurs (e.g., remotely logging onto a government computer).

[0126] The use of the heuristics 530 in combination with confidence levels and transactions values may be used as will be described below to allow the comparator to provide a dynamic context-sensitive authentication system.

[0127] According to another embodiment of the invention, the comparator 515 may advantageously track authentication attempts for a particular transaction. For example, when a transaction fails, the trust engine 110 may request the user to re-enter his or her current authentication data. The comparator 515 of the authentication engine 215 may advantageously employ an attempt limiter 535 to limit the number of authentication attempts, thereby prohibiting brute-force attempts to impersonate a user's authentication data. According to one embodiment, the attempt limiter 535 comprises a software module monitoring transactions for repeating authentication attempts and, for example, limiting the authentication attempts for a given

transaction to three. Thus, the attempt limiter 535 will limit an automated attempt to impersonate an individual's authentication data to, for example, simply three "guesses." Upon three failures, the attempt limiter 535 may advantageously deny additional authentication attempts. Such denial may advantageously be implemented through, for example, the comparator 515 returning a negative result regardless of the current authentication data being transmitted. On the other hand, the transaction engine 205 may advantageously block any additional authentication attempts pertaining to a transaction in which three attempts have previously failed.

[0128] The authentication engine 215 also includes the data splitting module 520 and the data assembling module 525. The data splitting module 520 advantageously comprises a software, hardware, or combination module having the ability to mathematically operate on various data so as to substantially randomize and split the data into portions. According to one embodiment, original data is not recreatable from an individual portion. The data assembling module 525 advantageously comprises a software, hardware, or combination module configured to mathematically operate on the foregoing substantially randomized portions, such that the combination thereof provides the original deciphered data. According to one embodiment, the authentication engine 215 employs the data splitting module 520 to randomize and split enrollment authentication data into portions, and employs the data assembling module 525 to reassemble the portions into usable enrollment authentication data.

[0129] FIGURE 6 illustrates a block diagram of the cryptographic engine 220 of the trust engine 200 of FIGURE 2 according to aspects of one embodiment of the invention. Similar to the transaction engine 205 of FIGURE 3, the cryptographic engine 220 comprises an operating system 605 having at least a listening and a handling thread of a modified version of a conventional Web server, such as, for example, Web servers available from Apache. As shown in FIGURE 6, the cryptographic engine 220 comprises a data splitting module 610 and a data assembling module 620 that function similar to those of FIGURE 5. However, according to one embodiment, the data splitting module 610 and the data assembling module 620 process cryptographic key data, as opposed to the foregoing enrollment authentication data. Although, a skilled artisan will recognize from the disclosure herein that the data splitting module 910 and the data splitting module 620 may be combined with those of the authentication engine 215.

[0130] The cryptographic engine 220 also comprises a cryptographic handling module 625 configured to perform one, some or all of a wide number of cryptographic functions. According to one embodiment, the cryptographic handling module 625 may comprise software modules or programs, hardware, or both. According to another embodiment, the cryptographic handling module 625 may perform data comparisons, data parsing, data splitting, data separating, data hashing, data encryption or decryption, digital signature verification or creation, digital certificate generation, storage, or requests, cryptographic key generation, or the like. Moreover, a skilled artisan will recognize from the disclosure herein that the cryptographic handling module 825 may advantageously comprises a public-key infrastructure, such as Pretty Good Privacy (PGP), an RSA-based public-key system, or a wide number of alternative key management systems. In addition, the cryptographic handling module 625 may perform public-key encryption, symmetric-key encryption, or both. In addition to the foregoing, the cryptographic handling module 625 may include one or more computer programs or modules, hardware, or both, for implementing seamless, transparent, interoperability functions.

[0131] A skilled artisan will also recognize from the disclosure herein that the cryptographic functionality may include a wide number or variety of functions generally relating to cryptographic key management systems.

[0132] FIGURE 7 illustrates a simplified block diagram of a depository system 700 according to aspects of an embodiment of the invention. As shown in FIGURE 7, the depository system 700 advantageously comprises multiple data storage facilities, for example, data storage facilities D1, D2, D3, and D4. However, it is readily understood by those of ordinary skill in the art that the depository system may have only one data storage facility. According to one embodiment of the invention, each of the data storage facilities D1 through D4 may advantageously comprise some or all of the elements disclosed with reference to the depository 210 of FIGURE 4. Similar to the depository 210, the data storage facilities D1 through D4 communicate with the transaction engine 205, the authentication engine 215, and the cryptographic engine 220, preferably through conventional SSL. Communication links transferring, for example, XML documents. Communications from the transaction engine 205 may advantageously include requests for data, wherein the request is advantageously broadcast to the IP address of each data storage facility D1 through D4. On the other hand, the transaction engine 205 may broadcast

requests to particular data storage facilities based on a wide number of criteria, such as, for example, response time, server loads, maintenance schedules, or the like.

[0133] In response to requests for data from the transaction engine 205, the depository system 700 advantageously forwards stored data to the authentication engine 215 and the cryptographic engine 220. The respective data assembling modules receive the forwarded data and assemble the data into useable formats. On the other hand, communications from the authentication engine 215 and the cryptographic engine 220 to the data storage facilities D1 through D4 may include the transmission of sensitive data to be stored. For example, according to one embodiment, the authentication engine 215 and the cryptographic engine 220 may advantageously employ their respective data splitting modules to divide sensitive data into undecipherable portions, and then transmit one or more undecipherable portions of the sensitive data to a particular data storage facility.

[0134] According to one embodiment, each data storage facility, D1 through D4, comprises a separate and independent storage system, such as, for example, a directory server. According to another embodiment of the invention, the depository system 700 comprises multiple geographically separated independent data storage systems. By distributing the sensitive data into distinct and independent storage facilities D1 through D4, some or all of which may be advantageously geographically separated, the depository system 700 provides redundancy along with additional security measures. For example, according to one embodiment, only data from two of the multiple data storage facilities, D1 through D4, are needed to decipher and reassemble the sensitive data. Thus, as many as two of the four data storage facilities D1 through D4 may be inoperative due to maintenance, system failure, power failure, or the like, without affecting the functionality of the trust engine 110. In addition, because, according to one embodiment, the data stored in each data storage facility is randomized and undecipherable, compromise of any individual data storage facility does not necessarily compromise the sensitive data. Moreover, in the embodiment having geographical separation of the data storage facilities, a compromise of multiple geographically remote facilities becomes increasingly difficult. In fact, even a rogue employee will be greatly challenged to subvert the needed multiple independent geographically remote data storage facilities.

[0135] Although the depository system 700 is disclosed with reference to its preferred and alternative embodiments, the invention is not intended to be limited thereby. Rather, a skilled

artisan will recognize from the disclosure herein, a wide number of alternatives for the depository system 700. For example, the depository system 700 may comprise one, two or more data storage facilities. In addition, sensitive data may be mathematically operated such that portions from two or more data storage facilities are needed to reassemble and decipher the sensitive data.

[0136] As mentioned in the foregoing, the authentication engine 215 and the cryptographic engine 220 each include a data splitting module 520 and 610, respectively, for splitting any type or form of sensitive data, such as, for example, text, audio, video, the authentication data and the cryptographic key data. FIGURE 8 illustrates a flowchart of a data splitting process 800 performed by the data splitting module according to aspects of an embodiment of the invention. As shown in FIGURE 8, the data splitting process 800 begins at step 805 when sensitive data "S" is received by the data splitting module of the authentication engine 215 or the cryptographic engine 220. Preferably, in step 810, the data splitting module then generates a substantially random number, value, or string or set of bits, "A." For example, the random number A may be generated in a wide number of varying conventional techniques available to one of ordinary skill in the art, for producing high quality random numbers suitable for use in cryptographic applications. In addition, according to one embodiment, the random number A comprises a bit length which may be any suitable length, such as shorter, longer or equal to the bit length of the sensitive data, S.

[0137] In addition, in step 820 the data splitting process 800 generates another statistically random number "C." According to the preferred embodiment, the generation of the statistically random numbers A and C may advantageously be done in parallel. The data splitting module then combines the numbers A and C with the sensitive data S such that new numbers "B" and "D" are generated. For example, number B may comprise the binary combination of A XOR S and number D may comprise the binary combination of C XOR S. The XOR function, or the "exclusive-or" function, is well known to those of ordinary skill in the art. The foregoing combinations preferably occur in steps 825 and 830, respectively, and, according to one embodiment, the foregoing combinations also occur in parallel. The data splitting process 800 then proceeds to step 835 where the random numbers A and C and the numbers B and D are paired such that none of the pairings contain sufficient data, by themselves, to reorganize and decipher the original sensitive data S. For example, the numbers may be paired as follows: AC,

AD, BC, and BD. According to one embodiment, each of the foregoing pairings is distributed to one of the depositories D1 through D4 of FIGURE 7. According to another embodiment, each of the foregoing pairings is randomly distributed to one of the depositories D1 through D4. For example, during a first data splitting process 800, the pairing AC may be sent to depository D2, through, for example, a random selection of D2's IP address. Then, during a second data splitting process 800, the pairing AC may be sent to depository D4, through, for example, a random selection of D4's IP address. In addition, the pairings may all be stored on one depository, and may be stored in separate locations on said depository.

[0138] Based on the foregoing, the data splitting process 800 advantageously places portions of the sensitive data in each of the four data storage facilities D1 through D4, such that no single data storage facility D1 through D4 includes sufficient encrypted data to recreate the original sensitive data S. As mentioned in the foregoing, such randomization of the data into individually unusable encrypted portions increases security and provides for maintained trust in the data even if one of the data storage facilities, D1 through D4, is compromised.

[0139] Although the data splitting process 800 is disclosed with reference to its preferred embodiment, the invention is not intended to be limited thereby. Rather a skilled artisan will recognize from the disclosure herein, a wide number of alternatives for the data splitting process 800. For example, the data splitting process may advantageously split the data into two numbers, for example, random number A and number B and, randomly distribute A and B through two data storage facilities. Moreover, the data splitting process 800 may advantageously split the data among a wide number of data storage facilities through generation of additional random numbers. The data may be split into any desired, selected, predetermined, or randomly assigned size unit, including but not limited to, a bit, bits, bytes, kilobytes, megabytes or larger, or any combination or sequence of sizes. In addition, varying the sizes of the data units resulting from the splitting process may render the data more difficult to restore to a useable form, thereby increasing security of sensitive data. It is readily apparent to those of ordinary skill in the art that the split data unit sizes may be a wide variety of data unit sizes or patterns of sizes or combinations of sizes. For example, the data unit sizes may be selected or predetermined to be all of the same size, a fixed set of different sizes, a combination of sizes, or randomly generates sizes. Similarly, the data units may be distributed into one or more shares according to a fixed or

predetermined data unit size, a pattern or combination of data unit sizes, or a randomly generated data unit size or sizes per share.

[0140] As mentioned in the foregoing, in order to recreate the sensitive data S, the data portions need to be derandomized and reorganized. This process may advantageously occur in the data assembling modules, 525 and 620, of the authentication engine 215 and the cryptographic engine 220, respectively. The data assembling module, for example, data assembly module 525, receives data portions from the data storage facilities D1 through D4, and reassembles the data into useable form. For example, according to one embodiment where the data splitting module 520 employed the data splitting process 800 of FIGURE 8, the data assembling module 525 uses data portions from at least two of the data storage facilities D1 through D4 to recreate the sensitive data S. For example, the pairings of AC, AD, BC, and BD, were distributed such that any two provide one of A and B, or, C and D. Noting that S = A XOR B or S = C XOR D indicates that when the data assembling module receives one of A and B, or, C and D, the data assembling module 525 can advantageously reassemble the sensitive data S. Thus, the data assembling module 525 may assemble the sensitive data S, when, for example, it receives data portions from at least the first two of the data storage facilities D1 through D4 to respond to an assemble request by the trust engine 110.

[0141] Based on the above data splitting and assembling processes, the sensitive data S exists in usable format only in a limited area of the trust engine 110. For example, when the sensitive data S includes enrollment authentication data, usable, nonrandomized enrollment authentication data is available only in the authentication engine 215. Likewise, when the sensitive data S includes private cryptographic key data, usable, nonrandomized private cryptographic key data is available only in the cryptographic engine 220.

[0142] Although the data splitting and assembling processes are disclosed with reference to their preferred embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize from the disclosure herein, a wide number of alternatives for splitting and reassembling the sensitive data S. For example, public-key encryption may be used to further secure the data at the data storage facilities D1 through D4. In addition, it is readily apparent to those of ordinary skill in the art that the data splitting module described herein is also a separate and distinct embodiment of the present invention that may be incorporated into, combined with or otherwise made part of any pre-existing computer systems, software suites,

database, or combinations thereof, or other embodiments of the present invention, such as the trust engine, authentication engine, and transaction engine disclosed and described herein.

[0143] FIGURE 9A illustrates a data flow of an enrollment process 900 according to aspects of an embodiment of the invention. As shown in FIGURE 9A, the enrollment process 900 begins at step 905 when a user desires to enroll with the trust engine 110 of the cryptographic system 100. According to this embodiment, the user system 105 advantageously includes a client-side applet, such as a Java-based, that queries the user to enter enrollment data, such as demographic data and enrollment authentication data. According to one embodiment, the enrollment authentication data includes user ID, password(s), biometric(s), or the like. According to one embodiment, during the querying process, the client-side applet preferably communicates with the trust engine 110 to ensure that a chosen user ID is unique. When the user ID is nonunique, the trust engine 110 may advantageously suggest a unique user ID. The client-side applet gathers the enrollment data and transmits the enrollment data, for example, through and XML document, to the trust engine 110, and in particular, to the transaction engine 205. According to one embodiment, the transmission is encoded with the public key of the authentication engine 215.

[0144] According to one embodiment, the user performs a single enrollment during step 905 of the enrollment process 900. For example, the user enrolls himself or herself as a particular person, such as Joe User. When Joe User desires to enroll as Joe User, CEO of Mega Corp., then according to this embodiment, Joe User enrolls a second time, receives a second unique user ID and the trust engine 110 does not associate the two identities. According to another embodiment of the invention, the enrollment process 900 provides for multiple user identities for a single user ID. Thus, in the above example, the trust engine 110 will advantageously associate the two identities of Joe User. As will be understood by a skilled artisan from the disclosure herein, a user may have many identities, for example, Joe User the head of household, Joe User the member of the Charitable Foundations, and the like. Even though the user may have multiple identities, according to this embodiment, the trust engine 110 preferably stores only one set of enrollment data. Moreover, users may advantageously add, edit/update, or delete identities as they are needed.

[0145] Although the enrollment process 900 is disclosed with reference to its preferred embodiment, the invention is not intended to be limited thereby. Rather, a skilled artisan will

recognize from the disclosure herein, a wide number of alternatives for gathering of enrollment data, and in particular, enrollment authentication data. For example, the applet may be common object model (COM) based applet or the like.

[0146] On the other hand, the enrollment process may include graded enrollment. For example, at a lowest level of enrollment, the user may enroll over the communication link 125 without producing documentation as to his or her identity. According to an increased level of enrollment, the user enrolls using a trusted third party, such as a digital notary. For example, and the user may appear in person to the trusted third party, produce credentials such as a birth certificate, driver's license, military ID, or the like, and the trusted third party may advantageously include, for example, their digital signature in enrollment submission. The trusted third party may include an actual notary, a government agency, such as the Post Office or Department of Motor Vehicles, a human resources person in a large company enrolling an employee, or the like. A skilled artisan will understand from the disclosure herein that a wide number of varying levels of enrollment may occur during the enrollment process 900.

[0147] After receiving the enrollment authentication data, at step 915, the transaction engine 205, using conventional FULL SSL technology forwards the enrollment authentication data to the authentication engine 215. In step 920, the authentication engine 215 decrypts the enrollment authentication data using the private key of the authentication engine 215. In addition, the authentication engine 215 employs the data splitting module to mathematically operate on the enrollment authentication data so as to split the data into at least two independently undecipherable, randomized, numbers. As mentioned in the foregoing, at least two numbers may comprise a statistically random number and a binary X0Red number. In step 925, the authentication engine 215 forwards each portion of the randomized numbers to one of the data storage facilities D1 through D4. As mentioned in the foregoing, the authentication engine 215 may also advantageously randomize which portions are transferred to which depositories.

[0148] Often during the enrollment process 900, the user will also desire to have a digital certificate issued such that he or she may receive encrypted documents from others outside the cryptographic system 100. As mentioned in the foregoing, the certificate authority 115 generally issues digital certificates according to one or more of several conventional standards. Generally, the digital certificate includes a public key of the user or system, which is known to everyone.

[0149] Whether the user requests a digital certificate at enrollment, or at another time, the request is transferred through the trust engine 110 to the authentication engine 215. According to one embodiment, the request includes an XML document having, for example, the proper name of the user. According to step 935, the authentication engine 215 transfers the request to the cryptographic engine 220 instructing the cryptographic engine 220 to generate a cryptographic key or key pair.

[0150] Upon request, at step 935, the cryptographic engine 220 generates at least one cryptographic key. According to one embodiment, the cryptographic handling module 625 generates a key pair, where one key is used as a private key, and one is used as a public key. The cryptographic engine 220 stores the private key and, according to one embodiment, a copy of the public key. In step 945, the cryptographic engine 220 transmits a request for a digital certificate to the transaction engine 205. According to one embodiment, the request advantageously includes a standardized request, such as PKCS10, embedded in, for example, an XML document. The request for a digital certificate may advantageously correspond to one or more certificate authorities and the one or more standard formats the certificate authorities require.

[0151] In step 950 the transaction engine 205 forwards this request to the certificate authority 115, who, in step 955, returns a digital certificate. The return digital certificate may advantageously be in a standardized format, such as PKCS7, or in a proprietary format of one or more of the certificate authorities 115. In step 960, the digital certificate is received by the transaction engine 205, and a copy is forwarded to the user and a copy is stored with the trust engine 110. The trust engine 110 stores a copy of the certificate such that the trust engine 110 will not need to rely on the availability of the certificate authority 115. For example, when the user desires to send a digital certificate, or a third party requests the user's digital certificate, the request for the digital certificate is typically sent to the certificate authority 115. However, if the certificate authority 115 is conducting maintenance or has been victim of a failure or security compromise, the digital certificate may not be available.

[0152] At any time after issuing the cryptographic keys, the cryptographic engine 220 may advantageously employ the data splitting process 800 described above such that the cryptographic keys are split into independently undecipherable randomized numbers. Similar to the authentication data, at step 965 the cryptographic engine 220 transfers the randomized numbers to the data storage facilities D1 through D4.

[0153] A skilled artisan will recognize from the disclosure herein that the user may request a digital certificate anytime after enrollment. Moreover, the communications between systems may advantageously include FULL SSL or public-key encryption technologies. Moreover, the enrollment process may issue multiple digital certificates from multiple certificate authorities, including one or more proprietary certificate authorities internal or external to the trust engine 110.

[0154] As disclosed in steps 935 through 960, one embodiment of the invention includes the request for a certificate that is eventually stored on the trust engine 110. Because, according to one embodiment, the cryptographic handling module 625 issues the keys used by the trust engine 110, each certificate corresponds to a private key. Therefore, the trust engine 110 may advantageously provide for interoperability through monitoring the certificates owned by, or associated with, a user. For example, when the cryptographic engine 220 receives a request for a cryptographic function, the cryptographic handling module 625 may investigate the certificates owned by the requesting user to determine whether the user owns a private key matching the attributes of the request. When such a certificate exists, the cryptographic handling module 625 may use the certificate or the public or private keys associated therewith, to perform the requested function. When such a certificate does not exist, the cryptographic handling module 625 may advantageously and transparently perform a number of actions to attempt to remedy the lack of an appropriate key. For example, FIGURE 9B illustrates a flowchart of an interoperability process 970, which according to aspects of an embodiment of the invention, discloses the foregoing steps to ensure the cryptographic handling module 625 performs cryptographic functions using appropriate keys.

[0155] As shown in FIGURE 9B, the interoperability process 970 begins with step 972 where the cryptographic handling module 925 determines the type of certificate desired. According to one embodiment of the invention, the type of certificate may advantageously be specified in the request for cryptographic functions, or other data provided by the requestor. According to another embodiment, the certificate type may be ascertained by the data format of the request. For example, the cryptographic handling module 925 may advantageously recognize the request corresponds to a particular type.

[0156] According to one embodiment, the certificate type may include one or more algorithm standards, for example, RSA, ELGAMAL, or the like. In addition, the certificate type may

include one or more key types, such as symmetric keys, public keys, strong encryption keys such as 256 bit keys, less secure keys, or the like. Moreover, the certificate type may include upgrades or replacements of one or more of the foregoing algorithm standards or keys, one or more message or data formats, one or more data encapsulation or encoding schemes, such as Base 32 or Base 64. The certificate type may also include compatibility with one or more third-party cryptographic applications or interfaces, one or more communication protocols, or one or more certificate standards or protocols. A skilled artisan will recognize from the disclosure herein that other differences may exist in certificate types, and translations to and from those differences may be implemented as disclosed herein.

[0157] Once the cryptographic handling module 625 determines the certificate type, the interoperability process 970 proceeds to step 974, and determines whether the user owns a certificate matching the type determined in step 974. When the user owns a matching certificate, for example, the trust engine 110 has access to the matching certificate through, for example, prior storage thereof, the cryptographic handling module 825 knows that a matching private key is also stored within the trust engine 110. For example, the matching private key may be stored within the depository 210 or depository system 700. The cryptographic handling module 625 may advantageously request the matching private key be assembled from, for example, the depository 210, and then in step 976, use the matching private key to perform cryptographic actions or functions. For example, as mentioned in the foregoing, the cryptographic handling module 625 may advantageously perform hashing, hash comparisons, data encryption or decryption, digital signature verification or creation, or the like.

[0158] When the user does not own a matching certificate, the interoperability process 970 proceeds to step 978 where the cryptographic handling module 625 determines whether the users owns a cross-certified certificate. According to one embodiment, cross-certification between certificate authorities occurs when a first certificate authority determines to trust certificates from a second certificate authority. In other words, the first certificate authority determines that certificates from the second certificate authority meets certain quality standards, and therefore, may be "certified" as equivalent to the first certificate authority's own certificates. Cross-certification becomes more complex when the certificate authorities issue, for example, certificates having levels of trust. For example, the first certificate authority may provide three levels of trust for a particular certificate, usually based on the degree of reliability in the

enrollment process, while the second certificate authority may provide seven levels of trust. Cross-certification may advantageously track which levels and which certificates from the second certificate authority may be substituted for which levels and which certificates from the first. When the foregoing cross-certification is done officially and publicly between two certification authorities, the mapping of certificates and levels to one another is often called "chaining."

[0159] According to another embodiment of the invention, the cryptographic handling module 625 may advantageously develop cross-certifications outside those agreed upon by the certificate authorities. For example, the cryptographic handling module 625 may access a first certificate authority's certificate practice statement (CPS), or other published policy statement, and using, for example, the authentication tokens required by particular trust levels, match the first certificate authority's certificates to those of another certificate authority.

[0160] When, in step 978, the cryptographic handling module 625 determines that the users owns a cross-certified certificate, the interoperability process 970 proceeds to step 976, and performs the cryptographic action or function using the cross-certified public key, private key, or both. Alternatively, when the cryptographic handling module 625 determines that the users does not own a cross-certified certificate, the interoperability process 970 proceeds to step 980, where the cryptographic handling module 625 selects a certificate authority that issues the requested certificate type, or a certificate cross-certified thereto. In step 982, the cryptographic handling module 625 determines whether the user enrollment authentication data, discussed in the foregoing, meets the authentication requirements of the chosen certificate authority. For example, if the user enrolled over a network by, for example, answering demographic and other questions, the authentication data provided may establish a lower level of trust than a user providing biometric data and appearing before a third-party, such as, for example, a notary. According to one embodiment, the foregoing authentication requirements may advantageously be provided in the chosen authentication authority's CPS.

[0161] When the user has provided the trust engine 110 with enrollment authentication data meeting the requirements of chosen certificate authority, the interoperability process 970 proceeds to step 984, where the cryptographic handling module 825 acquires the certificate from the chosen certificate authority. According to one embodiment, the cryptographic handling module 625 acquires the certificate by following steps 945 through 960 of the enrollment process

900. For example, the cryptographic handling module 625 may advantageously employ one or more public keys from one or more of the key pairs already available to the cryptographic engine 220, to request the certificate from the certificate authority. According to another embodiment, the cryptographic handling module 625 may advantageously generate one or more new key pairs, and use the public keys corresponding thereto, to request the certificate from the certificate authority.

[0162] According to another embodiment, the trust engine 110 may advantageously include one or more certificate issuing modules capable of issuing one or more certificate types. According to this embodiment, the certificate issuing module may provide the foregoing certificate. When the cryptographic handling module 625 acquires the certificate, the interoperability process 970 proceeds to step 976, and performs the cryptographic action or function using the public key, private key, or both corresponding to the acquired certificate.

[0163] When the user, in step 982, has not provided the trust engine 110 with enrollment authentication data meeting the requirements of chosen certificate authority, the cryptographic handling module 625 determines, in step 986 whether there are other certificate authorities that have different authentication requirements. For example, the cryptographic handling module 625 may look for certificate authorities having lower authentication requirements, but still issue the chosen certificates, or cross-certifications thereof.

[0164] When the foregoing certificate authority having lower requirements exists, the interoperability process 970 proceeds to step 980 and chooses that certificate authority. Alternatively, when no such certificate authority exists, in step 988, the trust engine 110 may request additional authentication tokens from the user. For example, the trust engine 110 may request new enrollment authentication data comprising, for example, biometric data. Also, the trust engine 110 may request the user appear before a trusted third party and provide appropriate authenticating credentials, such as, for example, appearing before a notary with a drivers license, social security card, bank card, birth certificate, military ID, or the like. When the trust engine 110 receives updated authentication data, the interoperability process 970 proceeds to step 984 and acquires the foregoing chosen certificate.

[0165] Through the foregoing interoperability process 970, the cryptographic handling module 625 advantageously provides seamless, transparent, translations and conversions between differing cryptographic systems. A skilled artisan will recognize from the disclosure herein, a

wide number of advantages and implementations of the foregoing interoperable system. For example, the foregoing step 986 of the interoperability process 970 may advantageously include aspects of trust arbitrage, discussed in further detail below, where the certificate authority may under special circumstances accept lower levels of cross-certification. In addition, the interoperability process 970 may include ensuring interoperability between and employment of standard certificate revocations, such as employing certificate revocation lists (CRL), online certificate status protocols (OCSP), or the like.

[0166] FIGURE 10 illustrates a data flow of an authentication process 1000 according to aspects of an embodiment of the invention. According to one embodiment, the authentication process 1000 includes gathering current authentication data from a user and comparing that to the enrollment authentication data of the user. For example, the authentication process 1000 begins at step 1005 where a user desires to perform a transaction with, for example, a vendor. Such transactions may include, for example, selecting a purchase option, requesting access to a restricted area or device of the vendor system 120, or the like. At step 1010, a vendor provides the user with a transaction ID and an authentication request. The transaction ID may advantageously include a 192 bit quantity having a 32 bit timestamp concatenated with a 128 bit random quantity, or a "nonce," concatenated with a 32 bit vendor specific constant. Such a transaction ID uniquely identifies the transaction such that copycat transactions can be refused by the trust engine 110.

[0167] The authentication request may advantageously include what level of authentication is needed for a particular transaction. For example, the vendor may specify a particular level of confidence that is required for the transaction at issue. If authentication cannot be made to this level of confidence, as will be discussed below, the transaction will not occur without either further authentication by the user to raise the level of confidence, or a change in the terms of the authentication between the vendor and the server. These issues are discussed more completely below.

[0168] According to one embodiment, the transaction ID and the authentication request may be advantageously generated by a vendor-side applet or other software program. In addition, the transmission of the transaction ID and authentication data may include one or more XML documents encrypted using conventional SSL technology, such as, for example, ½ SSL, or, in other words vendor-side authenticated SSL.

[0169] After the user system 105 receives the transaction ID and authentication request, the user system 105 gathers the current authentication data, potentially including current biometric information, from the user. The user system 105, at step 1015, encrypts at least the current authentication data "B" and the transaction ID, with the public key of the authentication engine 215, and transfers that data to the trust engine 110. The transmission preferably comprises XML documents encrypted with at least conventional ½ SSL technology. In step 1020, the transaction engine 205 receives the transmission, preferably recognizes the data format or request in the URL or URI, and forwards the transmission to the authentication engine 215.

[0170] During steps 1015 and 1020, the vendor system 120, at step 1025, forwards the transaction ID and the authentication request to the trust engine 110, using the preferred FULL SSL technology. This communication may also include a vendor ID, although vendor identification may also be communicated through a non-random portion of the transaction ID. At steps 1030 and 1035, the transaction engine 205 receives the communication, creates a record in the audit trail, and generates a request for the user's enrollment authentication data to be assembled from the data storage facilities D1 through D4. At step 1040, the depository system 700 transfers the portions of the enrollment authentication data corresponding to the user to the authentication engine 215. At step 1045, the authentication engine 215 decrypts the transmission using its private key and compares the enrollment authentication data to the current authentication data provided by the user.

[0171] The comparison of step 1045 may advantageously apply heuristical context sensitive authentication, as referred to in the forgoing, and discussed in further detail below. For example, if the biometric information received does not match perfectly, a lower confidence match results. In particular embodiments, the level of confidence of the authentication is balanced against the nature of the transaction and the desires of both the user and the vendor. Again, this is discussed in greater detail below.

[0172] At step 1050, the authentication engine 215 fills in the authentication request with the result of the comparison of step 1045. According to one embodiment of the invention, the authentication request is filled with a YES/NO or TRUE/FALSE result of the authentication process 1000. In step 1055 the filled-in authentication request is returned to the vendor for the vendor to act upon, for example, allowing the user to complete the transaction that initiated the

authentication request. According to one embodiment, a confirmation message is passed to the user.

[0173] Based on the foregoing, the authentication process 1000 advantageously keeps sensitive data secure and produces results configured to maintain the integrity of the sensitive data. For example, the sensitive data is assembled only inside the authentication engine 215. For example, the enrollment authentication data is undecipherable until it is assembled in the authentication engine 215 by the data assembling module, and the current authentication data is undecipherable until it is unwrapped by the conventional SSL technology and the private key of the authentication engine 215. Moreover, the authentication result transmitted to the vendor does not include the sensitive data, and the user may not even know whether he or she produced valid authentication data.

[0174] Although the authentication process 1000 is disclosed with reference to its preferred and alternative embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize from the disclosure herein, a wide number of alternatives for the authentication process 1000. For example, the vendor may advantageously be replaced by almost any requesting application, even those residing with the user system 105. For example, a client application, such as Microsoft Word, may use an application program interface (API) or a cryptographic API (CAPI) to request authentication before unlocking a document. Alternatively, a mail server, a network, a cellular phone, a personal or mobile computing device, a workstation, or the like, may all make authentication requests that can be filled by the authentication process 1000. In fact, after providing the foregoing trusted authentication process 1000, the requesting application or device may provide access to or use of a wide number of electronic or computer devices or systems.

[0175] Moreover, the authentication process 1000 may employ a wide number of alternative procedures in the event of authentication failure. For example, authentication failure may maintain the same transaction ID and request that the user reenter his or her current authentication data. As mentioned in the foregoing, use of the same transaction ID allows the comparator of the authentication engine 215 to monitor and limit the number of authentication attempts for a particular transaction, thereby creating a more secure cryptographic system 100.

[0176] In addition, the authentication process 1000 may be advantageously be employed to develop elegant single sign-on solutions, such as, unlocking a sensitive data vault. For example,

successful or positive authentication may provide the authenticated user the ability to automatically access any number of passwords for an almost limitless number of systems and applications. For example, authentication of a user may provide the user access to password, login, financial credentials, or the like, associated with multiple online vendors, a local area network, various personal computing devices, Internet service providers, auction providers, investment brokerages, or the like. By employing a sensitive data vault, users may choose truly large and random passwords because they no longer need to remember them through association. Rather, the authentication process 1000 provides access thereto. For example, a user may choose a random alphanumeric string that is twenty plus digits in length rather than something associated with a memorable data, name, etc.

[0177] According to one embodiment, a sensitive data vault associated with a given user may advantageously be stored in the data storage facilities of the depository 210, or split and stored in the depository system 700. According to this embodiment, after positive user authentication, the trust engine 110 serves the requested sensitive data, such as, for example, to the appropriate password to the requesting application. According to another embodiment, the trust engine 110 may include a separate system for storing the sensitive data vault. For example, the trust engine 110 may include a stand-alone software engine implementing the data vault functionality and figuratively residing "behind" the foregoing front-end security system of the trust engine 110. According to this embodiment, the software engine serves the requested sensitive data after the software engine receives a signal indicating positive user authentication from the trust engine 110.

[0178] In yet another embodiment, the data vault may be implemented by a third-party system. Similar to the software engine embodiment, the third-party system may advantageously serve the requested sensitive data after the third-party system receives a signal indicating positive user authentication from the trust engine 110. According to yet another embodiment, the data vault may be implemented on the user system 105. A user-side software engine may advantageously serve the foregoing data after receiving a signal indicating positive user authentication from the trust engine 110.

[0179] Although the foregoing data vaults are disclosed with reference to alternative embodiments, a skilled artisan will recognize from the disclosure herein, a wide number of additional implementations thereof. For example, a particular data vault may include aspects

from some or all of the foregoing embodiments. In addition, any of the foregoing data vaults may employ one or more authentication requests at varying times. For example, any of the data vaults may require authentication every one or more transactions, periodically, every one or more sessions, every access to one or more Webpages or Websites, at one or more other specified intervals, or the like.

[0180] FIGURE 11 illustrates a data flow of a signing process 1100 according to aspects of an embodiment of the invention. As shown in FIGURE 11, the signing process 1100 includes steps similar to those of the authentication process 1000 described in the foregoing with reference to FIGURE 10. According to one embodiment of the invention, the signing process 1100 first authenticates the user and then performs one or more of several digital signing functions as will be discussed in further detail below. According to another embodiment, the signing process 1100 may advantageously store data related thereto, such as hashes of messages or documents, or the like. This data may advantageously be used in an audit or any other event, such as for example, when a participating party attempts to repudiate a transaction.

[0181] As shown in FIGURE 11, during the authentication steps, the user and vendor may advantageously agree on a message, such as, for example, a contract. During signing, the signing process 1100 advantageously ensures that the contract signed by the user is identical to the contract supplied by the vendor. Therefore, according to one embodiment, during authentication, the vendor and the user include a hash of their respective copies of the message or contract, in the data transmitted to the authentication engine 215. By employing only a hash of a message or contract, the trust engine 110 may advantageously store a significantly reduced amount of data, providing for a more efficient and cost effective cryptographic system. In addition, the stored hash may be advantageously compared to a hash of a document in question to determine whether the document in question matches one signed by any of the parties. The ability to determine whether the document is identical to one relating to a transaction provides for additional evidence that can be used against a claim for repudiation by a party to a transaction.

[0182] In step 1103, the authentication engine 215 assembles the enrollment authentication data and compares it to the current authentication data provided by the user. When the comparator of the authentication engine 215 indicates that the enrollment authentication data matches the current authentication data, the comparator of the authentication engine 215 also

compares the hash of the message supplied by the vendor to the hash of the message supplied by the user. Thus, the authentication engine 215 advantageously ensures that the message agreed to by the user is identical to that agreed to by the vendor.

[0183] In step 1105, the authentication engine 215 transmits a digital signature request to the cryptographic engine 220. According to one embodiment of the invention, the request includes a hash of the message or contract. However, a skilled artisan will recognize from the disclosure herein that the cryptographic engine 220 may encrypt virtually any type of data, including, but not limited to, video, audio, biometrics, images or text to form the desired digital signature. Returning to step 1105, the digital signature request preferably comprises an XML document communicated through conventional SSL technologies.

[0184] In step 1110, the authentication engine 215 transmits a request to each of the data storage facilities D1 through D4, such that each of the data storage facilities D1 through D4 transmit their respective portion of the cryptographic key or keys corresponding to a signing party. According to another embodiment, the cryptographic engine 220 employs some or all of the steps of the interoperability process 970 discussed in the foregoing, such that the cryptographic engine 220 first determines the appropriate key or keys to request from the depository 210 or the depository system 700 for the signing party, and takes actions to provide appropriate matching keys. According to still another embodiment, the authentication engine 215 or the cryptographic engine 220 may advantageously request one or more of the keys associated with the signing party and stored in the depository 210 or depository system 700.

[0185] According to one embodiment, the signing party includes one or both the user and the vendor. In such case, the authentication engine 215 advantageously requests the cryptographic keys corresponding to the user and/or the vendor. According to another embodiment, the signing party includes the trust engine 110. In this embodiment, the trust engine 110 is certifying that the authentication process 1000 properly authenticated the user, vendor, or both. Therefore, the authentication engine 215 requests the cryptographic key of the trust engine 110, such as, for example, the key belonging to the cryptographic engine 220, to perform the digital signature. According to another embodiment, the trust engine 110 performs a digital notary-like function. In this embodiment, the signing party includes the user, vendor, or both, along with the trust engine 110. Thus, the trust engine 110 provides the digital signature of the user and/or vendor, and then indicates with its own digital signature that the user and/or vendor were properly

authenticated. In this embodiment, the authentication engine 215 may advantageously request assembly of the cryptographic keys corresponding to the user, the vendor, or both. According to another embodiment, the authentication engine 215 may advantageously request assembly of the cryptographic keys corresponding to the trust engine 110.

[0186] According to another embodiment, the trust engine 110 performs power of attorney-like functions. For example, the trust engine 110 may digitally sign the message on behalf of a third party. In such case, the authentication engine 215 requests the cryptographic keys associated with the third party. According to this embodiment, the signing process 1100 may advantageously include authentication of the third party, before allowing power of attorney-like functions. In addition, the authentication process 1000 may include a check for third party constraints, such as, for example, business logic or the like dictating when and in what circumstances a particular third-party's signature may be used.

[0187] Based on the foregoing, in step 1110, the authentication engine requested the cryptographic keys from the data storage facilities D1 through D4 corresponding to the signing party. In step 1115, the data storage facilities D1 through D4 transmit their respective portions of the cryptographic key corresponding to the signing party to the cryptographic engine 220. According to one embodiment, the foregoing transmissions include SSL technologies. According to another embodiment, the foregoing transmissions may advantageously be super-encrypted with the public key of the cryptographic engine 220.

[0188] In step 1120, the cryptographic engine 220 assembles the foregoing cryptographic keys of the signing party and encrypts the message therewith, thereby forming the digital signature(s). In step 1125 of the signing process 1100, the cryptographic engine 220 transmits the digital signature(s) to the authentication engine 215. In step 1130, the authentication engine 215 transmits the filled-in authentication request along with a copy of the hashed message and the digital signature(s) to the transaction engine 205. In step 1135, the transaction engine 205 transmits a receipt comprising the transaction ID, an indication of whether the authentication was successful, and the digital signature(s), to the vendor. According to one embodiment, the foregoing transmission may advantageously include the digital signature of the trust engine 110. For example, the trust engine 110 may encrypt the hash of the receipt with its private key, thereby forming a digital signature to be attached to the transmission to the vendor.

[0189] According to one embodiment, the transaction engine 205 also transmits a confirmation message to the user. Although the signing process 1100 is disclosed with reference to its preferred and alternative embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize from the disclosure herein, a wide number of alternatives for the signing process 1100. For example, the vendor may be replaced with a user application, such as an email application. For example, the user may wish to digitally sign a particular email with his or her digital signature. In such an embodiment, the transmission throughout the signing process 1100 may advantageously include only one copy of a hash of the message. Moreover, a skilled artisan will recognize from the disclosure herein that a wide number of client applications may request digital signatures. For example, the client applications may comprise word processors, spreadsheets, emails, voicemail, access to restricted system areas, or the like.

[0190] In addition, a skilled artisan will recognize from the disclosure herein that steps 1105 through 1120 of the signing process 1100 may advantageously employ some or all of the steps of the interoperability process 970 of FIGURE 9B, thereby providing interoperability between differing cryptographic systems that may, for example, need to process the digital signature under differing signature types.

[0191] FIGURE 12 illustrates a data flow of an encryption/decryption process 1200 according to aspects of an embodiment of the invention. As shown in FIGURE 12, the decryption process 1200 begins by authenticating the user using the authentication process 1000. According to one embodiment, the authentication process 1000 includes in the authentication request, a synchronous session key. For example, in conventional PKI technologies, it is understood by skilled artisans that encrypting or decrypting data using public and private keys is mathematically intensive and may require significant system resources. However, in symmetric key cryptographic systems, or systems where the sender and receiver of a message share a single common key that is used to encrypt and decrypt a message, the mathematical operations are significantly simpler and faster. Thus, in the conventional PKI technologies, the sender of a message will generate synchronous session key, and encrypt the message using the simpler, faster symmetric key system. Then, the sender will encrypt the session key with the public key of the receiver. The encrypted session key will be attached to the synchronously encrypted message and both data are sent to the receiver. The receiver uses his or her private key to decrypt the session key, and then uses the session key to decrypt the message. Based on the

foregoing, the simpler and faster symmetric key system is used for the majority of the encryption/decryption processing. Thus, in the decryption process 1200, the decryption advantageously assumes that a synchronous key has been encrypted with the public key of the user. Thus, as mentioned in the foregoing, the encrypted session key is included in the authentication request.

[0192] Returning to the decryption process 1200, after the user has been authenticated in step 1205, the authentication engine 215 forwards the encrypted session key to the cryptographic engine 220. In step 1210, the authentication engine 215 forwards a request to each of the data storage facilities, D1 through D4, requesting the cryptographic key data of the user. In step 1215, each data storage facility, D1 through D4, transmits their respective portion of the cryptographic key to the cryptographic engine 220. According to one embodiment, the foregoing transmission is encrypted with the public key of the cryptographic engine 220.

[0193] In step 1220 of the decryption process 1200, the cryptographic engine 220 assembles the cryptographic key and decrypts the session key therewith. In step 1225, the cryptographic engine forwards the session key to the authentication engine 215. In step 1227, the authentication engine 215 fills in the authentication request including the decrypted session key, and transmits the filled-in authentication request to the transaction engine 205. In step 1230, the transaction engine 205 forwards the authentication request along with the session key to the requesting application or vendor. Then, according to one embodiment, the requesting application or vendor uses the session key to decrypt the encrypted message.

[0194] Although the decryption process 1200 is disclosed with reference to its preferred and alternative embodiments, a skilled artisan will recognize from the disclosure herein, a wide number of alternatives for the decryption process 1200. For example, the decryption process 1200 may forego synchronous key encryption and rely on full public-key technology. In such an embodiment, the requesting application may transmit the entire message to the cryptographic engine 220, or, may employ some type of compression or reversible hash in order to transmit the message to the cryptographic engine 220. A skilled artisan will also recognize from the disclosure herein that the foregoing communications may advantageously include XML documents wrapped in SSL technology.

[0195] The encryption/decryption process 1200 also provides for encryption of documents or other data. Thus, in step 1235, a requesting application or vendor may advantageously transmit

to the transaction engine 205 of the trust engine 110, a request for the public key of the user. The requesting application or vendor makes this request because the requesting application or vendor uses the public key of the user, for example, to encrypt the session key that will be used to encrypt the document or message. As mentioned in the enrollment process 900, the transaction engine 205 stores a copy of the digital certificate of the user, for example, in the mass storage 225. Thus, in step 1240 of the encryption process 1200, the transaction engine 205 requests the digital certificate of the user from the mass storage 225. In step 1245, the mass storage 225 transmits the digital certificate corresponding to the user, to the transaction engine 205. In step 1250, the transaction engine 205 transmits the digital certificate to the requesting application or vendor. According to one embodiment, the encryption portion of the encryption process 1200 does not include the authentication of a user. This is because the requesting vendor needs only the public key of the user, and is not requesting any sensitive data.

[0196] A skilled artisan will recognize from the disclosure herein that if a particular user does not have a digital certificate, the trust engine 110 may employ some or all of the enrollment process 900 in order to generate a digital certificate for that particular user. Then, the trust engine 110 may initiate the encryption/decryption process 1200 and thereby provide the appropriate digital certificate. In addition, a skilled artisan will recognize from the disclosure herein that steps 1220 and 1235 through 1250 of the encryption/decryption process 1200 may advantageously employ some or all of the steps of the interoperability process of FIGURE 9B, thereby providing interoperability between differing cryptographic systems that may, for example, need to process the encryption.

[0197] FIGURE 13 illustrates a simplified block diagram of a trust engine system 1300 according to aspects of yet another embodiment of the invention. As shown in FIGURE 13, the trust engine system 1300 comprises a plurality of distinct trust engines 1305, 1310, 1315, and 1320, respectively. To facilitate a more complete understanding of the invention, FIGURE 13 illustrates each trust engine, 1305, 1310, 1315, and 1320 as having a transaction engine, a depository, and an authentication engine. However, a skilled artisan will recognize that each transaction engine may advantageously comprise some, a combination, or all of the elements and communication channels disclosed with reference to FIGURES 1-8. For example, one embodiment may advantageously include trust engines having one or more transaction engines, depositories, and cryptographic servers or any combinations thereof.

[0198] According to one embodiment of the invention, each of the trust engines 1305, 1310, 1315 and 1320 are geographically separated, such that, for example, the trust engine 1305 may reside in a first location, the trust engine 1310 may reside in a second location, the trust engine 1315 may reside in a third location, and the trust engine 1320 may reside in a fourth location. The foregoing geographic separation advantageously decreases system response time while increasing the security of the overall trust engine system 1300.

[0199] For example, when a user logs onto the cryptographic system 100, the user may be nearest the first location and may desire to be authenticated. As described with reference to FIGURE 10, to be authenticated, the user provides current authentication data, such as a biometric or the like, and the current authentication data is compared to that user's enrollment authentication data. Therefore, according to one example, the user advantageously provides current authentication data to the geographically nearest trust engine 1305. The transaction engine 1321 of the trust engine 1305 then forwards the current authentication data to the authentication engine 1322 also residing at the first location. According to another embodiment, the transaction engine 1321 forwards the current authentication data to one or more of the authentication engines of the trust engines 1310, 1315, or 1320.

[0200] The transaction engine 1321 also requests the assembly of the enrollment authentication data from the depositories of, for example, each of the trust engines, 1305 through 1320. According to this embodiment, each depository provides its portion of the enrollment authentication data to the authentication engine 1322 of the trust engine 1305. The authentication engine 1322 then employs the encrypted data portions from, for example, the first two depositories to respond, and assembles the enrollment authentication data into deciphered form. The authentication engine 1322 compares the enrollment authentication data with the current authentication data and returns an authentication result to the transaction engine 1321 of the trust engine 1305.

[0201] Based on the above, the trust engine system 1300 employs the nearest one of a plurality of geographically separated trust engines, 1305 through 1320, to perform the authentication process. According to one embodiment of the invention, the routing of information to the nearest transaction engine may advantageously be performed at client-side applets executing on one or more of the user system 105, vendor system 120, or certificate authority 115. According to an alternative embodiment, a more sophisticated decision process may be employed to select

from the trust engines 1305 through 1320. For example, the decision may be based on the availability, operability, speed of connections, load, performance, geographic proximity, or a combination thereof, of a given trust engine.

[0202] In this way, the trust engine system 1300 lowers its response time while maintaining the security advantages associated with geographically remote data storage facilities, such as those discussed with reference to FIGURE 7 where each data storage facility stores randomized portions of sensitive data. For example, a security compromise at, for example, the depository 1325 of the trust engine 1315 does not necessarily compromise the sensitive data of the trust engine system 1300. This is because the depository 1325 contains only non-decipherable randomized data that, without more, is entirely useless.

[0203] According to another embodiment, the trust engine system 1300 may advantageously include multiple cryptographic engines arranged similar to the authentication engines. The cryptographic engines may advantageously perform cryptographic functions such as those disclosed with reference to FIGURES 1-8. According to yet another embodiment, the trust engine system 1300 may advantageously replace the multiple authentication engines with multiple cryptographic engines, thereby performing cryptographic functions such as those disclosed with reference to FIGURES 1-8. According to yet another embodiment of the invention, the trust engine system 1300 may replace each multiple authentication engine with an engine having some or all of the functionality of the authentication engines, cryptographic engines, or both, as disclosed in the foregoing,

[0204] Although the trust engine system 1300 is disclosed with reference to its preferred and alternative embodiments, a skilled artisan will recognize that the trust engine system 1300 may comprise portions of trust engines 1305 through 1320. For example, the trust engine system 1300 may include one or more transaction engines, one or more depositories, one or more authentication engines, or one or more cryptographic engines or combinations thereof.

[0205] FIGURE 14 illustrates a simplified block diagram of a trust engine System 1400 according to aspects of yet another embodiment of the invention. As shown in FIGURE 14, the trust engine system 1400 includes multiple trust engines 1405, 1410, 1415 and 1420. According to one embodiment, each of the trust engines 1405, 1410, 1415 and 1420, comprise some or all of the elements of trust engine 110 disclosed with reference to FIGURES 1-8. According to this embodiment, when the client side applets of the user system 105, the vendor system 120, or the

certificate authority 115, communicate with the trust engine system 1400, those communications are sent to the IP address of each of the trust engines 1405 through 1420. Further, each transaction engine of each of the trust engines, 1405, 1410, 1415, and 1420, behaves similar to the transaction engine 1321 of the trust engine 1305 disclosed with reference to FIGURE 13. For example, during an authentication process, each transaction engine of each of the trust engines 1405, 1410, 1415, and 1420 transmits the current authentication data to their respective authentication engines and transmits a request to assemble the randomized data stored in each of the depositories of each of the trust engines 1405 through 1420. FIGURE 14 does not illustrate all of these communications; as such illustration would become overly complex. Continuing with the authentication process, each of the depositories then communicates its portion of the randomized data to each of the authentication engines of the each of the trust engines 1405 through 1420. Each of the authentication engines of the each of the trust engines employs its comparator to determine whether the current authentication data matches the enrollment authentication data provided by the depositories of each of the trust engines 1405 through 1420. According to this embodiment, the result of the comparison by each of the authentication engines is then transmitted to a redundancy module of the other three trust engines. For example, the result of the authentication engine from the trust engine 1405 is transmitted to the redundancy modules of the trust engines 1410, 1415, and 1420. Thus, the redundancy module of the trust engine 1405 likewise receives the result of the authentication engines from the trust engines 1410, 1415, and 1420.

[0206] FIGURE 15 illustrates a block diagram of the redundancy module of FIGURE 14. The redundancy module comprises a comparator configured to receive the authentication result from three authentication engines and transmit that result to the transaction engine of the fourth trust engine. The comparator compares the authentication result form the three authentication engines, and if two of the results agree, the comparator concludes that the authentication result should match that of the two agreeing authentication engines. This result is then transmitted back to the transaction engine corresponding to the trust engine not associated with the three authentication engines.

[0207] Based on the foregoing, the redundancy module determines an authentication result from data received from authentication engines that are preferably geographically remote from the trust engine of that the redundancy module. By providing such redundancy functionality, the

trust engine system 1400 ensures that a compromise of the authentication engine of one of the trust engines 1405 through 1420, is insufficient to compromise the authentication result of the redundancy module of that particular trust engine. A skilled artisan will recognize that redundancy module functionality of the trust engine system 1400 may also be applied to the cryptographic engine of each of the trust engines 1405 through 1420. However, such cryptographic engine communication was not shown in FIGURE 14 to avoid complexity. Moreover, a skilled artisan will recognize a wide number of alternative authentication result conflict resolution algorithms for the comparator of FIGURE 15 are suitable for use in the present invention.

[0208] According to yet another embodiment of the invention, the trust engine system 1400 may advantageously employ the redundancy module during cryptographic comparison steps. For example, some or all of the foregoing redundancy module disclosure with reference to FIGURES 14 and 15 may advantageously be implemented during a hash comparison of documents provided by one or more parties during a particular transaction.

[0209] Although the foregoing invention has been described in terms of certain preferred and alternative embodiments, other embodiments will be apparent to those of ordinary skill in the art from the disclosure herein. For example, the trust engine 110 may issue short-term certificates, where the private cryptographic key is released to the user for a predetermined period of time. For example, current certificate standards include a validity field that can be set to expire after a predetermined amount of time. Thus, the trust engine 110 may release a private key to a user where the private key would be valid for, for example, 24 hours. According to such an embodiment, the trust engine 110 may advantageously issue a new cryptographic key pair to be associated with a particular user and then release the private key of the new cryptographic key pair. Then, once the private cryptographic key is released, the trust engine 110 immediately expires any internal valid use of such private key, as it is no longer securable by the trust engine 110.

[0210] In addition, a skilled artisan will recognize that the cryptographic system 100 or the trust engine 110 may include the ability to recognize any type of devices, such as, but not limited to, a laptop, a cell phone, a network, a biometric device or the like. According to one embodiment, such recognition may come from data supplied in the request for a particular service, such as, a request for authentication leading to access or use, a request for cryptographic

functionality, or the like. According to one embodiment, the foregoing request may include a unique device identifier, such as, for example, a processor ID. Alternatively, the request may include data in a particular recognizable data format. For example, mobile and satellite phones often do not include the processing power for full X509.v3 heavy encryption certificates, and therefore do not request them. According to this embodiment, the trust engine 110 may recognize the type of data format presented, and respond only in kind.

[0211] In an additional aspect of the system described above, context sensitive authentication can be provided using various techniques as will be described below. Context sensitive authentication, for example as shown in FIGURE 16, provides the possibility of evaluating not only the actual data which is sent by the user when attempting to authenticate himself, but also the circumstances surrounding the generation and delivery of that data. Such techniques may also support transaction specific trust arbitrage between the user and trust engine 110 or between the vendor and trust engine 110, as will be described below.

[0212] As discussed above, authentication is the process of proving that a user is who he says he is. Generally, authentication requires demonstrating some fact to an authentication authority. The trust engine 110 of the present invention represents the authority to which a user must authenticate himself. The user must demonstrate to the trust engine 110 that he is who he says he is by either: knowing something that only the user should know (knowledge-based authentication), having something that only the user should have (token-based authentication), or by being something that only the user should be (biometric-based authentication).

[0213] Examples of knowledge-based authentication include without limitation a password, PIN number, or lock combination. Examples of token-based authentication include without limitation a house key, a physical credit card, a driver's license, or a particular phone number. Examples of biometric-based authentication include without limitation a fingerprint, handwriting analysis, facial scan, hand scan, ear scan, iris scan, vascular pattern, DNA, a voice analysis, or a retinal scan.

[0214] Each type of authentication has particular advantages and disadvantages, and each provides a different level of security. For example, it is generally harder to create a false fingerprint that matches someone else's than it is to overhear someone's password and repeat it. Each type of authentication also requires a different type of data to be known to the authenticating authority in order to verify someone using that form of authentication.

[0215] As used herein, "authentication" will refer broadly to the overall process of verifying someone's identity to be who he says he is. An "authentication technique" will refer to a particular type of authentication based upon a particular piece of knowledge, physical token, or biometric reading. "Authentication data" refers to information which is sent to or otherwise demonstrated to an authentication authority in order to establish identity. "Enrollment data" will refer to the data which is initially submitted to an authentication authority in order to establish a baseline for comparison with authentication data. An "authentication instance" will refer to the data associated with an attempt to authenticate by an authentication technique.

[0216] The internal protocols and communications involved in the process of authenticating a user is described with reference to FIGURE 10 above. The part of this process within which the context sensitive authentication takes place occurs within the comparison step shown as step 1045 of FIGURE 10. This step takes place within the authentication engine 215 and involves assembling the enrollment data 410 retrieved from the depository 210 and comparing the authentication data provided by the user to it. One particular embodiment of this process is shown in FIGURE 16 and described below.

[0217] The current authentication data provided by the user and the enrollment data retrieved from the depository 210 are received by the authentication engine 215 in step 1600 of FIGURE 16. Both of these sets of data may contain data which is related to separate techniques of authentication. The authentication engine 215 separates the authentication data associated with each individual authentication instance in step 1605. This is necessary so that the authentication data is compared with the appropriate subset of the enrollment data for the user (e.g. fingerprint authentication data should be compared with fingerprint enrollment data, rather than password enrollment data).

[0218] Generally, authenticating a user involves one or more individual authentication instances, depending on which authentication techniques are available to the user. These methods are limited by the enrollment data which were provided by the user during his enrollment process (if the user did not provide a retinal scan when enrolling, he will not be able to authenticate himself using a retinal scan), as well as the means which may be currently available to the user (e.g. if the user does not have a fingerprint reader at his current location, fingerprint authentication will not be practical). In some cases, a single authentication instance may be sufficient to authenticate a user; however, in certain circumstances a combination of

multiple authentication instances may be used in order to more confidently authenticate a user for a particular transaction.

[0219] Each authentication instance consists of data related to a particular authentication technique (e.g. fingerprint, password, smart card, etc.) and the circumstances which surround the capture and delivery of the data for that particular technique. For example, a particular instance of attempting to authenticate via password will generate not only the data related to the password itself, but also circumstantial data, known as "metadata", related to that password attempt. This circumstantial data includes information such as: the time at which the particular authentication instance took place, the network address from which the authentication information was delivered, as well as any other information as is known to those of skill in the art which may be determined about the origin of the authentication data (the type of connection, the processor serial number, etc.).

[0220] In many cases, only a small amount of circumstantial metadata will be available. For example, if the user is located on a network which uses proxies or network address translation or another technique which masks the address of the originating computer, only the address of the proxy or router may be determined. Similarly, in many cases information such as the processor serial number will not be available because of either limitations of the hardware or operating system being used, disabling of such features by the operator of the system, or other limitations of the connection between the user's system and the trust engine 110.

[0221] As shown in FIGURE 16, once the individual authentication instances represented within the authentication data are extracted and separated in step 1605, the authentication engine 215 evaluates each instance for its reliability in indicating that the user is who he claims to be. The reliability for a single authentication instance will generally be determined based on several factors. These may be grouped as factors relating to the reliability associated with the authentication technique, which are evaluated in step 1610, and factors relating to the reliability of the particular authentication data provided, which are evaluated in step 1815. The first group includes without limitation the inherent reliability of the authentication technique being used, and the reliability of the enrollment data being used with that method. The second group includes without limitation the degree of match between the enrollment data and the data provided with the authentication instance, and the metadata associated with that authentication instance. Each of these factors may vary independently of the others.

[0222] The inherent reliability of an authentication technique is based on how hard it is for an imposter to provide someone else's correct data, as well as the overall error rates for the authentication technique. For passwords and knowledge based authentication methods, this reliability is often fairly low because there is nothing that prevents someone from revealing their password to another person and for that second person to use that password. Even a more complex knowledge based system may have only moderate reliability since knowledge may be transferred from person to person fairly easily. Token based authentication, such as having a proper smart card or using a particular terminal to perform the authentication, is similarly of low reliability used by itself, since there is no guarantee that the right person is in possession of the proper token.

[0223] However, biometric techniques are more inherently reliable because it is generally difficult to provide someone else with the ability to use your fingerprints in a convenient manner, even intentionally. Because subverting biometric authentication techniques is more difficult, the inherent reliability of biometric methods is generally higher than that of purely knowledge or token based authentication techniques. However, even biometric techniques may have some occasions in which a false acceptance or false rejection is generated. These occurrences may be reflected by differing reliabilities for different implementations of the same biometric technique. For example, a fingerprint matching system provided by one company may provide a higher reliability than one provided by a different company because one uses higher quality optics or a better scanning resolution or some other improvement which reduces the occurrence of false acceptances or false rejections.

[0224] Note that this reliability may be expressed in different manners. The reliability is desirably expressed in some metric which can be used by the heuristics 530 and algorithms of the authentication engine 215 to calculate the confidence level of each authentication. One preferred mode of expressing these reliabilities is as a percentage or fraction. For instance, fingerprints might be assigned an inherent reliability of 97%, while passwords might only be assigned an inherent reliability of 50%. Those of skill in the art will recognize that these particular values are merely exemplary and may vary between specific implementations.

[0225] The second factor for which reliability must be assessed is the reliability of the enrollment. This is part of the "graded enrollment" process referred to above. This reliability factor reflects the reliability of the identification provided during the initial enrollment process.

For instance, if the individual initially enrolls in a manner where they physically produce evidence of their identity to a notary or other public official, and enrollment data is recorded at that time and notarized, the data will be more reliable than data which is provided over a network during enrollment and only vouched for by a digital signature or other information which is not truly tied to the individual.

[0226] Other enrollment techniques with varying levels of reliability include without limitation: enrollment at a physical office of the trust engine 110 operator; enrollment at a user's place of employment; enrollment at a post office or passport office; enrollment through an affiliated or trusted party to the trust engine 110 operator; anonymous or pseudonymous enrollment in which the enrolled identity is not yet identified with a particular real individual, as well as such other means as are known in the art.

[0227] These factors reflect the trust between the trust engine 110 and the source of identification provided during the enrollment process. For instance, if enrollment is performed in association with an employer during the initial process of providing evidence of identity, this information may be considered extremely reliable for purposes within the company, but may be trusted to a lesser degree by a government agency, or by a competitor. Therefore, trust engines operated by each of these other organizations may assign different levels of reliability to this enrollment.

[0228] Similarly, additional data which is submitted across a network, but which is authenticated by other trusted data provided during a previous enrollment with the same trust engine 110 may be considered as reliable as the original enrollment data was, even though the latter data were submitted across an open network. In such circumstances, a subsequent notarization will effectively increase the level of reliability associated with the original enrollment data. In this way for example, an anonymous or pseudonymous enrollment may then be raised to a full enrollment by demonstrating to some enrollment official the identity of the individual matching the enrolled data.

[0229] The reliability factors discussed above are generally values which may be determined in advance of any particular authentication instance. This is because they are based upon the enrollment and the technique, rather than the actual authentication. In one embodiment, the step of generating reliability based upon these factors involves looking up previously determined values for this particular authentication technique and the enrollment data of the user. In a

further aspect of an advantageous embodiment of the present invention, such reliabilities may be included with the enrollment data itself. In this way, these factors are automatically delivered to the authentication engine 215 along with the enrollment data sent from the depository 210.

[0230] While these factors may generally be determined in advance of any individual authentication instance, they still have an effect on each authentication instance which uses that particular technique of authentication for that user. Furthermore, although the values may change over time (e.g. if the user re-enrolls in a more reliable fashion), they are not dependent on the authentication data itself. By contrast, the reliability factors associated with a single specific instance's data may vary on each occasion. These factors, as discussed below, must be evaluated for each new authentication in order to generate reliability scores in step 1815.

[0231] The reliability of the authentication data reflects the match between the data provided by the user in a particular authentication instance and the data provided during the authentication enrollment. This is the fundamental question of whether the authentication data matches the enrollment data for the individual the user is claiming to be. Normally, when the data do not match, the user is considered to not be successfully authenticated, and the authentication fails. The manner in which this is evaluated may change depending on the authentication technique used. The comparison of such data is performed by the comparator 515 function of the authentication engine 215 as shown in FIGURE 5.

[0232] For instance, matches of passwords are generally evaluated in a binary fashion. In other words, a password is either a perfect match, or a failed match. It is usually not desirable to accept as even a partial match a password which is close to the correct password if it is not exactly correct. Therefore, when evaluating a password authentication, the reliability of the authentication returned by the comparator 515 is typically either 100% (correct) or 0% (wrong), with no possibility of intermediate values.

[0233] Similar rules to those for passwords are generally applied to token based authentication methods, such as smart cards. This is because having a smart card which has a similar identifier or which is similar to the correct one, is still just as wrong as having any other incorrect token. Therefore tokens tend also to be binary authenticators: a user either has the right token, or he doesn't.

[0234] However, certain types of authentication data, such as questionnaires and biometrics, are generally not binary authenticators. For example, a fingerprint may match a reference

fingerprint to varying degrees. To some extent, this may be due to variations in the quality of the data captured either during the initial enrollment or in subsequent authentications. (A fingerprint may be smudged or a person may have a still healing scar or burn on a particular finger.) In other instances the data may match less than perfectly because the information itself is somewhat variable and based upon pattern matching. (A voice analysis may seem close but not quite right because of background noise, or the acoustics of the environment in which the voice is recorded, or because the person has a cold.) Finally, in situations where large amounts of data are being compared, it may simply be the case that much of the data matches well, but some doesn't. (A ten-question questionnaire may have resulted in eight correct answers to personal questions, but two incorrect answers.) For any of these reasons, the match between the enrollment data and the data for a particular authentication instance may be desirably assigned a partial match value by the comparator 515. In this way, the fingerprint might be said to be a 85% match, the voice print a 65% match, and the questionnaire an 80% match, for example.

[0235] This measure (degree of match) produced by the comparator 515 is the factor representing the basic issue of whether an authentication is correct or not. However, as discussed above, this is only one of the factors which may be used in determining the reliability of a given authentication instance. Note also that even though a match to some partial degree may be determined, that ultimately, it may be desirable to provide a binary result based upon a partial match. In an alternate mode of operation, it is also possible to treat partial matches as binary, i.e. either perfect (100%) or failed (0%) matches, based upon whether or not the degree of match passes a particular threshold level of match. Such a process may be used to provide a simple pass/fail level of matching for systems which would otherwise produce partial matches.

[0236] Another factor to be considered in evaluating the reliability of a given authentication instance concerns the circumstances under which the authentication data for this particular instance are provided. As discussed above, the circumstances refer to the metadata associated with a particular authentication instance. This may include without limitation such information as: the network address of the authenticator, to the extent that it can be determined; the time of the authentication; the mode of transmission of the authentication data (phone line, cellular, network, etc.); and the serial number of the system of the authenticator.

[0237] These factors can be used to produce a profile of the type of authentication that is normally requested by the user. Then, this information can be used to assess reliability in at least

two manners. One manner is to consider whether the user is requesting authentication in a manner which is consistent with the normal profile of authentication by this user. If the user normally makes authentication requests from one network address during business days (when she is at work) and from a different network address during evenings or weekends (when she is at home), an authentication which occurs from the home address during the business day is less reliable because it is outside the normal authentication profile. Similarly, if the user normally authenticates using a fingerprint biometric and in the evenings, an authentication which originates during the day using only a password is less reliable.

[0238] An additional way in which the circumstantial metadata can be used to evaluate the reliability of an instance of authentication is to determine how much corroboration the circumstance provides that the authenticator is the individual he claims to be. For instance, if the authentication comes from a system with a serial number known to be associated with the user, this is a good circumstantial indicator that the user is who they claim to be. Conversely, if the authentication is coming from a network address which is known to be in Los Angeles when the user is known to reside in London, this is an indication that this authentication is less reliable based on its circumstances.

[0239] It is also possible that a cookie or other electronic data may be placed upon the system being used by a user when they interact with a vendor system or with the trust engine 110. This data is written to the storage of the system of the user and may contain an identification which may be read by a Web browser or other software on the user system. If this data is allowed to reside on the user system between sessions (a "persistent cookie"), it may be sent with the authentication data as further evidence of the past use of this system during authentication of a particular user. In effect, the metadata of a given instance, particularly a persistent cookie, may form a sort of token based authenticator itself.

[0240] Once the appropriate reliability factors based on the technique and data of the authentication instance are generated as described above in steps 1610 and 1615 respectively, they are used to produce an overall reliability for the authentication instance provided in step 1620. One means of doing this is simply to express each reliability as a percentage and then to multiply them together.

[0241] For example, suppose the authentication data is being sent in from a network address known to be the user's home computer completely in accordance with the user's past

authentication profile (100%), and the technique being used is fingerprint identification (97%), and the initial finger print data was roistered through the user's employer with the trust engine 110 (90%), and the match between the authentication data and the original fingerprint template in the enrollment data is very good (99%). The overall reliability of this authentication instance could then be calculated as the product of these reliabilities: 100% * 97% * 90% * 99% - 86.4% reliability.

[0242] This calculated reliability represents the reliability of one single instance of authentication. The overall reliability of a single authentication instance may also be calculated using techniques which treat the different reliability factors differently, for example by using formulas where different weights are assigned to each reliability factor. Furthermore, those of skill in the art will recognize that the actual values used may represent values other than percentages and may use non-arithmetic systems. One embodiment may include a module used by an authentication requestor to set the weights for each factor and the algorithms used in establishing the overall reliability of the authentication instance.

[0243] The authentication engine 215 may use the above techniques and variations thereof to determine the reliability of a single authentication instance, indicated as step 1620. However, it may be useful in many authentication situations for multiple authentication instances to be provided at the same time. For example, while attempting to authenticate himself using the system of the present invention, a user may provide a user identification, fingerprint authentication data, a smart card, and a password. In such a case, three independent authentication instances are being provided to the trust engine 110 for evaluation. Proceeding to step 1625, if the authentication engine 215 determines that the data provided by the user includes more than one authentication instance, then each instance in turn will be selected as shown in step 1630 and evaluated as described above in steps 1610, 1615 and 1620.

[0244] Note that many of the reliability factors discussed may vary from one of these instances to another. For instance, the inherent reliability of these techniques is likely to be different, as well as the degree of match provided between the authentication data and the enrollment data. Furthermore, the user may have provided enrollment data at different times and under different circumstances for each of these techniques, providing different enrollment reliabilities for each of these instances as well. Finally, even though the circumstances under which the data for each of these instances is being submitted is the same, the use of such techniques may each fit the

profile of the user differently, and so may be assigned different circumstantial reliabilities. (For example, the user may normally use their password and fingerprint, but not their smart card.)

[0245] As a result, the final reliability for each of these authentication instances may be different from One another. However, by using multiple instances together, the overall confidence level for the authentication will tend to increase.

[0246] Once the authentication engine has performed steps 1610 through 1620 for all of the authentication instances provided in the authentication data, the reliability of each instance is used in step 1635 to evaluate the overall authentication confidence level. This process of combining the individual authentication instance reliabilities into the authentication confidence level may be modeled by various methods relating the individual reliabilities produced, and may also address the particular interaction between some of these authentication techniques. (For example, multiple knowledge-based systems such as passwords may produce less confidence than a single password and even a fairly weak biometric, such as a basic voice analysis.)

[0247] One means in which the authentication engine 215 may combine the reliabilities of multiple concurrent authentication instances to generate a final confidence level is to multiply the unreliability of each instance to arrive at a total unreliability. The unreliability is generally the complementary percentage of the reliability. For example, a technique which is 84% reliable is 16% unreliable. The three authentication instances described above (fingerprint, smart card, password)which produce reliabilities of 86%, 75%, and 72% would have corresponding unreliabilities of (100- 86)%, (100- 75)% and (100- 72)%, or 14%, 25%, and 28%, respectively. By multiplying these unreliabilities, we get a cumulative unreliability of 14% * 25% * 28% - .98% unreliability, which corresponds to a reliability of 99.02%.

[0248] In an additional mode of operation, additional factors and heuristics 530 may be applied within the authentication engine 215 to account for the interdependence of various authentication techniques. For example, if someone has unauthorized access to a particular home computer, they probably have access to the phone line at that address as well. Therefore, authenticating based on an originating phone number as well as upon the serial number of the authenticating system does not add much to the overall confidence in the authentication. However, knowledge based authentication is largely independent of token based authentication (i.e. if someone steals your cellular phone or keys, they are no more likely to know your PIN or password than if they hadn't).

[0249] Furthermore, different vendors or other authentication requestors may wish to weigh different aspects of the authentication differently. This may include the use of separate weighing factors or algorithms used in calculating the reliability of individual instances as well as the use of different means to evaluate authentication events with multiple instances.

[0250] For instance, vendors for certain types of transactions, for instance corporate email systems, may desire to authenticate primarily based upon heuristics and other circumstantial data by default. Therefore, they may apply high weights to factors related to the metadata and other profile related information associated with the circumstances surrounding authentication events. This arrangement could be used to ease the burden on users during normal operating hours, by not requiring more from the user than that he be logged on to the correct machine during business hours. However, another vendor may weigh authentications coming from a particular technique most heavily, for instance fingerprint matching, because of a policy decision that such a technique is most suited to authentication for the particular vendor's purposes.

[0251] Such varying weights may be defined by the authentication requestor in generating the authentication request and sent to the trust engine 110 with the authentication request in one mode of operation. Such options could also be set as preferences during an initial enrollment process for the authentication requestor and stored within the authentication engine in another mode of operation.

[0252] Once the authentication engine 215 produces an authentication confidence level for the authentication data provided, this confidence level is used to complete the authentication request in step 1640, and this information is forwarded from the authentication engine 215 to the transaction engine 205 for inclusion in a message to the authentication requestor.

[0253] The process described above is merely exemplary, and those of skill in the art will recognize that the steps need not be performed in the order shown or that only certain of the steps are desired to be performed, or that a variety of combinations of steps may be desired. Furthermore, certain steps, such as the evaluation of the reliability of each authentication instance provided, may be carried out in parallel with one another if circumstances permit.

[0254] In a further aspect of this invention, a method is provided to accommodate conditions when the authentication confidence level produced by the process described above fails to meet the required trust level of the vendor or other party requiring the authentication. In circumstances such as these where a gap exists between the level of confidence provided and the

level of trust desired, the operator of the trust engine 110 is in a position to provide opportunities for one or both parties to provide alternate data or requirements in order to close this trust gap. This process will be referred to as "trust arbitrage" herein.

[0255] Trust arbitrage may take place within a framework of cryptographic authentication as described above with reference to FIGURES 10 and 11. As shown therein, a vendor or other party will request authentication of a particular user in association with a particular transaction. In one circumstance, the vendor simply requests an authentication, either positive or negative, and after receiving appropriate data from the user, the trust engine 110 will provide such a binary authentication. In circumstances such as these, the degree of confidence required in order to secure a positive authentication is determined based upon preferences set within the trust engine 110.

[0256] However, it is also possible that the vendor may request a particular level of trust in order to complete a particular transaction. This required level may be included with the authentication request (e.g. authenticate this user to 98% confidence) or may be determined by the trust engine 110 based on other factors associated with the transaction (i.e. authenticate this user as appropriate for this transaction). One such factor might be the economic value of the transaction. For transactions which have greater economic value, a higher degree of trust may be required. Similarly, for transactions with high degrees of risk a high degree of trust may be required. Conversely, for transactions which are either of low risk or of low value, lower trust levels may be required by the vendor or other authentication requestor.

[0257] The process of trust arbitrage occurs between the steps of the trust engine 110 receiving the authentication data in step 1050 of FIGURE 10 and the return of an authentication result to the vendor in step 1055 of FIGURE 10. Between these steps, the process which leads to the evaluation of trust levels and the potential trust arbitrage occurs as shown in FIGURE 17. In circumstances where simple binary authentication is performed, the process shown in FIGURE 17 reduces to having the transaction engine 205 directly compare the authentication data provided with the enrollment data for the identified user as discussed above with reference to FIGURE 10, flagging any difference as a negative authentication.

[0258] As shown in FIGURE 17, the first step after receiving the data in step 1050 is for the transaction engine 205 to determine the trust level which is required for a positive authentication for this particular transaction in step 1710. This step may be performed by one of several

different methods. The required trust level may be specified to the trust engine 110 by the authentication requestor at the time when the authentication request is made. The authentication requestor may also set a preference in advance which is stored within the depository 210 or other storage which is accessible by the transaction engine 205. This preference may then be read and used each time an authentication request is made by this authentication requestor. The preference may also be associated with a particular user as a security measure such that a particular level of trust is always required in order to authenticate that user, the user preference being stored in the depository 210 or other storage media accessible by the transaction engine 205. The required level may also be derived by the transaction engine 205 or authentication engine 215 based upon information provided in the authentication request, such as the value and risk level of the transaction to be authenticated.

[0259] In one mode of operation, a policy management module or other software which is used when generating the authentication request is used to specify the required degree of trust for the authentication of the transaction. This may be used to provide a series of rules to follow when assigning the required level of trust based upon the policies which are specified within the policy management module. One advantageous mode of operation is for such a module to be incorporated with the web server of a vendor in order to appropriately determine required level of trust for transactions initiated with the vendor's web server. In this way, transaction requests from users may be assigned a required trust level in accordance with the policies of the vendor and such information may be forwarded to the trust engine 110 along with the authentication request.

[0260] This required trust level correlates with the degree of certainty that the vendor wants to have that the individual authenticating is in fact who he identifies himself as. For example, if the transaction is one where the vendor wants a fair degree of certainty because goods are changing hands, the vendor may require a trust level of 85%. For situation where the vendor is merely authenticating the user to allow him to view members only content or exercise privileges on a chat room, the downside risk may be small enough that the vendor requires only a 60% trust level. However, to enter into a production contract with a value of tens of thousands of dollars, the vendor may require a trust level of 99% or more.

[0261] This required trust level represents a metric to which the user must authenticate himself in order to complete the transaction. If the required trust level is 85% for example, the user must

provide authentication to the trust engine 110 sufficient for the trust engine 110 to say with 85% confidence that the user is who they say they are. It is the balance between this required trust level and the authentication confidence level which produces either a positive authentication (to the satisfaction of the vendor) or a possibility of trust arbitrage.

[0262] As shown in FIGURE 17, after the transaction engine 205 receives the required trust level, it compares in step 1720 the required trust level to the authentication confidence level which the authentication engine 215 calculated for the current authentication (as discussed with reference to FIGURE 16). If the authentication confidence level is higher than the required trust level for the transaction in step 1730, then the process moves to step 1740 where a positive authentication for this transaction is produced by the transaction engine 205. A message to this effect will then be inserted into the authentication results and returned to the vendor by the transaction engine 205 as shown in step 1055 (see FIGURE 10).

[0263] However, if the authentication confidence level does not fulfill the required trust level in step 1730, then a confidence gap exists for the current authentication, and trust arbitrage is conducted in step 1750. Trust arbitrage is described more completely with reference to FIGURE 18 below. This process as described below takes place within the transaction engine 205 of the trust engine 110. Because no authentication or other cryptographic operations are needed to execute trust arbitrage (other than those required for the SSL communication between the transaction engine 205 and other components), the process may be performed outside the authentication engine 215. However, as will be discussed below, any reevaluation of authentication data or other cryptographic or authentication events will require the transaction engine 205 to resubmit the appropriate data to the authentication engine 215. Those of skill in the art will recognize that the trust arbitrage process could alternately be structured to take place partially or entirely within the authentication engine 215 itself.

[0264] As mentioned above, trust arbitrage is a process where the trust engine 110 mediates a negotiation between the vendor and user in an attempt to secure a positive authentication where appropriate. As shown in step 1805, the transaction engine 205 first determines whether or not the current situation is appropriate for trust arbitrage. This may be determined based upon the circumstances of the authentication, e.g. whether this authentication has already been through multiple cycles of arbitrage, as well as upon the preferences of either the vendor or user, as will be discussed further below.

[0265] In such circumstances where arbitrage is not possible, the process proceeds to step 1810 where the transaction engine 205 generates a negative authentication and then inserts it into the authentication results which are sent to the vendor in step 1055 (see FIGURE 10). One limit which may be advantageously used to prevent authentications from pending indefinitely is to set a time-out period from the initial authentication request. In this way, any transaction which is not positively authenticated within the time limit is denied further arbitrage and negatively authenticated. Those of skill in the art will recognize that such a time limit may vary depending upon the circumstances of the transaction and the desires of the user and vendor. Limitations may also be placed upon the number of attempts that may be made at providing a successful authentication. Such limitations may be handled by an attempt limiter 535 as shown in FIGURE 5.

[0266] If arbitrage is not prohibited in step 1805, the transaction engine 205 will then engage in negotiation with one or both of the transacting parties. The transaction engine 205 may send a message to the user requesting some form of additional authentication in order to boost the authentication confidence level produced as shown in step 1820. In the simplest form, this may simply indicates that authentication was insufficient. A request to produce one or more additional authentication instances to improve the overall confidence level of the authentication may also be sent.

[0267] If the user provides some additional authentication instances in step 1825, then the transaction engine 205 adds these authentication instances to the authentication data for the transaction and forwards it to the authentication engine 215 as shown in step 1015 (see FIGURE 10), and the authentication is reevaluated based upon both the pre-existing authentication instances for this transaction and the newly provided authentication instances.

[0268] An additional type of authentication may be a request from the trust engine 110 to make some form of person-to-person contact between the trust engine 110 operator (or a trusted associate) and the user, for example, by phone call. This phone call or other non-computer authentication can be used to provide personal contact with the individual and also to conduct some form of questionnaire based authentication. This also may give the opportunity to verify an originating telephone number and potentially a voice analysis of the user when he calls in. Even if no additional authentication data can be provided, the additional context associated with the user's phone number may improve the reliability of the authentication context. Any revised

data or circumstances based upon this phone call are fed into the trust engine 110 for use in consideration of the authentication request.

[0269] Additionally, in step 1820 the trust engine 110 may provide an opportunity for the user to purchase insurance, effectively buying a more confident authentication. The operator of the trust engine 110 may, at times, only want to make such an option available if the confidence level of the authentication is above a certain threshold to begin with. In effect, this user side insurance is a way for the trust engine 110 to vouch for the user when the authentication meets the normal required trust level of the trust engine 110 for authentication, but does not meet the required trust level of the vendor for this transaction. In this way, the user may still successfully authenticate to a high level as may be required by the vendor, even though he only has authentication instances which produce confidence sufficient for the trust engine 110.

[0270] This function of the trust engine 110 allows the trust engine 110 to vouch for someone who is authenticated to the satisfaction of the trust engine 110, but not of the vendor. This is analogous to the function performed by a notary in adding his signature to a document in order to indicate to someone reading the document at a later time that the person whose signature appears on the document is in fact the person who signed it. The signature of the notary testifies to the act of signing by the user. In the same way, the trust engine is providing an indication that the person transacting is who they say they are.

[0271] However, because the trust engine 110 is artificially boosting the level of confidence provided by the user, there is a greater risk to the trust engine 110 operator, since the user is not actually meeting the required trust level of the vendor. The cost of the insurance is designed to offset the risk of a false positive authentication to the trust engine 110 (who may be effectively notarizing the authentications of the user). The user pays the trust engine 110 operator to take the risk of authenticating to a higher level of confidence than has actually been provided.

[0272] Because such an insurance system allows someone to effectively buy a higher confidence rating from the trust engine 110, both vendors and users may wish to prevent the use of user side insurance in certain transactions. Vendors may wish to limit positive authentications to circumstances where they know that actual authentication data supports the degree of confidence which they require and so may indicate to the trust engine 110 that user side insurance is not to be allowed. Similarly, to protect his online identity, a user may wish to prevent the use of user side insurance on his account, or may wish to limit its use to situations

where the authentication confidence level without the insurance is higher than a certain limit. This may be used as a security measure to prevent someone from overhearing a password or stealing a smart card and using them to falsely authenticate to a low level of confidence, and then purchasing insurance to produce a very high level of (false) confidence. These factors may be evaluated in determining whether user side insurance is allowed.

[0273] If user purchases insurance in step 1840, then the authentication confidence level is adjusted based upon the insurance purchased in step 1845, and the authentication confidence level and required trust level are again compared in step 1730 (see FIGURE 17). The process continues from there, and may lead to either a positive authentication in step 1740 (see FIGURE 17), or back into the trust arbitrage process in step 1750 for either further arbitrage (if allowed) or a negative authentication in step 1810 if further arbitrage is prohibited.

[0274] In addition to sending a message to the user in step 1820, the transaction engine 205 may also send a message to the vendor in step 1830 which indicates that a pending authentication is currently below the required trust level. The message may also offer various options on how to proceed to the vendor. One of these Options is to simply inform the vendor of what the current authentication confidence level is and ask if the vendor wishes to maintain their current unfulfilled required trust level. This may be beneficial because in some cases, the vendor may have independent means for authenticating the transaction or may have been using a default set of requirements which generally result in a higher required level being initially specified than is actually needed for the particular transaction at hand.

[0275] For instance, it may be standard practice that all incoming purchase order transactions with the vendor are expected to meet a 98% trust level. However, if an order was recently discussed by phone between the vendor and a long-standing customer, and immediately thereafter the transaction is authenticated, but only to a 93% confidence level, the vendor may wish to simply lower the acceptance threshold for this transaction, because the phone call effectively provides additional authentication to the vendor. In certain circumstances, the vendor may be willing to lower their required trust level, but not all the way to the level of the current authentication confidence. For instance, the vendor in the above example might consider that the phone call prior to the order might merit a 4% reduction in the degree of trust needed; however, this is still greater than the 93% confidence produced by the user.

[0276] If the vendor does adjust their required trust level in step 1835, then the authentication confidence level produced by the authentication and the required trust level are compared in step 1730 (see FIGURE 17). If the confidence level now exceeds the required trust level, a positive authentication may be generated in the transaction engine 205 in step 1740 (see FIGURE 17). If not, further arbitrage may be attempted as discussed above if it is permitted.

[0277] In addition to requesting an adjustment to the required trust level, the transaction engine 205 may also offer vendor side insurance to the vendor requesting the authentication. This insurance serves a similar purpose to that described above for the user side insurance. Here, however, rather than the cost corresponding to the risk being taken by the trust engine 110 in authenticating above the actual authentication confidence level produced, the cost of the insurance corresponds to the risk being taken by the vendor in accepting a lower trust level in the authentication.

[0278] Instead of just lowering their actual required trust level, the vendor has the option of purchasing insurance to protect itself from the additional risk associated with a lower level of trust in the authentication of the user. As described above, it may be advantageous for the vendor to only consider purchasing such insurance to cover the trust gap in conditions where the existing authentication is already above a certain threshold.

[0279] The availability of such vendor side insurance allows the vendor the option to either: lower his trust requirement directly at no additional cost to himself, bearing the risk of a false authentication himself (based on the lower trust level required); or, buying insurance for the trust gap between the authentication confidence level and his requirement, with the trust engine 110 operator bearing the risk of the lower confidence level which has been provided. By purchasing the insurance, the vendor effectively keeps his high trust level requirement; because the risk of a false authentication is shifted to the trust engine 110 operator.

[0280] If the vendor purchases insurance in step 1840, the authentication confidence level and required trust levels are compared in step 1730 (see FIGURE 17), and the process continues as described above.

[0281] Note that it is also possible that both the user and the vendor respond to messages from the trust engine 110. Those of skill in the art will recognize that there are multiple ways in which such situations can be handled. One advantageous mode of handling the possibility of multiple responses is simply to treat the responses in a first-come, first-served manner. For example, if

the vendor responds with a lowered required trust level and immediately thereafter the user also purchases insurance to raise his authentication level, the authentication is first reevaluated based upon the lowered trust requirement from the vendor. If the authentication is now positive, the user's insurance purchase is ignored. In another advantageous mode of operation, the user might only be charged for the level of insurance required to meet the new, lowered trust requirement of the vendor (if a trust gap remained even with the lowered vendor trust requirement).

[0282] If no response from either party is received during the trust arbitrage process at step 1850 within the time limit set for the authentication, the arbitrage is reevaluated in step 1805. This effectively begins the arbitrage process again. If the time limit was final or other circumstances prevent further arbitrage in step 1805, a negative authentication is generated by the transaction engine 205 in step 1810 and returned to the vendor in step 1055 (see FIGURE 10). If not, new messages may be sent to the user and vendor, and the process may be repeated as desired.

[0283] Note that for certain types of transactions, for instance, digitally signing documents which are not part of a transaction, there may not necessarily be a vendor or other third party; therefore the transaction is primarily between the user and the trust engine 110. In circumstances such as these, the trust engine 110 will have its own required trust level which must be satisfied in order to generate a positive authentication. However, in such circumstances, it will often not be desirable for the trust engine 110 to offer insurance to the user in order for him to raise the confidence of his own signature.

[0284] The process described above and shown in FIGURES 16-18 may be carried out using various communications modes as described above with reference to the trust engine 110. For instance, the messages may be web-based and sent using SSL connections between the trust engine 110 and applets downloaded in real time to browsers running on the user or vendor systems. In an alternate mode of operation, certain dedicated applications may be in use by the user and vendor which facilitate such arbitrage and insurance transactions. In another alternate mode of operation, secure email operations may be used to mediate the arbitrage described above, thereby allowing deferred evaluations and batch processing of authentications. Those of skill in the art will recognize that different communications modes may be used as are appropriate for the circumstances and authentication requirements of the vendor.

[0285] The following description with reference to FIGURE 19 describes a sample transaction which integrates the various aspects of the present invention as described above. This example illustrates the overall process between a user and a vendor as mediates by the trust engine 110. Although the various steps and components as described in detail above may be used to carry out the following transaction, the process illustrated focuses on the interaction between the trust engine 110, user and vendor.

[0286] The transaction begins when the user, while viewing web pages online, fills out an order form on the web site of the vendor in step 1900. The user wishes to submit this order form to the vendor, signed with his digital signature. In order to do this, the user submits the order form with his request for a signature to the trust engine 110 in step 1905. The user will also provide authentication data which will be used as described above to authenticate his identity.

[0287] In step 1910 the authentication data is compared to the enrollment data by the trust engine 110 as discussed above, and if a positive authentication is produced, the hash of the order form, signed with the private key of the user, is forwarded to the vendor along with the order form itself.

[0288] The vendor receives the signed form in step 1915, and then the vendor will generate an invoice or other contract related to the purchase to be made in step 1920. This contract is sent back to the user with a request for a signature in step 1925. The vendor also sends an authentication request for this contract transaction to the trust engine 110 in step 1930 including a hash of the contract which will be signed by both parties. To allow the contract to be digitally signed by both parties, the vendor also includes authentication data for itself so that the vendor's signature upon the contract can later be verified if necessary.

[0289] As discussed above, the trust engine 110 then verifies the authentication data provided by the vendor to confirm the vendor's identity, and if the data produces a positive authentication in step 1935, continues with step 1955 when the data is received from the user. If the vendor's authentication data does not match the enrollment data of the vendor to the desired degree, a message is returned to the vendor requesting further authentication. Trust arbitrage may be performed here if necessary, as described above, in order for the vendor to successfully authenticate itself to the trust engine 110.

[0290] When the user receives the contract in step 1940, he reviews it, generates authentication data to sign it if it is acceptable in step 1945, and then sends a hash of the contract and his

authentication data to the trust engine 110 in step 1950. The trust engine 110 verifies the authentication data in step 1955 and if the authentication is good, proceeds to process the contract as described below. As discussed above with reference to FIGURES 17 and 18, trust arbitrage may be performed as appropriate to close any trust gap which exists between the authentication confidence level and the required authentication level for the transaction.

[0291] The trust engine 110 signs the hash of the contract with the user's private key, and sends this signed hash to the vendor in step 1960, signing the complete message on its own behalf, i.e., including a hash of the complete message (including the user's signature) encrypted with the private key 510 of the trust engine 110. This message is received by the vendor in step 1965. The message represents a signed contract (hash of contract encrypted using user's private key) and a receipt from the trust engine 110 (the hash of the message including the signed contract, encrypted using the trust engine 110's private key).

[0292] The trust engine 110 similarly prepares a hash of the contract with the vendor's private key in step 1970, and forwards this to the user, signed by the trust engine 110. In this way, the user also receives a copy of the contract, signed by the vendor, as well as a receipt, signed by the trust engine 110, for delivery of the signed contract in step 1975.

[0293] In addition to the foregoing, an additional aspect of the invention provides a cryptographic Service Provider Module (SPM) which may be available to a client side application as a means to access functions provided by the trust engine 110 described above. One advantageous way to provide such a service is for the cryptographic SPM is to mediate communications between a third party Application Programming Interface (API) and a trust engine 110 which is accessible via a network or other remote connection. A sample cryptographic SPM is described below with reference to FIGURE 20.

[0294] For example, on a typical system, a number of API's are available to programmers. Each API provides a set of function calls which may be made by an application 2000 running upon the system. Examples of API's which provide programming interfaces suitable for cryptographic functions, authentication functions, and other security function include the Cryptographic API (CAPI) 2010 provided by Microsoft with its Windows operating systems, and the Common Data Security Architecture (CDSA), sponsored by IBM, Intel and other members of the Open Group. CAPI will be used as an exemplary security API in the discussion

that follows. However, the cryptographic SPM described could be used with CDSA or other security API's as are known in the art.

[0295] This API is used by a user system 105 or vendor system 120 when a call is made for a cryptographic function. Included among these functions may be requests associated with performing various cryptographic operations, such as encrypting a document with a particular key, signing a document, requesting a digital certificate, verifying a signature upon a signed document, and such other cryptographic functions as are described herein or known to those of skill in the art.

[0296] Such cryptographic functions are normally performed locally to the system upon which CAPI 2010 is located. This is because generally the functions called require the use of either resources of the local user system 105, such as a fingerprint reader, or software functions which are programmed using libraries which are executed on the local machine. Access to these local resources is normally provided by one or more Service Provider Modules (SPM's) 2015, 2020 as referred to above which provide resources with which the cryptographic functions are carried out. Such SPM's may include software libraries 2015 to perform encrypting or decrypting operations, or drivers and applications 2020 which are capable of accessing specialized hardware 2025, such as biometric scanning devices. In much the way that CAPI 2010 provides functions which may be used by an application 2000 of the system 105, the SPM's 2015, 2020 provide CAPI with access to the lower level functions and resources associated with the available services upon the system.

[0297] In accordance with the invention, it is possible to provide a cryptographic SPM 2030 which is capable of accessing the cryptographic functions provided by the trust engine 110 and making these functions available to an application 2000 through CAPI 2010. Unlike embodiments where CAPI 2010 is only able to access resources which are locally available through SPM's 2015, 2020, a cryptographic SPM 2030 as described herein would be able to submit requests for cryptographic operations to a remotely-located, network-accessible trust engine 110 in order to perform the operations desired.

[0298] For instance, if an application 2000 has a need for a cryptographic operation, such as signing a document, the application 2000 makes a function call to the appropriate CAPI 2010 function. CAPI 2010 in turn will execute this function, making use of the resources which are made available to it by the SPM's 2015, 2020 and the cryptographic SPM 2030. In the case of a

digital signature function, the cryptographic SPM 2030 will generate an appropriate request which will be sent to the trust engine 110 across the communication link 125.

[0299] The operations which occur between the cryptographic SPM 2030 and the trust engine 110 are the same operations that would be possible between any other system and the trust engine 110. However, these functions are effectively made available to a user system 105 through CAPI 2010 such that they appear to be locally available upon the user system 105 itself. However, unlike ordinary SPM's 2015, 2020, the functions are being carried out on the remote trust engine 110 and the results relayed to the cryptographic SPM 2030 in response to appropriate requests across the communication link 125.

[0300] This cryptographic SPM 2030 makes a number of operations available to the user system 105 or a vendor system 120 which might not otherwise be available. These functions include without limitation: encryption and decryption of documents; issuance of digital certificates; digital signing of documents; verification of digital signatures; and such other operations as will be apparent to those of skill in the art.

[0301] In a separate embodiment, the present invention comprises a complete system for performing the data securing methods of the present invention on any data set. The computer system of this embodiment comprises a data splitting module that comprises the functionality shown in FIGURE 8 and described herein. In one embodiment of the present invention, the data splitting module, sometimes referred to herein as a secure data parser, comprises a parser program or software suite which comprises data splitting, encryption and decryption, reconstitution or reassembly functionality. This embodiment may further comprise a data storage facility or multiple data storage facilities, as well. The data splitting module, or secure data parser, comprises a cross-platform software module suite which integrates within an electronic infrastructure, or as an add-on to any application which requires the ultimate security of its data elements. This parsing process operates on any type of data set, and on any and all file types, or in a database on any row, column or cell of data in that database.

[0302] The parsing process of the present invention may, in one embodiment, be designed in a modular tiered fashion, and any encryption process is suitable for use in the process of the present invention. The modular tiers of the parsing and splitting process of the present invention may include, but are not limited to, 1) cryptographic split, dispersed and securely stored in multiple locations; 2) encrypt, cryptographically split, dispersed and securely stored in multiple

locations; 3) encrypt, cryptographically split, encrypt each share, then dispersed and securely stored in multiple locations; and 4) encrypt, cryptographically split, encrypt each share with a different type of encryption than was used in the first step, then dispersed and securely stored in multiple locations.

[0303] The process comprises, in one embodiment, splitting of the data according to the contents of a generated random number, or key and performing the same cryptographic splitting of the key used in the encryption of splitting of the data to be secured into two or more portions, or shares, of parsed and split data, and in one embodiment, preferably into four or more portions of parsed and split data, encrypting all of the portions, then scattering and storing these portions back into the database, or relocating them to any named device, fixed or removable, depending on the requestor's need for privacy and security. Alternatively, in another embodiment, encryption may occur prior to the splitting of the data set by the splitting module or secure data parser. The original data processed as described in this embodiment is encrypted and obfuscated and is secured. The dispersion of the encrypted elements, if desired, can be virtually anywhere, including, but not limited to, a single server or data storage device, or among separate data storage facilities or devices. Encryption key management in one embodiment may be included within the software suite, or in another embodiment may be integrated into an existing infrastructure or any other desired location.

[0304] A cryptographic split (cryptosplit) partitions the data into N number of shares. The partitioning can be on any size unit of data, including an individual bit, bits, bytes, kilobytes, megabytes, or larger units, as well as any pattern or combination of data unit sizes whether predetermined or randomly generated. The units can also be of different sized, based on either a random or predetermined set of values. This means the data can be viewed as a sequence of these units. In this manner the size of the data units themselves may render the data more secure, for example by using one or more predetermined or randomly generated pattern, sequence or combination of data unit sizes. The units are then distributed (either randomly or by a predetermined set of values) into the N shares. This distribution could also involve a shuffling of the order of the units in the shares. It is readily apparent to those of ordinary skill in the art that the distribution of the data units into the shares may be performed according to a wide variety of possible selections, including but not limited to size-fixed, predetermined sizes, or one or more

combination, pattern or sequence of data unit sizes that are predetermined or randomly generated.

[0305] One example of this cryptographic split process, or cryptosplit, would be to consider the data to be 23 bytes in size, with the data unit size chosen to be one byte, and with the number of shares selected to be 4. Each byte would be distributed into one of the 4 shares. Assuming a random distribution, a key would be obtained to create a sequence of 23 random numbers (r1, r2, r3 through r23), each with a value between 1 and 4 corresponding to the four shares. Each of the units of data (in this example 23 individual bytes of data) is associated with one of the 23 random numbers corresponding to one of the four shares. The distribution of the bytes of data into the four shares would occur by placing the first byte of the data into share number r1, byte two into share r2, byte three into share r3, through the 23rd byte of data into share r23. It is readily apparent to those of ordinary skill in the art that a wide variety of other possible steps or combination or sequence of steps, including the size of the data units, may be used in the cryptosplit process of the present invention, and the above example is a non-limiting description of one process for cryptosplitting data. To recreate the original data, the reverse operation would be performed.

[0306] In another embodiment of the cryptosplit process of the present invention, an option for the cryptosplitting process is to provide sufficient redundancy in the shares such that only a subset of the shares are needed to reassemble or restore the data to its original or useable form. As a non-limiting example, the cryptosplit may be done as a "3 of 4" cryptosplit such that only three of the four shares are necessary to reassemble or restore the data to its original or useable form. This is also referred to as a "M of N cryptosplit" wherein N is the total number of shares, and M is at least one less than N. It is readily apparent to those of ordinary skill in the art that there are many possibilities for creating this redundancy in the cryptosplitting process of the present invention.

[0307] In one embodiment of the cryptosplitting process of the present invention, each unit of data is stored in two shares, the primary share and the backup share. Using the "3 of 4" cryptosplitting process described above, any one share can be missing, and this is sufficient to reassemble or restore the original data with no missing data units since only three of the total four shares are required. As described herein, a random number is generated that corresponds to one of the shares. The random number is associated with a data unit, and stored in the

corresponding share, based on a key. One key is used, in this embodiment, to generate the primary and backup share random number. As described herein for the cryptosplitting process of the present invention, a set of random numbers (also referred to as primary share numbers) from 0 to 3 are generated equal to the number of data units. Then another set of random numbers is generated (also referred to as backup share numbers) from 1 to 3 equal to the number of data units. Each unit of data is then associated with a primary share number and a backup share number. Alternatively, a set of random numbers may be generated that is fewer than the number of data units, and repeating the random number set, but this may reduce the security of the sensitive data. The primary share number is used to determine into which share the data unit is stored. The backup share number is combined with the primary share number to create a third share number between 0 and 3, and this number is used to determine into which share the data unit is stored. In this example, the equation to determine the third share number is:
(primary share number + backup share number) MOD 4 = third share number.

[0308] In the embodiment described above where the primary share number is between 0 and 3, and the backup share number is between 1 and 3 ensures that the third share number is different from the primary share number. This results in the data unit being stored in two different shares. It is readily apparent to those of ordinary skill in the art that there are many ways of performing redundant cryptosplitting and non-redundant cryptosplitting in addition to the embodiments disclosed herein. For example, the data units in each share could be shuffled utilizing a different algorithm. This data unit shuffling may be performed as the original data is split into the data units, or after the data units are placed into the shares, or after the share is full, for example.

[0309] The various cryptosplitting processes and data shuffling processes described herein, and all other embodiments of the cryptosplitting and data shuffling methods of the present invention may be performed on data units of any size, including but not limited to, as small as an individual bit, bits, bytes, kilobytes, megabytes or larger.

[0310] An example of one embodiment of source code that would perform the cryptosplitting process described herein is:

DATA [1:24] – array of bytes with the data to be split
SHARES[0:3; 1:24] – 2-dimensionalarray with each row representing one of the shares
RANDOM[1:24] – array random numbers in the range of 0..3

```
S1 = 1;
S2 = 1;
S3 = 1;
S4 = 1;

For J = 1 to 24 do
        Begin
        IF RANDOM[J[ ==0 then
                Begin
                SHARES[1,S1] = DATA [J];
                S1 = S1 + 1;
                End
        ELSE IF RANDOM[J[ ==1 then
                Begin
                SHARES[2,S2] = DATA [J];
                S2 = S2 + 1;
                END
        ELSE IF RANDOM[J[ ==2 then
                Begin
                Shares[3,S3] = data [J];
                S3 = S3 + 1;
                End
        Else   begin
                Shares[4,S4] = data [J];
                S4 = S4 + 1;
                End;
        END;
```

[0311] An example of one embodiment of source code that would perform the cryptosplitting RAID process described herein is:

[0312] Generate two sets of numbers, PrimaryShare is 0 to 3, BackupShare is 1 to 3. Then put each data unit into share[primaryshare[1]] and share[(primaryshare[1]+backupshare[1]) mod 4,

with the same process as in cryptosplitting described above. This method will be scalable to any size N, where only N-1 shares are necessary to restore the data.

[0313] The retrieval, recombining, reassembly or reconstituting of the encrypted data elements may utilize any number of authentication techniques, including, but not limited to, biometrics, such as fingerprint recognition, facial scan, hand scan, iris scan, retinal scan, ear scan, vascular pattern recognition or DNA analysis. The data splitting and/or parser modules of the present invention may be integrated into a wide variety of infrastructure products or applications as desired.

[0314] Traditional encryption technologies known in the art rely on one or more key used to encrypt the data and render it unusable without the key. The data, however, remains whole and intact and subject to attack. The secure data parser of the present invention, in one embodiment, addresses this problem by performing a cryptographic parsing and splitting of the encrypted file into two or more portions or shares, and in another embodiment, preferably four or more shares, adding another layer of encryption to each share of the data, then storing the shares in different physical and/or logical locations. When one or more data shares are physically removed from the system, either by using a removable device, such as a data storage device, or by placing the share under another party's control, any possibility of compromise of secured data is effectively removed.

[0315] An example of one embodiment of the secure data parser of the present invention and an example of how it may be utilized is shown in FIGURE 21 and described below. However, it is readily apparent to those of ordinary skill in the art that the secure data parser of the present invention may be utilized in a wide variety of ways in addition to the non-limiting example below. As a deployment option, and in one embodiment, the secure data parser may be implemented with external session key management or secure internal storage of session keys. Upon implementation, a Parser Master Key will be generated which will be used for securing the application and for encryption purposes. It should be also noted that the incorporation of the Parser Master key in the resulting secured data allows for a flexibility of sharing of secured data by individuals within a workgroup, enterprise or extended audience.

[0316] As shown in Figure 21, this embodiment of the present invention shows the steps of the process performed by the secure data parser on data to store the session master key with the parsed data:

[0317] 1. Generating a session master key and encrypt the data using RS1 stream cipher.

[0318] 2. Separating the resulting encrypted data into four shares or portions of parsed data according to the pattern of the session master key.

[0319] 3. In this embodiment of the method, the session master key will be stored along with the secured data shares in a data depository. Separating the session master key according to the pattern of the Parser Master Key and append the key data to the encrypted parsed data.

[0320] 4. The resulting four shares of data will contain encrypted portions of the original data and portions of the session master key. Generate a stream cipher key for each of the four data shares.

[0321] 5. Encrypting each share, then store the encryption keys in different locations from the encrypted data portions or shares: Share 1 gets Key 4, Share 2 gets Key 1, Share 3 gets Key 2, Share 4 gets Key 3.

[0322] To restore the original data format, the steps are reversed.

[0323] It is readily apparent to those of ordinary skill in the art that certain steps of the methods described herein may be performed in different order, or repeated multiple times, as desired. It is also readily apparent to those skilled in the art that the portions of the data may be handled differently from one another. For example, multiple parsing steps may be performed on only one portion of the parsed data. Each portion of parsed data may be uniquely secured in any desirable way provided only that the data may be reassembled, reconstituted, reformed, decrypted or restored to its original or other usable form.

[0324] As shown in FIGURE 22 and described herein, another embodiment of the present invention comprises the steps of the process performed by the secure data parser on data to store the session master key data in one or more separate key management table:

[0325] 1. Generating a session master key and encrypt the data using RS1 stream cipher.

[0326] 2. Separating the resulting encrypted data into four shares or portions of parsed data according to the pattern of the session master key.

[0327] 3. In this embodiment of the method of the present invention, the session master key will be stored in a separate key management table in a data depository. Generating a unique transaction ID for this transaction. Storing the transaction ID and session master key in a separate key management table. Separating the transaction ID according to the pattern of the Parser Master Key and append the data to the encrypted parsed or separated data.

[0328] 4. The resulting four shares of data will contain encrypted portions of the original data and portions of the transaction ID.

[0329] 5. Generating a stream cipher key for each of the four data shares.

[0330] 6. Encrypting each share, then store the encryption keys in different locations from the encrypted data portions or shares: Share 1 gets Key 4, Share 2 gets Key 1, Share 3 gets Key 2, Share 4 gets Key 3.

[0331] To restore the original data format, the steps are reversed.

[0332] It is readily apparent to those of ordinary skill in the art that certain steps of the method described herein may be performed in different order, or repeated multiple times, as desired. It is also readily apparent to those skilled in the art that the portions of the data may be handled differently from one another. For example, multiple separating or parsing steps may be performed on only one portion of the parsed data. Each portion of parsed data may be uniquely secured in any desirable way provided only that the data may be reassembled, reconstituted, reformed, decrypted or restored to its original or other usable form.

[0333] As shown in Figure 23, this embodiment of the present invention shows the steps of the process performed by the secure data parser on data to store the session master key with the parsed data:

[0334] 1. Accessing the parser master key associated with the authenticated user

[0335] 2. Generating a unique Session Master key

[0336] 3. Derive an Intermediary Key from an exclusive OR function of the Parser Master Key and Session Master key

[0337] 4. Optional encryption of the data using an existing or new encryption algorithm keyed with the Intermediary Key.

[0338] 5. Separating the resulting optionally encrypted data into four shares or portions of parsed data according to the pattern of the Intermediary key.

[0339] 6. In this embodiment of the method, the session master key will be stored along with the secured data shares in a data depository. Separating the session master key according to the pattern of the Parser Master Key and append the key data to the optionally encrypted parsed data shares.

[0340] 7. The resulting multiple shares of data will contain optionally encrypted portions of the original data and portions of the session master key.

[0341] 8. Optionally generate an encryption key for each of the four data shares.

[0342] 9. Optionally encrypting each share with an existing or new encryption algorithm, then store the encryption keys in different locations from the encrypted data portions or shares: for example, Share 1 gets Key 4, Share 2 gets Key 1, Share 3 gets Key 2, Share 4 gets Key 3.

[0343] To restore the original data format, the steps are reversed.

[0344] It is readily apparent to those of ordinary skill in the art that certain steps of the methods described herein may be performed in different order, or repeated multiple times, as desired. It is also readily apparent to those skilled in the art that the portions of the data may be handled differently from one another. For example, multiple parsing steps may be performed on only one portion of the parsed data. Each portion of parsed data may be uniquely secured in any desirable way provided only that the data may be reassembled, reconstituted, reformed, decrypted or restored to its original or other usable form.

[0345] As shown in FIGURE 24 and described herein, another embodiment of the present invention comprises the steps of the process performed by the secure data parser on data to store the session master key data in one or more separate key management table:

[0346] 1. Accessing the Parser Master Key associated with the authenticated user

[0347] 2. Generating a unique Session Master Key

[0348] 3. Derive an Intermediary Key from an exclusive OR function of the Parser Master Key and Session Master key

[0349] 4. Optionally encrypt the data using an existing or new encryption algorithm keyed with the Intermediary Key.

[0350] 5. Separating the resulting optionally encrypted data into four shares or portions of parsed data according to the pattern of the Intermediary Key.

[0351] 6. In this embodiment of the method of the present invention, the session master key will be stored in a separate key management table in a data depository. Generating a unique transaction ID for this transaction. Storing the transaction ID and session master key in a separate key management table or passing the Session Master Key and transaction ID back to the calling program for external management. Separating the transaction ID according to the pattern of the Parser Master Key and append the data to the optionally encrypted parsed or separated data.

[0352] 7. The resulting four shares of data will contain optionally encrypted portions of the original data and portions of the transaction ID.

[0353] 8. Optionally generate an encryption key for each of the four data shares.

[0354] 9. Optionally encrypting each share, then store the encryption keys in different locations from the encrypted data portions or shares. For example: Share 1 gets Key 4, Share 2 gets Key 1, Share 3 gets Key 2, Share 4 gets Key 3.

[0355] To restore the original data format, the steps are reversed.

[0356] It is readily apparent to those of ordinary skill in the art that certain steps of the method described herein may be performed in different order, or repeated multiple times, as desired. It is also readily apparent to those skilled in the art that the portions of the data may be handled differently from one another. For example, multiple separating or parsing steps may be performed on only one portion of the parsed data. Each portion of parsed data may be uniquely secured in any desirable way provided only that the data may be reassembled, reconstituted, reformed, decrypted or restored to its original or other usable form.

[0357] A wide variety of encryption methodologies are suitable for use in the methods of the present invention, as is readily apparent to those skilled in the art. The One Time Pad algorithm, is often considered one of the most secure encryption methods, and is suitable for use in the method of the present invention. Using the One Time Pad algorithm requires that a key be generated which is as long as the data to be secured. The use of this method may be less desirable in certain circumstances such as those resulting in the generation and management of very long keys because of the size of the data set to be secured. In the One-Time Pad (OTP) algorithm, the simple exclusive-or function, XOR, is used. For two binary streams x and y of the same length, x XOR y means the bitwise exclusive-or of x and y.

[0358] At the bit level is generated:

0 XOR 0 = 0

0 XOR 1 = 1

1 XOR 0 = 1

1 XOR 1 = 0

[0359] An example of this process is described herein for an n-byte secret, s, (or data set) to be split. The process will generate an n-byte random value, a, and then set:

b = a XOR s.

[0360] Note that one can derive "s" via the equation:

s = a XOR b.

[0361] The values a and b are referred to as shares or portions and are placed in separate depositories. Once the secret s is split into two or more shares, it is discarded in a secure manner.

[0362] The secure data parser of the present invention may utilize this function, performing multiple XOR functions incorporating multiple distinct secret key values: K1, K2, K3, Kn, K5. At the beginning of the operation, the data to be secured is passed through the first encryption operation, secure data = data XOR secret key 5:

S = D XOR K5

[0363] In order to securely store the resulting encrypted data in, for example, four shares, S1, S2, S3, Sn, the data is parsed and split into "n" segments, or shares, according to the value of K5. This operation results in "n" pseudorandom shares of the original encrypted data. Subsequent XOR functions may then be performed on each share with the remaining secret key values, for example: Secure data segment 1 = encrypted data share 1 XOR secret key 1:

SD1 = S1 XOR K1

SD2 = S2 XOR K2

SD3 = S3 XOR K3

SDn = Sn XOR Kn.

[0364] In one embodiment, it may not be desired to have any one depository contain enough information to decrypt the information held there, so the key required to decrypt the share is stored in a different data depository:

Depository 1: SD1, Kn

Depository 2: SD2, K1

Depository 3: SD3, K2

Depository n: SDn, K3.

[0365] Additionally, appended to each share may be the information required to retrieve the original session encryption key, K5. Therefore, in the key management example described herein, the original session master key is referenced by a transaction ID split into "n" shares according to the contents of the installation dependant Parser Master Key (TID1, TID2, TID3, TIDn):

Depository 1: SD1, Kn, TID1

Depository 2: SD2, K1, TID2

Depository 3: SD3, K2, TID3

Depository n: SDn, K3, TIDn.

[0366] In the incorporated session key example described herein, the session master key is split into "n" shares according to the contents of the installation dependant Parser Master Key (SK1, SK2, SK3, SKn):

Depository 1: SD1, Kn, SK1

Depository 2: SD2, K1, SK2

Depository 3: SD3, K2, SK3

Depository n: SDn, K3, SKn.

[0367] Unless all four shares are retrieved, the data cannot be reassembled according to this example. Even if all four shares are captured, there is no possibility of reassembling or restoring the original information without access to the session master key and the Parser Master Key.

[0368] This example has described an embodiment of the method of the present invention, and also describes, in another embodiment, the algorithm used to place shares into depositories so that shares from all depositories can be combined to form the secret authentication material. The computations needed are very simple and fast. However, with the One Time Pad (OTP) algorithm there may be circumstances that cause it to be less desirable, such as a large data set to be secured, because the key size is the same size as the data to be stored. Therefore, there would be a need to store and transmit about twice the amount of the original data which may be less desirable under certain circumstances.

Stream Cipher RS1

[0369] The stream cipher RS1 splitting technique is very similar to the OTP splitting technique described herein. Instead of an n-byte random value, an n' = min(n, 16)-byte random value is generated and used to key the RS1 Stream Cipher algorithm. The advantage of the RS1 Stream Cipher algorithm is that a pseudorandom key is generated from a much smaller seed number. The speed of execution of the RS1 Stream Cipher encryption is also rated at approximately 10 times the speed of the well known in the art Triple DES encryption without compromising security. The RS1 Stream Cipher algorithm is well known in the art, and may be used to

generate the keys used in the XOR function. The RS1 Stream Cipher algorithm is interoperable with other commercially available stream cipher algorithms, such as the RC4™ stream cipher algorithm of RSA Security, Inc and is suitable for use in the methods of the present invention.

[0370] Using the key notation above, K1 thru K5 are now an n' byte random values and we set:

SD1 = S1 XOR E(K1)

SD2 = S2 XOR E(K2)

SD3 = S3 XOR E(K3)

SDn = Sn XOR E(Kn)

where E(K1) thru E(Kn) are the first n' bytes of output from the RS1 Stream Cipher algorithm keyed by K1 thru Kn. The shares are now placed into data depositories as described herein.

[0371] In this stream cipher RS1 algorithm, the required computations needed are nearly as simple and fast as the OTP algorithm. The benefit in this example using the RS1 Stream Cipher is that the system needs to store and transmit on average only about 16 bytes more than the size of the original data to be secured per share. When the size of the original data is more than 16 bytes, this RS1 algorithm is more efficient than the OTP algorithm because it is simply shorter. It is readily apparent to those of ordinary skill in the art that a wide variety of encryption methods or algorithms are suitable for use in the present invention, including, but not limited to RS1, OTP, RC4™, Triple DES and AES.

[0372] There are major advantages provided by the data security methods and computer systems of the present invention over traditional encryption methods. One advantage is the security gained from moving shares of the data to different locations on one or more data depositories or storage devices, that may be in different logical, physical or geographical locations. When the shares of data are split physically and under the control of different personnel, for example, the possibility of compromising the data is greatly reduced.

[0373] Another advantage provided by the methods and system of the present invention is the combination of the steps of the method of the present invention for securing data to provide a comprehensive process of maintaining security of sensitive data. The data is encrypted with a secure key and split into one or more shares, and in one embodiment, four shares, according to the secure key. The secure key is stored safely with a reference pointer which is secured into four shares according to a secure key. The data shares are then encrypted individually and the keys are stored safely with different encrypted shares. When combined, the entire process for

securing data according to the methods disclosed herein becomes a comprehensive package for data security.

[0374] The data secured according to the methods of the present invention is readily retrievable and restored, reconstituted, reassembled, decrypted, or otherwise returned into its original or other suitable form for use. In order to restore the original data, the following items may be utilized:

[0375] 1. All shares or portions of the data set.

[0376] 2. Knowledge of and ability to reproduce the process flow of the method used to secure the data.

[0377] 3. Access to the session master key.

[0378] 4. Access to the Parser Master Key.

[0379] Therefore, it may be desirable to plan a secure installation wherein at least one of the above elements may be physically separated from the remaining components of the system (under the control of a different system administrator for example).

[0380] Protection against a rogue application invoking the data securing methods application may be enforced by use of the Parser Master Key. A mutual authentication handshake between the secure data parser and the application may be required in this embodiment of the present invention prior to any action taken.

[0381] The security of the system dictates that there be no "backdoor" method for recreation of the original data. For installations where data recovery issues may arise, the secure data parser can be enhanced to provide a mirror of the four shares and session master key depository. Hardware options such as RAID (redundant array of inexpensive disks, used to spread information over several disks) and software options such as replication can assist as well in the data recovery planning.

Key Management

[0382] In one embodiment of the present invention, the data securing method uses three sets of keys for an encryption operation. Each set of keys may have individual key storage, retrieval, security and recovery options, based on the installation. The keys that may be used, include, but are not limited to:

The Parser Master Key

[0383] This key is an individual key associated with the installation of the secure data parser. It is installed on the server on which the secure data parser has been deployed. There are a variety of options suitable for securing this key including, but not limited to, a smart card, separate hardware key store, standard key stores, custom key stores or within a secured database table, for example.

The Session Master Key

[0384] A Session Master Key may be generated each time data is secured. The Session Master Key is used to encrypt the data prior to the parsing and splitting operations. It may also be incorporated (if the Session Master Key is not integrated into the parsed data) as a means of parsing the encrypted data. The Session Master Key may be secured in a variety of manners, including, but not limited to, a standard key store, custom key store, separate database table, or secured within the encrypted shares, for example.

The Share Encryption Keys

[0385] For each share or portions of a data set that is created, an individual Share Encryption Key may be generated to further encrypt the shares. The Share Encryption Keys may be stored in different shares than the share that was encrypted.

[0386] It is readily apparent to those of ordinary skill in the art that the data securing methods and computer system of the present invention are widely applicable to any type of data in any setting or environment. In addition to commercial applications conducted over the Internet or between customers and vendors, the data securing methods and computer systems of the present invention are highly applicable to non-commercial or private settings or environments. Any data set that is desired to be kept secure from any unauthorized user may be secured using the methods and systems described herein. For example, access to a particular database within a company or organization may be advantageously restricted to only selected users by employing the methods and systems of the present invention for securing data. Another example is the generation, modification or access to documents wherein it is desired to restrict access or prevent unauthorized or accidental access or disclosure outside a group of selected individuals, computers or workstations. These and other examples of the ways in which the methods and systems of data securing of the present invention are applicable to any non-commercial or commercial environment or setting for any setting, including, but not limited to any organization, government agency or corporation.

[0387] In another embodiment of the present invention, the data securing method uses three sets of keys for an encryption operation. Each set of keys may have individual key storage, retrieval, security and recovery options, based on the installation. The keys that may be used, include, but are not limited to:

1. The Parser Master Key

[0388] This key is an individual key associated with the installation of the secure data parser. It is installed on the server on which the secure data parser has been deployed. There are a variety of options suitable for securing this key including, but not limited to, a smart card, separate hardware key store, standard key stores, custom key stores or within a secured database table, for example.

2. The Session Master Key

[0389] A Session Master Key may be generated each time data is secured. The Session Master Key is used in conjunction with the Parser Master key to derive the Intermediary Key. The Session Master Key may be secured in a variety of manners, including, but not limited to, a standard key store, custom key store, separate database table, or secured within the encrypted shares, for example.

3. The Intermediary Key

[0390] An Intermediary Key may be generated each time data is secured. The Intermediary Key is used to encrypt the data prior to the parsing and splitting operation. It may also be incorporated as a means of parsing the encrypted data.

4. The Share Encryption Keys

[0391] For each share or portions of a data set that is created, an individual Share Encryption Key may be generated to further encrypt the shares. The Share Encryption Keys may be stored in different shares than the share that was encrypted.

[0392] It is readily apparent to those of ordinary skill in the art that the data securing methods and computer system of the present invention are widely applicable to any type of data in any setting or environment. In addition to commercial applications conducted over the Internet or between customers and vendors, the data securing methods and computer systems of the present invention are highly applicable to non-commercial or private settings or environments. Any data set that is desired to be kept secure from any unauthorized user may be secured using the methods and systems described herein. For example, access to a particular database within a

company or organization may be advantageously restricted to only selected users by employing the methods and systems of the present invention for securing data. Another example is the generation, modification or access to documents wherein it is desired to restrict access or prevent unauthorized or accidental access or disclosure outside a group of selected individuals, computers or workstations. These and other examples of the ways in which the methods and systems of data securing of the present invention are applicable to any non-commercial or commercial environment or setting for any setting, including, but not limited to any organization, government agency or corporation.

Workgroup, Project, Individual PC/Laptop or Cross Platform Data Security

[0393] The data securing methods and computer systems of the present invention are also useful in securing data by workgroup, project, individual PC/Laptop and any other platform that is in use in, for example, businesses, offices, government agencies, or any setting in which sensitive data is created, handled or stored. The present invention provides methods and computer systems to secure data that is known to be sought after by organizations, such as the U.S. Government, for implementation across the entire government organization or between governments at a state or federal level.

[0394] The data securing methods and computer systems of the present invention provide the ability to not only parse and split flat files but also data fields, sets and or table of any type. Additionally, all forms of data are capable of being secured under this process, including, but not limited to, text, video, images, biometrics and voice data. Scalability, speed and data throughput of the methods of securing data of the present invention are only limited to the hardware the user has at their disposal.

[0395] In one embodiment of the present invention, the data securing methods are utilized as described below in a workgroup environment. In one embodiment, as shown in FIGURE 23 and described below, the Workgroup Scale data securing method of the present invention uses the private key management functionality of the TrustEngine to store the user/group relationships and the associated private keys (Parser Group Master Keys) necessary for a group of users to share secure data. The method of the present invention has the capability to secure data for an enterprise, workgroup, or individual user, depending on how the Parser Master Key was deployed.

[0396] In one embodiment, additional key management and user/group management programs may be provided, enabling wide scale workgroup implementation with a single point of administration and key management. Key generation, management and revocation are handled by the single maintenance program, which all become especially important as the number of users increase. In another embodiment, key management may also be set up across one or several different system administrators, which may not allow any one person or group to control data as needed. This allows for the management of secured data to be obtained by roles, responsibilities, membership, rights, etc., as defined by an organization, and the access to secured data can be limited to just those who are permitted or required to have access only to the portion they are working on, while others, such as managers or executives, may have access to all of the secured data. This embodiment allows for the sharing of secured data among different groups within a company or organization while at the same time only allowing certain selected individuals, such as those with the authorized and predetermined roles and responsibilities, to observe the data as a whole. In addition, this embodiment of the methods and systems of the present invention also allows for the sharing of data among, for example, separate companies, or separate departments or divisions of companies, or any separate organization departments, groups, agencies, or offices, or the like, of any government or organization or any kind, where some sharing is required, but not any one party may be permitted to have access to all the data. Particularly apparent examples of the need and utility for such a method and system of the present invention are to allow sharing, but maintain security, in between government areas, agencies and offices, and between different divisions, departments or offices of a large company, or any other organization, for example.

[0397] An example of the applicability of the methods of the present invention on a smaller scale is as follows. A Parser Master key is used as a serialization or branding of the secure data parser to an organization. As the scale of use of the Parser Master key is reduced from the whole enterprise to a smaller workgroup, the data securing methods described herein are used to share files within groups of users.

[0398] In the example shown in FIGURE 25 and described below, there are six users defined along with their title or role within the organization. The side bar represents five possible groups that the users can belong to according to their role. The arrow represents membership by the user in one or more of the groups.

[0399] When configuring the secure data parser for use in this example, the system administrator accesses the user and group information from the operating system by a maintenance program. This maintenance program generates and assigns Parser Group Master Keys to users based on their membership in groups.

[0400] In this example, there are three members in the Senior Staff group. For this group, the actions would be:

[0401] 1. Access Parser Group Master Key for the Senior Staff group (generate a key if not available);

[0402] 2. Generate a digital certificate associating CEO with the Senior Staff group;

[0403] 3. Generate a digital certificate associating CFO with the Senior Staff group;

[0404] 4. Generate a digital certificate associating Vice President, Marketing with the Senior Staff group.

[0405] The same set of actions would be done for each group, and each member within each group. When the maintenance program is complete, the Parser Group Master Key becomes a shared credential for each member of the group. Revocation of the assigned digital certificate may be done automatically when a user is removed from a group through the maintenance program without affecting the remaining members of the group.

[0406] Once the shared credentials have been defined, the parsing and splitting process remains the same. When a file, document or data element is to be secured, the user is prompted for the target group to be used when securing the data. The resulting secured data is only accessible by other members of the target group. This functionality of the methods and systems of the present invention may be used with any other computer system or software platform, any may be, for example, integrated into existing application programs or used standalone for file security.

[0407] It is readily apparent to those of ordinary skill in the art that any one or combination of encryption algorithms are suitable for use in the methods and systems of the present invention. For example, the encryption steps may, in one embodiment, be repeated to produce a multi-layered encryption scheme. In addition, a different encryption algorithm, or combination of encryption algorithms, may be used in repeat encryption steps such that different encryption algorithms are applied to the different layers of the multi-layered encryption scheme. As such,

the encryption scheme itself may become a component of the methods of the present invention for securing sensitive data from unauthorized use or access.

[0408] The secure data parser may include as an internal component, as an external component, or as both an error-checking component. For example, in one suitable approach, as portions of data are created using the secure data parser in accordance with the present invention, to assure the integrity of the data within a portion, a hash value is taken at preset intervals within the portion and is appended to the end of the interval. The hash value is a predictable and reproducible numeric representation of the data. If any bit within the data changes, the hash value would be different. A scanning module (either as a stand-alone component external to the secure data parser or as an internal component) may then scan the portions of data generated by the secure data parser. Each portion of data (or alternatively, less than all portions of data according to some interval or by a random or pseudo-random sampling) is compared to the appended hash value or values and an action may be taken. This action may include a report of values that match and do not match, an alert for values that do not match, or invoking of some external or internal program to trigger a recovery of the data. For example, recovery of the data could be performed by invoking a recovery module based on the concept that fewer than all portions may be needed to generate original data in accordance with the present invention.

[0409] Any other suitable integrity checking may be implemented using any suitable integrity information appended anywhere in all or a subset of data portions. Integrity information may include any suitable information that can be used to determine the integrity of data portions. Examples of integrity information may include hash values computed based on any suitable parameter (e.g., based on respective data portions), digital signature information, message authentication code (MAC) information, any other suitable information, or any combination thereof.

[0410] The secure data parser of the present invention may be used in any suitable application. Namely, the secure data parser described herein has a variety of applications in different areas of computing and technology. Several such areas are discussed below. It will be understood that these are merely illustrative in nature and that any other suitable applications may make use of the secure data parser. It will further be understood that the examples described are merely illustrative embodiments that may be modified in any suitable way in order to satisfy any suitable desires. For example, parsing and splitting may be based on any suitable units, such as

by bits, by bytes, by kilobytes, by megabytes, by any combination thereof, or by any other suitable unit.

[0411] The secure data parser of the present invention may be used to implement secure physical tokens, whereby data stored in a physical token may be required in order to access additional data stored in another storage area. In one suitable approach, a physical token, such as a compact USB flash drive, a floppy disk, an optical disk, a smart card, or any other suitable physical token, may be used to store one of at least two portions of parsed data in accordance with the present invention. In order to access the original data, the USB flash drive would need to be accessed. Thus, a personal computer holding one portion of parsed data would need to have the USB flash drive, having the other portion of parsed data, attached before the original data can be accessed. FIGURE 26 illustrates this application. Storage area 2500 includes a portion of parsed data 2502. Physical token 2504, having a portion of parsed data 2506 would need to be coupled to storage area 2500 using any suitable communications interface 2508 (e.g., USB, serial, parallel, Bluetooth, IR, IEEE 1394, Ethernet, or any other suitable communications interface) in order to access the original data. This is useful in a situation where, for example, sensitive data on a computer is left alone and subject to unauthorized access attempts. By removing the physical token (e.g., the USB flash drive), the sensitive data is inaccessible. It will be understood that any other suitable approach for using physical tokens may be used.

[0412] The secure data parser of the present invention may be used to implement a secure authentication system whereby user enrollment data (e.g., passwords, private encryption keys, fingerprint templates, biometric data or any other suitable user enrollment data) is parsed and split using the secure data parser. The user enrollment data may be parsed and split whereby one or more portions are stored on a smart card, a government Common Access Card, any suitable physical storage device (e.g., magnetic or optical disk, USB key drive, etc.), or any other suitable device. One or more other portions of the parsed user enrollment data may be stored in the system performing the authentication. This provides an added level of security to the authentication process (e.g., in addition to the biometric authentication information obtained from the biometric source, the user enrollment data must also be obtained via the appropriate parsed and split data portion).

[0413] The secure data parser of the present invention may be integrated into any suitable existing system in order to provide the use of its functionality in each system's respective

environment. FIGURE 27 shows a block diagram of an illustrative system 2600, which may include software, hardware, or both for implementing any suitable application. System 2600 may be an existing system in which secure data parser 2602 may be retrofitted as an integrated component. Alternatively, secure data parser 2602 may be integrated into any suitable system 2600 from, for example, its earliest design stage. Secure data parser 2600 may be integrated at any suitable level of system 2600. For example, secure data parser 2602 may be integrated into system 2600 at a sufficiently back-end level such that the presence of secure data parser 2602 may be substantially transparent to an end user of system 2600. Secure data parser 2602 may be used for parsing and splitting data among one or more storage devices 2604 in accordance with the present invention. Some illustrative examples of systems having the secure data parser integrated therein are discussed below.

[0414] The secure data parser of the present invention may be integrated into an operating system kernel (e.g., Linux, Unix, or any other suitable commercial or proprietary operating system). This integration may be used to protect data at the device level whereby, for example, data that would ordinarily be stored in one or more devices is separated into a certain number of portions by the secure data parser integrated into the operating system and stored among the one or more devices. When original data is attempted to be accessed, the appropriate software, also integrated into the operating system, may recombine the parsed data portions into the original data in a way that may be transparent to the end user.

[0415] The secure data parser of the present invention may be integrated into a volume manager or any other suitable component of a storage system to protect local and networked data storage across any or all supported platforms. For example, with the secure data parser integrated, a storage system may make use of the redundancy offered by the secure data parser (i.e., which is used to implement the feature of needing fewer than all separated portions of data in order to reconstruct the original data) to protect against data loss. The secure data parser also allows all data written to storage devices, whether using redundancy or not, to be in the form of multiple portions that are generated according to the parsing of the present invention. When original data is attempted to be accessed, the appropriate software, also integrated into the volume manager or other suitable component of the storage system, may recombine the parsed data portions into the original data in a way that may be transparent to the end user.

[0416] In one suitable approach, the secure data parser of the present invention may be integrated into a RAID controller (as either hardware or software). This allows for the secure storage of data to multiple drives while maintaining fault tolerance in case of drive failure.

[0417] The secure data parser of the present invention may be integrated into a database in order to, for example, protect sensitive table information. For example, in one suitable approach, data associated with particular cells of a database table (e.g., individual cells, one or more particular columns, one or more particular rows, any combination thereof, or an entire database table) may be parsed and separated according to the present invention (e.g., where the different portions are stored on one or more storage devices at one or more locations or on a single storage device). Access to recombine the portions in order to view the original data may be granted by traditional authentication methods (e.g., username and password query).

[0418] The secure parser of the present invention may be integrated in any suitable system that involves data in motion (i.e., transfer of data from one location to another). Such systems include, for example, email, streaming data broadcasts, and wireless (e.g., WiFi) communications. With respect to email, in one suitable approach, the secure parser may be used to parse outgoing messages (i.e., containing text, binary data, or both (e.g., files attached to an email message)) and sending the different portions of the parsed data along different paths thus creating multiple streams of data. If any one of these streams of data is compromised, the original message remains secure because the system may require that more than one of the portions be combined, in accordance with the present invention, in order to generate the original data. In another suitable approach, the different portions of data may be communicated along one path sequentially so that if one portion is obtained, it may not be sufficient to generate the original data. The different portions arrive at the intended recipient's location and may be combined to generate the original data in accordance with the present invention.

[0419] FIGURES 28 and 29 are illustrative block diagrams of such email systems. FIGURE 28 shows a sender system 2700, which may include any suitable hardware, such as a computer terminal, personal computer, handheld device (e.g., PDA, Blackberry), cellular telephone, computer network, any other suitable hardware, or any combination thereof. Sender system 2700 is used to generate and/or store a message 2704, which may be, for example, an email message, a binary data file (e.g., graphics, voice, video, etc.), or both. Message 2704 is parsed and split by secure data parser 2702 in accordance with the present invention. The resultant data

portions may be communicated across one or more separate communications paths 2706 over network 2708 (e.g., the Internet, an intranet, a LAN, WiFi, Bluetooth, any other suitable hard-wired or wireless communications means, or any combination thereof) to recipient system 2710. The data portions may be communicated parallel in time or alternatively, according to any suitable time delay between the communication of the different data portions. Recipient system 2710 may be any suitable hardware as described above with respect to sender system 2700. The separate data portions carried along communications paths 2706 are recombined at recipient system 2710 to generate the original message or data in accordance with the present invention.

[0420] FIGURE 29 shows a sender system 2800, which may include any suitable hardware, such as a computer terminal, personal computer, handheld device (e.g., PDA), cellular telephone, computer network, any other suitable hardware, or any combination thereof. Sender system 2800 is used to generate and/or store a message 2804, which may be, for example, an email message, a binary data file (e.g., graphics, voice, video, etc.), or both. Message 2804 is parsed and split by secure data parser 2802 in accordance with the present invention. The resultant data portions may be communicated across a single communications paths 2806 over network 2808 (e.g., the Internet, an intranet, a LAN, WiFi, Bluetooth, any other suitable communications means, or any combination thereof) to recipient system 2810. The data portions may be communicated serially across communications path 2806 with respect to one another. Recipient system 2810 may be any suitable hardware as described above with respect to sender system 2800. The separate data portions carried along communications path 2806 are recombined at recipient system 2810 to generate the original message or data in accordance with the present invention.

[0421] It will be understood that the arrangement of FIGURES 28 and 29 are merely illustrative. Any other suitable arrangement may be used. For example, in another suitable approach, the features of the systems of FIGURES 28 and 29 may be combined whereby the multi-path approach of FIGURE 28 is used and in which one or more of communications paths 2706 are used to carry more than one portion of data as communications path 2806 does in the context of FIGURE 29.

[0422] The secure data parser may be integrated at any suitable level of a data-in motion system. For example, in the context of an email system, the secure parser may be integrated at the user-interface level (e.g., into Microsoft® Outlook), in which case the user may have control

over the use of the secure data parser features when using email. Alternatively, the secure parser may be implemented in a back-end component such as at the exchange server, in which case messages may be automatically parsed, split, and communicated along different paths in accordance with the present invention without any user intervention.

[0423] Similarly, in the case of streaming broadcasts of data (e.g., audio, video), the outgoing data may be parsed and separated into multiple streams each containing a portion of the parsed data. The multiple streams may be transmitted along one or more paths and recombined at the recipient's location in accordance with the present invention. One of the benefits of this approach is that it avoids the relatively large overhead associated with traditional encryption of data followed by transmission of the encrypted data over a single communications channel. The secure parser of the present invention allows data in motion to be sent in multiple parallel streams, increasing speed and efficiency.

[0424] It will be understand that the secure data parser may be integrated for protection of and fault tolerance of any type of data in motion through any transport medium, including, for example, wired, wireless, or physical. For example, voice over Internet protocol (VoIP) applications may make use of the secure data parser of the present invention. Wireless or wired data transport from or to any suitable personal digital assistant (PDA) devices such as Blackberries and SmartPhones may be secured using the secure data parser of the present invention. Communications using wireless 802.11 protocols for peer to peer and hub based wireless networks, satellite communications, point to point wireless communications, Internet client/server communications, or any other suitable communications may involve the data in motion capabilities of the secure data parser in accordance with the present invention. Data communication between computer peripheral device (e.g., printer, scanner, monitor, keyboard, network router, biometric authentication device (e.g., fingerprint scanner), or any other suitable peripheral device) between a computer and a computer peripheral device, between a computer peripheral device and any other suitable device, or any combination thereof may make use of the data in motion features of the present invention.

[0425] The data in motion features of the present invention may also apply to physical transportation of secure shares using for example, separate routes, vehicles, methods, any other suitable physical transportation, or any combination thereof. For example, physical transportation of data may take place on digital/magnetic tapes, floppy disks, optical disks,

physical tokens, USB drives, removable hard drives, consumer electronic devices with flash memory (e.g., Apple IPODs or other MP3 players), flash memory, any other suitable medium used for transporting data, or any combination thereof.

[0426] The secure data parser of the present invention may provide security with the ability for disaster recovery. According to the present invention, fewer than all portions of the separated data generated by the secure data parser may be necessary in order to retrieve the original data. That is, out of m portions stored, n may be the minimum number of these m portions necessary to retrieve the original data, where n <= m. For example, if each of four portions is stored in a different physical location relative to the other three portions, then, if n=2 in this example, two of the locations may be compromised whereby data is destroyed or inaccessible, and the original data may still be retrieved from the portions in the other two locations. Any suitable value for n or m may be used.

[0427] In addition, the n of m feature of the present invention may be used to create a "two man rule" whereby in order to avoid entrusting a single individual or any other entity with full access to what may be sensitive data, two or more distinct entities, each with a portion of the separated data parsed by the secure parser of the present invention may need to agree to put their portions together in order to retrieve the original data.

[0428] The secure data parser of the present invention may be used to provide a group of entities with a group-wide key that allows the group members to access particular information authorized to be accessed by that particular group. The group key may be one of the data portions generated by the secure parser in accordance with the present invention that may be required to be combined with another portion centrally stored, for example in order to retrieve the information sought. This feature allows for, for example, secure collaboration among a group. It may be applied in for example, dedicated networks, virtual private networks, intranets, or any other suitable network.

[0429] Specific applications of this use of the secure parser include, for example, coalition information sharing in which, for example, multi-national friendly government forces are given the capability to communicate operational and otherwise sensitive data on a security level authorized to each respective country over a single network or a dual network (i.e., as compared to the many networks involving relatively substantial manual processes currently used). This capability is also applicable for companies or other organizations in which information needed to

be known by one or more specific individuals (within the organization or without) may be communicated over a single network without the need to worry about unauthorized individuals viewing the information.

[0430] Another specific application includes a multi-level security hierarchy for government systems. That is, the secure parser of the present invention may provide for the ability to operate a government system at different levels of classified information (e.g., unclassified, classified, secret, top secret) using a single network. If desired, more networks may be used (e.g., a separate network for top secret), but the present invention allows for substantially fewer than current arrangement in which a separate network is used for each level of classification.

[0431] It will be understood that any combination of the above described applications of the secure parser of the present invention may be used. For example, the group key application can be used together with the data in motion security application (i.e., whereby data that is communicated over a network can only be accessed by a member of the respective group and where, while the data is in motion, it is split among multiple paths (or sent in sequential portions) in accordance with the present invention).

[0432] The secure data parser of the present invention may be integrated into any middleware application to enable applications to securely store data to different database products or to different devices without modification to either the applications or the database. Middleware is a general term for any product that allows two separate and already existing programs to communicate. For example, in one suitable approach, middleware having the secure data parser integrated, may be used to allow programs written for a particular database to communicate with other databases without custom coding.

[0433] The secure data parser of the present invention may be implemented having any combination of any suitable capabilities, such as those discussed herein. In some embodiments of the present invention, for example, the secure data parser may be implemented having only certain capabilities whereas other capabilities may be obtained through the use of external software, hardware, or both interfaced either directly or indirectly with the secure data parser.

[0434] FIGURE 30, for example, shows an illustrative implementation of the secure data parser as secure data parser 3000. Secure data parser 3000 may be implemented with very few built-in capabilities. As illustrated, secure data parser 3000 may include built-in capabilities for parsing and splitting data into portions (also referred to herein as shares) of data using module

3002 in accordance with the present invention. Secure data parser 3000 may also include built in capabilities for performing redundancy in order to be able to implement, for example, the m of n feature described above (i.e., recreating the original data using fewer than all shares of parsed and split data) using module 3004. Secure data parser 3000 may also include share distribution capabilities using module 3006 for placing the shares of data into buffers from which they are sent for communication to a remote location, for storage, etc. in accordance with the present invention. It will be understood that any other suitable capabilities may be built into secure data parser 3000.

[0435] Assembled data buffer 3008 may be any suitable memory used to store the original data (although not necessarily in its original form) that will be parsed and split by secure data parser 3000. In a splitting operation, assembled data buffer 3008 provides input to secure data parser 3008. In a restore operation, assembled data buffer 3008 may be used to store the output of secure data parser 3000.

[0436] Split shares buffers 3010 may be one or more memory modules that may be used to store the multiple shares of data that resulted from the parsing and splitting of original data. In a splitting operation, split shares buffers 3010 hold the output of the secure data parser. In a restore operation, split shares buffers hold the input to secure data parser 3000.

[0437] It will be understood that any other suitable arrangement of capabilities may be built-in for secure data parser 3000. Any additional features may be built-in and any of the features illustrated may be removed, made more robust, made less robust, or may otherwise be modified in any suitable way. Buffers 3008 and 3010 are likewise merely illustrative and may be modified, removed, or added to in any suitable way.

[0438] Any suitable modules implemented in software, hardware or both may be called by or may call to secure data parser 3000. If desired, even capabilities that are built into secure data parser 3000 may be replaced by one or more external modules. As illustrated, some external modules include random number generator 3012, cipher feedback key generator 3014, hash algorithm 3016, any one or more types of encryption 3018, and key management 3020. It will be understood that these are merely illustrative external modules. Any other suitable modules may be used in addition to or in place of those illustrated.

[0439] Cipher feedback key generator 3014 may, externally to secure data parser 3000, generate for each secure data parser operation, a unique key, or random number (using, for

example, random number generator 3012), to be used as a seed value for an operation that extends an original session key size (e.g., a value of 128, 256, 512, or 1024 bits) into a value equal to the length of the data to be parsed and split. Any suitable algorithm may be used for the cipher feedback key generation, including, for example, the AES cipher feedback key generation algorithm.

[0440] In order to facilitate integration of secure data parser 3000 and its external modules (i.e., secure data parser layer 3026) into an application layer 3024 (e.g., email application, database application, etc.), a wrapping layer that may make use of, for example, API function calls may be used. Any other suitable arrangement for facilitating integration of secure data parser layer 3026 into application layer 3024 may be used.

[0441] FIGURE 31 illustratively shows how the arrangement of FIGURE 30 may be used when a write (e.g., to a storage device), insert (e.g., in a database field), or transmit (e.g., across a network) command is issued in application layer 3024. At step 3100 data to be secured is identified and a call is made to the secure data parser. The call is passed through wrapper layer 3022 where at step 3102, wrapper layer 3022 streams the input data identified at step 3100 into assembled data buffer 3008. Also at step 3102, any suitable share information, filenames, any other suitable information, or any combination thereof may be stored (e.g., as information 3106 at wrapper layer 3022). Secure data processor 3000 then parses and splits the data it takes as input from assembled data buffer 3008 in accordance with the present invention. It outputs the data shares into split shares buffers 3010. At step 3104, wrapper layer 3022 obtains from stored information 3106 any suitable share information (i.e., stored by wrapper 3022 at step 3102) and share location(s) (e.g., from one or more configuration files). Wrapper layer 3022 then writes the output shares (obtained from split shares buffers 3010) appropriately (e.g., written to one or more storage devices, communicated onto a network, etc.).

[0442] FIGURE 32 illustratively shows how the arrangement of FIGURE 30 may be used when a read (e.g., from a storage device), select (e.g., from a database field), or receive (e.g., from a network) occurs. At step 3200, data to be restored is identified and a call to secure data parser 3000 is made from application layer 3024. At step 3202, from wrapper layer 3022, any suitable share information is obtained and share location is determined. Wrapper layer 3022 loads the portions of data identified at step 3200 into split shares buffers 3010. Secure data parser 3000 then processes these shares in accordance with the present invention (e.g., if only

three of four shares are available, then the redundancy capabilities of secure data parser 3000 may be used to restore the original data using only the three shares). The restored data is then stored in assembled data buffer 3008. At step 3204, application layer 3022 converts the data stored in assembled data buffer 3008 into its original data format (if necessary) and provides the original data in its original format to application layer 3024.

[0443] It will be understood that the parsing and splitting of original data illustrated in FIGURE 31 and the restoring of portions of data into original data illustrated in FIGURE 32 is merely illustrative. Any other suitable processes, components, or both may be used in addition to or in place of those illustrated.

[0444] FIGURE 33 is a block diagram of an illustrative process flow for parsing and splitting original data into two or more portions of data in accordance with one embodiment of the present invention. As illustrated, the original data desired to be parsed and split is plain text 3306 (i.e., the word "SUMMIT" is used as an example). It will be understood that any other type of data may be parsed and split in accordance with the present invention. A session key 3300 is generated. If the length of session key 3300 is not compatible with the length of original data 3306, then cipher feedback session key 3304 may be generated.

[0445] In one suitable approach, original data 3306 may be encrypted prior to parsing, splitting, or both. For example, as FIGURE 33 illustrates, original data 3306 may be XORed with any suitable value (e.g., with cipher feedback session key 3304, or with any other suitable value). It will be understood that any other suitable encryption technique may be used in place of or in addition to the XOR technique illustrate. It will further be understood that although FIGURE 33 is illustrated in terms of byte by byte operations, the operation may take place at the bit level or at any other suitable level. It will further be understood that, if desired, there need not be any encryption whatsoever of original data 3306.

[0446] The resultant encrypted data (or original data if no encryption took place) is then hashed to determine how to split the encrypted (or original) data among the output buckets (e.g., of which there are four in the illustrated example). In the illustrated example, the hashing takes place by bytes and is a function of cipher feedback session key 3304. It will be understood that this is merely illustrative. The hashing may be performed at the bit level, if desired. The hashing may be a function of any other suitable value besides cipher feedback session key 3304.

In another suitable approach, hashing need not be used. Rather, any other suitable technique for splitting data may be employed.

[0447] FIGURE 34 is a block diagram of an illustrative process flow for restoring original data 3306 from two or more parsed and split portions of original data 3306 in accordance with one embodiment of the present invention. The process involves hashing the portions in reverse (i.e., to the process of FIGURE 33) as a function of cipher feedback session key 3304 to restore the encrypted original data (or original data if there was no encryption prior to the parsing and splitting). The encryption key may then be used to restore the original data (i.e., in the illustrated example, cipher feedback session key 3304 is used to decrypt the XOR encryption by XORing it with the encrypted data). This the restores original data 3306.

[0448] FIGURE 35 shows how bit-splitting may be implemented in the example of FIGURES 33 and 34. A hash may be used (e.g., as a function of the cipher feedback session key, as a function of any other suitable value) to determine a bit value at which to split each byte of data. It will be understood that this is merely one illustrative way in which to implement splitting at the bit level. Any other suitable technique may be used.

[0449] It will be understood that any reference to hash functionality made herein may be made with respect to any suitable hash algorithm. These include for example, MD5 and SHA-1. Different hash algorithms may be used at different times and by different components of the present invention.

[0450] After a split point has been determined in accordance with the above illustrative procedure or through any other procedure or algorithm, a determination may be made with regard to which data portions to append each of the left and right segments. Any suitable algorithm may be used for making this determination. For example, in one suitable approach, a table of all possible distributions (e.g., in the form of pairings of destinations for the left segment and for the right segment) may be created, whereby a destination share value for each of the left and right segment may be determined by using any suitable hash function on corresponding data in the session key, cipher feedback session key, or any other suitable random or pseudo-random value, which may be generated and extended to the size of the original data. For example, a hash function of a corresponding byte in the random or pseudo-random value may be made. The output of the hash function is used to determine which pairing of destinations (i.e., one for the left segment and one for the right segment) to select from the table of all the destination

combinations. Based on this result, each segment of the split data unit is appended to the respective two shares indicated by the table value selected as a result of the hash function.

[0451] Redundancy information may be appended to the data portions in accordance with the present invention to allow for the restoration of the original data using fewer than all the data portions. For example, if two out of four portions are desired to be sufficient for restoration of data, then additional data from the shares may be accordingly appended to each share in, for example, a round-robin manner (e.g., where the size of the original data is 4MB, then share 1 gets its own shares as well as those of shares 2 and 3; share 2 gets its own share as well as those of shares 3 and 4; share 3 gets its own share as well as those of shares 4 and 1; and share 4 gets its own shares as well as those of shares 1 and 2). Any such suitable redundancy may be used in accordance with the present invention.

[0452] It will be understood that any other suitable parsing and splitting approach may be used to generate portions of data from an original data set in accordance with the present invention. For example, parsing and splitting may be randomly or pseudo-randomly processed on a bit by bit basis. A random or pseudo-random value may be used (e.g., session key, cipher feedback session key, etc.) whereby for each bit in the original data, the result of a hash function on corresponding data in the random or pseudo-random value may indicate to which share to append the respective bit. In one suitable approach the random or pseudo-random value may be generated as, or extended to, 8 times the size of the original data so that the hash function may be performed on a corresponding byte of the random or pseudo-random value with respect to each bit of the original data. Any other suitable algorithm for parsing and splitting data on a bit by bit level may be used in accordance with the present invention. It will further be appreciated that redundancy data may be appended to the data shares such as, for example, in the manner described immediately above in accordance with the present invention.

[0453] In one suitable approach, parsing and splitting need not be random or pseudo-random. Rather, any suitable deterministic algorithm for parsing and splitting data may be used. For example, breaking up the original data into sequential shares may be employed as a parsing and splitting algorithm. Another example is to parse and split the original data bit by bit, appending each respective bit to the data shares sequentially in a round-robin manner. It will further be appreciated that redundancy data may be appended to the data shares such as, for example, in the manner described above in accordance with the present invention.

[0454] In one embodiment of the present invention, after the secure data parser generates a number of portions of original data, in order to restore the original data, certain one or more of the generated portions may be mandatory. For example, if one of the portions is used as an authentication share (e.g., saved on a physical token device), and if the fault tolerance feature of the secure data parser is being used (i.e., where fewer than all portions are necessary to restore the original data), then even though the secure data parser may have access to a sufficient number of portions of the original data in order to restore the original data, it may require the authentication share stored on the physical token device before it restores the original data. It will be understood that any number and types of particular shares may be required based on, for example, application, type of data, user, any other suitable factors, or any combination thereof.

[0455] In one suitable approach, the secure data parser or some external component to the secure data parser may encrypt one or more portions of the original data. The encrypted portions may be required to be provided and decrypted in order to restore the original data. The different encrypted portions may be encrypted with different encryption keys. For example, this feature may be used to implement a more secure "two man rule" whereby a first user would need to have a particular share encrypted using a first encryption and a second user would need to have a particular share encrypted using a second encryption key. In order to access the original data, both users would need to have their respective encryption keys and provide their respective portions of the original data. In one suitable approach, a public key may be used to encrypt one or more data portions that may be a mandatory share required to restore the original data. A private key may then be used to decrypt the share in order to be used to restore to the original data.

[0456] Any such suitable paradigm may be used that makes use of mandatory shares where fewer than all shares are needed to restore original data.

[0457] In one suitable embodiment of the present invention, distribution of data into a finite number of shares of data may be processed randomly or pseudo-randomly such that from a statistical perspective, the probability that any particular share of data receives a particular unit of data is equal to the probability that any one of the remaining shares will receive the unit of data. As a result, each share of data will have an approximately equal amount of data bits.

[0458] According to another embodiment of the present invention, each of the finite number of shares of data need not have an equal probability of receiving units of data from the parsing and

splitting of the original data. Rather certain one or more shares may have a higher or lower probability than the remaining shares. As a result, certain shares may be larger or smaller in terms of bit size relative to other shares. For example, in a two-share scenario, one share may have a 1% probability of receiving a unit of data whereas the second share has a 99% probability. It should follow, therefore that once the data units have been distributed by the secure data parser among the two share, the first share should have approximately 1% of the data and the second share 99%. Any suitable probabilities may be used in accordance with the present invention.

[0459] It will be understood that the secure data parser may be programmed to distribute data to shares according to an exact (or near exact) percentage as well. For example, the secure data parser may be programmed to distribute 80% of data to a first share and the remaining 20% of data to a second share.

[0460] According to another embodiment of the present invention, the secure data parser may generate data shares, one or more of which have predefined sizes. For example, the secure data parser may split original data into data portions where one of the portions is exactly 256 bits. In one suitable approach, if it is not possible to generate a data portion having the requisite size, then the secure data parser may pad the portion to make it the correct size. Any suitable size may be used.

[0461] In one suitable approach, the size of a data portion may be the size of an encryption key, a splitting key, any other suitable key, or any other suitable data element.

[0462] As previously discussed, the secure data parser may use keys in the parsing and splitting of data. For purposes of clarity and brevity, these keys shall be referred to herein as "splitting keys." For example, the Session Master Key, previously introduced, is one type of splitting key. Also, as previously discussed, splitting keys may be secured within shares of data generated by the secure data parser. Any suitable algorithms for securing splitting keys may be used to secure them among the shares of data. For example, the Shamir algorithm may be used to secure the splitting keys whereby information that may be used to reconstruct a splitting key is generated and appended to the shares of data. Any other such suitable algorithm may be used in accordance with the present invention.

[0463] Similarly, any suitable encryption keys may be secured within one or more shares of data according to any suitable algorithm such as the Shamir algorithm. For example, encryption keys used to encrypt a data set prior to parsing and splitting, encryption keys used to encrypt a

data portions after parsing and splitting, or both may be secured using, for example, the Shamir algorithm or any other suitable algorithm.

[0464] According to one embodiment of the present invention, an All or Nothing Transform (AoNT), such as a Full Package Transform, may be used to further secure data by transforming splitting keys, encryption keys, any other suitable data elements, or any combination thereof. For example, an encryption key used to encrypt a data set prior to parsing and splitting in accordance with the present invention may be transformed by an AoNT algorithm. The transformed encryption key may then be distributed among the data shares according to, for example, the Shamir algorithm or any other suitable algorithm. In order to reconstruct the encryption key, the encrypted data set must be restored (e.g., not necessarily using all the data shares if redundancy was used in accordance with the present invention) in order to access the necessary information regarding the transformation in accordance with AoNTs as is well known by one skilled in the art. When the original encryption key is retrieved, it may be used to decrypt the encrypted data set to retrieve the original data set. It will be understood that the fault tolerance features of the present invention may be used in conjunction with the AoNT feature. Namely, redundancy data may be included in the data portions such that fewer than all data portions are necessary to restore the encrypted data set.

[0465] It will be understood that the AoNT may be applied to encryption keys used to encrypt the data portions following parsing and splitting either in place of or in addition to the encryption and AoNT of the respective encryption key corresponding to the data set prior to parsing and splitting. Likewise, AoNT may be applied to splitting keys.

[0466] In one embodiment of the present invention, encryption keys, splitting keys, or both as used in accordance with the present invention may be further encrypted using, for example, a workgroup key in order to provide an extra level of security to a secured data set.

[0467] In one embodiment of the present invention, an audit module may be provided that tracks whenever the secure data parser is invoked to split data.

[0468] FIGURE 36 illustrates possible options 3600 for using the components of the secure data parser in accordance with the invention. Each combination of options is outlined below and labeled with the appropriate step numbers from FIGURE 36. The secure data parser may be modular in nature, allowing for any known algorithm to be used within each of the function blocks shown in FIGURE 36. For example, other key splitting (e.g., secret sharing) algorithms

such as Blakely may be used in place of Shamir, or the AES encryption could be replaced by other known encryption algorithms such as Triple DES. The labels shown in the example of FIGURE 36 merely depict one possible combination of algorithms for use in one embodiment of the invention. It should be understood that any suitable algorithm or combination of algorithms may be used in place of the labeled algorithms.

[0469] 1) 3610, 3612, 3614, 3615, 3616, 3617, 3618, 3619

[0470] Using previously encrypted data at step 3610, the data may be eventually split into a predefined number of shares. If the split algorithm requires a key, a split encryption key may be generated at step 3612 using a cryptographically secure pseudo-random number generator. The split encryption key may optionally be transformed using an All or Nothing Transform (AoNT) into a transform split key at step 3614 before being key split to the predefined number of shares with fault tolerance at step 3615. The data may then be split into the predefined number of shares at step 3616. A fault tolerant scheme may be used at step 3617 to allow for regeneration of the data from less than the total number of shares. Once the shares are created, authentication/integrity information may be embedded into the shares at step 3618. Each share may be optionally post-encrypted at step 3619.

[0471] 2) 3111, 3612, 3614, 3615, 3616, 3617, 3618, 3619

[0472] In some embodiments, the input data may be encrypted using an encryption key provided by a user or an external system. The external key is provided at step 3611. For example, the key may be provided from an external key store. If the split algorithm requires a key, the split encryption key may be generated using a cryptographically secure pseudo-random number generator at step 3612. The split key may optionally be transformed using an All or Nothing Transform (AoNT) into a transform split encryption key at step 3614 before being key split to the predefined number of shares with fault tolerance at step 3615. The data is then split to a predefined number of shares at step 3616. A fault tolerant scheme may be used at step 3617 to allow for regeneration of the data from less than the total number of shares. Once the shares are created, authentication/integrity information may be embedded into the shares at step 3618. Each share may be optionally post-encrypted at step 3619.

[0473] 3) 3612, 3613, 3614, 3615, 3612, 3614, 3615, 3616, 3617, 3618, 3619

[0474] In some embodiments, an encryption key may be generated using a cryptographically secure pseudo-random number generator at step 3612 to transform the data. Encryption of the

data using the generated encryption key may occur at step 3613. The encryption key may optionally be transformed using an All or Nothing Transform (AoNT) into a transform encryption key at step 3614. The transform encryption key and/or generated encryption key may then be split into the predefined number of shares with fault tolerance at step 3615. If the split algorithm requires a key, generation of the split encryption key using a cryptographically secure pseudo-random number generator may occur at step 3612. The split key may optionally be transformed using an All or Nothing Transform (AoNT) into a transform split encryption key at step 3614 before being key split to the predefined number of shares with fault tolerance at step 3615. The data may then be split into a predefined number of shares at step 3616. A fault tolerant scheme may be used at step 3617 to allow for regeneration of the data from less than the total number of shares. Once the shares are created, authentication/integrity information will be embedded into the shares at step 3618. Each share may then be optionally post-encrypted at step 3619.

[0475] 4) 3612, 3614, 3615, 3616, 3617, 3618, 3619

[0476] In some embodiments, the data may be split into a predefined number of shares. If the split algorithm requires a key, generation of the split encryption key using a cryptographically secure pseudo-random number generator may occur at step 3612. The split key may optionally be transformed using an All or Nothing Transform (AoNT) into a transformed split key at step 3614 before being key split into the predefined number of shares with fault tolerance at step 3615. The data may then be split at step 3616. A fault tolerant scheme may be used at step 3617 to allow for regeneration of the data from less than the total number of shares. Once the shares are created, authentication/integrity information may be embedded into the shares at step 3618. Each share may be optionally post-encrypted at step 3619.

[0477] Although the above four combinations of options are preferably used in some embodiments of the invention, any other suitable combinations of features, steps, or options may be used with the secure data parser in other embodiments.

[0478] The secure data parser may offer flexible data protection by facilitating physical separation. Data may be first encrypted, then split into shares with "m of n" fault tolerance. This allows for regeneration of the original information when less than the total number of shares is available. For example, some shares may be lost or corrupted in transmission. The lost or corrupted shares may be recreated from fault tolerance or integrity information appended to the shares, as discussed in more detail below.

[0479] In order to create the shares, a number of keys are optionally utilized by the secure data parser. These keys may include one or more of the following:

[0480] Pre-encryption key: When pre-encryption of the shares is selected, an external key may be passed to the secure data parser. This key may be generated and stored externally in a key store (or other location) and may be used to optionally encrypt data prior to data splitting.

[0481] Split encryption key: This key may be generated internally and used by the secure data parser to encrypt the data prior to splitting. This key may then be stored securely within the shares using a key split algorithm.

[0482] Split session key: This key is not used with an encryption algorithm; rather, it may be used to key the data partitioning algorithms when random splitting is selected. When a random split is used, a split session key may be generated internally and used by the secure data parser to partition the data into shares. This key may be stored securely within the shares using a key splitting algorithm.

[0483] Post encryption key: When post encryption of the shares is selected, an external key may be passed to the secure data parser and used to post encrypt the individual shares. This key may be generated and stored externally in a key store or other suitable location.

[0484] In some embodiments, when data is secured using the secure data parser in this way, the information may only be reassembled provided that all of the required shares and external encryption keys are present.

[0485] FIGURE 37 shows illustrative overview process 3700 for using the secure data parser of the present invention in some embodiments. As described above, two well-suited functions for secure data parser 3706 may include encryption 3702 and backup 3704. As such, secure data parser 3706 may be integrated with a RAID or backup system or a hardware or software encryption engine in some embodiments.

[0486] The primary key processes associated with secure data parser 3706 may include one or

more of pre-encryption process 3708, encrypt/transform process 3710, key secure process 3712, parse/distribute process 3714, fault tolerance process 3716, share authentication process 3716, and post-encryption process 3720. These processes may be executed in several suitable orders or combinations, as detailed in FIGURE 36. The combination and order of processes used may depend on the particular application or use, the level of security desired, whether optional pre-encryption, post-encryption, or both, are desired, the redundancy desired, the capabilities or performance of an underlying or integrated system, or any other suitable factor or combination of factors.

[0487] The output of illustrative process 3700 may be two or more shares 3722. As described above, data may be distributed to each of these shares randomly (or pseudo-randomly) in some embodiments. In other embodiments, a deterministic algorithm (or some suitable combination of random, pseudo-random, and deterministic algorithms) may be used.

[0488] In addition to the individual protection of information assets, there is sometimes a requirement to share information among different groups of users or communities of interest. It may then be necessary to either control access to the individual shares within that group of users or to share credentials among those users that would only allow members of the group to reassemble the shares. To this end, a workgroup key may be deployed to group members in some embodiments of the invention. The workgroup key should be protected and kept confidential, as compromise of the workgroup key may potentially allow those outside the group to access information. Some systems and methods for workgroup key deployment and protection are discussed below.

[0489] The workgroup key concept allows for enhanced protection of information assets by encrypting key information stored within the shares. Once this operation is performed, even if all required shares and external keys are discovered, an attacker has no hope of recreating the information without access to the workgroup key.

[0490] FIGURE 38 shows illustrative block diagram 3800 for storing key and data components within the shares. In the example of diagram 3800, the optional pre-encrypt and post-encrypt steps are omitted, although these steps may be included in other embodiments.

[0491] The simplified process to split the data includes encrypting the data using encryption key 3804 at encryption stage 3802. Portions of encryption key 3804 may then be split and stored within shares 3810 in accordance with the present invention. Portions of split encryption key

3806 may also be stored within shares 3810. Using the split encryption key, data 3808 is then split and stored in shares 3810.

[0492] In order to restore the data, split encryption key 3806 may be retrieved and restored in accordance with the present invention. The split operation may then be reversed to restore the ciphertext. Encryption key 3804 may also be retrieved and restored, and the ciphertext may then be decrypted using the encryption key.

[0493] When a workgroup key is utilized, the above process may be changed slightly to protect the encryption key with the workgroup key. The encryption key may then be encrypted with the workgroup key prior to being stored within the shares. The modified steps are shown in illustrative block diagram 3900 of FIGURE 39.

[0494] The simplified process to split the data using a workgroup key includes first encrypting the data using the encryption key at stage 3902. The encryption key may then be encrypted with the workgroup key at stage 3904. The encryption key encrypted with the workgroup key may then be split into portions and stored with shares 3912. Split key 3908 may also be split and stored in shares 3912. Finally, portions of data 3910 are split and stored in shares 3912 using split key 3908.

[0495] In order to restore the data, the split key may be retrieved and restored in accordance with the present invention. The split operation may then be reversed to restore the ciphertext in accordance with the present invention. The encryption key (which was encrypted with the workgroup key) may be retrieved and restored. The encryption key may then be decrypted using the workgroup key. Finally, the ciphertext may be decrypted using the encryption key.

[0496] There are several secure methods for deploying and protecting workgroup keys. The selection of which method to use for a particular application depends on a number of factors. These factors may include security level required, cost, convenience, and the number of users in the workgroup. Some commonly used techniques used in some embodiments are provided below:

[0497] Hardware-based Key Storage

Hardware-based solutions generally provide the strongest guarantees for the security of encryption/decryption keys in an encryption system. Examples of hardware-based storage solutions include tamper-resistant key token devices which store keys in a portable device (e.g., smartcard/dongle), or non-portable key storage peripherals. These devices are designed to

prevent easy duplication of key material by unauthorized parties. Keys may be generated by a trusted authority and distributed to users, or generated within the hardware. Additionally, many key storage systems provide for multi-factor authentication, where use of the keys requires access both a physical object (token) and a passphrase or biometric.

[0498] Software-based Key Storage

While dedicated hardware-based storage may be desirable for high-security deployments or applications, other deployments may elect to store keys directly on local hardware (e.g., disks, RAM or non-volatile RAM stores such as USB drives). This provides a lower level of protection against insider attacks, or in instances where an attacker is able to directly access the encryption machine.

[0499] To secure keys on disk, software-based key management often protects keys by storing them in encrypted form under a key derived from a combination of other authentication metrics, including: passwords and passphrases, presence of other keys (e.g., from a hardware-based solution), biometrics, or any suitable combination of the foregoing. The level of security provided by such techniques may range from the relatively weak key protection mechanisms provided by some operating systems (e.g., MS Windows and Linux), to more robust solutions implemented using multi-factor authentication.

[0500] The secure data parser of the present invention may be advantageously used in a number of applications and technologies. For example, email system, RAID systems, video broadcasting systems, database systems, tape backup systems, or any other suitable system may have the secure data parser integrated at any suitable level. As previously discussed, it will be understand that the secure data parser may also be integrated for protection and fault tolerance of any type of data in motion through any transport medium, including, for example, wired, wireless, or physical transport mediums. As one example, voice over Internet protocol (VoIP) applications may make use of the secure data parser of the present invention to solve problems relating to echoes and delays that are commonly found in VoIP. The need for network retry on dropped packets may be eliminated by using fault tolerance, which guarantees packet delivery even with the loss of a predetermined number of shares. Packets of data (e.g., network packets) may also be efficiently split and restored "on-the-fly" with minimal delay and buffering, resulting in a comprehensive solution for various types of data in motion. The secure data parser may act on network data packets, network voice packets, file system data blocks, or any other

suitable unit of information. In addition to being integrated with a VoIP application, the secure data parser may be integrated with a file-sharing application (e.g., a peer-to-peer file-sharing application), a video broadcasting application, an electronic voting or polling application (which may implement an electronic voting protocol and blind signatures, such as the Sensus protocol), an email application, or any other network application that may require or desire secure communication.

[0501] In some embodiments, support for network data in motion may be provided by the secure data parser of the present invention in two distinct phases -- a header generation phase and a data partitioning phase. Simplified header generation process 4000 and simplified data partitioning process 4010 are shown in FIGURES 40A and 40B, respectively. One or both of these processes may be performed on network packets, file system blocks, or any other suitable information.

[0502] In some embodiments, header generation process 4000 may be performed one time at the initiation of a network packet stream. At step 4002, a random (or pseudo-random) split encryption key, K, may be generated. The split encryption key, K, may then be optionally encrypted (e.g., using the workgroup key described above) at AES key wrap step 4004. Although an AES key wrap may be used in some embodiments, any suitable key encryption or key wrap algorithm may be used in other embodiments. AES key wrap step 4004 may operate on the entire split encryption key, K, or the split encryption key may be parsed into several blocks (e.g., 64-bit blocks). AES key wrap step 4004 may then operate on blocks of the split encryption key, if desired.

[0503] At step 4006, a secret sharing algorithm (e.g., Shamir) may be used to split the split encryption key, K, into key shares. Each key share may then be embedded into one of the output shares (e.g., in the share headers). Finally, a share integrity block and (optionally) a post-authentication tag (e.g., MAC) may be appended to the header block of each share. Each header block may be designed to fit within a single data packet.

[0504] After header generation is complete (e.g., using simplified header generation process 4000), the secure data parser may enter the data partitioning phase using simplified data splitting process 4010. Each incoming data packet or data block in the stream is encrypted using the split encryption key, K, at step 4012. At step 4014, share integrity information (e.g., a hash H) may be computed on the resulting ciphertext from step 4012. For example, a SHA-256 hash may be

computed. At step 4106, the data packet or data block may then be partitioned into two or more data shares using one of the data splitting algorithms described above in accordance with the present invention. In some embodiments, the data packet or data block may be split so that each data share contains a substantially random distribution of the encrypted data packet or data block. The integrity information (e.g., hash H) may then be appended to each data share. An optional post-authentication tag (e.g., MAC) may also be computed and appended to each data share in some embodiments.

[0505] Each data share may include metadata, which may be necessary to permit correct reconstruction of the data blocks or data packets. This information may be included in the share header. The metadata may include such information as cryptographic key shares, key identities, share nonces, signatures/MAC values, and integrity blocks. In order to maximize bandwidth efficiency, the metadata may be stored in a compact binary format.

[0506] For example, in some embodiments, the share header includes a cleartext header chunk, which is not encrypted and may include such elements as the Shamir key share, per-session nonce, per-share nonce, key identifiers (e.g., a workgroup key identifier and a post-authentication key identifier). The share header may also include an encrypted header chunk, which is encrypted with the split encryption key. An integrity header chunk, which may include integrity checks for any number of the previous blocks (e.g., the previous two blocks) may also be included in the header. Any other suitable values or information may also be included in the share header.

[0507] As shown in illustrative share format 4100 of FIGURE 41, header block 4102 may be associated with two or more output blocks 4104. Each header block, such as header block 4102, may be designed to fit within a single network data packet. In some embodiments, after header block 4102 is transmitted from a first location to a second location, the output blocks may then be transmitted. Alternatively, header block 4102 and output blocks 4104 may be transmitted at the same time in parallel. The transmission may occur over one or more similar or dissimilar communications paths.

[0508] Each output block may include data portion 4106 and integrity/authenticity portion 4108. As described above, each data share may be secured using a share integrity portion including share integrity information (e.g., a SHA-256 hash) of the encrypted, pre-partitioned data. To verify the integrity of the outputs blocks at recovery time, the secure data parser may

compare the share integrity blocks of each share and then invert the split algorithm. The hash of the recovered data may then be verified against the share hash.

[0509] As previously mentioned, in some embodiments of the present invention, the secure date parser may be used in conjunction with a tape backup system. For example, an individual tape may be used as a node (*i.e.*, portion/share) in accordance with the present invention. Any other suitable arrangement may be used. For example, a tape library or subsystem, which is made up of two or more tapes, may be treated as a single node.

[0510] Redundancy may also be used with the tapes in accordance with the present invention. For example, if a data set is apportioned among four tapes (*i.e.*, portions/shares), then two of the four tapes may be necessary in order to restore the original data. It will be understood that any suitable number of nodes (*i.e.*, less than the total number of nodes) may be required to restore the original data in accordance with the redundancy features of the present invention. This substantially increases the probability for restoration when one or more tapes expire.

[0511] Each tape may also be digitally protected with a SHA-256, HMAC hash value, any other suitable value, or any combination thereof to insure against tampering. Should any data on the tape or the hash value change, that tape would not be a candidate for restoration and any minimum required number of tapes of the remaining tapes would be used to restore the data.

[0512] In conventional tape backup systems, when a user calls for data to be written to or read from a tape, the tape management system (TMS) presents a number that corresponds to a physical tape mount. This tape mount points to a physical drive where the data will be mounted. The tape is loaded either by a human tape operator or by a tape robot in a tape silo.

[0513] Under the present invention, the physical tape mount may be considered a logical mount point that points to a number of physical tapes. This not only increases the data capacity but also improves the performance because of the parallelism.

[0514] For increased performance the tape nodes may be or may include a RAID array of disks used for storing tape images. This allows for high-speed restoration because the data may always be available in the protected RAID.

[0515] In any of the foregoing embodiments, the data to be secured may be distributed into a plurality of shares using deterministic, probabilistic, or both deterministic and probabilistic data distribution techniques. In order to prevent an attacker from beginning a crypto attack on any cipher block, the bits from cipher blocks may be deterministically distributed to the shares. For

example, the distribution may be performed using the BitSegment routine, or the BlockSegment routine may be modified to allow for distribution of portions of blocks to multiple shares. This strategy may defend against an attacker who has accumulated less than "M" shares.

[0516] In some embodiments, a keyed secret sharing routine may be employed using keyed information dispersal (*e.g.*, through the use of a keyed information dispersal algorithm or "IDA"). The key for the keyed IDA may also be protected by one or more external workgroup keys, one or more shared keys, or any combination of workgroup keys and shared keys. In this way, a multi-factor secret sharing scheme may be employed. To reconstruct the data, at least "M" shares plus the workgroup key(s) (and/or shared key(s)) may be required in some embodiments. The IDA (or the key for the IDA) may also be driven into the encryption process. For example, the transform may be driven into the clear text (*e.g.*, during the pre-processing layer before encrypting) and may further protect the clear text before it is encrypted.

[0517] For example, in some embodiments, keyed information dispersal is used to distribute unique portions of data from a data set into two or more shares. The keyed information dispersal may use a session key to first encrypt the data set, to distribute unique portions of encrypted data from the data set into two or more encrypted data set shares, or both encrypt the data set and distribute unique portions of encrypted data from the data set into the two or more encrypted data set shares. For example, to distribute unique portions of the data set or encrypted data set, secret sharing (or the methods described above, such as BitSegment or BlockSegment) may be used. The session key may then optionally be transformed (for example, using a full package transform or AoNT) and shared using, for example, secret sharing (or the keyed information dispersal and session key).

[0518] In some embodiments, the session key may be encrypted using a shared key (e.g., a workgroup key) before unique portions of the key are distributed or shared into two or more session key shares. Two or more user shares may then be formed by combining at least one encrypted data set share and at least one session key share. In forming a user share, in some embodiments, the at least one session key share may be interleaved into an encrypted data set share. In other embodiments, the at least one session key share may be inserted into an encrypted data set share at a location based at least in part on the shared workgroup key. For example, keyed information dispersal may be used to distribute each session key share into a unique encrypted data set share to form a user share. Interleaving or inserting a session key

share into an encrypted data set share at a location based at least in part on the shared workgroup may provide increased security in the face of cryptographic attacks. In other embodiments, one or more session key shares may be appended to the beginning or end of an encrypted data set share to form a user share. The collection of user shares may then be stored separately on at least one data depository. The data depository or depositories may be located in the same physical location (for example, on the same magnetic or tape storage device) or geographically separated (for example, on physically separated servers in different geographic locations). To reconstruct the original data set, an authorized set of user shares and the shared workgroup key may be required.

[0519] Keyed information dispersal may be secure even in the face of key-retrieval oracles. For example, take a blockcipher E and a key-retrieval oracle for E that takes a list $(X_1, Y_1), \ldots, (X_c, Y_c)$ of input/output pairs to the blockcipher, and returns a key K that is consistent with the input/output examples (*e.g.*, $Y_i = E_K(X_i)$ for all i). The oracle may return the distinguished value $\perp$ if there is no consistent key. This oracle may model a cryptanalytic attack that may recover a key from a list of input/output examples.

[0520] Standard blockcipher-based schemes may fail in the presence of a key-retrieval oracle. For example, CBC encryption or the CBC MAC may become completely insecure in the presence of a key-retrieval oracle.

[0521] If Π^{IDA} is an IDA scheme and Π^{Enc} is an encryption scheme given by a mode of operation of some blockcipher E, then (Π^{IDA}, Π^{Enc}) provides security in the face of a key-retrieval attack if the two schemes, when combined with an arbitrary perfect secret-sharing scheme (PSS) as per HK1 or HK2, achieve the robust computational secret sharing (RCSS) goal, but in the model in which the adversary has a key-retrieval oracle.

[0522] If there exists an IDA scheme Π^{IDA} and an encryption scheme Π^{Enc} such that the pair of schemes provides security in the face of key-retrieval attacks, then one way to achieve this pair may be to have a "clever" IDA and a "dumb" encryption scheme. Another way to achieve this pair of schemes may be to have a "dumb" IDA and a "clever" encryption scheme.

[0523] To illustrate the use of a clever IDA and a dumb encryption scheme, in some embodiments, the encryption scheme may be CBC and the IDA may have a "weak privacy" property. The weak privacy property means, for example, that if the input to the IDA is a random sequence of blocks $M = M_1 \ldots M_l$ and the adversary obtains shares from a non-

authorized collection, then there is some block index i such that it is infeasible for the adversary to compute M_i. Such a weakly-private IDA may be built by first applying to M an information-theoretic AoNT, such as Stinson's AoNT, and then applying a simple IDA such as BlockSegment, or a bit-efficient IDA like Rabin's scheme (*e.g.,* Reed-Solomon encoding).

[0524] To illustrate the use of a dumb IDA and a clever encryption scheme, in some embodiments, one may use a CBC mode with double encryption instead of single encryption. Now any IDA may be used, even replication. Having the key-retrieval oracle for the blockcipher would be useless to an adversary, as the adversary will be denied any singly-enciphered input/output example.

[0525] While a clever IDA has value, it may also be inessential in some contexts, in the sense that the "smarts" needed to provide security in the face of a key-retrieval attack could have been "pushed" elsewhere. For example, in some embodiments, no matter how smart the IDA, and for whatever goal is trying to be achieved with the IDA in the context of HK1/HK2, the smarts may be pushed out of the IDA and into the encryption scheme, being left with a fixed and dumb IDA.

[0526] Based on the above, in some embodiments, a "universally sound" clever IDA Π^{IDA} may be used. For example, an IDA is provided such that, for all encryption schemes Π^{Enc}, the pair (Π^{IDA}, Π^{Enc}) universally provides security in the face of key-retrieval attacks.

[0527] In some embodiments, an encryption scheme is provided that is RCSS secure in the face of a key-retrieval oracle. The scheme may be integrated with HK1/HK2, with *any* IDA, to achieve security in the face of key-retrieval. Using the new scheme may be particularly useful, for example, for making symmetric encryption schemes more secure against key-retrieval attacks.

[0528] As mentioned above, classical secret-sharing notions are typically unkeyed. Thus, a secret is broken into shares, or reconstructed from them, in a way that requires neither the dealer nor the party reconstructing the secret to hold any kind of symmetric or asymmetric key. The secure data parser described herein, however, is optionally keyed. The dealer may provide a symmetric key that, if used for data sharing, may be required for data recovery. The secure data parser may use the symmetric key to disperse or distribute unique portions of the message to be secured into two or more shares.

[0529] The shared key may enable multi-factor or two-factor secret-sharing (2FSS). The adversary may then be required to navigate through two fundamentally different types of security

in order to break the security mechanism. For example, to violate the secret-sharing goals, the adversary (1) may need to obtain the shares of an authorized set of players, *and* (2) may need to obtain a secret key that it should not be able to obtain (or break the cryptographic mechanism that is keyed by that key).

[0530] In some embodiments, a new set of additional requirements is added to the RCSS goal. The additional requirements may include the "second factor"—key possession. These additional requirements may be added without diminishing the original set of requirements. One set of requirements may relate to the adversary's inability to break the scheme if it knows the secret key but does not obtain enough shares (*e.g.*, the *classical* or *first-factor* requirements) while the other set of requirements may relate to the adversary's inability to break the scheme if it does have the secret key but manages to get hold of all of the shares (*e.g.*, the *new* or *second-factor* requirements).

[0531] In some embodiments, there may be two second-factor requirements: a privacy requirement and an authenticity requirement. In the privacy requirement, a game may be involved where a secret key K and a bit b are selected by the environment. The adversary now supplies a pair of equal-length messages in the domain of the secret-sharing scheme, M_1^0 and M_1^1. The environment computes the shares of M_1^b to get a vector of shares, $\boldsymbol{S}_1 = (\boldsymbol{S}_1[1], \ldots, \boldsymbol{S}_1[n])$, and it gives the shares $\boldsymbol{S}_1$ (all of them) to the adversary. The adversary may now choose another pair of messages (M_2^0, M_2^1) and everything proceeds as before, using the same key K and hidden bit b. The adversary's job is to output the bit b' that it believes to be b. The adversary privacy advantage is one less than twice the probability that $b = b'$. This games captures the notion that, even learning all the shares, the adversary still cannot learn anything about the shared secret if it lacks the secret key.

[0532] In the authenticity requirement, a game may be involved where the environment chooses a secret key K and uses this in the subsequent calls to *Share* and *Recover*. *Share* and *Recover* may have their syntax modified, in some embodiments, to reflect the presence of this key. Then the adversary makes *Share* requests for whatever messages $M_1, \ldots, M_q$ it chooses in the domain of the secret-sharing scheme. In response to each *Share* request it gets the corresponding n-vector of shares, $\boldsymbol{S}_1, \ldots, \boldsymbol{S}_q$. The adversary's aim is to *forge* a new plaintext; it wins if it outputs a vector of shares $\boldsymbol{S}'$ such that, when fed to the *Recover* algorithm, results in something *not* in $\{M_1, \ldots, M_q\}$. This is an "integrity of plaintext" notion.

[0533] There are two approaches to achieve multi-factor secret-sharing. The first is a generic approach -- generic in the sense of using an underlying (R)CSS scheme in a black-box way. An authenticated-encryption scheme is used to encrypt the message that is to be CSS-shared, and then the resulting ciphertext may be shared out, for example, using a secret sharing algorithm, such as Blakely or Shamir.

[0534] A potentially more efficient approach is to allow the shared key to be the workgroup key. Namely, (1) the randomly generated session key of the (R)CSS scheme may be encrypted using the shared key, and (2) the encryption scheme applied to the message (*e.g.,* the file) may be replaced by an authenticated-encryption scheme. This approach may entail only a minimal degradation in performance.

[0535] Although some applications of the secure data parser are described above, it should be clearly understood that the present invention may be integrated with any network application in order to increase security, fault-tolerance, anonymity, or any suitable combination of the foregoing.

[0536] The secure data parser of the present invention may be used to implement a cloud computing data security solution. Cloud computing is network-based computing, storage, or both where computing and storage resources may be provided to computer systems and other devices over a network. Cloud computing resources are generally accessed over the Internet, but cloud computing may be performed over any suitable public or private network. Cloud computing may provide a level of abstraction between computing resources and their underlying hardware components (e.g., servers, storage devices, networks), enabling remote access to a pool of computing resources. These cloud computing resources may be collectively referred to as the "cloud." Cloud computing may be used to provide dynamically scalable and often virtualized resources as a service over the Internet or any other suitable network or combination of networks.

[0537] Security is an important concern with cloud computing because private data (e.g., from an enterprises' private network) may be transferred over public networks and may be processed and stored within publicly accessible or shared systems (e.g., Google (e.g., Google Apps Storage), Dropbox, or Amazon (e.g., Amazon's S3 storage facility)). These publicly accessible systems do not necessarily provided encrypted storage space, however, they do provide user's with the capability of storing a set of files on their servers. The secure data parser may be used

to protect cloud computing resources and the data being communicated between the cloud and an end-user or device. For example, the secure data parser may be used to secure data storage in the cloud, data-in-motion to/from the cloud, network access in the cloud, data services in the cloud, access to high-performance computing resources in the cloud, and any other operations in the cloud.

[0538] FIGURE 42 is an illustrative block diagram of a cloud computing security solution. System 4200, including secure data parser 4210, is coupled to cloud 4250 including cloud resources 4260. System 4200 may include any suitable hardware, such as a computer terminal, personal computer, handheld device (e.g., PDA, Blackberry, smart phone, tablet device), cellular telephone, computer network, any other suitable hardware, or any combination thereof. Secure data parser 4210 may be integrated at any suitable level of system 4200. For example, secure data parser 4210 may be integrated into the hardware and/or software of system 4200 at a sufficiently back-end level such that the presence of secure data parser 4210 may be substantially transparent to an end user of system 4200. The integration of the secure data parser within suitable systems is described in greater detail above with respect to, for example, FIGURES 27 and 28. Cloud 4250 includes multiple illustrative cloud resources 4260 including, data storage resources 4260a and 4260e, data service resources 4260b and 4260g, network access control resources 4260c and 4260h, and high performing computing resources 4260d and 4260f. The cloud resources may be provided by a plurality of cloud resource providers, e.g., Amazon, Google, or Dropbox. Each of these cloud computing resources will be described in greater detail below with respect to FIGURES 43-56. These cloud computing resources are merely illustrative. It should be understood that any suitable number and type of cloud computing resources may be accessible from system 4200.

[0539] One advantage of cloud computing is that the user of system 4200 may be able to access multiple cloud computing resources without having to invest in dedicated storage hardware. The user may have the ability to dynamically control the number and type of cloud computing resources accessible to system 4200. For example, system 4200 may be provided with on-demand storage resources in the cloud having capacities that are dynamically adjustable based on current needs. In some embodiments, one or more software applications executed on system 4200 may couple system 4200 to cloud resources 4260. For example, an Internet web browser may be used to couple system 4200 to one or more cloud resources 4260 over the

Internet. In some embodiments, hardware integrated with or connected to system 4200 may couple system 4200 to cloud resources 4260. In both embodiments, secure data parser 4210 may secure communications with cloud resources 4260 and/or the data stored within cloud resources 4260. The coupling of cloud resources 4260 to system 4200 may be transparent to system 4200 or the users of system 4200 such that cloud resources 4260 appear to system 4200 as local hardware resources. Furthermore shared cloud resources 4260 may appear to system 4200 as dedicated hardware resources.

[0540] In some embodiments, secure data parser 4210 may encrypt and split data such that no forensically discernable data will traverse or will be stored within the cloud. The underlying hardware components of the cloud (e.g., servers, storage devices, networks) may be geographically disbursed to ensure continuity of cloud resources in the event of a power grid failure, weather event or other man-made or natural event. As a result, even if some of the hardware components within the cloud suffer a catastrophic failure, the cloud resources may still be accessible. Cloud resources 4260 may be designed with redundancies to provide uninterrupted service in spite of one or more hardware failures.

[0541] In some embodiments, the secure parser of the present invention may first randomize the original data and then split the data according to either a randomized or deterministic technique. For example, if randomizing at the bit level, the secure parser of the present invention may jumble the bits of original data according to a randomized technique (e.g., according to a random or pseudo-random session key) to form a sequence of randomized bits. The secure parser may then split the bits into a predetermined number of shares by any suitable technique (e.g., a suitable information dispersal algorithm (IDA)) as previously discussed.

[0542] FIGURE 43 is an illustrative block diagram of a cloud computing security solution for securing data in motion (i.e., during the transfer of data from one location to another) through the cloud. FIGURE 43 shows a sender system 4300, which may include any suitable hardware, such as a computer terminal, personal computer, handheld device (e.g., PDA, Blackberry), cellular telephone, computer network, any other suitable hardware, or any combination thereof. Sender system 4300 is used to generate and/or store data, which may be, for example, an email message, a binary data file (e.g., graphics, voice, video, etc.), or both. The data is parsed and split by secure data parser 4310 in accordance with the present invention. The resultant data portions may be communicated over cloud 4350 to recipient system 4370.

[0543] Cloud 4350 may include any suitable combination of public and private cloud storage shown illustratively as clouds 4350a, 4350b, and 4350c. For instance, clouds 4350a and 4350c may be cloud storage resources that are publically accessible, such as those provided by Amazon, Google, or Dropbox. Cloud 4350b may be a private cloud that is inaccessible to any individual or group outside of a particular organization, e.g., an enterprise or educational institution. In other embodiments, a cloud may be a hybrid of a public and private cloud.

[0544] Recipient system 4370 of system 4300 may be any suitable hardware as described above with respect to sender system 4300. The separate data portions may be recombined at recipient system 4370 to generate the original data in accordance with the present invention. When traveling through cloud 4310 the data portions may be communicated across one or more communications paths including the Internet and/or one or more intranets, LANs, WiFi, Bluetooth, any other suitable hard-wired or wireless communications networks, or any combination thereof. As described above with respect to FIGURES 28 and 29, the original data is secured by the secure data parser even if some of the
data portions are compromised.

[0545] FIGURE 44 is an illustrative block diagram of a cloud computing security solution for securing data services in the cloud. In this embodiment, a user 4400 may provide data services 4420 to an end user 4440 over cloud 4430. Secure parser 4410 may secure the data services in accordance with the disclosed embodiments. Data service 4420 may be any suitable application or software service that is accessible over cloud 4430. For example, data service 4420 may be a web-based application implemented as part of a service-oriented architecture (SOA) system. Data service 4420 may be stored and executed on one or more systems within cloud 4430. The abstraction provided by this cloud computing implementation allows data service 4420 to appear as a virtualized resource to end user 4440 irrespective of the underlying hardware resources. Secure parser 4410 may secure data in motion between data service 4420 and end user 4440. Secure parser 4410 may also secure stored data associated with data service 4420. The stored data associated with data service 4420 may be secured within the system or systems implementing data service 4420 and/or within separate secure cloud data storage devices, which will be described in greater detail below. Although data service 4420 and other portions of FIGURE 44 are shown outside of cloud 4430, it should be understood that any of these elements may be incorporated within cloud 4430.

[0546] FIGURE 45 is an illustrative block diagram of a cloud computing security solution for securing data storage resources in the cloud. System 4500, including secure data parser 4510, is coupled to cloud 4550 which includes data storage resources 4560. Secure data parser 4510 may be used for parsing and splitting data among one or more data storage resources 4560. Each data storage resource 4560 may represent a one or more networked storage devices. These storage devices may be assigned to a single user/system of may be shared by multiple users/systems. The security provided by secure data parser 4510 may allow data from multiple users/systems to securely co-exist on the same storage devices or resources of cloud storage providers. The abstraction provided by this cloud computing implementation allows data storage resources 4560 to appear as a single virtualized storage resource to system 4500 irrespective of the number and location of the underlying data storage resources. When data is written to or read from data storage resources 4560, secure data parser 4510 may split and recombine the data in a way that may be transparent to the end user. In this manner, an end user may be able to access to dynamically scalable storage on demand.

[0547] Data storage in the cloud using secure data parser 4510 is secure, resilient, persistent, and private. Secure data parser 4510 secures the data by ensuring that no forensically discernable data traverses the cloud or is stored in a single storage device. The cloud storage system is resilient because of the redundancy offered by the secure data parser (i.e., fewer than all separated portions of the data are needed to reconstruct the original data). Storing the separated portions within multiple storage devices and/or within multiple data storage resources 4560 ensures that the data may be reconstructed even if one or more of the storage devices fail or are inaccessible. The cloud storage system is persistent because loss of a storage device within data storage resources 4560 has no impact on the end user. If one storage device fails, the data portions that were stored within that storage device may be rebuilt at another storage device without having to expose the data. Furthermore, the storage resources 4560 (or even the multiple networked storage devices that make up a data storage resource 4560) may be geographically dispersed to limit the risk of multiple failures. Finally, the data stored in the cloud may be kept private using one or more keys. As described above, data may be assigned to a user or a community of interest by unique keys such that only that user or community will have access to the data.

[0548] Data storage in the cloud using the secure data parser may also provide a performance boost over traditional local or networked storage. The throughput of the system may be improved by writing and reading separate portions of data to multiple storage devices in parallel. This increase in throughput may allow slower, less expensive storage devices to be used without substantially affecting the overall speed of the storage system.

[0549] FIGURE 46 is an illustrative block diagram for securing network access using a secure data parser in accordance with the disclosed embodiments. Secure data parser 4610 may be used with network access control block 4620 to control access to network resources. As illustrated in FIGURE 46, network access control block 4620 may be used to provide secure network communications between user 4600 and end user 4640. In some embodiments, network access control block 4620 may provide secure network access for one or more network resources in the cloud (e.g., cloud 4250, FIGURE 42). Authorized users (e.g., user 4600 and end user 4640) may be provided with group-wide keys that provide the users with the ability to securely communicate over a network and/or to access secure network resources. The secured network resources will not respond unless the proper credentials (e.g., group keys) are presented. This may prevent common networking attacks such as, for example, denial of service attacks, port scanning attacks, man-in-the-middle attacks, and playback attacks.

[0550] In addition to providing security for data at rest stored within a communications network and security for data in motion through the communications network, network access control block 4620 may be used with secure data parser 4620 to share information among different groups of users or communities of interest. Collaboration groups may be set up to participate as secure communities of interest on secure virtual networks. A workgroup key may be deployed to group members to provide members of the group access to the network and networked resources. Systems and methods for workgroup key deployments have been discussed above.

[0551] FIGURE 47 is an illustrative block diagram for securing access to high performance computing resources using a secure data parser in accordance with the disclosed embodiments. Secure data parser 4710 may be used to provide secure access to high performance computing resources 4720. As illustrated in FIGURE 47 end user 4740 may access high performance computing resources 4720. In some embodiments, secure data parser 4710 may provide secure access to high performance resources in the cloud (e.g., cloud 4250, FIGURE 42). High

performance computing resources may be large computer servers or server farms. These high performance computing resources may provide flexible, scalable, and configurable data services and data storage services to users.

[0552] The secure data parser of the present invention may be configured to implement a server-based secure data solution. The server-based solution of the secure parser of the present invention refers to a backend server-based Data at Rest (DAR) solution. The server may be any Windows-based, Linux-based, Solaris-based, or any other suitable operating system. This server-based solution presents a transparent file system to a user, i.e., a user does not observe any indication of the splits of data. When data is presented to the backend server of the secure data parser of the present invention, the data is split into N shares and sent to N accessible (therefore, available) data storage locations mounted/attached to the server. However, only some number M of those shares is required to rebuild the data. In some embodiments, the server-based solution of the secure parser of the present invention may first randomize the original data and then split the data according to either a randomized or deterministic technique. For example, if randomizing at the bit level, the secure parser of the present invention may jumble the bits of original data according to a randomized technique (e.g., according to a random or pseudo-random session key) to form a sequence of randomized bits. The server-based solution of the secure parser of the present invention may then split the bits into a predetermined number of shares by any suitable technique (e.g., round robin) as previously discussed. For the embodiments of FIGURES 42-47 above, and the embodiments of the FIGURES below, it will be assumed that the secure parser of the present invention may first split the data according to either a randomized or deterministic technique. Furthermore, in the embodiments described below, splitting data may include splitting data using any suitable information dispersion algorithm (IDA), including round robin or random bit splitting, as described above.

[0553] The abovedescribed solutions enable the recovery of data from local storage or remote storage such as single or multiple clouds because data may be rebuilt from any M of the N data shares, even when the data is first randomized and then split according to either a randomized or deterministic technique. Further descriptions of the server-based solution of the secure parser of the present invention are provided below, particularly with respect to FIGURES 48-56. In some embodiments, the server-based solution may be used in conjunction with cloud computing embodiments described above with respect to FIGURES 42-47.

[0554] In the embodiments of FIGURES 48-50, embodiments of the server-based solution of the secure parser of the present invention will be described in relation to their implemented in connection with public clouds (e.g., Dropbox), as well as other private, public, and hybrid clouds or cloud computing resources.

[0555] FIGURE 48 is a schematic of an illustrative arrangement in which the secure data parser is used to secure data storage in a plurality of storage devices in a private and a public cloud in accordance with one embodiment of the present invention. Private cloud 4804 includes a processor 4808 which is configured to implement the server-based solution of a secure parser of the present invention and generate encrypted data shares 4816b, 4818b, 4814b, 4812b, 4820b, and 4822b. Private cloud 4804 may optionally be accessible, e.g., via an Internet connection, to an end user device 4800. Remote users may access their data stored on the private cloud 4804 via end user device 4800, and may also transmit commands relating to data share generation and management from end user device 4800 to processor 4804 of cloud 4804. A subset of these encrypted data shares are stored on storage devices within the private cloud 4804. In particular, data share 4814b is stored on storage device 4814a, while data share 4812b is stored on storage device 4812a. Processor 4808 is also configured to store other subsets of the encrypted data shares in other public, private, or hybrid clouds 4802, 4806, or 4810. For instance, cloud 4806 may include public cloud resources provided by Amazon, while cloud 4802 may include public cloud resources provided by Dropbox. In this illustrative embodiment, shares 4818b and 4816b are stored on storage devices 4818a and 4816a, respectively, in cloud 4802, share 4822b is stored on storage device 4822a in cloud 4806, and share 4820b is stored on storage device 4820a in cloud 4810. In this manner, the provider of private cloud 4804 may leverage the storage resources of other cloud storage providers to store data shares, thereby reducing the storage burden on the storage devices with cloud 4804. The secure parser of private cloud 4804 simultaneously secures data while providing robust data survivability from disasters because only M of N parsed shares will be required to rebuild the data, where M<N. For example, if access to one of the public or private clouds 4806, 4810, or 4802 is interrupted or lost, the data can still be accessed and recovered using the available subset of encrypted data shares. In general, only M of N parsed shares will be required to rebuild the data, where M<N. For example, if access to one of the public or private clouds 4806, 4810, or 4802 is interrupted or lost, the data can still be accessed and recovered using the available subset of encrypted data

shares. As a further illustrative example, if a storage resource within one or more of public or private clouds 4806, 4810, or 4802 is down or otherwise inaccessible, the data can still be accessed and recovered using the accessible subset of encrypted data shares within the cloud(s).

[0556] FIGURE 49 is a schematic of an illustrative arrangement in which the secure data parser is used to secure data storage in a plurality of private and public clouds similar to the arrangement of FIGURE 48, in accordance with one embodiment of the present invention. FIGURE 49 illustrates a private cloud 4904 which is coupled, e.g., via an Internet connection, to an end user device such as laptop 4902, and to public clouds 4906 and 4908, e.g., via an Internet connection. Public clouds include cloud storage resources that are publically accessible, such as those provided by Dropbox and Amazon (e.g., Amazon's S3 storage facility). The abovedescribed Internet connections may be secure or unsecure. In the illustrative embodiment of FIGURE 49, public cloud 4906 is provided by Dropbox, while public cloud 4908 is provided by Amazon. Data from the end user device 4902 may be transmitted to private cloud 4904. The processor 4905 of the private cloud 4904 may be configured to implement the server-based solution of a secure parser of the present invention and generate encrypted data shares 4910a, 4910b, 4910c, and 4910d. Shares 4910a and 4910b are stored on storage devices within private cloud 4904, while shares 4910c and 4910d are transmitted to and stored on public clouds 4906 and 4908, respectively. As with the arrangement of FIGURE 48, the provider of private cloud 4904 may leverage the storage resources of other cloud storage providers to store data shares, thereby reducing the storage burden on the storage devices with cloud 4904. The secure parser of private cloud 4904 simultaneously secures data while providing robust data survivability from disasters because only M of N parsed shares will be required to rebuild the data, where M<N. For example, if access to one of the public or private clouds 4906 or 4908 is interrupted or lost, the data can still be accessed and recovered using the available subset of encrypted data shares.

[0557] FIGURE 50 is a schematic of another illustrative arrangement in which the secure data parser is used to secure data storage in a plurality of private and public clouds via the Internet 5006 in accordance with one embodiment of the present invention. In the arrangement of FIGURE 50, similar to that of FIGURES 48 and 49, an end user device 5002 is coupled to a private cloud 5008 via the publicly-accessible Internet 5006. Private cloud 5008 includes a processor 5001 that is configured to implement the server-based solution of the secure parser of the present invention and generate two sets of encrypted data shares: 5014a-d and 5016a-d.

Some of these encrypted data shares are stored in the same storage device, e.g., shares 5014b and 5016a, and shares 5014c and 5016b, while other shares are stored in different storage devices, e.g., shares 5016c, and 5016d. Shares 5014a and 5014d are transmitted to and stored on public clouds 5010 and 5012, respectively, provided by public cloud storage providers Google, Amazon, and Dropbox, which were described above respectively. As with the arrangement of FIGURES 48 and 49, the provider of private cloud 5008 may leverage the storage resources of other cloud storage providers to store data shares, thereby reducing the storage burden on the storage devices within private cloud 5008. The secure parser of private cloud 5008 therefore simultaneously secures data while providing robust data survivability from disasters because only M of N parsed shares will be required to rebuild the data, where M<N. Thus, if access to one of the public or private clouds 5010 or 5012 is interrupted or lost, the data can still be accessed and recovered using the available subset of encrypted data shares. In some embodiments, a removable storage device such as USB access key 5004 may be required at the end user device 5002 for authenticating the identity of a remote user who wishes to view, encrypt, or decrypt data that is managed by processor 5001 of private cloud 5008. In some embodiments, a removable storage device such as USB token 5004 may be required at the end user device 5002 to initiate the encryption, decryption, or splitting of data by processor 5001 of private cloud 5008. In some embodiments, data is split using any suitable information dispersion algorithm (IDA). In some embodiments, data is first randomized prior to splitting. In some embodiments, a user may manage their cryptographic keys themselves. In these embodiments, a user's keys may be stored on a user's end device such as USB token 5004 or end-user device 5002. In other embodiments, any suitable centralized or dispersed key management system may be used to manage a user's or work groups' cryptographic keys.

[0558] In some embodiments, to allow for data viewing and/or reconstruction at each of a plurality of distinct end-user devices, one or more cryptographic keys and/or one or more data shares may be stored on the USB memory device 5004. In addition, one or more of the data shares may also be stored on a cloud 5010 and/or 5012. Thus, a user in possession of the portable user device may access the USB memory device 5004 from a different end user device than device 5002 to view and/or rebuild the data from the shares dispersed across the USB memory device 5004 and if necessary, the cloud. For instance, two data shares may be stored on USB memory device 5004 and two data shares may be stored in each of clouds 5010 and 5012.

A user in possession of USB memory device 5004 may use any computing device with the secure parser of the present invention coupled to USB memory device 5004 to access the two data shares stored on device 5004. For example, a user may use a first laptop computer to create and disperse the shares across the USB memory device 5004 and the cloud, and may then use a second, different laptop computer to retrieve the shares from the USB memory device 5004 and/or the clouds 5010 and 5012, and then reconstruct/rebuild the data from the retrieved shares.

[0559] In some embodiments, the secure parser of the present invention may provide confidentiality, availability, and integrity of stored data by ensuring that a lost or stolen device's data remains secure and undecipherable. In some embodiments, the present invention may include software running at the kernel level in the background of any Windows or Linux enabled PC or end user device (e.g., mobile phone, laptop computer, personal computer, tablet computer, smart phone, set-top box, etc.). In some embodiments, a secure parser such as Security First Corp.'s FIPS 140-2 certified, Suite B compliant, SecureParser Eχtended (SPχ) may be used to split the data to be secured. In some embodiments, FIPS 140-2 AES 256 encryption, random bit data splitting, integrity checking and re-encrypting split shares is performed. In some embodiments, the data is split using any suitable information dispersion algorithm (IDA). In some embodiments, the splitting is deterministic. In some embodiments, the data may also be randomized prior to the splitting. In some embodiments, any files stored to a secure location (e.g. the "C:" drive) on a user's end device are invisible without the proper credentials and access. In some embodiments, even the file names cannot be seen or recovered without the requisite cryptographic key and authentication process.

[0560] In some embodiments, a set of N shares are created and the secure parser of the present invention stores these N shares in N separate, possibly geographically-dispersed storage locations. For instance, four (4) encrypted shares may be created and the secure parser of the present invention then stores these four encrypted shares in four (4) separate storage locations. FIGURES 51-53 illustrate two such embodiments, in which four encrypted shares are created, of the secure parser of the present invention.

[0561] FIGURE 51 is a schematic of an illustrative arrangement in which the secure data parser is used to secure data storage in a user's removable storage device 5104 and on mass storage device 5106 in accordance with one embodiment of the present invention. FIGURE 51 shows an end user device such as a laptop computer 5102 that has generated four encrypted shares 5108a,

5108b, 5108c, and 5108d. Each of these encrypted shares 5108a-d is stored in a different storage sector within mass storage device 5106 of the end user device 5102. The secure parser of the end-user device simultaneously secures data while providing robust data survivability from disasters because only M of N parsed shares will be required to rebuild the data, where M<N. In the embodiment of FIGURE 51, there are 4 shares, and 2 or 3 of these shares would be required to re-construct the data. Assuming that only two of the four, or three of the four, encrypted shares are required to re-construct the data, the disaster recovery process is accelerated if one or two of the encrypted shares are lost, e.g., if one of the sectors of mass storage 5106 is corrupted. The removable storage device 5104 may be used to store one or more cryptographic access keys that may be required in order to view and/or decrypt and/or encrypt data within the mass storage 5106 of the end user device 5102. In some embodiments, without the cryptographic key on the removable storage device 5104, the encrypted data shares 5108a-d cannot be decrypted and/or reconstituted. In some embodiments, a user may manage their cryptographic keys themselves. In these embodiments, a user's keys may be stored on a user's end device such as removable storage device (e.g., USB memory) 5104 or end-user device 5102. In other embodiments, any suitable centralized or dispersed key management system may be used to manage a user's or work groups' cryptographic keys.

[0562] In some embodiments, to allow for data viewing and/or reconstruction at each of a plurality of distinct end-user devices, one or more cryptographic keys and/or one or more data shares may be stored on the USB memory device 5104. In addition, one or more of the data shares may also be stored on a cloud. Thus, a user in possession of the portable user device may access the USB memory device 5104 from a different end user device than device 5102 to view and/or rebuild the data from the shares dispersed across the USB memory device 5104 and if necessary, the cloud. For instance, two data shares may be stored on USB memory device 5104 and two data shares may be stored in end user device 5102. A user in possession of USB memory device 5104 may use any computing device with the secure parser of the present invention coupled to USB memory device 5104 to access the two data shares stored on USB memory device 5104. For example, a user may use a first laptop computer to create and disperse the shares across the USB memory device 5104 and the end user device 5102, and may then use a second, different laptop computer to retrieve the shares from the USB memory device 5104

and, assuming these two shares are sufficient for reconstructing the data, reconstruct/rebuild the data from these two shares.

[0563] FIGURE 52 is a schematic of an illustrative arrangement in which the secure data parser is used to secure data storage in a plurality of user storage devices in accordance with one embodiment of the present invention. FIGURE 52 shows an end user device such as a laptop computer 5202 that has generated four encrypted shares 5208a, 5208b, 5208c, and 5208d. Each of these encrypted shares 5208a-d is stored in geographically dispersed storage location and/or different parts of the same storage location. In particular, encrypted shares 5208c and 5208d are stored in two different sectors on mass storage device 5206 of the laptop computer 5202, while encrypted shares 5308a and 5308b are each stored on a removable storage device such as USB memory device 5204. The secure parser of the end-user device simultaneously secures data while providing robust data survivability from disasters because only M of N parsed shares will be required to rebuild the data, where M<N. In the embodiment of FIGURE 52, there are 4 shares, and 2 or 3 of these shares would be required to re-construct the data. Thus, the encrypted shares are geographically and physically dispersed, and assuming that only two of the four, or three of the four encrypted shares are required to re-construct the data, the disaster recovery process is accelerated if one or two of the encrypted shares are lost. Such a loss may occur, e.g., if one of the sectors of mass storage 5202 is corrupted, or if the removable storage device such as USB memory device 5204 is lost, or any combination thereof.

[0564] In some embodiments, instead of or in addition to storing encrypted shares on the USB memory device 5204, one or more keys (e.g., encryption key, split key, or authentication key) are stored on the USB memory device 5204. These keys may be used to split, encrypt/decrypt, or authenticate shares of data stored on the USB memory device 5204 itself, or elsewhere, e.g., in the end-user device mass storage 5202 or in a public or private cloud storage. For example, a user may store a key on USB memory device 5204 and use this key to decrypt encrypted shares of data stored on mass storage device 5202. As a further illustrative example, two data shares may be stored on USB memory device 5204 and two data shares may be stored in end user device mass storage 5202. A user in possession of USB memory device 5204 may use any computing device with the secure parser of the present invention coupled to USB memory device 5204 to access the key stored on USB memory device 5204. For example, a user may use a first laptop computer to store the key within USB memory device 5204, and may then use a second,

different laptop computer to retrieve the key from the USB memory device 5204. This key may then be used to encrypt/decrypt, split, or authenticate data.

[0565] In some embodiments, to allow for data viewing and/or reconstruction at each of a plurality of distinct end-user devices, one or more cryptographic keys and/or one or more data shares may be stored on the USB memory device 5204. In addition, one or more of the data shares may also be stored on a cloud. Thus, a user in possession of the portable user device may access the USB memory device 5204 from a different end user device than device 5202 to view and/or rebuild the data from the shares dispersed across the USB memory device 5204 and if necessary, the cloud. For instance, two data shares may be stored on USB memory device 5204 and two data shares may be stored in end user device 5202. A user in possession of USB memory device 5204 may use any computing device with the secure parser of the present invention coupled to USB memory device 5204 to access the two data shares stored on USB memory device 5204. For example, a user may use a first laptop computer to create and disperse the shares across the USB memory device 5204 and the end user device 5202, and may then use a second, different laptop computer to retrieve the shares from the USB memory device 5204 and, assuming these two shares are sufficient for reconstructing the data, reconstruct/rebuild the data from these two shares.

[0566] FIGURE 53 is a schematic of an illustrative arrangement in which the secure data parser is used to secure data storage in a plurality of public and private clouds and at least one user storage device in accordance with one embodiment of the present invention. FIGURE 53 shows an end user device such as a laptop computer 5302 that has generated four encrypted shares 5306a, 5306b, 5306c, and 5306d. Each of these encrypted shares 5306a-d is stored in geographically dispersed storage location and/or different parts of the same storage location. In particular, encrypted shares 5306a and 5306b are stored in two different sectors on mass storage device 5308 of the laptop computer 5302, while encrypted share 5306c is stored, by transmission over a secure network connection, in a publicly accessible cloud storage such as Amazon's S3 cloud storage 5310, and encrypted share 5306d is stored, by transmission over a secure network connection, in a publicly accessible cloud storage such as Dropbox's cloud storage 5312. In this manner, the encrypted shares are geographically and physically dispersed, and assuming that only two of the four, or three of the four, encrypted shares are required to re-construct the data, the disaster recovery process is accelerated if one or two of the encrypted shares are lost. Such a

loss may occur, e.g., if one of the sectors of mass storage 5308 is corrupted, or if the internet connection between the end user device 5302 and the clouds 5310 and 5312 is lost.

[0567] In each of the embodiments of FIGURES 51-53, the encrypted data share generation process splitting process is transparent to the user. Furthermore, the secure parser of the present invention simultaneously secures data while providing robust data survivability from disasters because only M of N parsed shares will be required to rebuild the data, where M<N. For example, in some of the embodiments described above, only two (2) or three (3) of the four (4) parsed shares would be needed to re-construct or rebuild the data. If a hard drive's sector fails, or a removable USB device is lost, or a remote storage location is down or inaccessible, the data can still be accessed and recovered. Furthermore, if a failed drive's share is recovered, or if a share is stolen, goes off-line or is hacked into, the data may remain safe and protected since any single parsed share contains no forensically discernable information. In other words, a single parsed share cannot be reconstituted, decrypted, hacked or recovered without first having the corresponding second and/or third shares, proper user authentication, the secure parser of the present invention, and in some cases, the USB key or USB memory device.

[0568] In some embodiments, the secure parser of the present invention may be used in a mobile device such as an Apple iPad, a RIM Blackberry, an Apple iPhone, a Motorola Droid phone, or any suitable mobile device. Those skilled in the art will come to realize that the systems and methods disclosed herein are application to a variety of end user devices, including but not limited to mobile devices, personal computers, tablet computers, smart phones and the like.

[0569] The secure parser of the present invention may be implemented using one or more processors, each of which performs one or more of the secure parser functions such as key generation, data encryption, share generation, data decryption, etc. In some embodiments, splitting data includes cryptographically splitting data, e.g., random bit splitting. In some embodiments, the data is split using any suitable information dispersal algorithm (IDA). The processor(s) may be any suitable processors, e.g., Intel or AMD, and may run a back end for a server based platform. In some embodiments, one or more dedicated coprocessors may be used to accelerate the operation of the secure parser of the present invention. In the embodiments of FIGURES 54-56 described below, one or more functions of secure parser of the present invention are implemented on one or more dedicated co-processors, which allows for the

acceleration of the secure parser functions. In some embodiments, the coprocessors may be included in a main motherboard or in a daughterboard, or any suitable combination thereof, of the secure parser hardware platform.

[0570] FIGURE 54 is a schematic of a co-processor acceleration device 5400 for the secure data parser in accordance with one embodiment of the present invention. Device 5400 includes two processors: central processing unit (CPU) or main processor 5402 and rapid processing unit (RPU) or auxiliary processor 5404. Processors 5402 and 5404 are coupled to one another, and also coupled to a memory device 5406 and mass storage device 5408. The coupling of these devices may include the use of an interconnect bus. Each of the CPU and RPU may include a single microprocessor or a plurality of microprocessors for configuring the CPU and/or RPU as a multiprocessor system. Memory 5406 may include dynamic random access memory (DRAM) and/or high-speed cache memory. Memory 5406 may include at least two dedicated memory devices, one for each of CPU 5402 and RPU 5404. Mass storage device 5408 may include one or more magnetic disk or tape drives or optical disk drives, for storing data and instructions for use by the CPU 5402 and/or RPU 5406. Mass storage device 5408 may also include one or more drives for various portable media, such as a floppy disk, a compact disc read only memory (CD-ROM), DVD, a FLASH drive, or an integrated circuit non-volatile memory adapter (i.e. PC-MCIA adapter) to input and output data and code to and from the CPU 5402 and/or RPU 5406. The CPU 5402 and/or RPU 5406 may each also include one or more input/output interfaces for communications, shown by way of example, as the communications bus 5410. Communications bus may also include an interface for data communications via the network 5412. The network 5412 may include one or more storage devices, e.g., cloud storage devices, NAS, SAN, etc. The interface to the network 5412 via the communications bus 5410 may be a modem, a network card, serial port, bus adapter, or any other suitable data communications mechanism for communicating with one or more systems on-board the aircraft or on the ground. The communication link to the network 5412 may be, for example, optical, wired, or wireless (e.g., via satellite or cellular network).

[0571] In some embodiments, RPU may include a redundant array of independent disks (RAID) processing unit that implements one or more RAID functions for one or more storage devices associated with the co-processor acceleration device 5400. In some embodiments, RPU 5404 may include a general purpose or special purpose integrated circuit (IC) to perform array

built calculations and/or RAID calculations. In some embodiments, the RPU 5404 may be coupled to the CPU 5402 via a PCIe connection such as a PCIe bus coupled to the RPU. If RPU includes a RAID processing unit, then the PCIe connection may include a specialized RAID adapter. In some embodiments, the PCIe card may run at 10 Gigabits/sec (Gb/s) or more. In some embodiments, the RPU 5404 may be coupled to the CPU 5402 via an HT connection, such as a socketed RPU connected to an HT bus. The processors 5402 and 5404 will typically access the same memory and mass storage devices such that the same data is accessible to both these processors. The coprocessor may perform dedicated secure parsing accelerated functions including, but not limited to, data splitting, encryption, and decryption. These functions are independent of one another, and may be performed using different algorithms. For example, encryption may be performed using any of the abovedescribed techniques, while splitting may be performed using any suitable information dispersal algorithm (IDA), such as those described above. In some embodiments, the RPU may be coupled to a Field Programmable Gate Array (FPGA) device that could also perform dedicated accelerated functions of the secure parser of the present invention external to the coprocessor acceleration device 5400.

[0572] FIGURE 55 is a first process flow diagram of an illustrative acceleration process using the co-processor acceleration device 5400 of FIGURE 54 for the secure data parser in accordance with one embodiment of the present invention. With continued reference to FIGURES 54 and 55, in this illustrative embodiment, the RPU 5510 may be coupled to the CPU 5520 via an HT connection, such as a socketed RPU via an HT bus. The left side of FIGURE 55 illustrates that certain functions of the secure parser such as the data splitting and share generation functions (3910 and 3912 in FIGURE 39) may be performed by the CPU, while other functions such as the encryption (e.g., the AES, IDA, SHA algorithms) (3902, 3904, 3906 in FIGURE 39) may be performed by the RPU. These functions of encryption and encryption share generation are shown on the right side of FIGURE 55, in which there is an indication of whether the CPU or RPU performs a particular secure parser function.

[0573] FIGURE 56 is a second process flow diagram of an illustrative acceleration process using the co-processor acceleration device 5400 of FIGURE 54 for the secure data parser in accordance with one embodiment of the present invention. With continued reference to FIGURES 54 and 56, in this illustrative embodiment, the RPU 5610 may be coupled to the CPU 5620 via an HT connection, such as a socketed RPU via an HT bus. The left side of FIGURE 56

illustrates that certain functions of the secure parser such as the data splitting and share generation functions (3910 and 3912 in FIGURE 39) may be performed by the CPU, while other functions such as the encryption (e.g., the AES, IDA, SHA algorithms) (3902, 3904, 3906 in FIGURE 39) may be performed by the RPU. These functions of encryption and encryption share generation are shown on the right side of FIGURE 55, in which there is an indication of whether the CPU or RPU performs a particular secure parser function.

[0574] With respect to the embodiments in FIGURES 48-56 describing the server-based solution of the secure parser of the present invention, there are several additional functions and characteristics of the secure parser of the present invention that may be enabled or provided by the server-based solution. In addition to performing cryptographic splitting and data share rebuilding, other functionality may be included such as block level updates of encrypted data shares and cryptographic key management. The description that follows will describe each of these functions. Those skilled in the art will come to realize that this functionality may be easily incorporated into any of the embodiments described with respect to FIGURES 48-56.

[0575] In some embodiments, the server-based solution of the secure parser of the present invention allows block level updates/changes to files, instead of updates/changes to the entire data file. In some embodiments, once a data share has been sent from the secure parser to a cloud storage device, in order to operate more efficiently, when the underlying data is updated by a user or workgroup, instead of restoring the entire data file, only the updates at the file block level of particular data shares may be transmitted to the cloud storage device using the cryptographic systems of the present invention. Thus, restoration of an entire data file is not performed nor required when only minor changes are made to the data file.

[0576] In some embodiments, the server-based solution of the secure parser of the present invention generates a stub for each of the data shares. In some embodiments, a stub may include a list of attributes for its associated data share, and is stored together with the data share. In some embodiments, a stub may include information about the data share including, e.g., the name of a data share, the date the data share was created, the last time the data share was modified, a pointer to the location of the data share within the file system of a storage device, etc. Such information could be used to quickly provide a user with information regarding the data shares. In some embodiments, a user may designate a stub directory which stores the stubs. For instance, a user may designate a particular virtual or physical drive on their storage device on

which the stub directory should be stored. For instance, a stub directory may be created for a user, wherein each of the stubs in the directory points the user to secure data stored by the secure parser in a mass storage device, removable storage device, public cloud, private cloud, or any combination thereof. In this manner, stubs may be utilized to generate a virtual file system of data shares for a user.

[0577] In some embodiments, the stubs may be stored in a separate location from the data shares, in the same location as the data shares, or both. In some embodiments, when a user wishes to view some information on the data shares, they may access the stub directory. In some embodiments, instead of directly viewing the stub directory, the stubs are retrieved from the stub directory, processed by the server-based solution of the secure parser of the present invention, and subsequently used to provide the aforementioned information to the user. In this manner, stubs may be utilized to generate a virtual file system of data shares for a user.

[0578] In some embodiments, the stubs are stored in the respective headers of the data shares. Thus, if a user wishes to view the information in a stub, the stub is retrieved from the header, processed by the server-based solution of the secure parser of the present invention, and subsequently a stub directory is generated and provided to the user.

[0579] In some embodiments, the server-based solution of the secure parser of the present invention frequently checks the stub(s) and/or encrypted data shares for data integrity using the above described techniques. The secure parser of the present invention is essentially proactive in retrieving and examining data shares for data integrity, even when not initiated or prompted by a user. If a data share or stub is missing or damaged, the secure parser of the present invention attempts to recreate and restore the stub or data share.

[0580] The server-based solution of the secure parser of the present invention may be configured to provide a centralized cryptographic key management facility. In particular, cryptographic keys used to encrypt/decrypt data, data shares and communication sessions across a plurality of storage devices and systems may be stored in a central location within an enterprises' storage facility, e.g., an enterprises' private cloud. This centralized key management facility may also interface with hardware-based key management based solutions such as those provided by SafeNet, Inc., Belcamp, MD, or with software-based key management systems. For instance, an existing private cloud may control access to encrypted shares of data via an authentication/access/authorization system, and the server-based solution may use the

authentication information to allow access to the cryptographic keys used to encrypt those shares, thereby allowing a user to cryptographically split data, or restore the encrypted shares of data. In other words, the server-based solution of the secure parser of the present invention may act in conjunction with an existing authentication/access/authorization system. In this manner, an enterprise is not forced to change its current way of managing users' and work groups' access to data.

[0581] In some embodiments, the server-based solution of the secure parser of the present invention may perform share rebuilding without decrypting any of the encrypted data shares. In some embodiments, the server-based solution of the secure parser of the present invention may re-generate splits of data using one or more new keys without decrypting any of the encrypted data shares. FIGURE 57 illustrates a process 5700 by which data is split into N shares and stored, according to an illustrative embodiment of the present invention. FIGURE 58 illustrates a process by which shares of data are rebuilt and/or re-keyed, according to an illustrative embodiment of the present invention. In each of FIGURES 57 and 58 each of the steps of the process may be optional. For instance, it is not necessary to encrypt data prior to splitting the data.

[0582] With reference to FIGURE 57, the secure parser first encrypts the data using an encryption key (5702). The encryption key may be generated internally within the secure parser of the present invention. The encryption key may be generated based at least in part on an external workgroup key. The secure parser then splits the data into N shares using a split key (5704). The split key may be generated internally within the secure parser of the present invention. The split key may be generated based at least in part on an external workgroup key. The secure parser then ensures that only M of N shares will be required to rebuild the data (5706) and authenticates the N shares using an authentication key (5708). The authentication key may be generated internally within the secure parser of the present invention. The authentication key may be generated based at least in part on an external workgroup key. The authentication, split, and encryption keys are each wrapped using a key encryption key (5710). The KEK is then split and stored within the headers of the N shares (5712). The N shares are then dispersed across N storage locations.

[0583] In some instances, it is desirable for a user or an enterprise to use a new split key and/or a new authentication key for a set of data shares. With the server-based solution of the secure

parser of the present invention, this re-keying of the data may be performed without decrypting any of the data shares. In other instances, it is desirable for a user or an enterprise to regenerate a set of new data shares because one or more existing data shares have been corrupted, lost or otherwise inaccessible. With the server-based solution of the secure parser of the present invention, this rebuilding of the lost data shares may be performed without decrypting any of the remaining, available data shares. With reference to FIGURE 58, assuming that N-M shares of data are corrupted or otherwise inaccessible, the secure parser retrieves the remaining M of N shares from their storage locations (5802). These M shares are authenticated using an authentication key (5804). Using the authenticated M shares, the encrypted data is reconstructed by the secure parser (5806). The split key is then used to regenerate the N shares (5808), and the authentication key is used to authenticate the N shares (5810). If a different split key or authentication key were used (5812) for steps 5808 or 5810, then the headers of each of the M shares are retrieved (5816), the key encryption key is reconstructed (5818), and similar to the processes of steps 5710 and 5712 (FIGURE 57), the new split key and/or authentication key are wrapped/encrypted using the key encryption key (5820). The N shares are then stored in one or more storage devices of the secure parser of the present invention (5822). If a different split key or authentication was not used (5812) in steps 5808 or 5810, then the lost/inaccessible N-M shares are stored in one or more storage devices of the secure parser of the present invention (5814).

[0584] The server-based solution of the secure parser of the present invention may be configured to secure the file name of a data share, such as the data shares described in relation to the embodiments of FIGURES 42-58 above. In some embodiments, when splitting a file into N data shares, e.g., using an IDA, the generated data shares are stored on one or more share locations in a storage network. The storage network may include a private cloud, a public cloud, a hybrid cloud, a removable storage device, a mass storage device, or any combination thereof. In many applications, there will be more than one file that is split and stored in a share location in the storage network. In other words, there may be several files, each of which may be split into N data shares (e.g., using an IDA), where each of the generated data shares may be stored as files on the share locations. In these applications, it is advantageous to have a unique identifier such as a file name that associates a data share in a share location with the file from it was generated.

[0585] In some embodiments, the secure parser of the present invention may be configured to use a portion of the file name of the original file (i.e., the file that is to be split) to name the data shares with the same name as the original file. As an illustrative example, if an original file "2010Budget.xls" is split into 4 data shares, these data shares may be named "2010Budget.xls.1", "2010Budget.xls.2", "2010Budget.xls.3" and "2010Budget.xls.4", thereby associating each generated data share with the original file. By this process, the secure parser of the present invention would efficiently be able to locate the data shares and associate them with the original file. The drawback of this process, however, is that it may expose information such as the fact that the budge information is for year 2010 to a third party. In many applications, exposing the file name in this manner is not acceptable, and thus the file name of a data share cannot be easily associated with the file name of the original file

[0586] In some embodiments, the secure parser of the present invention may be configured to first secure the file name would be to use an authentication algorithm such as HMAC-SHA256 to hash the file name of the original file into a value that cannot be reversed. The secure parser of the present invention would thus process the file name of the original file with the HMAC-SHA256 algorithm to obtain a "hashed" file name and receive an authentication value that is secure and may not be reversed to the file name of the original file. The file names of the data shares associated with the original file are then generated using this hashed file name instead of the file name of the original file. In these embodiments, in order to locate the data shares (on a storage network) associated with the file name of the original file, the secure parser of the present invention would once again use the HMAC-SHA256 algorithm on the original file name and regenerate the authentication value. In some embodiments, the authentication value for the original file name and the file names of the generated shares are substantially equal. The secure parser of the present invention would then search the share locations on the storage network for data share file names that match this authentication value. The storage network may include a private cloud, a public cloud, a hybrid cloud, a removable storage device, a mass storage device, or any combination thereof. In some embodiments, the full path of the original file name is used so that the authentication value generated for a file with full path, e.g., "\Marketing\2010Budget.xls" is different from the authentication value generated for the file with full path, e.g., "\Sales\2010Budget.xls". In some embodiments, the resulting data share filenames corresponding to each data share location are made different by hashing the full path

for a file, the full path including the share location. For instance, by appending the share number of a data share to the full path of the original file, for example "\Sales\2010Budget.xls.1", the resulting data share filenames are different for each data share location.

[0587] In some embodiments, the secure parser of the present invention secures the file name of a file by encrypting the full path of the original filename using an encryption algorithm such as AES, as described above. Such encryption ensures that the file name of the original file is secure until it is decrypted by the secure parser of the present invention based on authenticated access to the share locations on a storage network, the retrieved data shares and the encryption key. The storage network may include a private cloud, a public cloud, a hybrid cloud, a removable storage device, a mass storage device, or any combination thereof. As with the abovedescribed example, unique data share filenames for each share location can be created by first appending additional information such as the share number for a data share to the full path of the original file.

[0588] Although some applications of the secure data parser are described above, it should be clearly understood that the present invention may be integrated with any network application in order to increase security, fault-tolerance, anonymity, or any suitable combination of the foregoing.

[0589] Additionally, other combinations, additions, substitutions and modifications will be apparent to the skilled artisan in view of the disclosure herein.

What is claimed is:

1. A method for rebuilding a set of data shares that were generated from an encrypted data set by an information dispersal algorithm using a first split key, the method comprising:

receiving at least a minimum number of data shares necessary for rebuilding the set of data shares; and

rebuilding the set of data shares from the minimum number of data shares without decrypting the minimum number of data shares.

2. The method of claim 1, wherein the rebuilding is performed in response to a determination that one or more of the data shares have been compromised.

3. The method of claim 1, further comprising storing at least one of the rebuilt data shares on a storage network.

4. The method of claim 3, wherein the storage network includes one of a private cloud, a public cloud, a hybrid cloud, a removable storage device, and a mass storage device.

5. The method of claim 1, wherein rebuilding includes:

authenticating the minimum number of data shares with an authentication key;

reconstructing the encrypted data from the authenticated minimum number of data shares using the split key;

regenerating the set of data shares by splitting the encrypted data using the split key.

6. A method for rekeying a set of data shares that were generated from an encrypted data set by an information dispersal algorithm using a first encryption key, the method comprising:

receiving at least a minimum number of data shares necessary for rebuilding the set of data shares;

associating the minimum number of data shares with a first authentication key;

rebuilding the set of data shares from the minimum number of data shares without decrypting the minimum number of data shares; and

rekeying the rebuilt set of data shares by associating the rebuilt set of data shares with a second encryption key.

7. The method of claim 6, further comprising:

retrieving headers associated with the minimum number of data shares;

extracting a key encryption key from the retrieved headers;

encrypting the second encryption key with the key encryption key; and

storing the encrypted second authentication key within the headers of the rekeyed data shares.

8. The method of claim 6, further comprising storing at least one of the rekeyed data shares on a storage network.

9. The method of claim 8, wherein the storage network includes one of a private cloud, a public cloud, a hybrid cloud, a removable storage device, and a mass storage device.

10. A method for rekeying a set of data shares that were generated from an encrypted data set by an information dispersal algorithm using a first split key, the method comprising:

receiving at least a minimum number of data shares necessary for rekeying the set of data shares;

rebuilding the set of data shares from the minimum number of data shares without decrypting the minimum number of data shares; and

rekeying the rebuilt set of data shares by associating the rebuilt set of data shares with a second split key.

11. The method of claim 10, further comprising:

retrieving headers associated with the minimum number of data shares;

extracting a key encryption key from the retrieved headers;

encrypting the second split key with the key encryption key; and

storing the encrypted second split key within the headers of the rekeyed data shares.

12. The method of claim 10, further comprising storing at least one of the rekeyed data shares on a storage network.

13. The method of claim 11, wherein the storage network includes one of a private cloud, a public cloud, a hybrid cloud, a removable storage device, and a mass storage device.

14. A method for associating stubs with a set of data shares on the file system of a storage network, the method comprising

generating the set of data shares from an encrypted data set by an information dispersal algorithm;

generating a set of stubs associated with the generated data shares, wherein each stub corresponds to a respective data share, and wherein each stub includes information associated with the respective data share; and

storing the set of stubs in a location on the storage network.

15. The method of claim 14, wherein the information includes one of the name of the respective data share, a date the respective data share was created, a time at which the respective data share was last modified, a pointer to the location of the respective data share within the file system

16. The method of claim 14, wherein the storage network includes one or more storage devices associated with one of a private cloud, a public cloud, a hybrid cloud, a removable storage device, and a mass storage device.

17. The method of claim 14, further comprising:

receiving a command to view the information associated with the generated data shares;

retrieving the stubs from the location on the storage network;

extracting the information from the stubs to create a file system of data shares; and

displaying the file system of data shares.

18. The method of claim 14, wherein the stubs are stored within the headers of the generated data shares, and wherein retrieving includes retrieving the headers of the generated data shares.

19. The method of claim 18, wherein less than all of the headers are retrieved.

20. The method of claim 14, wherein the stubs are stored in a stub directory, and wherein retrieving includes retrieving the stubs from the stub directory.

21. The method of claim 14, further comprising:

receiving an indication of a virtual directory or a physical directory in the storage network in which to store the stubs.

22. The method of claim 21, wherein the indication is received from a user.

23. A coprocessor acceleration device for acceleration of secure data processing, comprising:

a memory for storing data;

a main processor coupled to the memory; and

a coprocessor coupled to the main processor and the memory configured to perform dedicated secure parsing functions including at least one of encrypting data, splitting data, and decrypting data.

24. The device of claim 23, wherein splitting data includes the use of an information dispersal algorithm (IDA).

25. The device of claim 23, further comprising a field programmable gate array coupled to the coprocessor.

26. The device of claim 25, wherein the FPGA performs at least one of encrypting the parsed data, or decrypting encrypted data

27. The device of claim 23, wherein the coprocessor is coupled to the main processor via a PCIe bus.

28. The device of claim 23, wherein the coprocessor is coupled to the main processor via an HT bus.

29. The device of claim 23, wherein the memory includes a dedicated memory for the main processor.

30. The device of claim 23, wherein the memory includes a dedicated memory for the coprocessor.

31. The device of claim 23, wherein the coprocessor is a redundant array of independent disks (RAID) processing unit that implements one or more RAID functions.

32. A method for securing data using a portable device, the method comprising:

generating at least two portions of data from a set of data based at least in part on a key, wherein the at least two portions of data and the key are sufficient to reconstruct the set of data; and

storing the key on the portable device.

33. The method of claim 32, wherein the portable device is a removable storage device.

34. The method of claim 33, wherein the removable storage device couples to an end user device via a Universal Serial Bus (USB) interface.

35. The method of claim 32, further comprising storing at least one of the generated data portions on the portable device.

36. The method of claim 32, wherein the key is one of an encryption key, a split key and an authentication key.

37. The method of claim 32, wherein the at least two portions of data are generated using an information dispersal algorithm (IDA) and a split key associated with the IDA.

38. A method for securing data using a portable device, the method comprising:

generating at least two portions of data from a set of data based at least in part on a key, wherein the at least two portions of data and the key are sufficient to reconstruct the set of data; and

storing at least one of the generated data portions on the portable device.

39. The method of claim 38, wherein the portable device is a removable storage device.

40. The method of claim 39, wherein the removable storage device couples to an end user device via a Universal Serial Bus (USB) interface.

41. The method of claim 38, further comprising storing the key on the portable device.

42. The method of claim 38, wherein the key is one of an encryption key, a split key and an authentication key.

43. The method of claim 38, wherein the at least two portions of data are generated using an information dispersal algorithm (IDA) and a split key associated with the IDA.

44. A method for securing the file name of a file to be split and stored on a storage network, the method comprising:

processing the file name of the file using an authentication algorithm to obtain an authentication value; and

retrieving the data shares corresponding to the file by searching the share locations on the storage network for file names of data shares with authentication values that match the authentication value of the file.

45. The method of claim 44, further comprising:

generating one or more data shares associated with the authenticated file name using an information dispersal algorithm; and

storing the generated data shares on one or more data share locations in the storage network.

46. The method of claim 44, wherein the storage network includes one of a private cloud, a public cloud, a hybrid cloud, a removable storage device, and a mass storage device.

47. The method of claim 44, wherein the authentication algorithm is an HMAC-SHA256 algorithm.

48. The method of claim 44, further comprising appending additional information to the file name of the file prior to the processing.

49. The method of claim 48, wherein the additional information includes a number associated with a data share location.

50. A method for securing the file name of a file to be split and stored on a storage network, the method comprising:

encrypting the file name of the file using an encryption algorithm;

generating one or more data shares associated with the encrypted file name using an information dispersal algorithm;

storing the generated data shares on one or more data share locations in the storage network; and

regenerating the file name of the file by decrypting the file name of one of the generated data shares.

51. The method of claim 50, wherein the storage network includes one of a private cloud, a public cloud, a hybrid cloud, a removable storage device, and a mass storage device.

52. The method of claim 50, wherein the encryption algorithm is an AES algorithm.

53. The method of claim 50, further comprising appending additional information to the file name of the file prior to the encrypting.

54. The method of claim 53, wherein the additional information includes a number associated with a data share location.

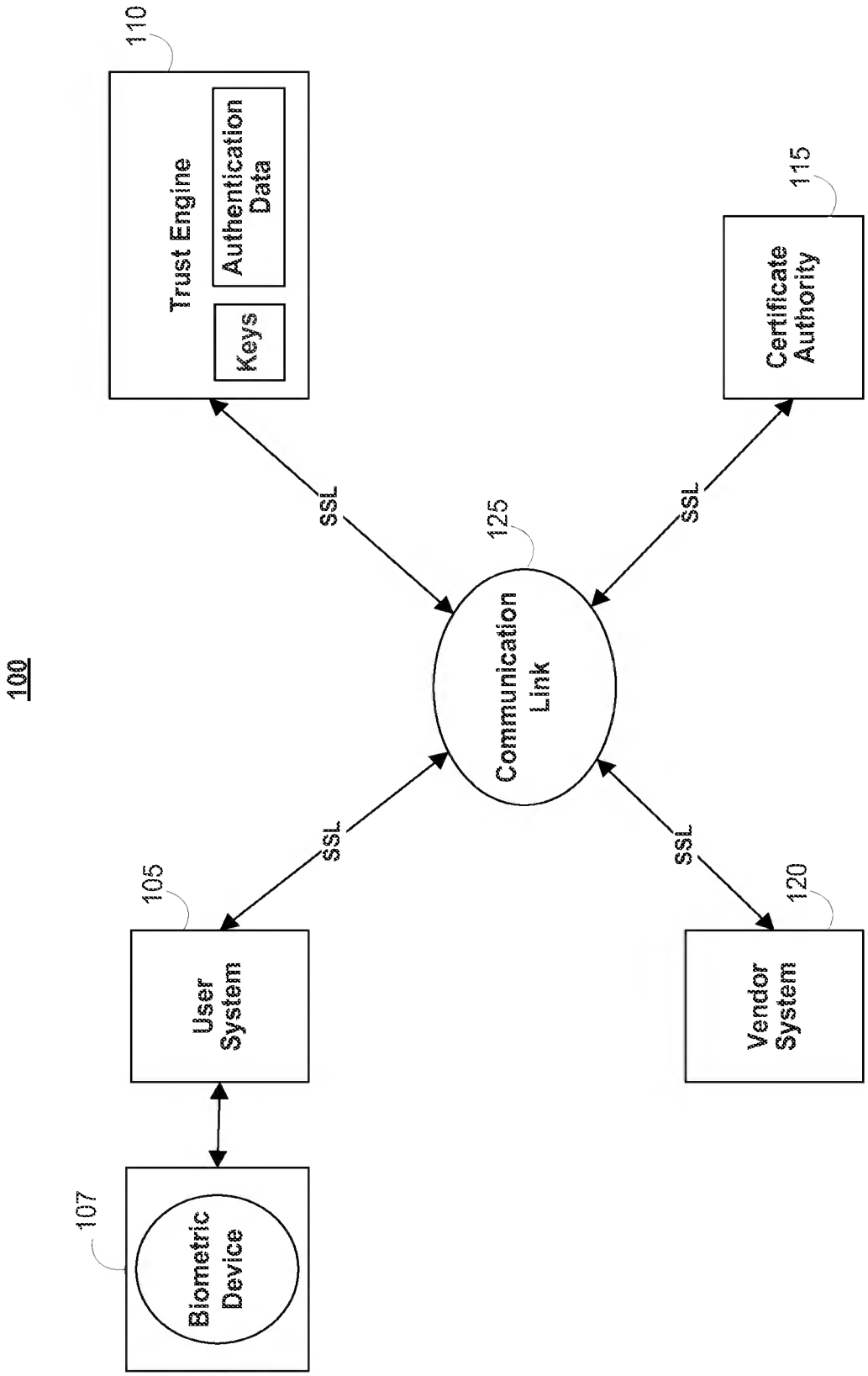

100
110
Trust Engine
Authentication Data
Keys
115
Certificate Authority
SSL
SSL
125
Communication Link
SSL
SSL
105
User System
120
Vendor System
107
Biometric Device

FIG. 1

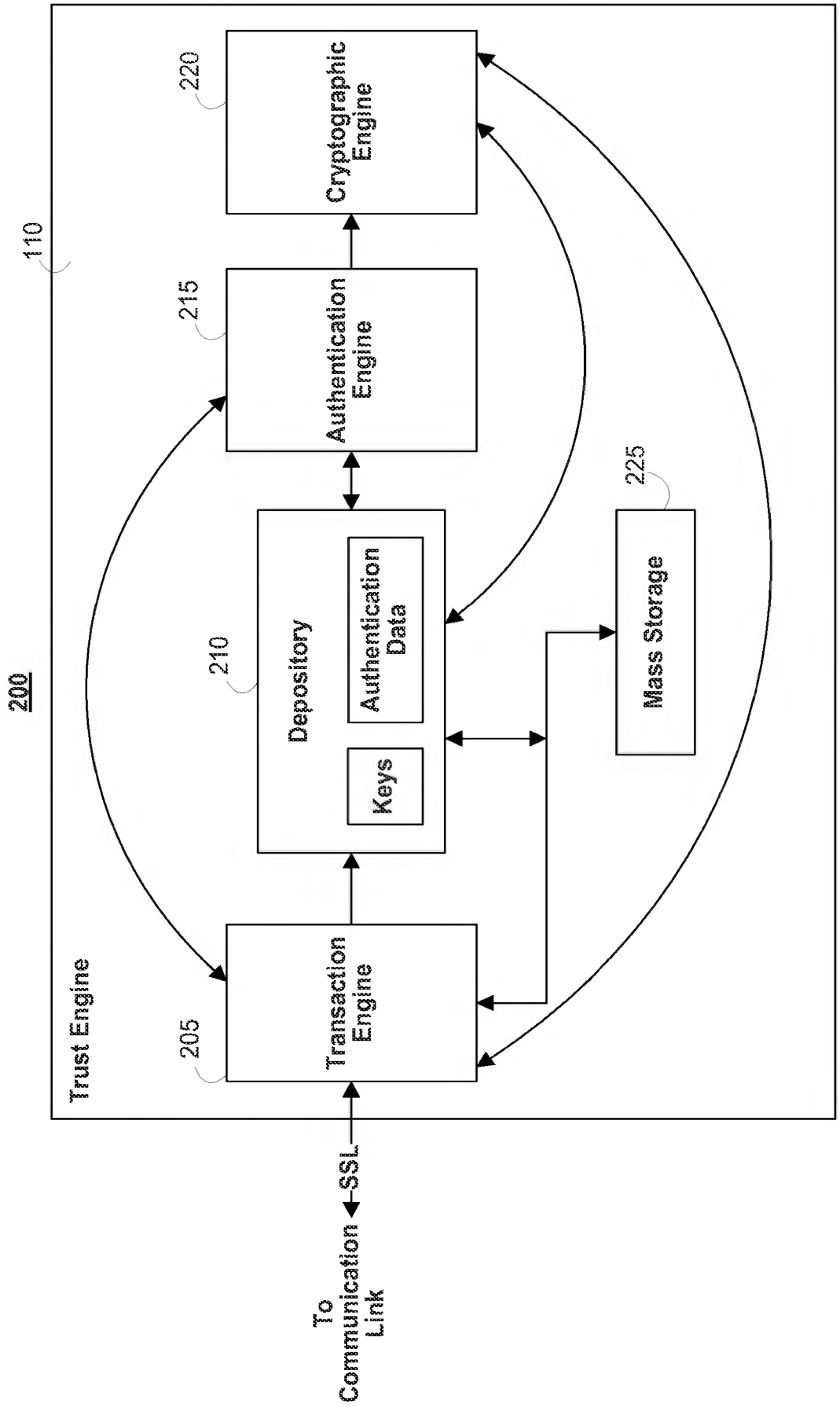

200
110
Trust Engine
205
Transaction Engine
210
Depository
Authentication Data
Keys
215
Authentication Engine
220
Cryptographic Engine
225
Mass Storage
SSL
To Communication Link

FIG. 2

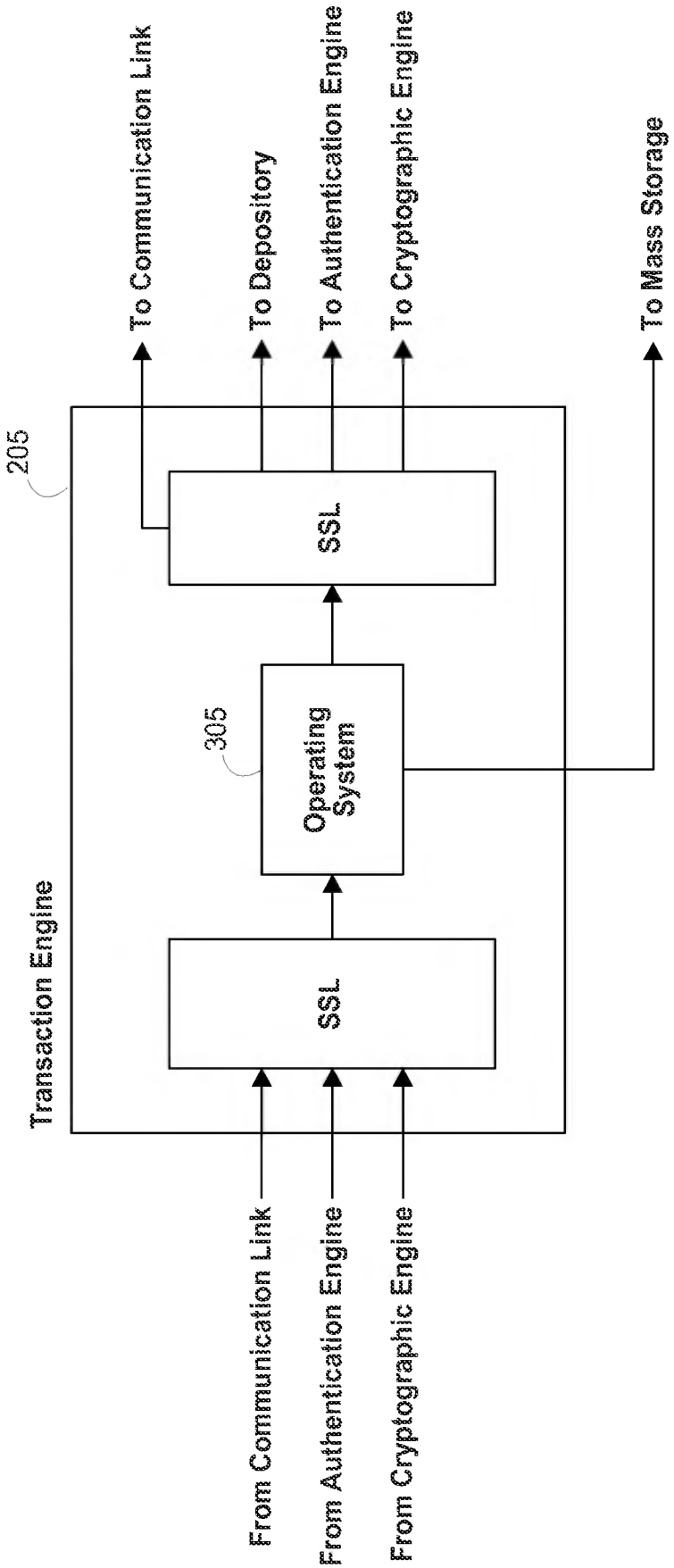

To Communication Link
To Depository
To Authentication Engine
To Cryptographic Engine
To Mass Storage
205
SSL
305
Operating System
SSL
Transaction Engine
From Communication Link
From Authentication Engine
From Cryptographic Engine

FIG. 3

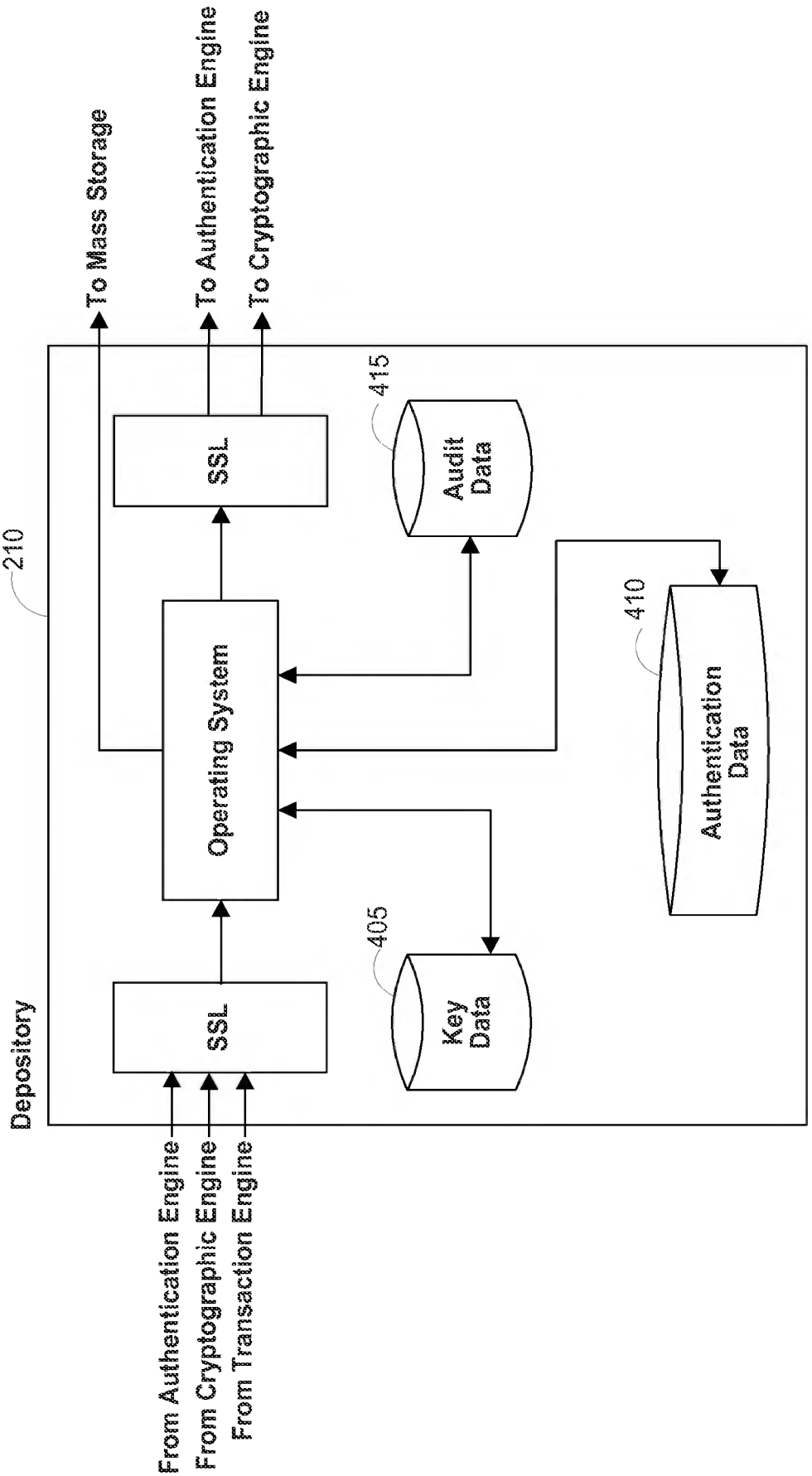

To Mass Storage
To Authentication Engine
To Cryptographic Engine
415
Audit Data
SSL
210
Operating System
410
Authentication Data
405
Key Data
SSL
Depository
From Authentication Engine
From Cryptographic Engine
From Transaction Engine

FIG. 4

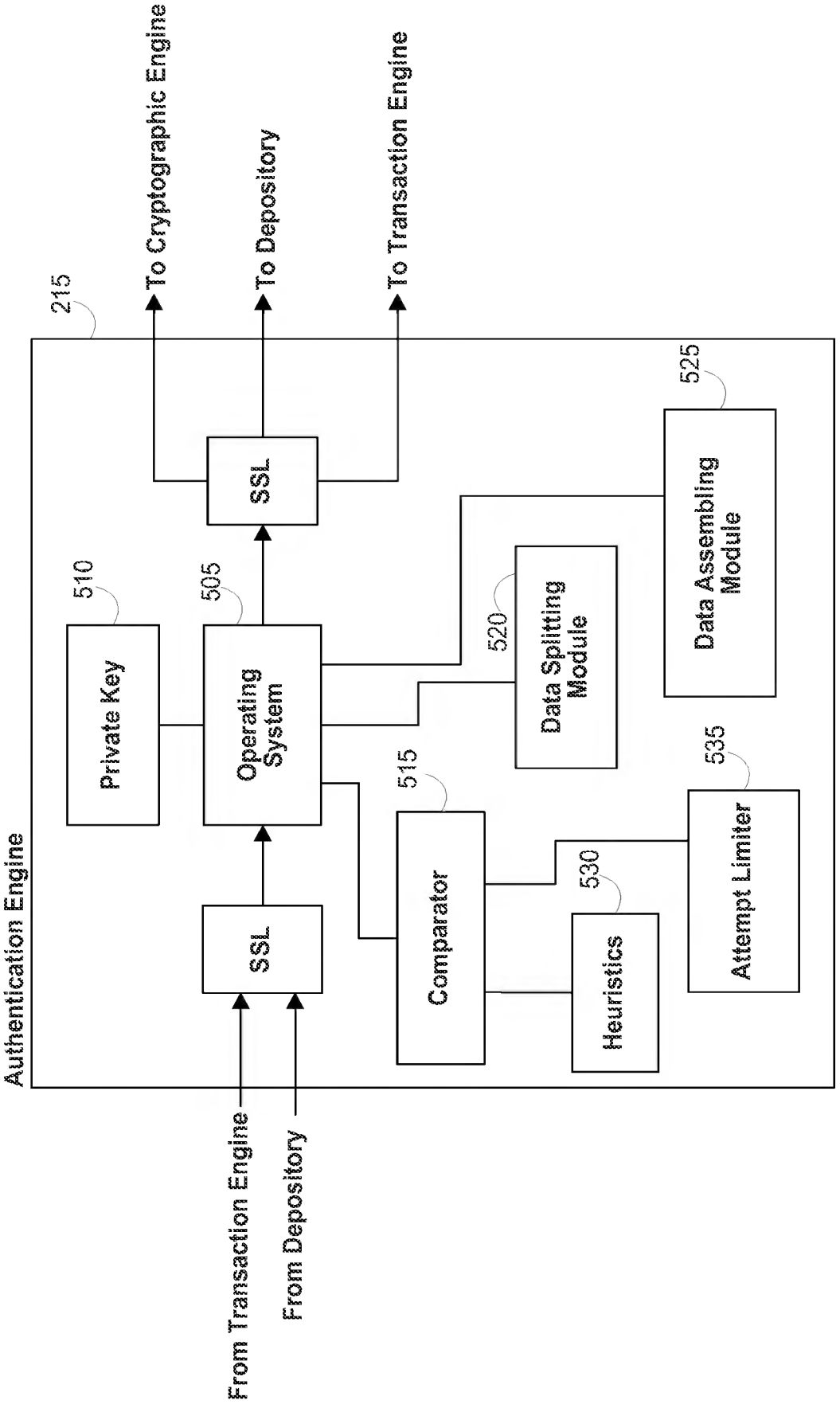

Authentication Engine
215
To Cryptographic Engine
To Depository
To Transaction Engine
SSL
SSL
Private Key
510
Operating System
505
Data Splitting Module
520
Data Assembling Module
525
Comparator
515
Heuristics
530
Attempt Limiter
535
From Transaction Engine
From Depository

FIG. 5

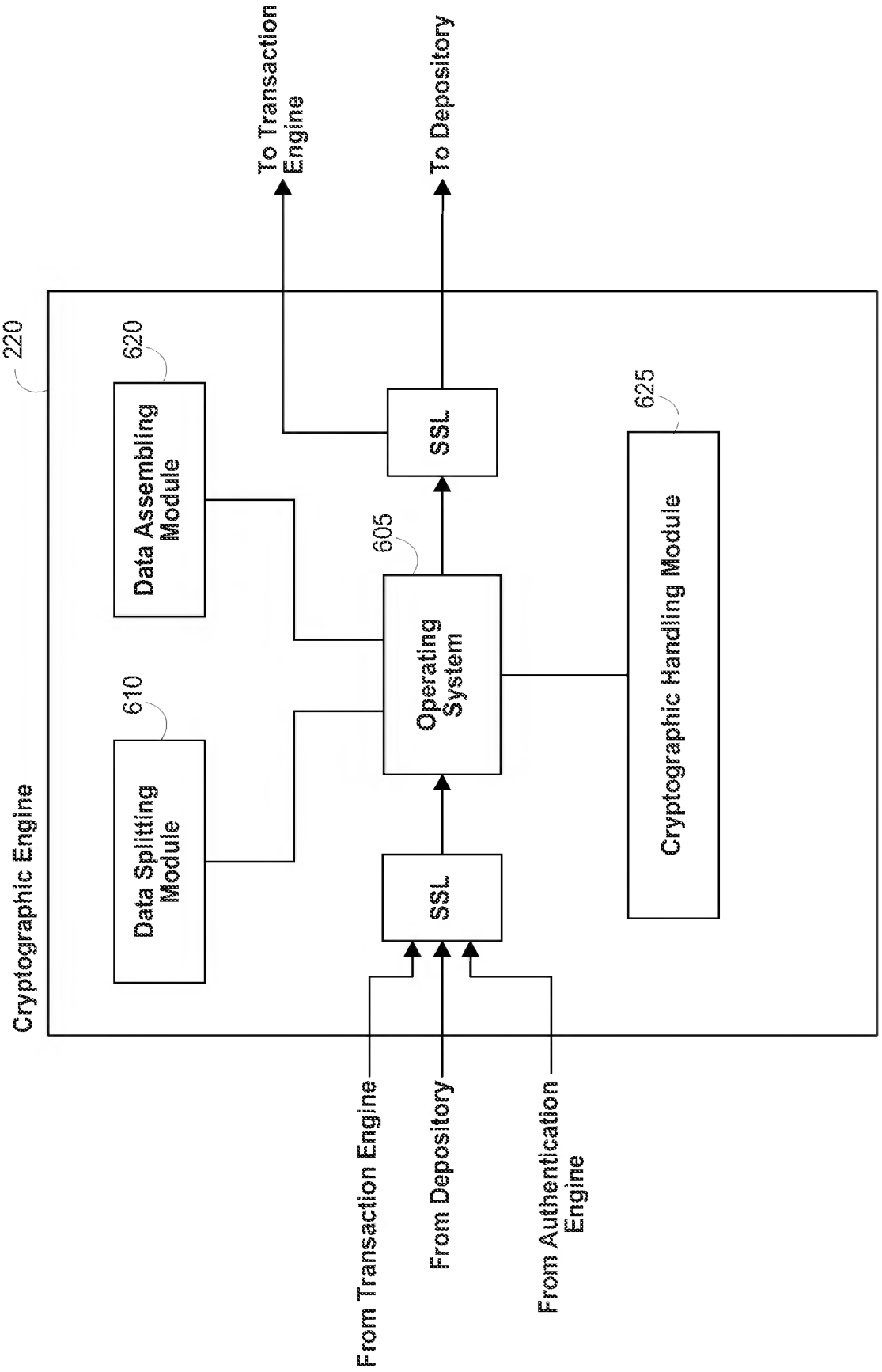

Cryptographic Engine
220
Data Splitting Module
610
Data Assembling Module
620
Operating System
605
SSL
SSL
Cryptographic Handling Module
625
From Transaction Engine
From Depository
From Authentication Engine
To Transaction Engine
To Depository

FIG. 6

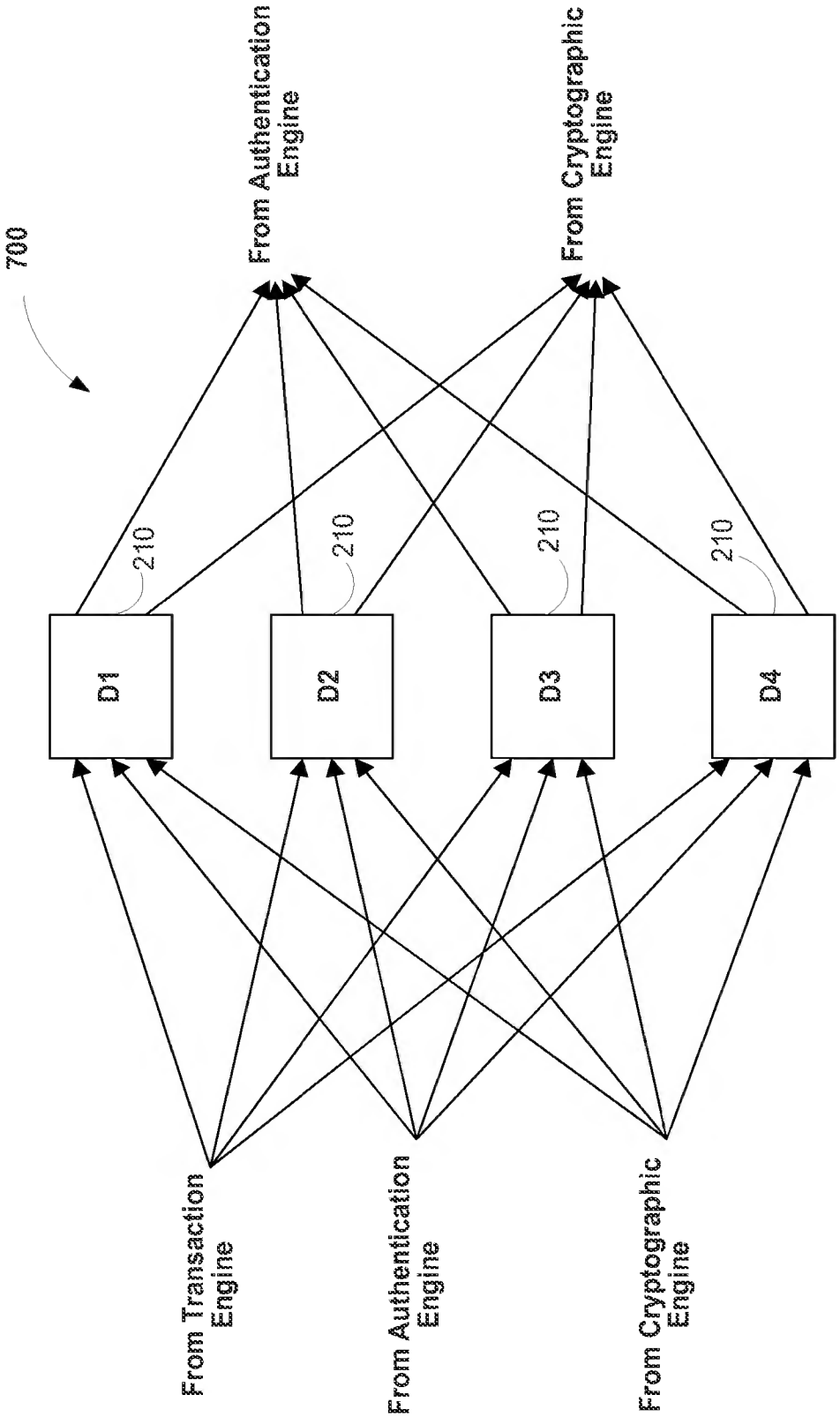

700
From Authentication Engine
From Cryptographic Engine
210
210
210
210
D1
D2
D3
D4
From Transaction Engine
From Authentication Engine
From Cryptographic Engine

FIG. 7

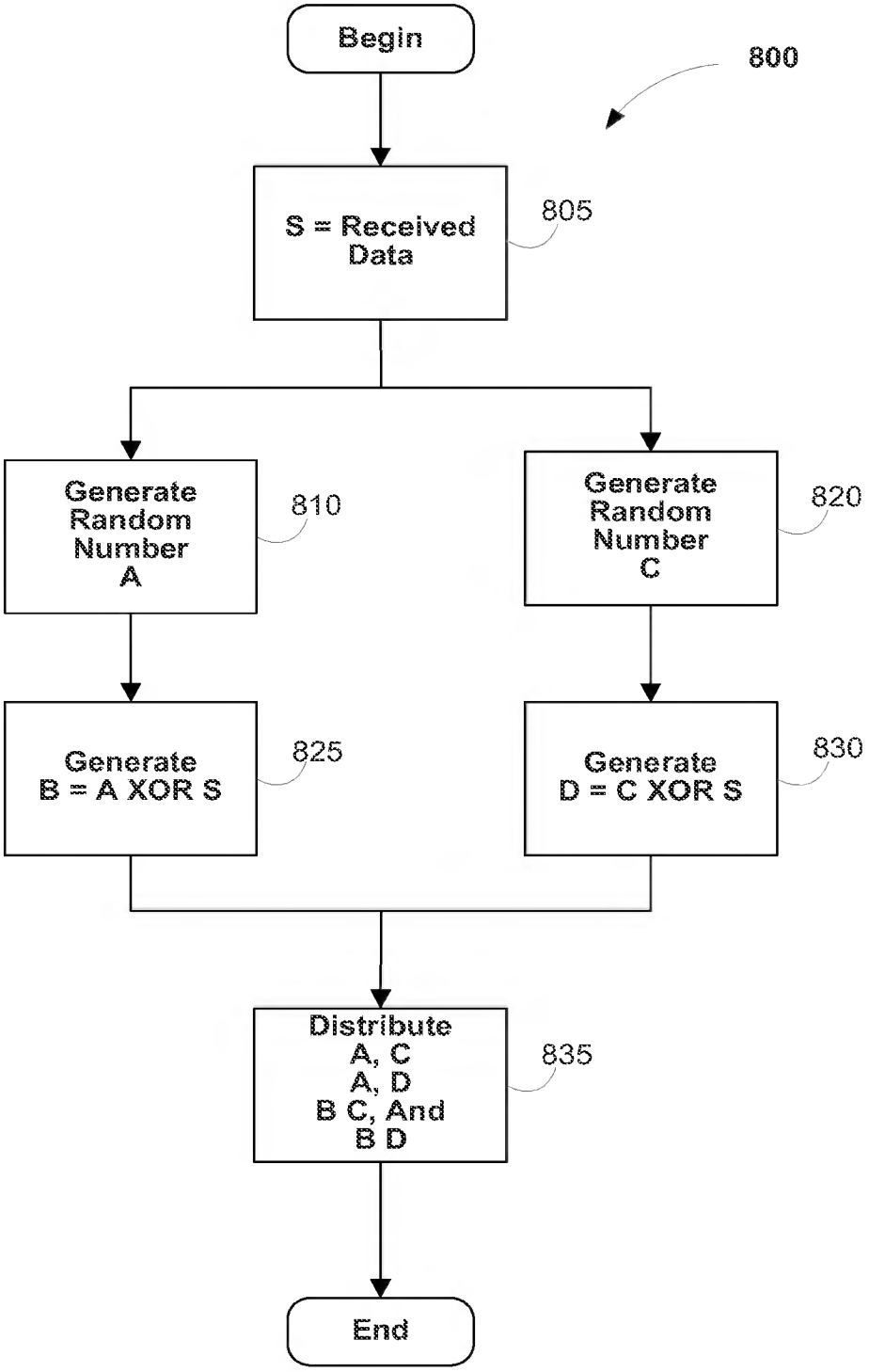

Begin
800
S = Received Data
805
Generate Random Number A
810
Generate Random Number C
820
Generate B = A XOR S
825
Generate D = C XOR S
830
Distribute A, C A, D B C, And B D
835
End

FIG. 8

900

Enrollment Data Flow				
Send	Receive	SSL	Action	
User	Transaction Engine (TE)	1/2	Transmit Enrollment Authentication Data (B) and the User ID (UID) encrypted with the Public Key of the Authentication Engine (AE) as (PUB_AE(UID,B))	905
TE	AE	Full	Forward Transmission	915
			AE Decrypts and Splits Forwarded Data	920
AE	The Xth Depository (DX)	Full	Store Respective Portion of Data	925
When Digital Certificate Requested				
AE	Cryptographic Engine (CE)	Full	Request Key Generation	930
			CE Generates and Splits Key	935
CE	TE	Full	Transmit Request for Digital Certificate	945
TE	Certification Authority (CA)	1/2	Transmit Request	950
CA	TE	1/2	Transmit Digital Certificate	955
TE	User	1/2	Transmit Digital Certificate	960
TE	MS	Full	Store Digital Certificate	
CE	DX	Full	Store Respective Portion of Key	965

FIG. 9, Panel A

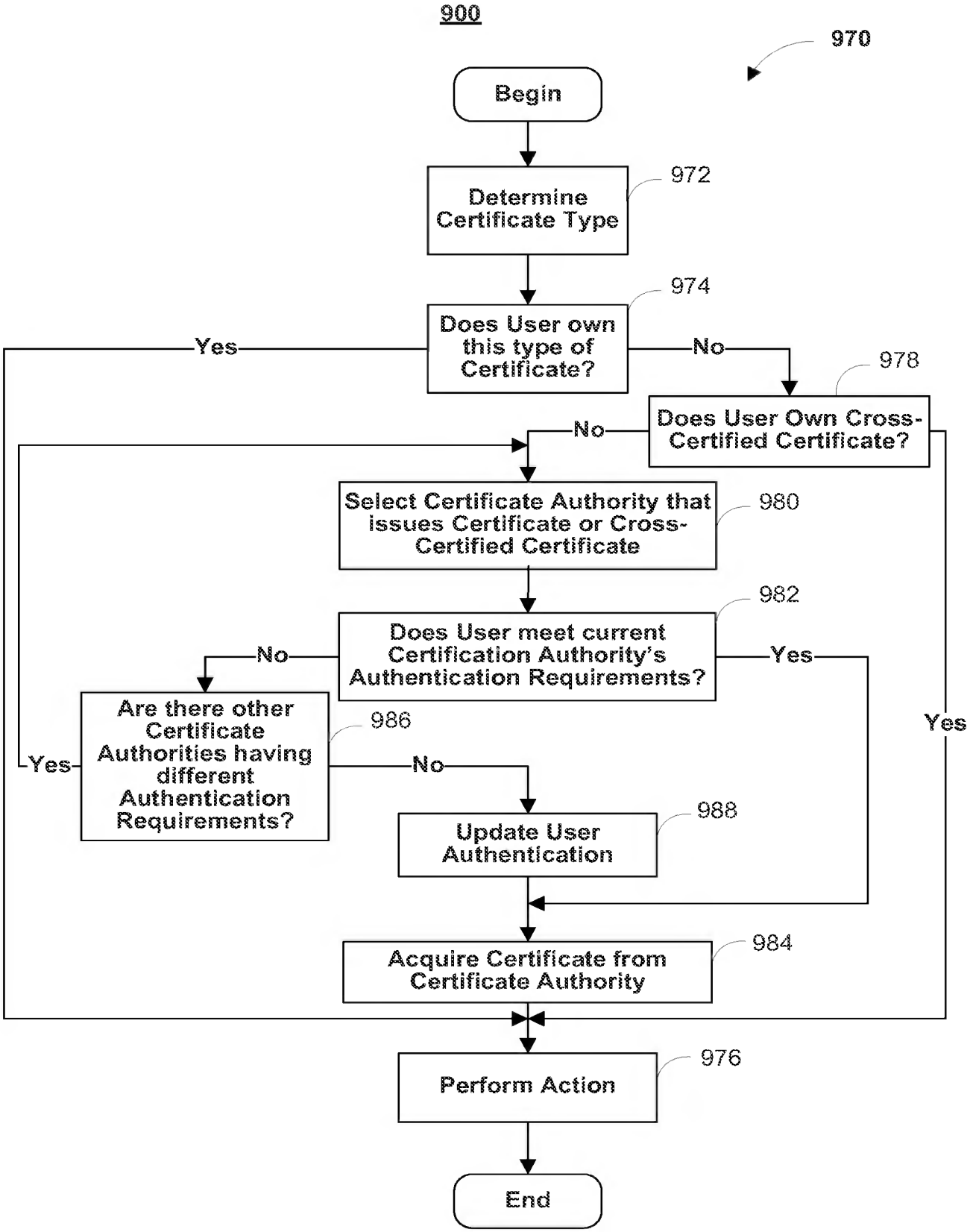

900
970
Begin
Determine Certificate Type
972
Does User own this type of Certificate?
974
Yes
No
Does User Own Cross-Certified Certificate?
978
No
Yes
Select Certificate Authority that issues Certificate or Cross-Certified Certificate
980
Does User meet current Certification Authority's Authentication Requirements?
982
No
Yes
Are there other Certificate Authorities having different Authentication Requirements?
986
Yes
No
Update User Authentication
988
Acquire Certificate from Certificate Authority
984
Perform Action
976
End

FIG. 9, Panel B

1000

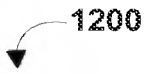
	Authentication Data Flow			
	SEND	RECEIVE	SSL	ACTION
1005	User	Vendor	1/2	Transaction occurs, such as selecting purchase
1010	Vendor	User	1/2	Transmit transaction ID (TID) and authentication request (AR)
				Authentication data (B') is gathered from User
1015	User	TE	1/2	Transmit TID and B' wrapped in the Public Key of the Authentication Engine (AE), as (PUB_AE(TID, B'))
1020	TE	AE	Full	Forward transmission
				Enrollment authentication data (B) is requested and gathered
1025	Vendor	Transaction Engine (TE)	Full	Transmits TID, AR
1030	TE	Mass Storage (MS)	Full	Create Record in database
1035	TE	The Xth Depository (DX)	Full	UID, TID
1040	DX	AE	Full	Transmit the TID and the portion of the authentication data stored at enrollment (BX) as (PUB_AE(TID, BX))
1045				AE assembles B and compares to B'
1050	AE	TE	Full	TID, the filled in AR
1055	TE	Vendor	Full	TID, Yes/No
	TE	User	1/2	TID, confirmation message

FIG. 10

1100

Signing Data Flow			
SEND	RECEIVE	SSL	ACTION
User	Vendor	1/2	Transaction occurs, such as agreeing on a deal
Vendor	User	1/2	Transmit transaction identification number (TID), authentication request (AR), and agreement or message (M)
			Current authentication data (B') and a hash of the message received by the User (h(M')) is is gathered from User
User	TE	1/2	Transmit TID, B', AR, and h(M') wrapped in the Public Key of the Authentication Engine (AE), as (PUB_AE(TID, B', h(M'))
TE	AE	Full	Forward transmission
			Gather enrollment authentication data
Vendor	Transaction Engine (TE)	Full	Transmits UID, TID, AR, and a hash of the message (h(M')).
TE	Mass Storage (MS)	Full	Create Record in database
TE	The Xth Depository (DX)	Full	UID, TID
DX	AE	Full	Transmit the TID and the portion of the authentication data stored at Enrollment (BX), as (PUB_AE(TID, BX))
			The original vendor message is transmitted to the AE
TE	AE	Full	Transmit h(M)
1103			AE assembles B, compares to B' and compares h(M) to h(M')
1105 AE	Cryptographic Engine (CE)	Full	Request for digital signature and a message to be signed, for example, the hashed message
1110 AE	DX	Full	TID, signing UID
1115 DX	CE	Full	Transmit the portion of the Crypto-graphic Key corresponding to the signing party
1120			CE assembles key and signs
1125 CE	AE	Full	Transmit the digital signature (S) of signing party
1130 AE	TE	Full	TID, the filled in AR, h(M), and S
1135 TE	Vendor	Full	TID, a receipt=(TID, Yes/No, and S), and the digital signature of the trust engine, for example, a hash of the receipt encrypted with the trust engine's Private Key (Priv_TE(h(receipt)))
1140 TE	User	1/2	TID, confirmation message

FIG. 11



Encryption/Decryption Data Flow				
Send	Receive	SSL	Action	
Decryption				
			Perform Authentication Data Process 1000, include the Session Key (sync) in the AR, where the sync has been encrypted with the Public Key of the User as PUB_USER(SYNC)	
			Authenticate the User	1205
AE	CE	Full	Forward PUB_USER(SYNC) to CE	1210
AE	DX	Full	UID, TID	1215
DX	CE	Full	Transmit the TID and the portion of the Private Key as (PUB_AE(TID, KEY_USER))	1220
			CE assembles the Cryptographic Key and decrypts the sync	1225
CE	AE	Full	TID, the filled in AR including decrypted sync	1230
AE	TE	Full	Forward to TE	
TE	Requesting APP/Vendor	1/2	TID, Yes/No, Sync	
Encryption				
Requesting APP/ Vendor	TE	1/2	Request for Public Key of User	1235
TE	MS	Full	Request Digital Certificate	1240
MS	TE	Full	Transmit Digital Certificate	1245
TE	Requesting APP/Vendor	1/2	Transmit Digital Certificate	1250

FIG. 12

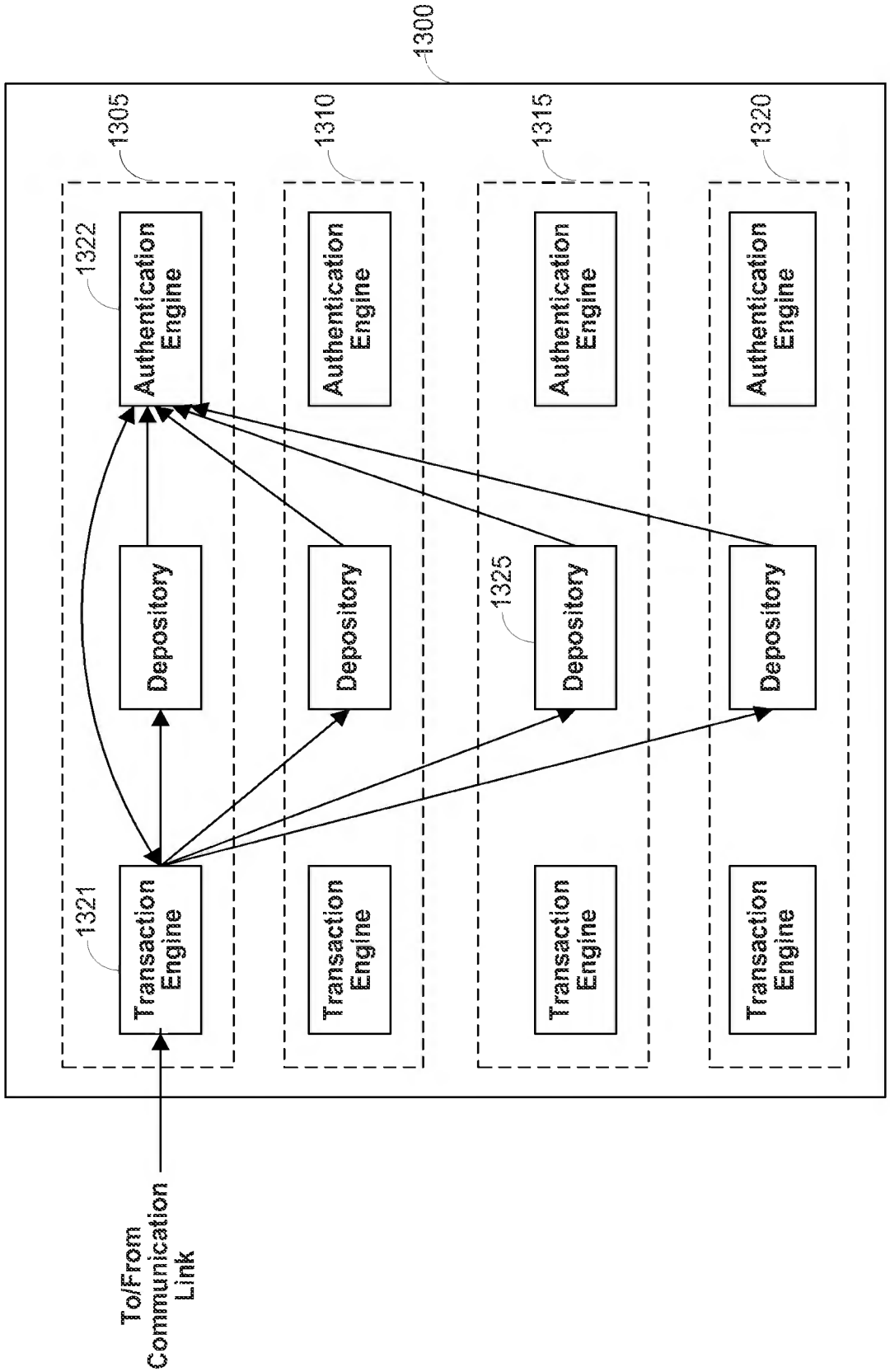

1300
1305
1310
1315
1320
1322
Authentication Engine
Depository
1325
Transaction Engine
1321
To/From Communication Link

FIG. 13

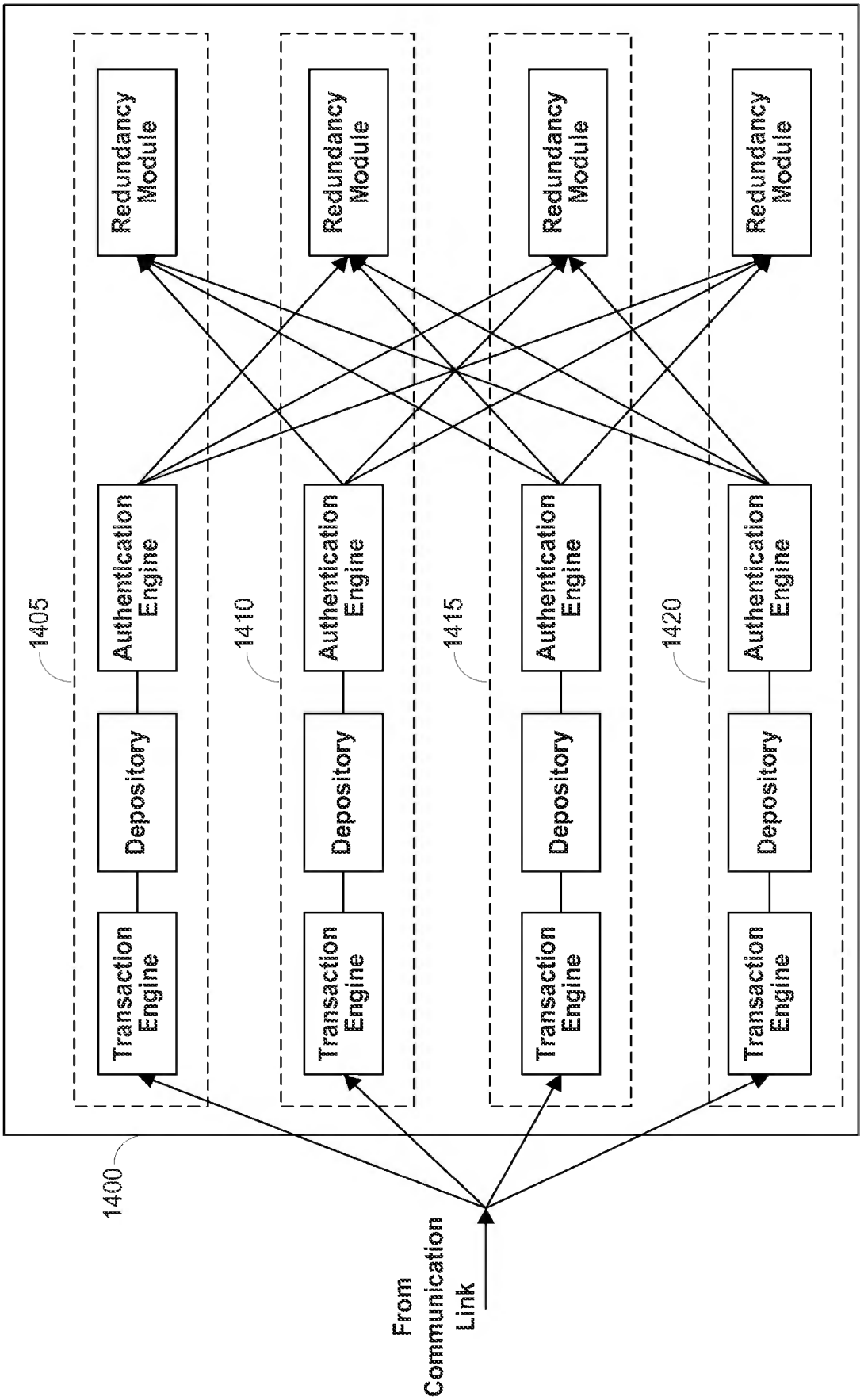

1400
From Communication Link
Transaction Engine
Depository
Authentication Engine
Redundancy Module
1405
1410
1415
1420

FIG. 14

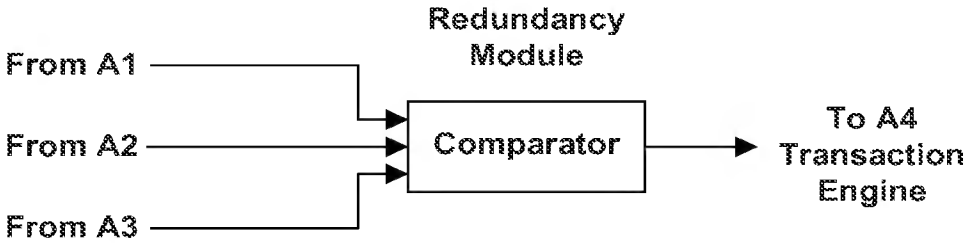

Redundancy
Module
From A1
From A2
From A3
Comparator
To A4
Transaction
Engine

FIG. 15

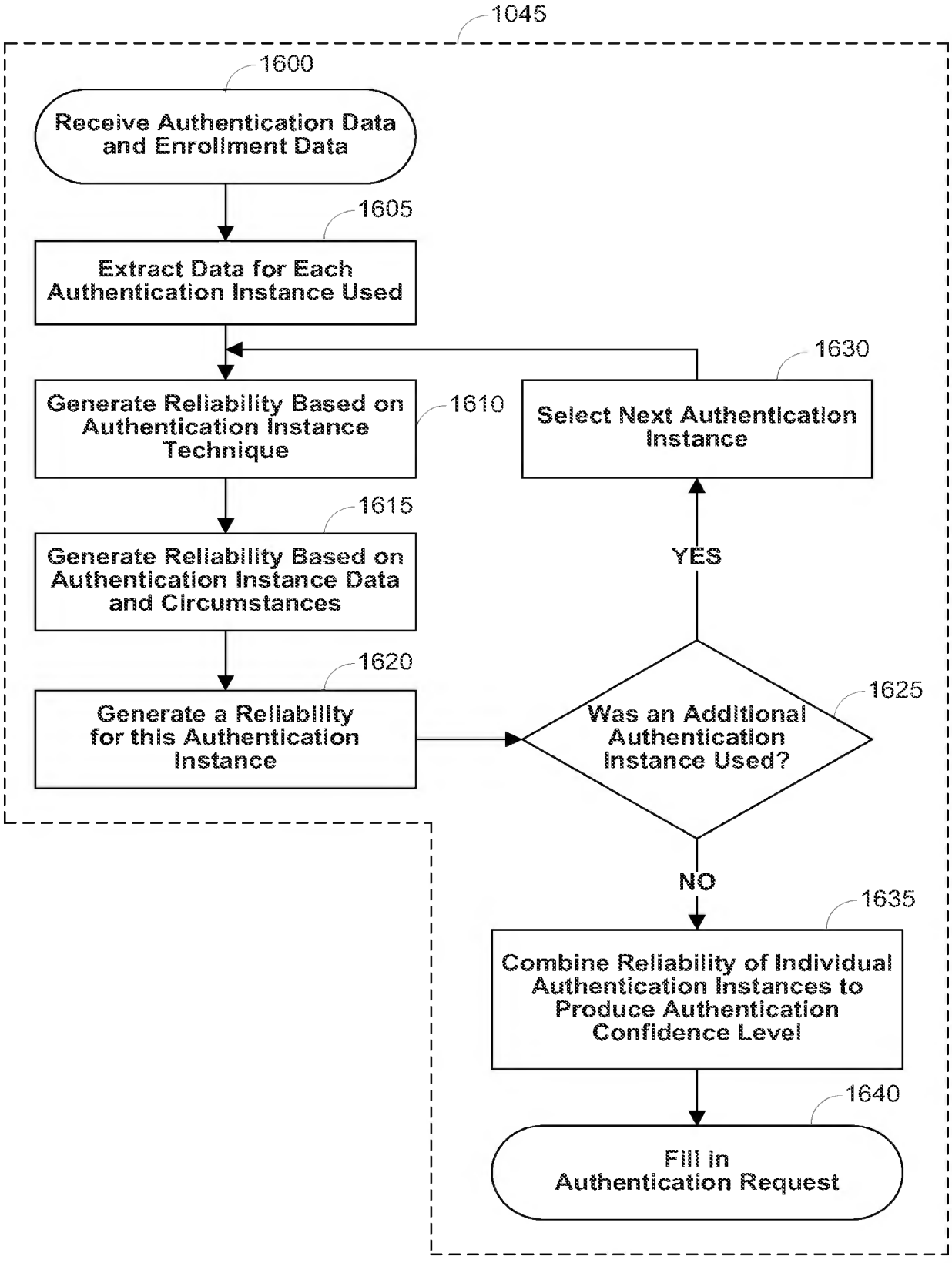

1045
1600
Receive Authentication Data and Enrollment Data
1605
Extract Data for Each Authentication Instance Used
Generate Reliability Based on Authentication Instance Technique
1610
Generate Reliability Based on Authentication Instance Data and Circumstances
1615
Generate a Reliability for this Authentication Instance
1620
Was an Additional Authentication Instance Used?
1625
YES
Select Next Authentication Instance
1630
NO
Combine Reliability of Individual Authentication Instances to Produce Authentication Confidence Level
1635
Fill in Authentication Request
1640

FIG. 16

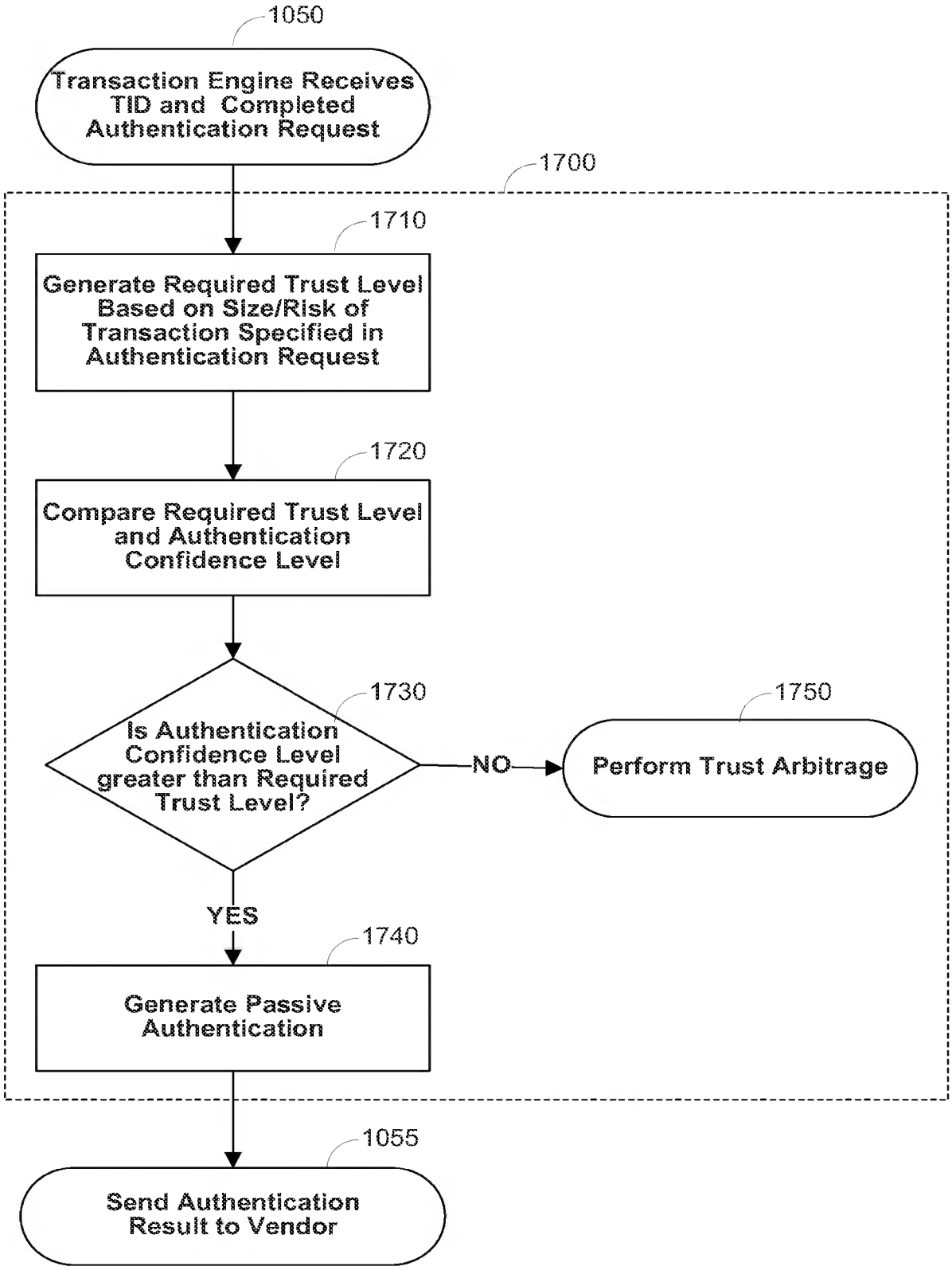

1050
Transaction Engine Receives TID and Completed Authentication Request
1700
1710
Generate Required Trust Level Based on Size/Risk of Transaction Specified in Authentication Request
1720
Compare Required Trust Level and Authentication Confidence Level
1730
Is Authentication Confidence Level greater than Required Trust Level?
NO
1750
Perform Trust Arbitrage
YES
1740
Generate Passive Authentication
1055
Send Authentication Result to Vendor

FIG. 17

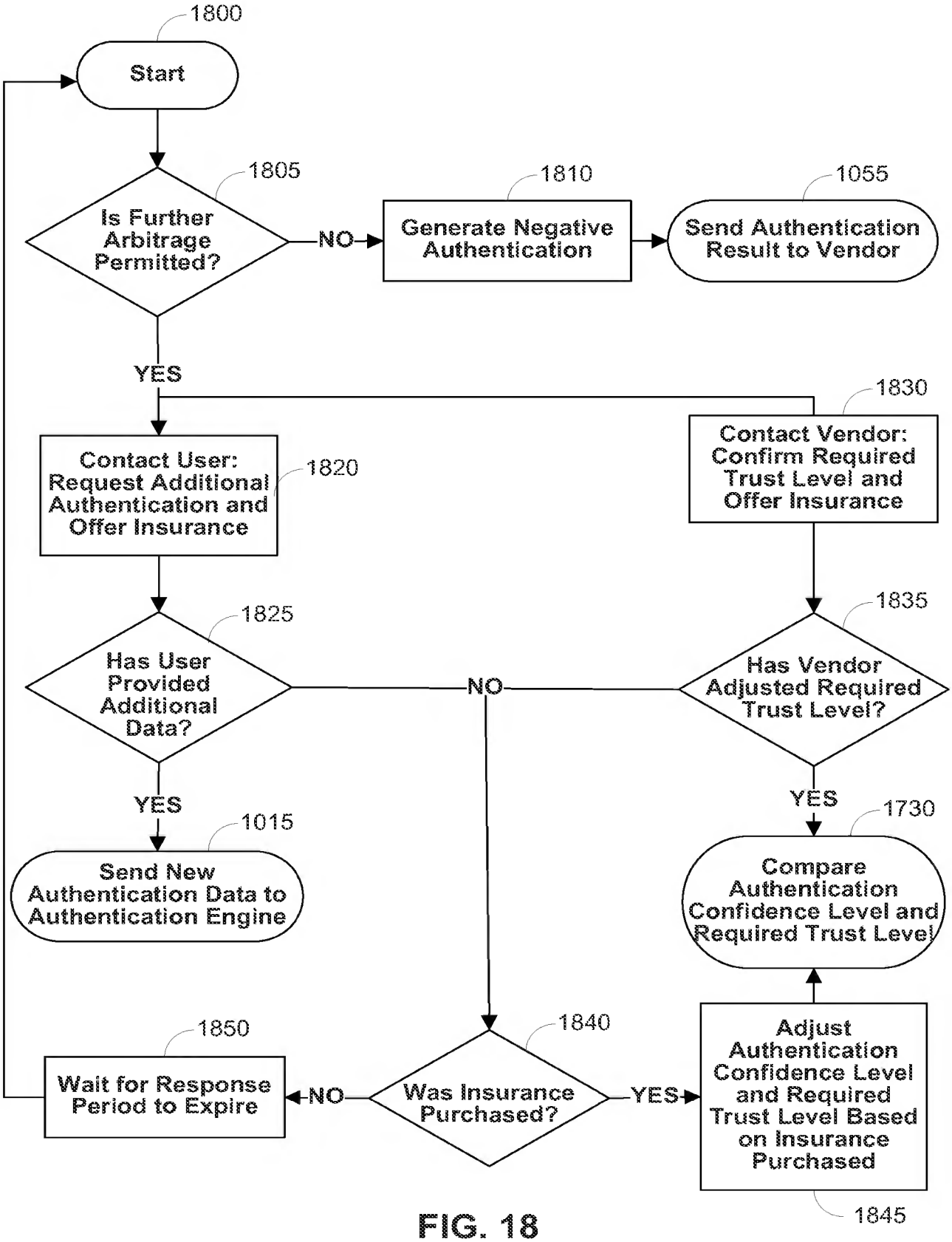

1800
Start
1805
Is Further Arbitrage Permitted?
NO
1810
Generate Negative Authentication
1055
Send Authentication Result to Vendor
YES
1830
Contact Vendor: Confirm Required Trust Level and Offer Insurance
Contact User: Request Additional Authentication and Offer Insurance
1820
1825
Has User Provided Additional Data?
NO
1835
Has Vendor Adjusted Required Trust Level?
YES
1015
Send New Authentication Data to Authentication Engine
YES
1730
Compare Authentication Confidence Level and Required Trust Level
1850
Wait for Response Period to Expire
NO
1840
Was Insurance Purchased?
YES
Adjust Authentication Confidence Level and Required Trust Level Based on Insurance Purchased
1845

FIG. 18

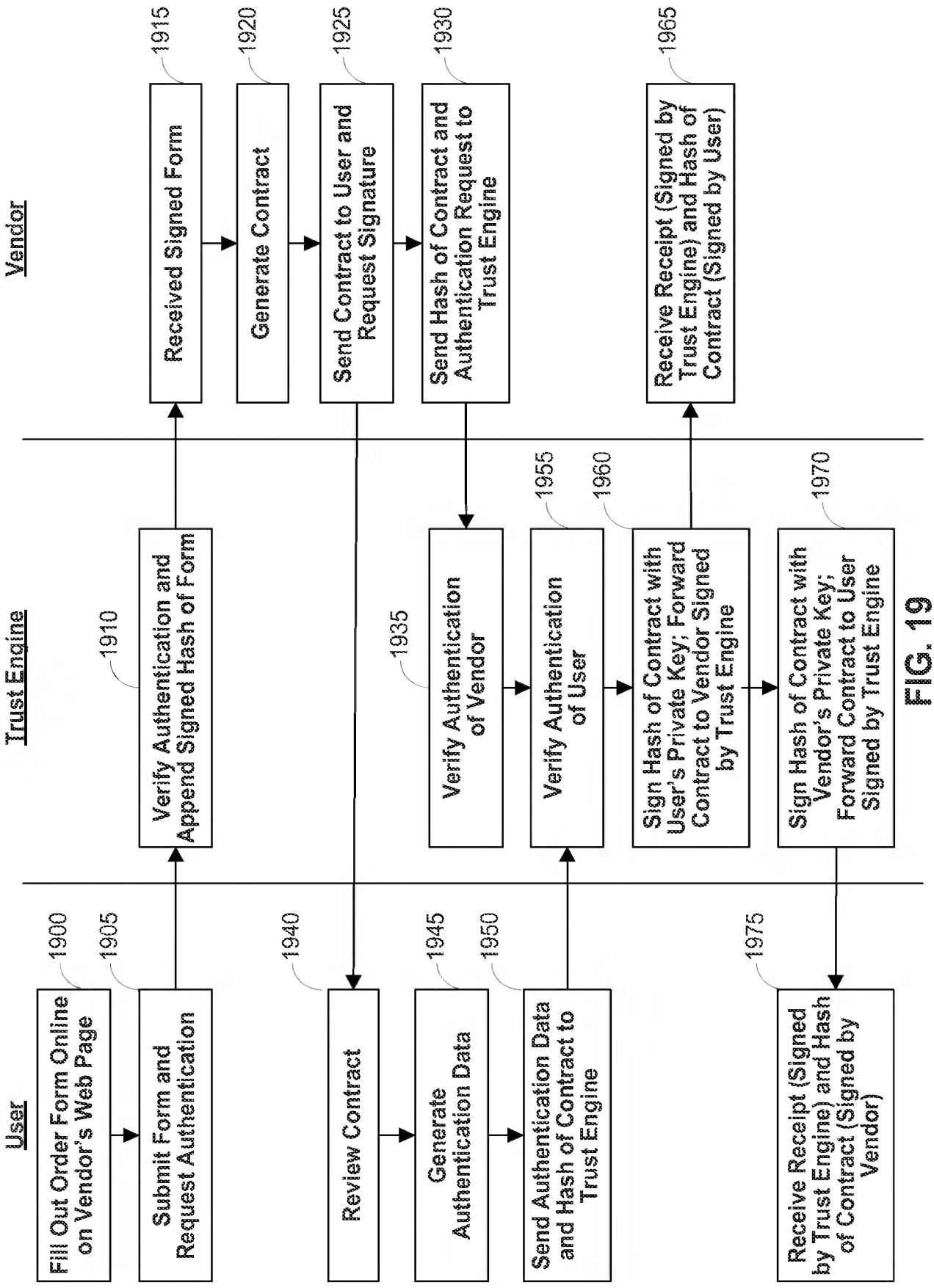

User
Trust Engine
Vendor
Fill Out Order Form Online on Vendor's Web Page
1900
Submit Form and Request Authentication
1905
Verify Authentication and Append Signed Hash of Form
1910
Received Signed Form
1915
Generate Contract
1920
Send Contract to User and Request Signature
1925
Send Hash of Contract and Authentication Request to Trust Engine
1930
Review Contract
1940
Generate Authentication Data
1945
Send Authentication Data and Hash of Contract to Trust Engine
1950
Verify Authentication of Vendor
1935
Verify Authentication of User
1955
Sign Hash of Contract with User's Private Key; Forward Contract to Vendor Signed by Trust Engine
1960
Receive Receipt (Signed by Trust Engine) and Hash of Contract (Signed by User)
1965
Sign Hash of Contract with Vendor's Private Key; Forward Contract to User Signed by Trust Engine
1970
Receive Receipt (Signed by Trust Engine) and Hash of Contract (Signed by Vendor)
1975

FIG. 19

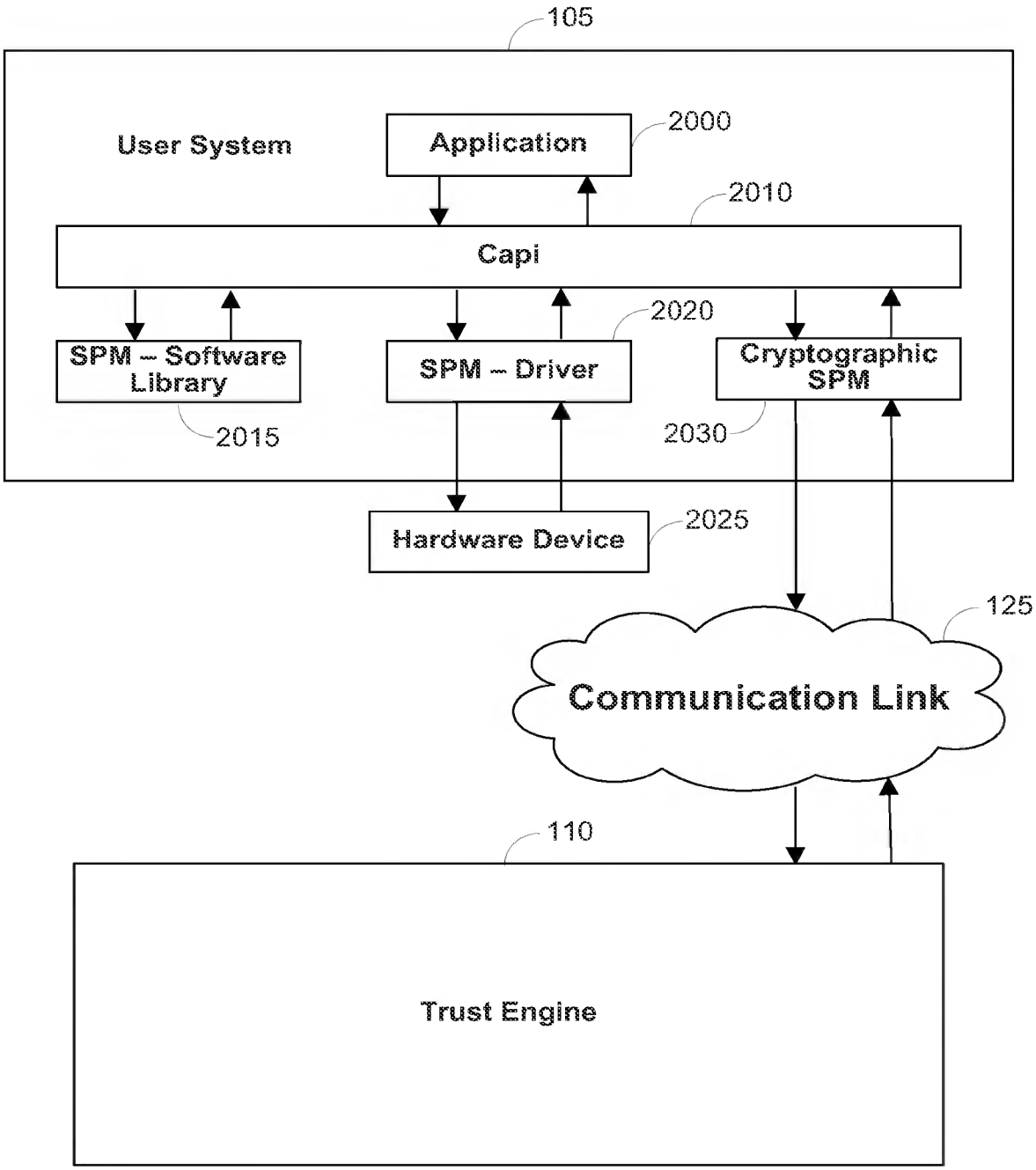

105
User System
Application
2000
2010
Capi
SPM – Software Library
2015
SPM – Driver
2020
Cryptographic SPM
2030
Hardware Device
2025
125
Communication Link
110
Trust Engine

FIG. 20

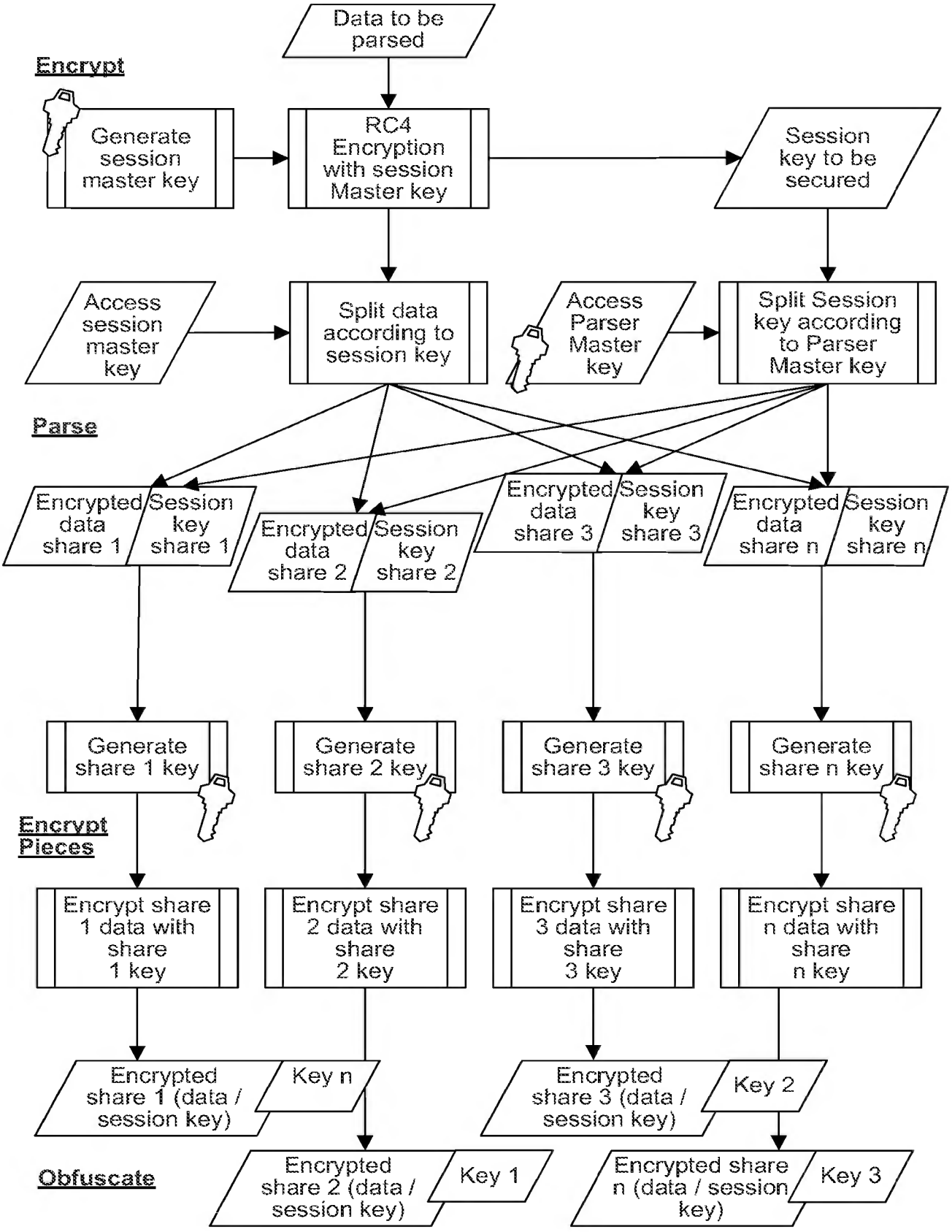

Data to be parsed
Encrypt
Generate session master key
RC4 Encryption with session Master key
Session key to be secured
Access session master key
Split data according to session key
Access Parser Master key
Split Session key according to Parser Master key
Parse
Encrypted data share 1
Session key share 1
Encrypted data share 2
Session key share 2
Encrypted data share 3
Session key share 3
Encrypted data share n
Session key share n
Generate share 1 key
Generate share 2 key
Generate share 3 key
Generate share n key
Encrypt Pieces
Encrypt share 1 data with share 1 key
Encrypt share 2 data with share 2 key
Encrypt share 3 data with share 3 key
Encrypt share n data with share n key
Encrypted share 1 (data / session key)
Key n
Encrypted share 3 (data / session key)
Key 2
Obfuscate
Encrypted share 2 (data / session key)
Key 1
Encrypted share n (data / session key)
Key 3

FIG. 21

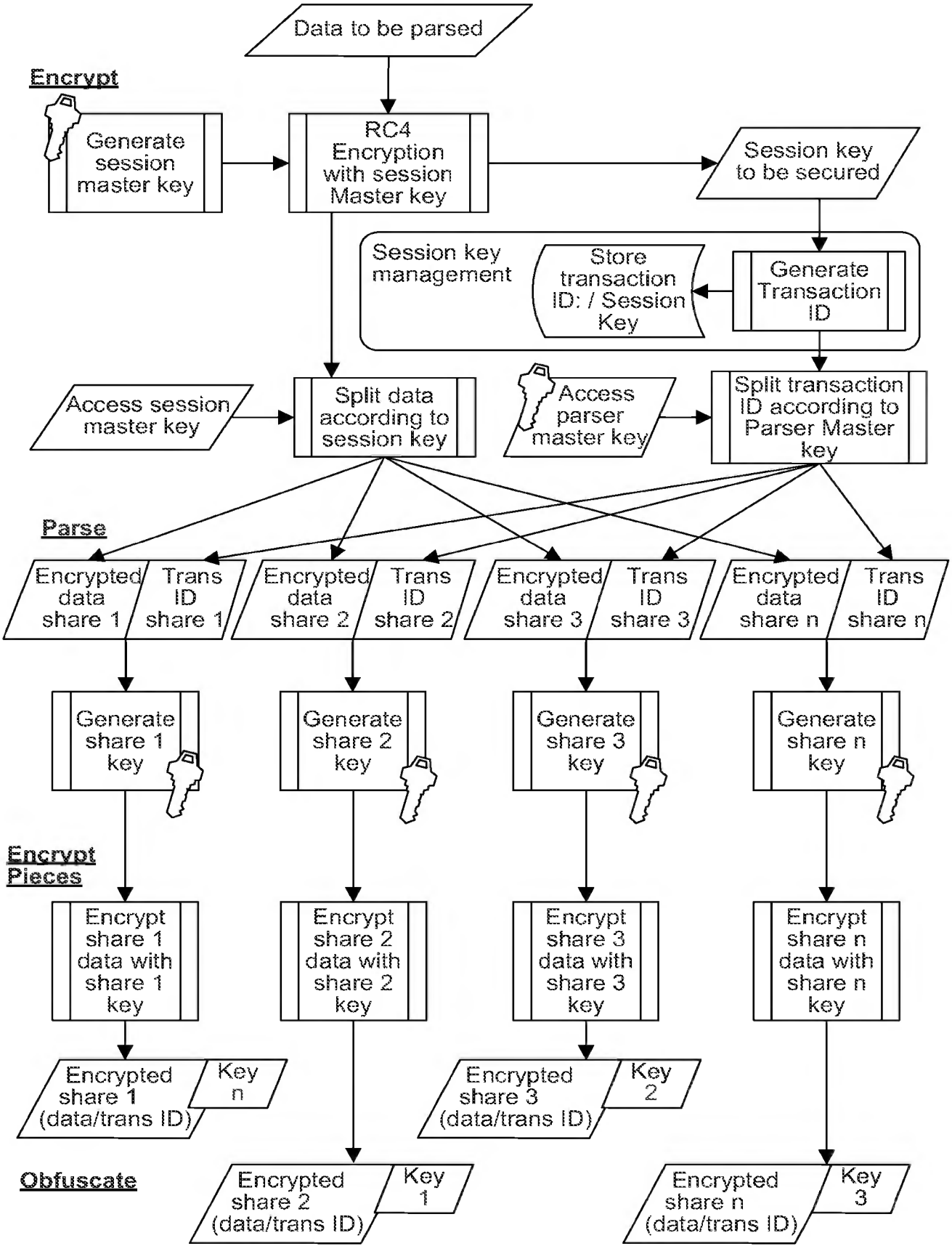

Data to be parsed
Encrypt
Generate session master key
RC4 Encryption with session Master key
Session key to be secured
Session key management
Store transaction ID: / Session Key
Generate Transaction ID
Access session master key
Split data according to session key
Access parser master key
Split transaction ID according to Parser Master key
Parse
Encrypted data share 1
Trans ID share 1
Encrypted data share 2
Trans ID share 2
Encrypted data share 3
Trans ID share 3
Encrypted data share n
Trans ID share n
Generate share 1 key
Generate share 2 key
Generate share 3 key
Generate share n key
Encrypt Pieces
Encrypt share 1 data with share 1 key
Encrypt share 2 data with share 2 key
Encrypt share 3 data with share 3 key
Encrypt share n data with share n key
Encrypted share 1 (data/trans ID)
Key n
Encrypted share 3 (data/trans ID)
Key 2
Obfuscate
Encrypted share 2 (data/trans ID)
Key 1
Encrypted share n (data/trans ID)
Key 3

FIG. 22

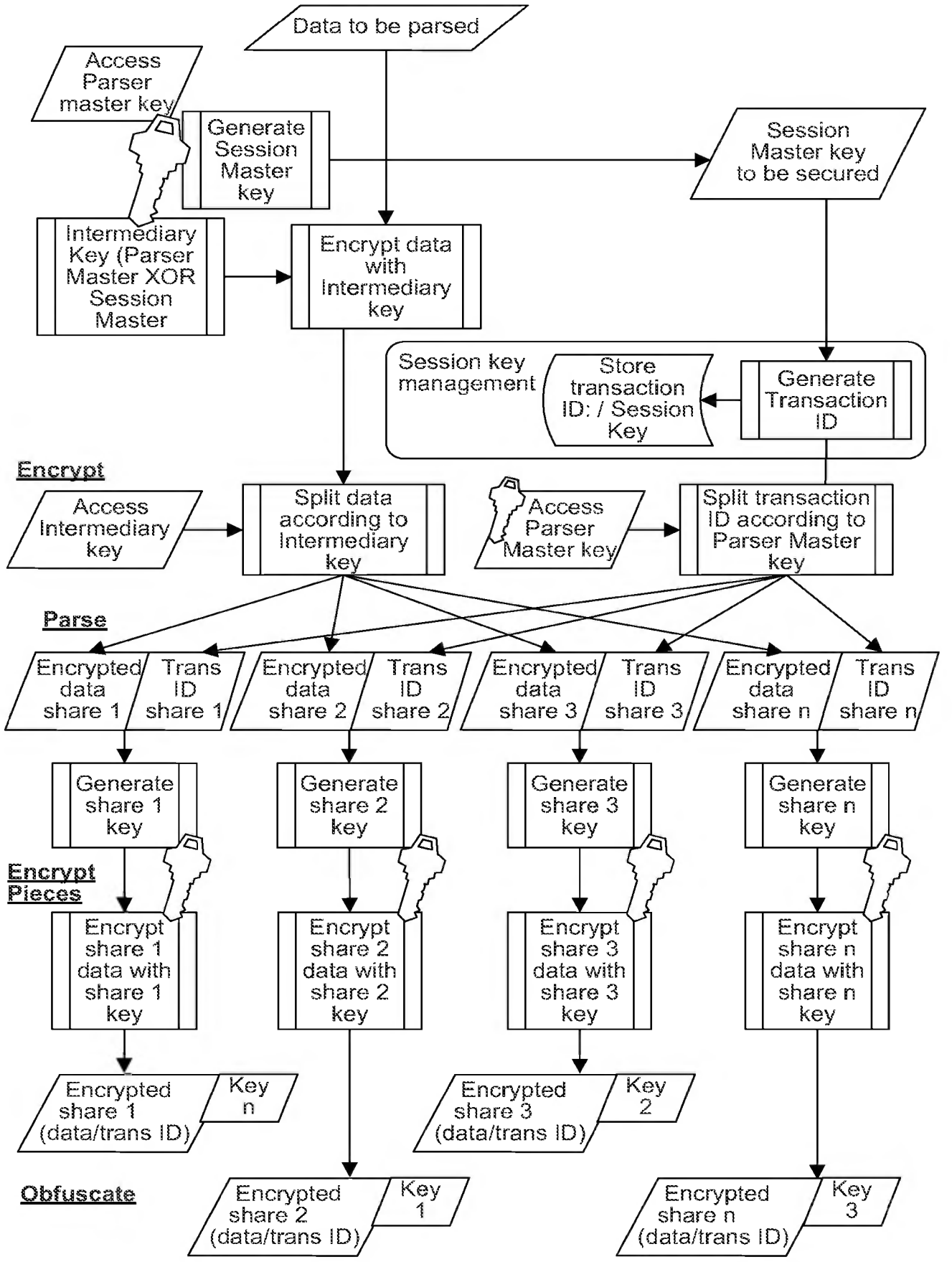

Data to be parsed
Access Parser master key
Generate Session Master key
Session Master key to be secured
Intermediary Key (Parser Master XOR Session Master
Encrypt data with Intermediary key
Session key management
Store transaction ID: / Session Key
Generate Transaction ID
Encrypt
Access Intermediary key
Split data according to Intermediary key
Access Parser Master key
Split transaction ID according to Parser Master key
Parse
Encrypted data share 1
Trans ID share 1
Encrypted data share 2
Trans ID share 2
Encrypted data share 3
Trans ID share 3
Encrypted data share n
Trans ID share n
Generate share 1 key
Generate share 2 key
Generate share 3 key
Generate share n key
Encrypt Pieces
Encrypt share 1 data with share 1 key
Encrypt share 2 data with share 2 key
Encrypt share 3 data with share 3 key
Encrypt share n data with share n key
Encrypted share 1 (data/trans ID)
Key n
Encrypted share 3 (data/trans ID)
Key 2
Obfuscate
Encrypted share 2 (data/trans ID)
Key 1
Encrypted share n (data/trans ID)
Key 3

FIG. 23

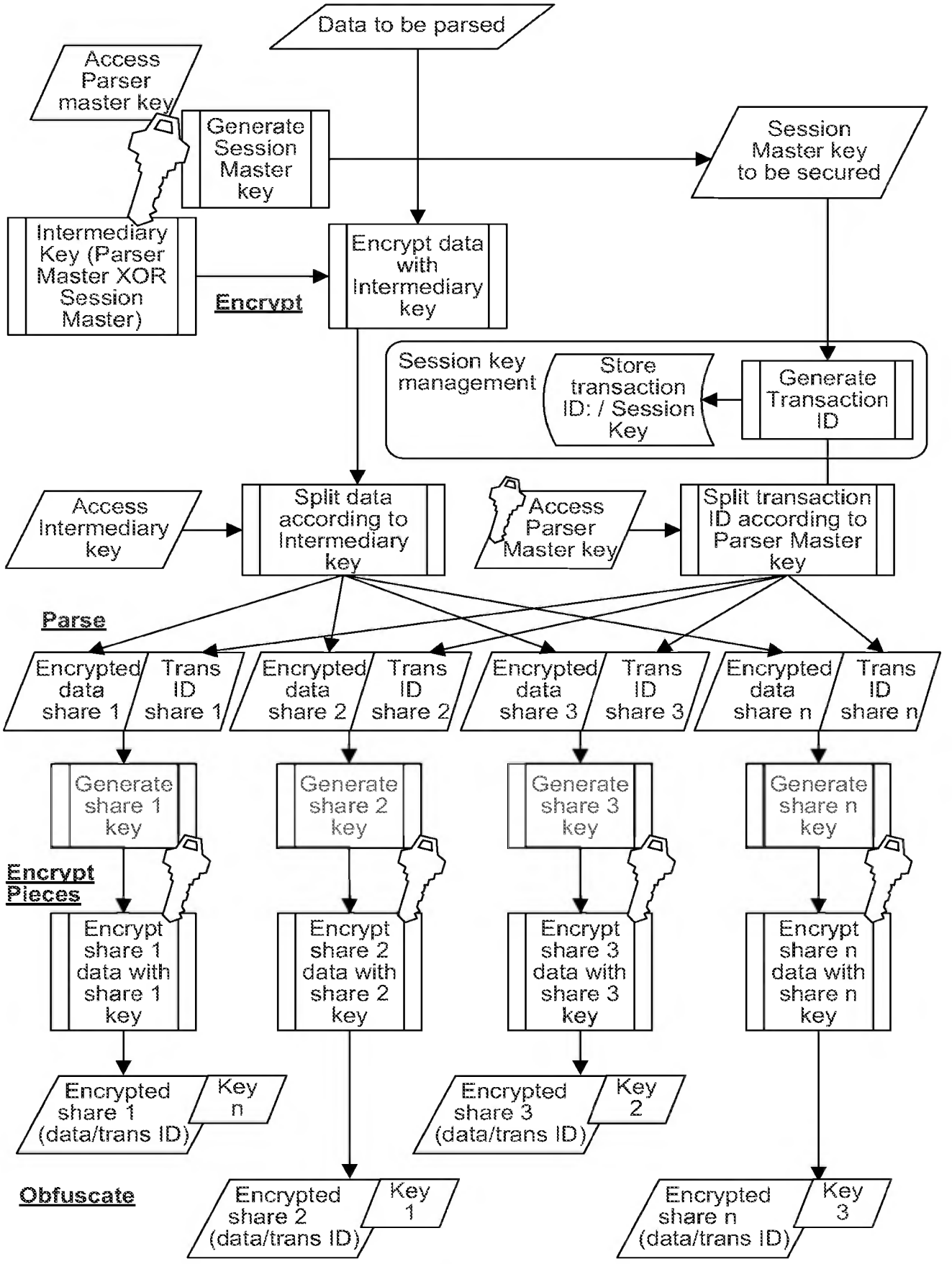

Data to be parsed
Access Parser master key
Generate Session Master key
Session Master key to be secured
Intermediary Key (Parser Master XOR Session Master)
Encrypt
Encrypt data with Intermediary key
Session key management
Store transaction ID: / Session Key
Generate Transaction ID
Access Intermediary key
Split data according to Intermediary key
Access Parser Master key
Split transaction ID according to Parser Master key
Parse
Encrypted data share 1
Trans ID share 1
Encrypted data share 2
Trans ID share 2
Encrypted data share 3
Trans ID share 3
Encrypted data share n
Trans ID share n
Generate share 1 key
Generate share 2 key
Generate share 3 key
Generate share n key
Encrypt Pieces
Encrypt share 1 data with share 1 key
Encrypt share 2 data with share 2 key
Encrypt share 3 data with share 3 key
Encrypt share n data with share n key
Encrypted share 1 (data/trans ID)
Key n
Encrypted share 3 (data/trans ID)
Key 2
Obfuscate
Encrypted share 2 (data/trans ID)
Key 1
Encrypted share n (data/trans ID)
Key 3

FIG. 24

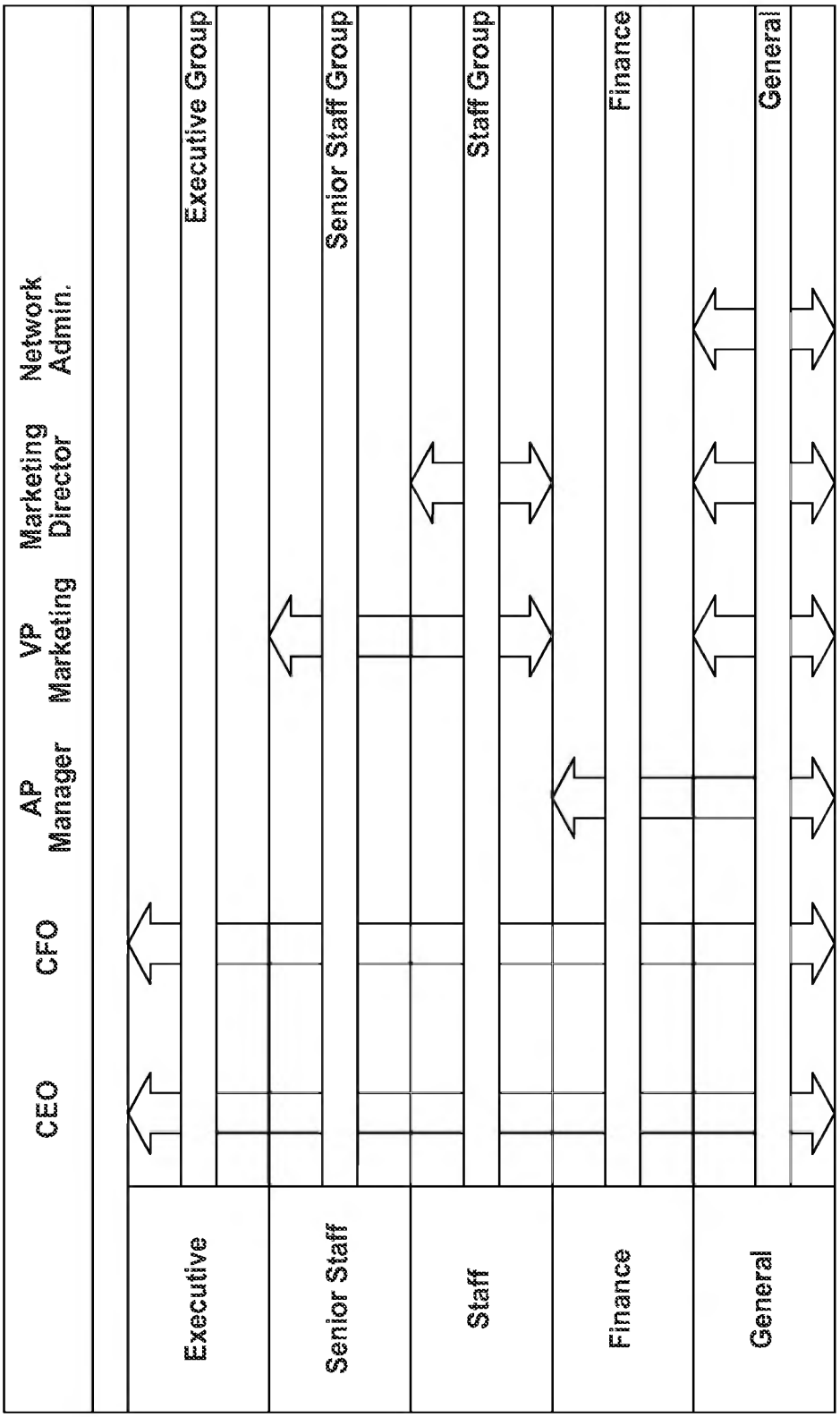

CEO
CFO
AP Manager
VP Marketing
Marketing Director
Network Admin.
Executive Group
Senior Staff Group
Staff Group
Finance
General
Executive
Senior Staff
Staff
Finance
General

FIG. 25

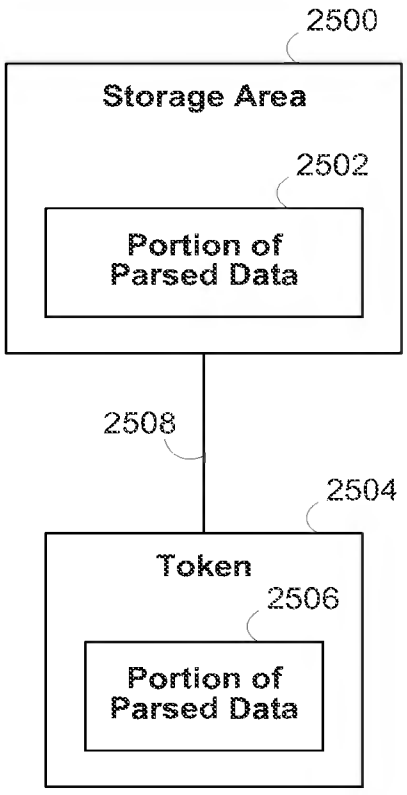

2500
Storage Area
2502
Portion of Parsed Data
2508
2504
Token
2506
Portion of Parsed Data

FIG. 26

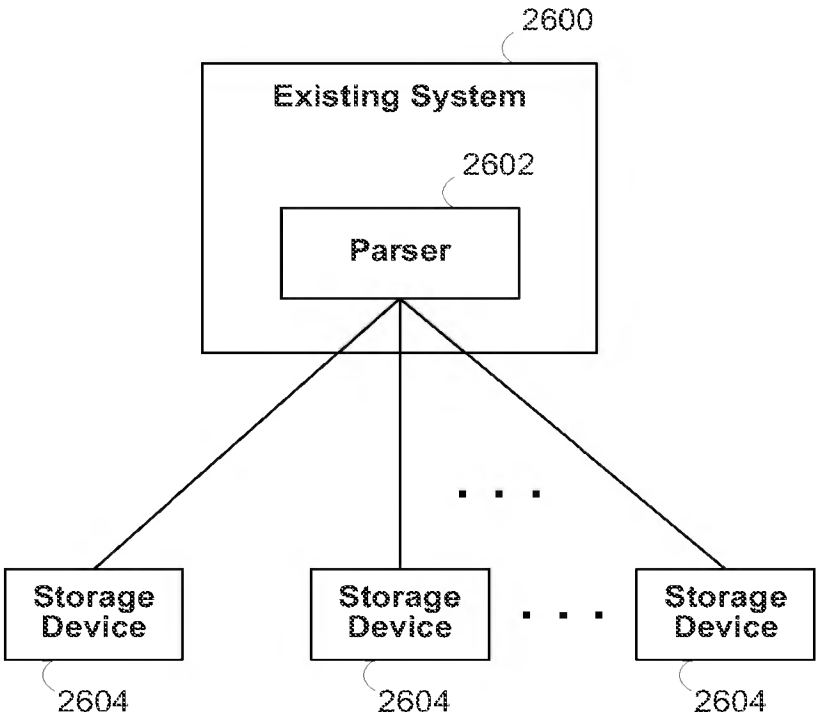

2600
Existing System
2602
Parser
Storage Device
Storage Device
Storage Device
2604
2604
2604

FIG. 27

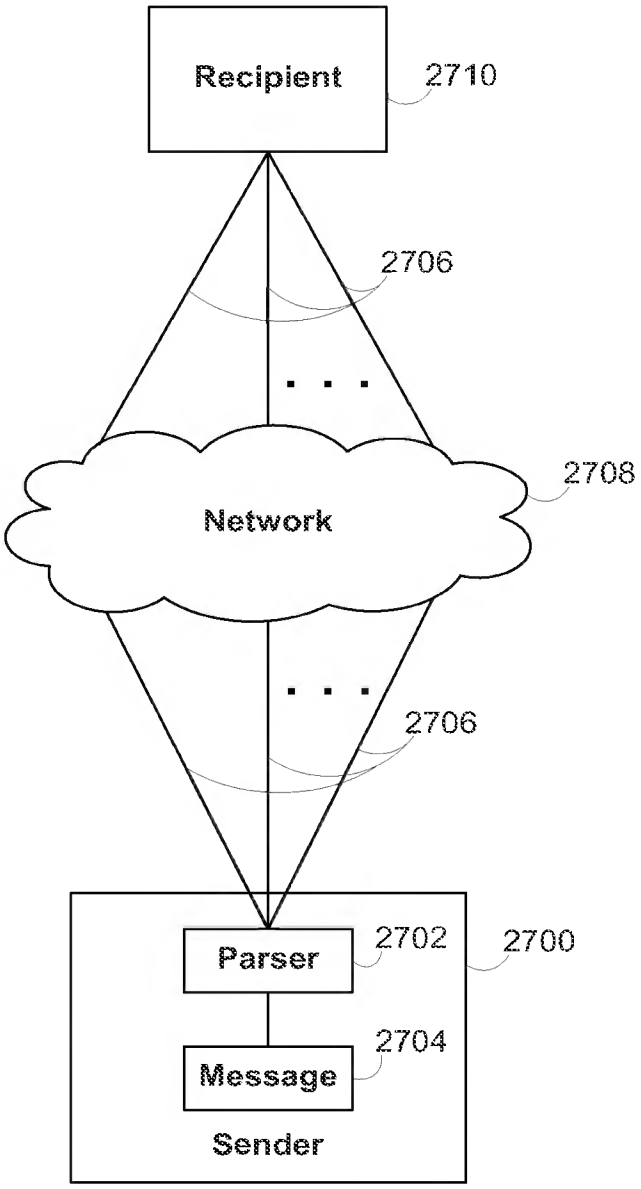

Recipient
2710
2706
2708
Network
2706
Parser
2702
2700
2704
Message
Sender

FIG. 28

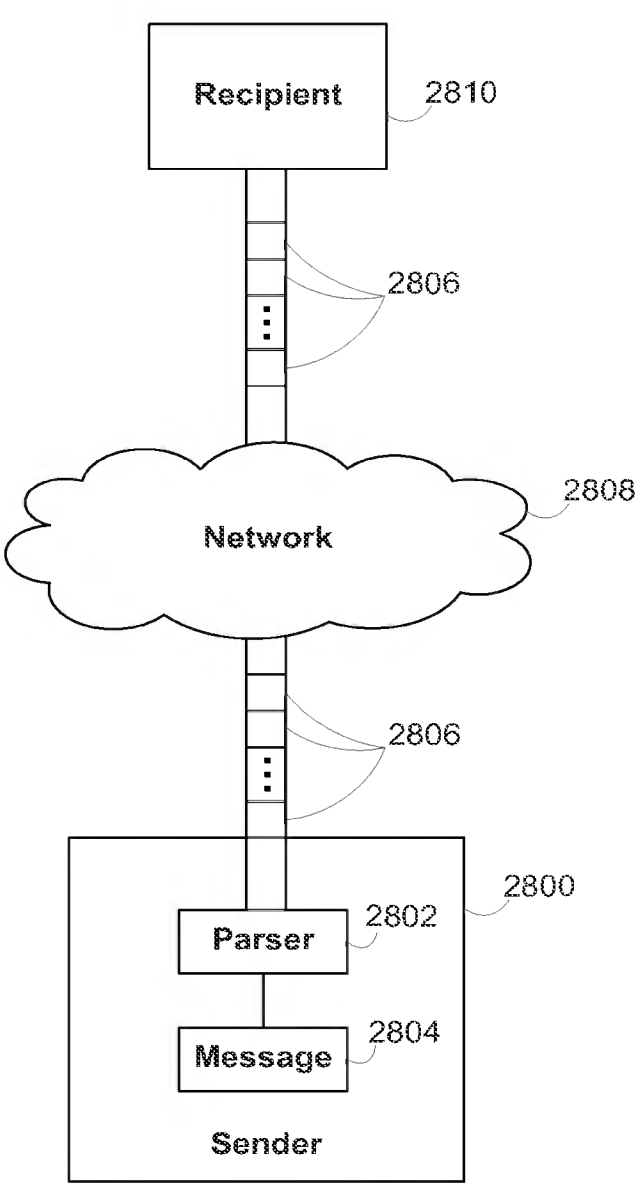

Recipient
2810
2806
Network
2808
2806
2800
Parser
2802
Message
2804
Sender

FIG. 29

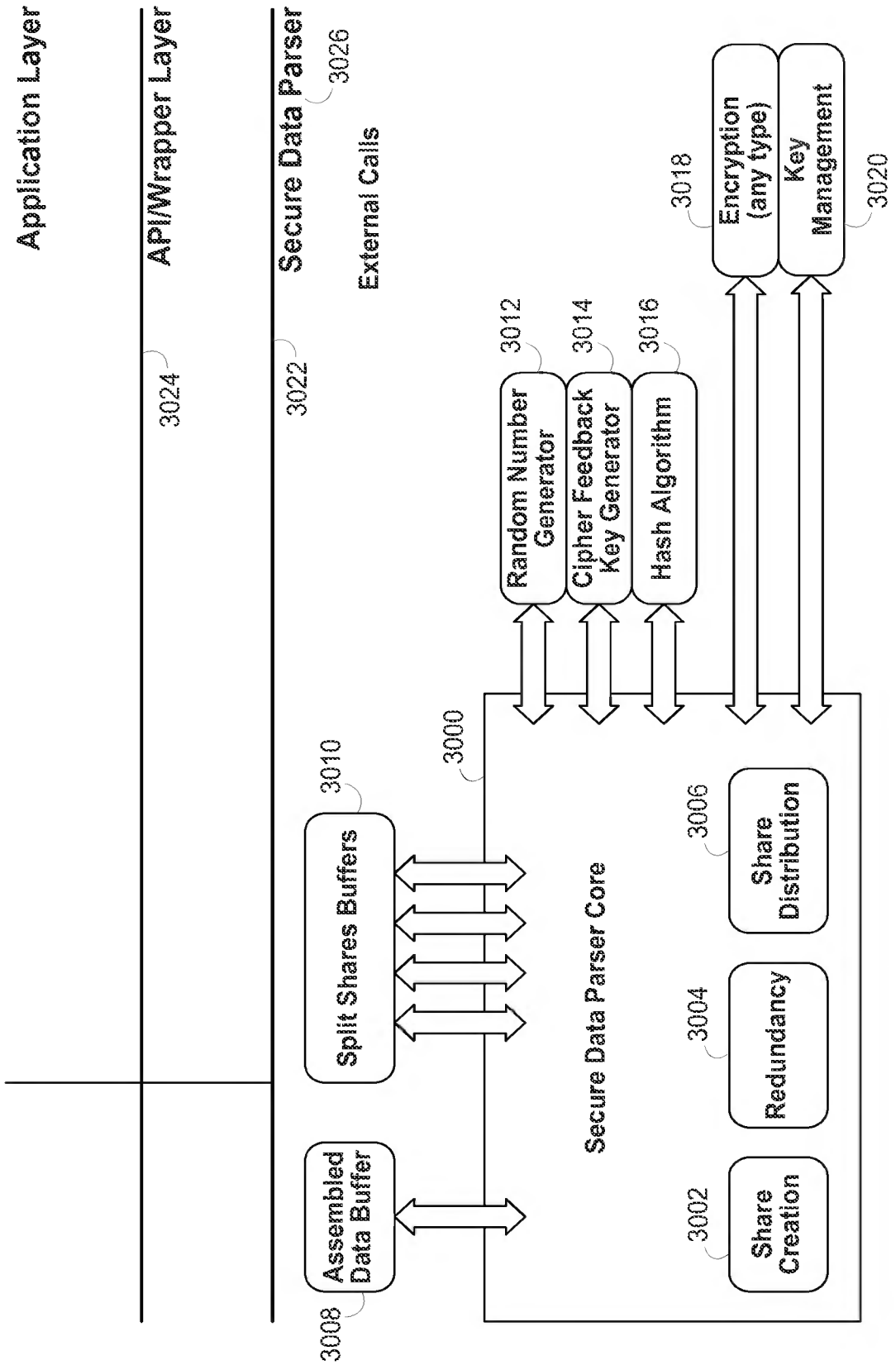

Application Layer
API/Wrapper Layer
Secure Data Parser
3026
External Calls
3024
3022
3012
3014
3016
3018
3020
Random Number Generator
Cipher Feedback Key Generator
Hash Algorithm
Encryption (any type)
Key Management
3000
3010
Split Shares Buffers
Secure Data Parser Core
3006
Share Distribution
3004
Redundancy
3002
Share Creation
3008
Assembled Data Buffer

FIG. 30

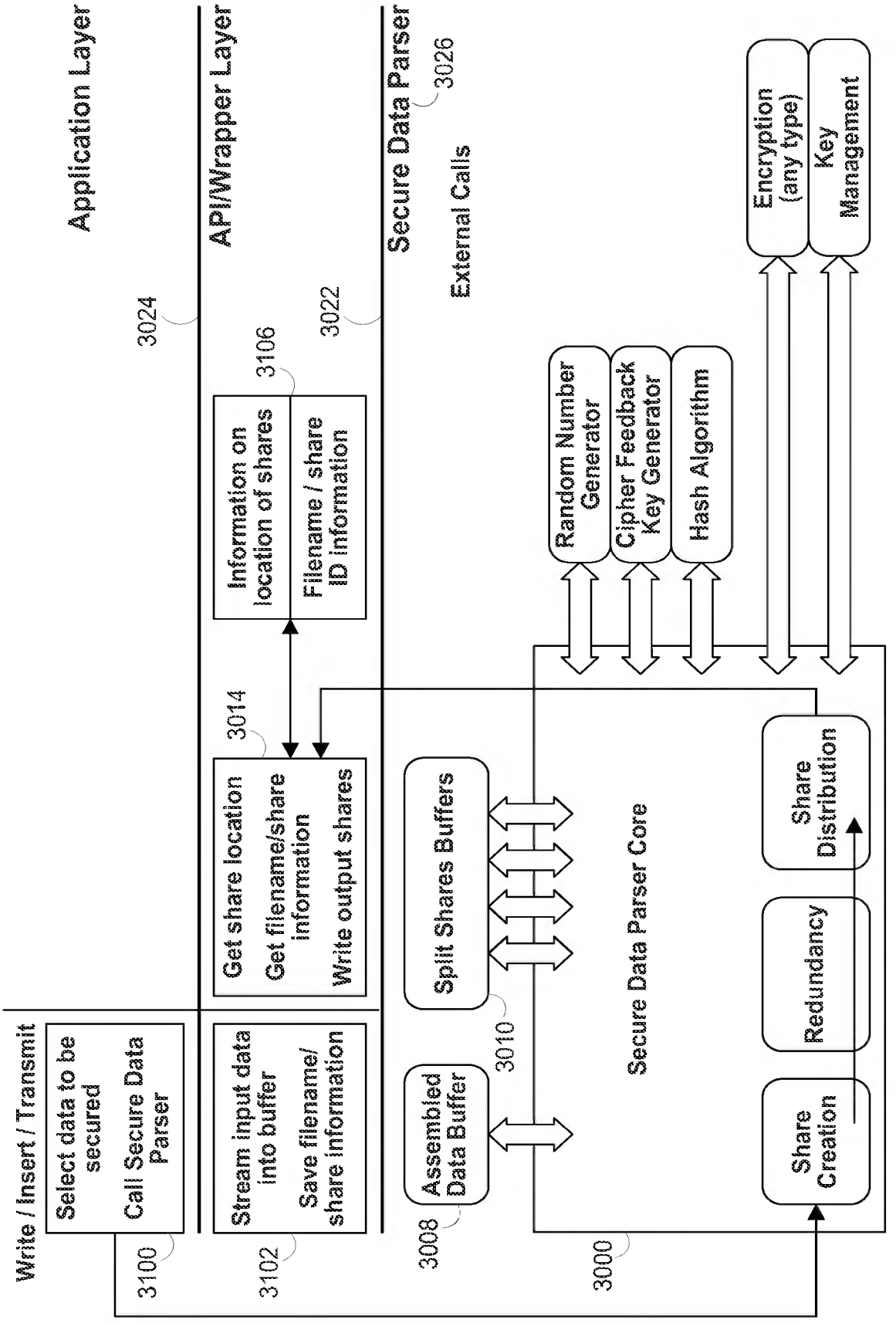

Application Layer
API/Wrapper Layer
Secure Data Parser
3026
External Calls
3024
3022
3106
Write / Insert / Transmit
Select data to be secured
Call Secure Data Parser
3100
Stream input data into buffer
Save filename/ share information
3102
Get share location
Get filename/share information
Write output shares
3014
Information on location of shares
Filename / share ID information
Split Shares Buffers
3010
Assembled Data Buffer
3008
Secure Data Parser Core
3000
Random Number Generator
Cipher Feedback Key Generator
Hash Algorithm
Encryption (any type)
Key Management
Share Distribution
Redundancy
Share Creation

FIG. 31

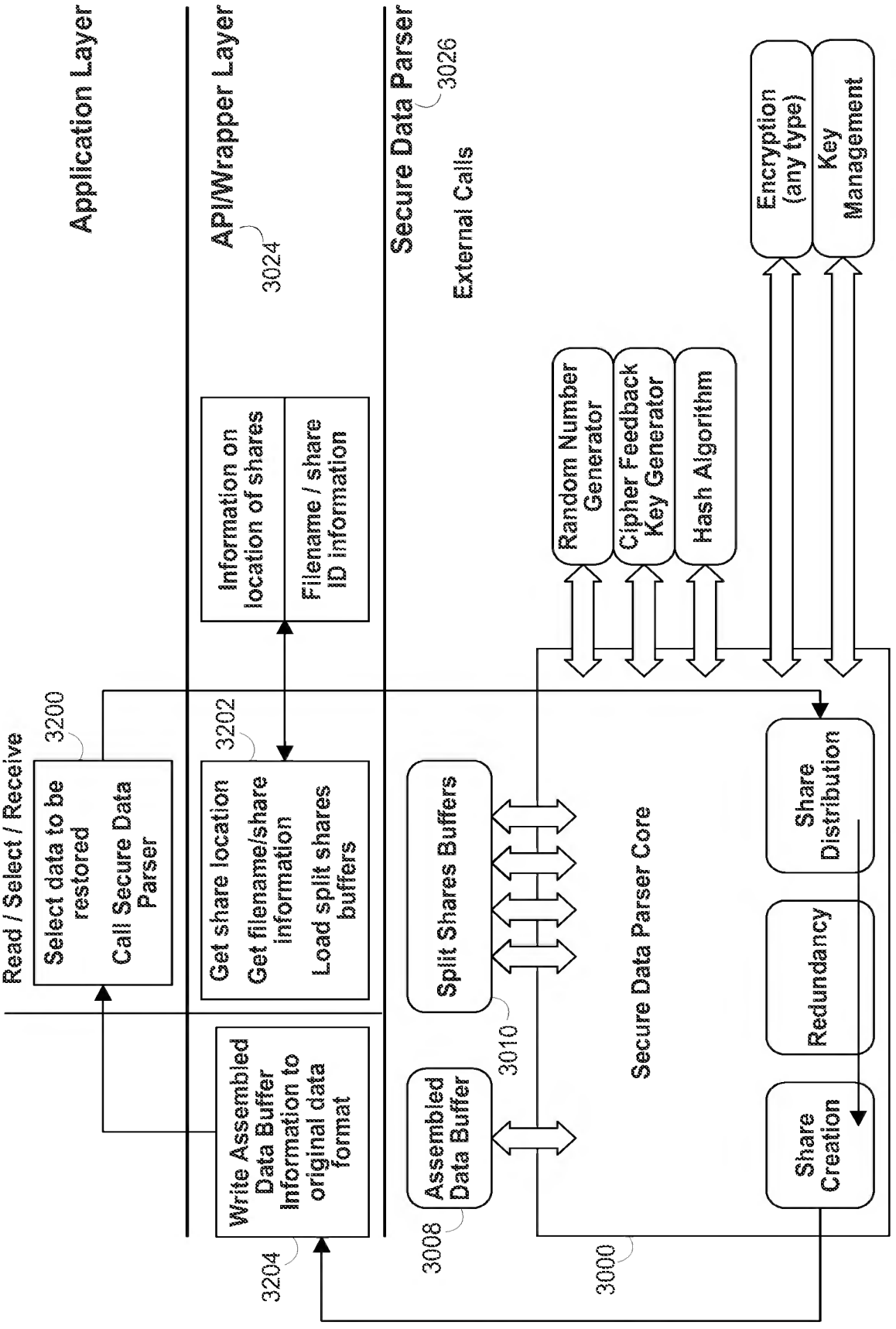

Application Layer
API/Wrapper Layer
3024
Secure Data Parser
3026
External Calls
Encryption (any type)
Key Management
Random Number Generator
Cipher Feedback Key Generator
Hash Algorithm
Read / Select / Receive
3200
Select data to be restored
Call Secure Data Parser
3202
Get share location
Get filename/share information
Load split shares buffers
Information on location of shares
Filename / share ID information
Split Shares Buffers
3010
Assembled Data Buffer
3008
Write Assembled Data Buffer Information to original data format
3204
Secure Data Parser Core
3000
Share Distribution
Redundancy
Share Creation

FIG. 32

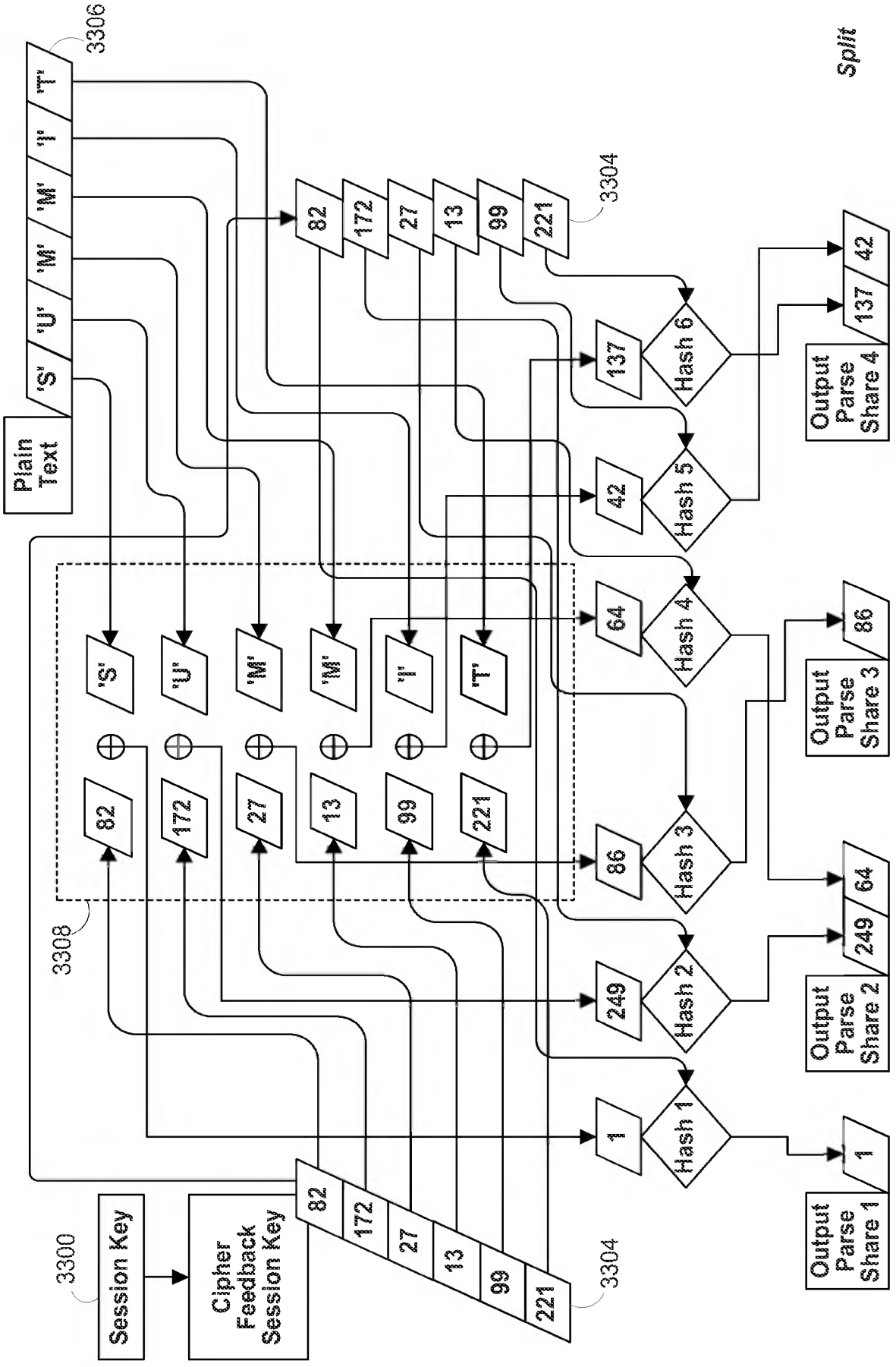

3306
Plain Text
'S' 'U' 'M' 'M' 'I' 'T'
3308
'S' 'U' 'M' 'M' 'I' 'T'
82 172 27 13 99 221
3304
Split
Hash 1
Hash 2
Hash 3
Hash 4
Hash 5
Hash 6
1
249
86
64
42
137
Output Parse Share 1
Output Parse Share 2
Output Parse Share 3
Output Parse Share 4
3300
Session Key
Cipher Feedback Session Key

FIG. 33

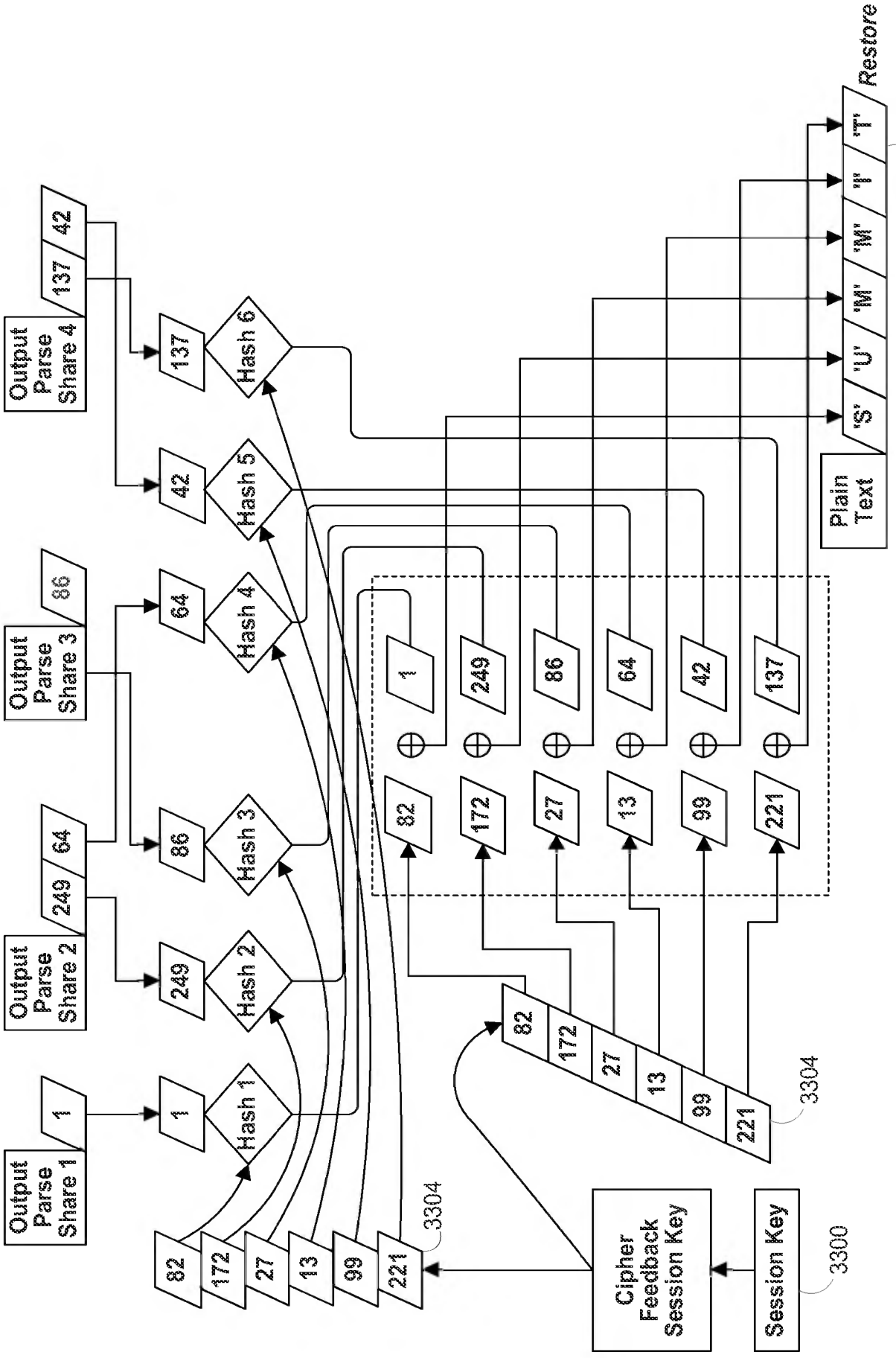

Output Parse Share 1
Output Parse Share 2
Output Parse Share 3
Output Parse Share 4
Hash 1
Hash 2
Hash 3
Hash 4
Hash 5
Hash 6
Cipher Feedback Session Key
Session Key
Plain Text
Restore
3300
3304
3306

FIG. 34

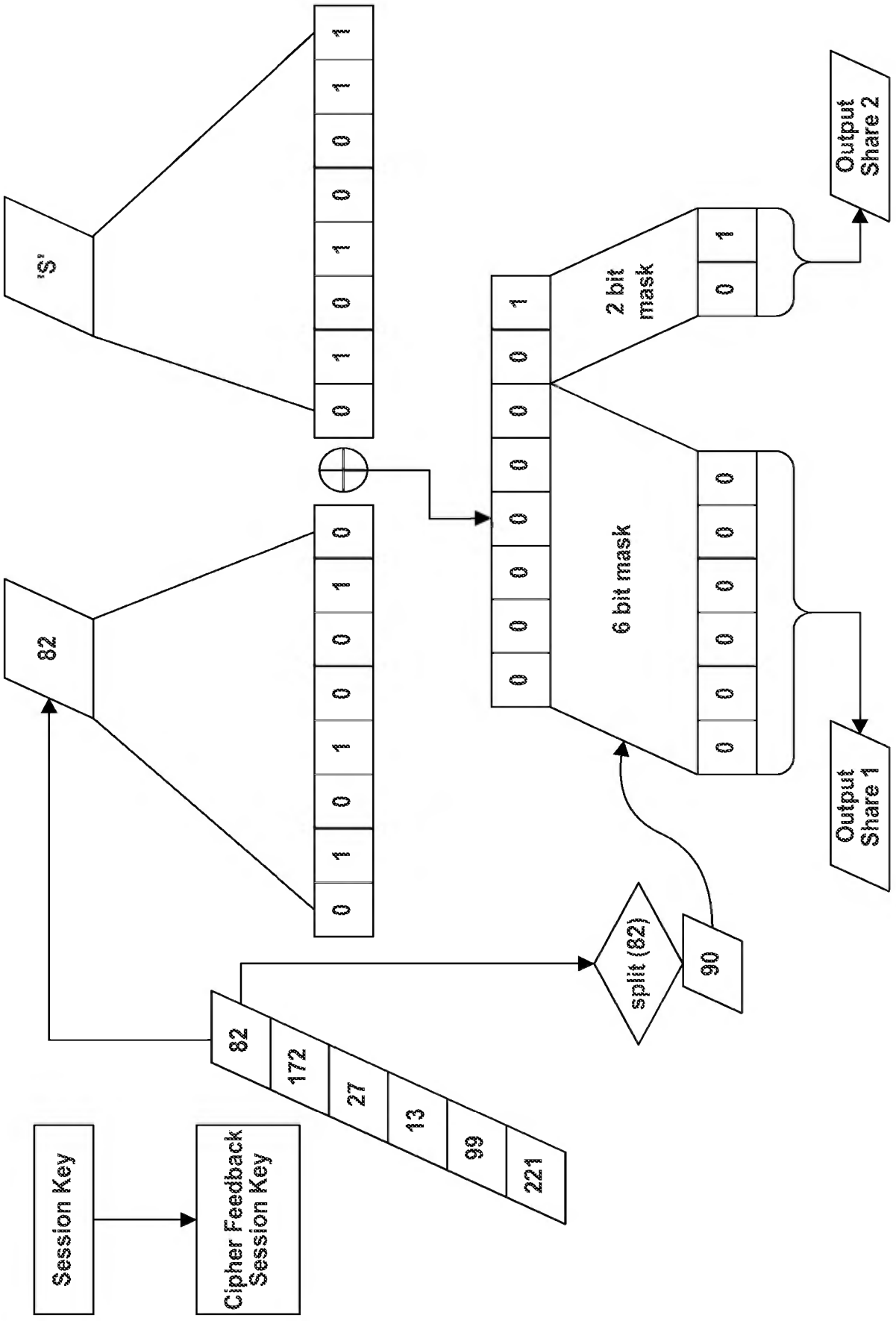

Session Key
Cipher Feedback Session Key
82
172
27
13
99
221
split (82)
90
82
'S'
0 1 0 1 0 0 1 0
0 1 1 0 1 0 0 1
0 0 0 0 0 0 0 1
6 bit mask
2 bit mask
0 0 0 0 0 0
0 1
Output Share 1
Output Share 2

FIG. 35

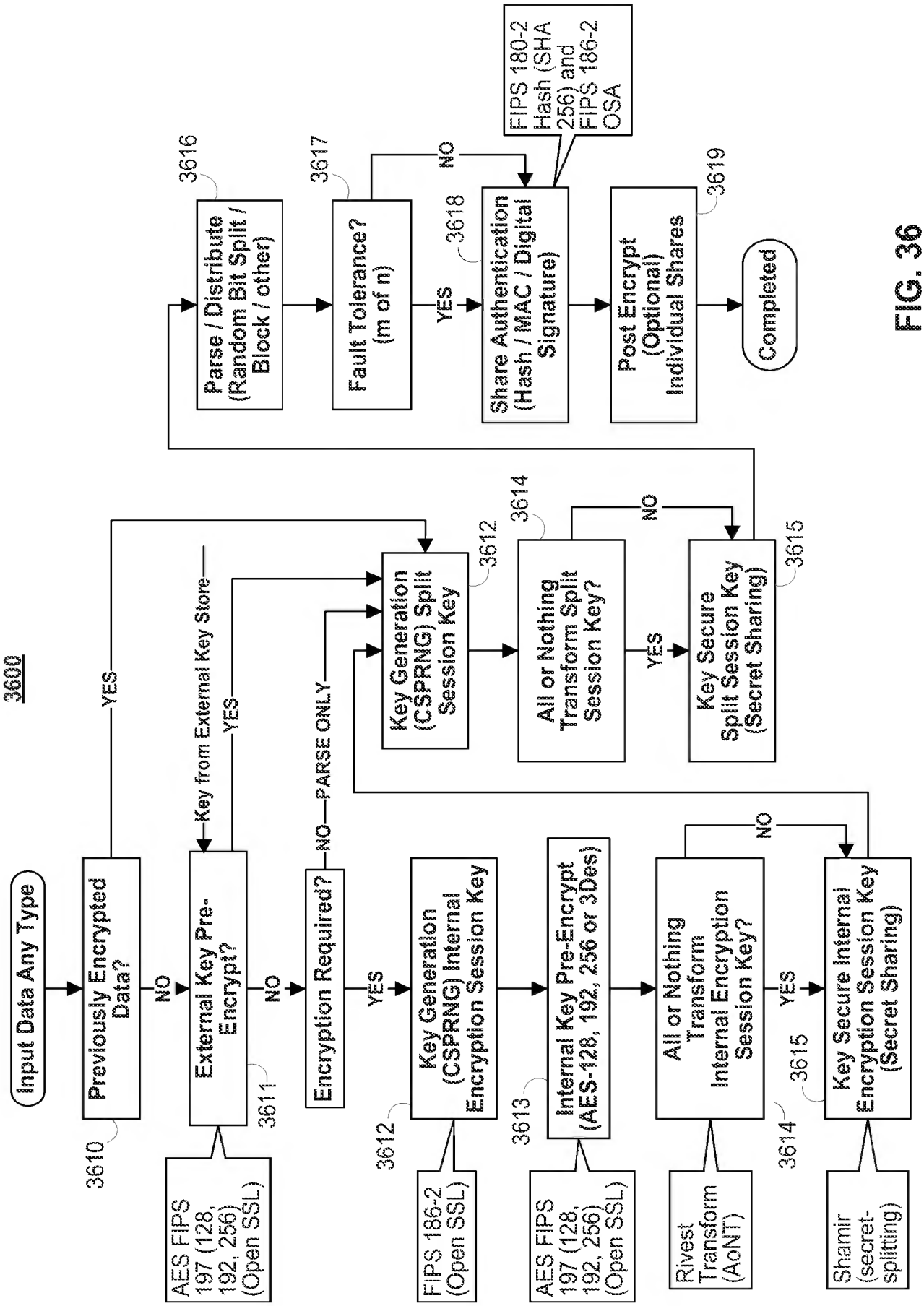

3600
Input Data Any Type
Previously Encrypted Data?
YES
NO
3610
External Key Pre-Encrypt?
Key from External Key Store
YES
NO
3611
AES FIPS 197 (128, 192, 256) (Open SSL)
Encryption Required?
PARSE ONLY
NO
YES
Key Generation (CSPRNG) Internal Encryption Session Key
3612
FIPS 186-2 (Open SSL)
Internal Key Pre-Encrypt (AES-128, 192, 256 or 3Des)
3613
AES FIPS 197 (128, 192, 256) (Open SSL)
All or Nothing Transform Internal Encryption Session Key?
NO
YES
3614
Rivest Transform (AoNT)
Key Secure Internal Encryption Session Key (Secret Sharing)
3615
Shamir (secret-splitting)
Key Generation (CSPRNG) Split Session Key
3612
All or Nothing Transform Split Session Key?
3614
NO
YES
Key Secure Split Session Key (Secret Sharing)
3615
Parse / Distribute (Random Bit Split / Block / other)
3616
Fault Tolerance? (m of n)
3617
NO
YES
3618
FIPS 180-2 Hash (SHA 256) and FIPS 186-2 OSA
Share Authentication (Hash / MAC / Digital Signature)
Post Encrypt (Optional) Individual Shares
3619
Completed

FIG. 36

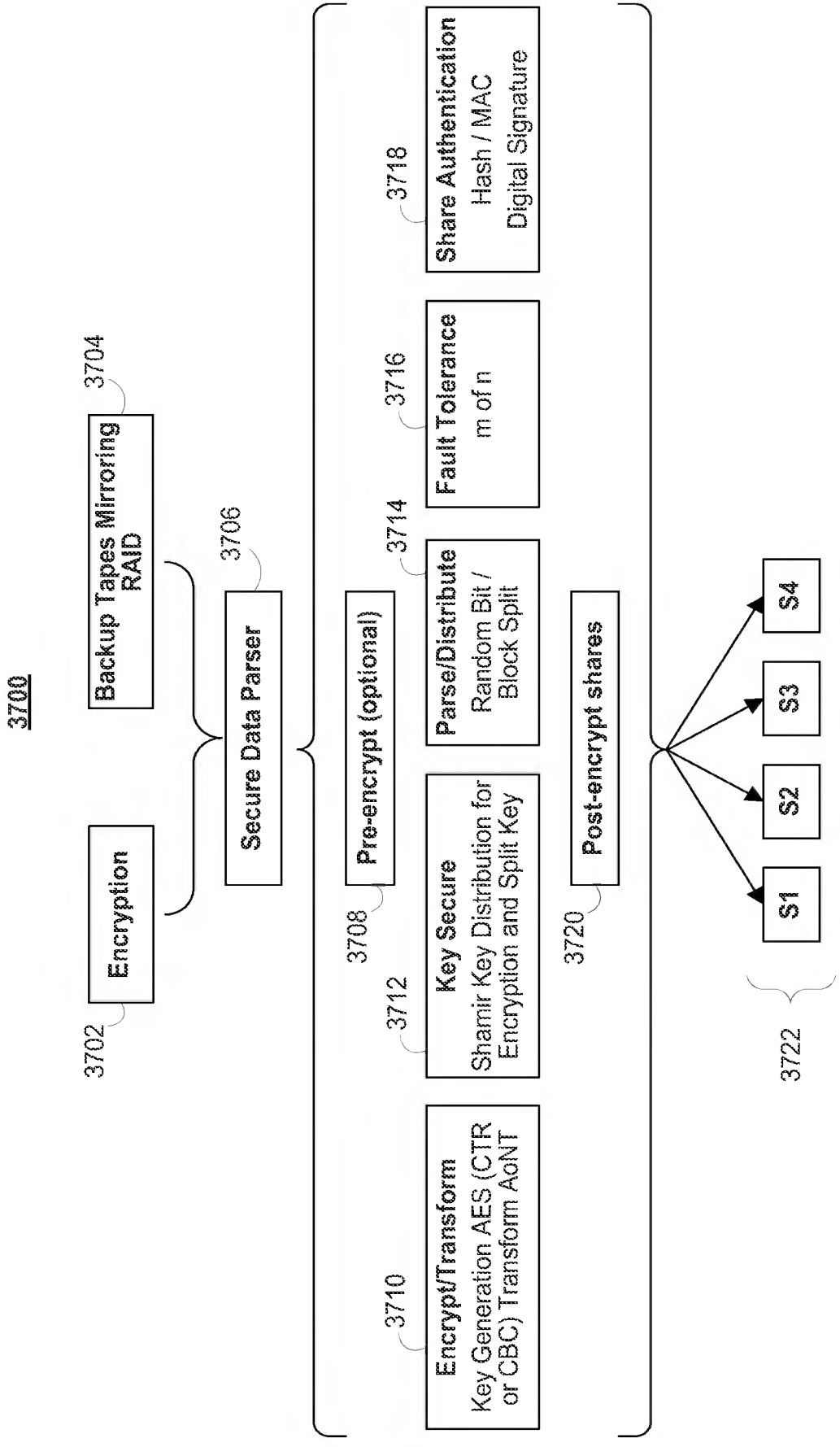

3700
Encryption
3702
Backup Tapes Mirroring RAID
3704
Secure Data Parser
3706
Pre-encrypt (optional)
3708
Encrypt/Transform
Key Generation AES (CTR or CBC) Transform AoNT
3710
Key Secure
Shamir Key Distribution for Encryption and Split Key
3712
Parse/Distribute
Random Bit / Block Split
3714
Fault Tolerance
m of n
3716
Share Authentication
Hash / MAC
Digital Signature
3718
Post-encrypt shares
3720
S1
S2
S3
S4
3722

FIG. 37

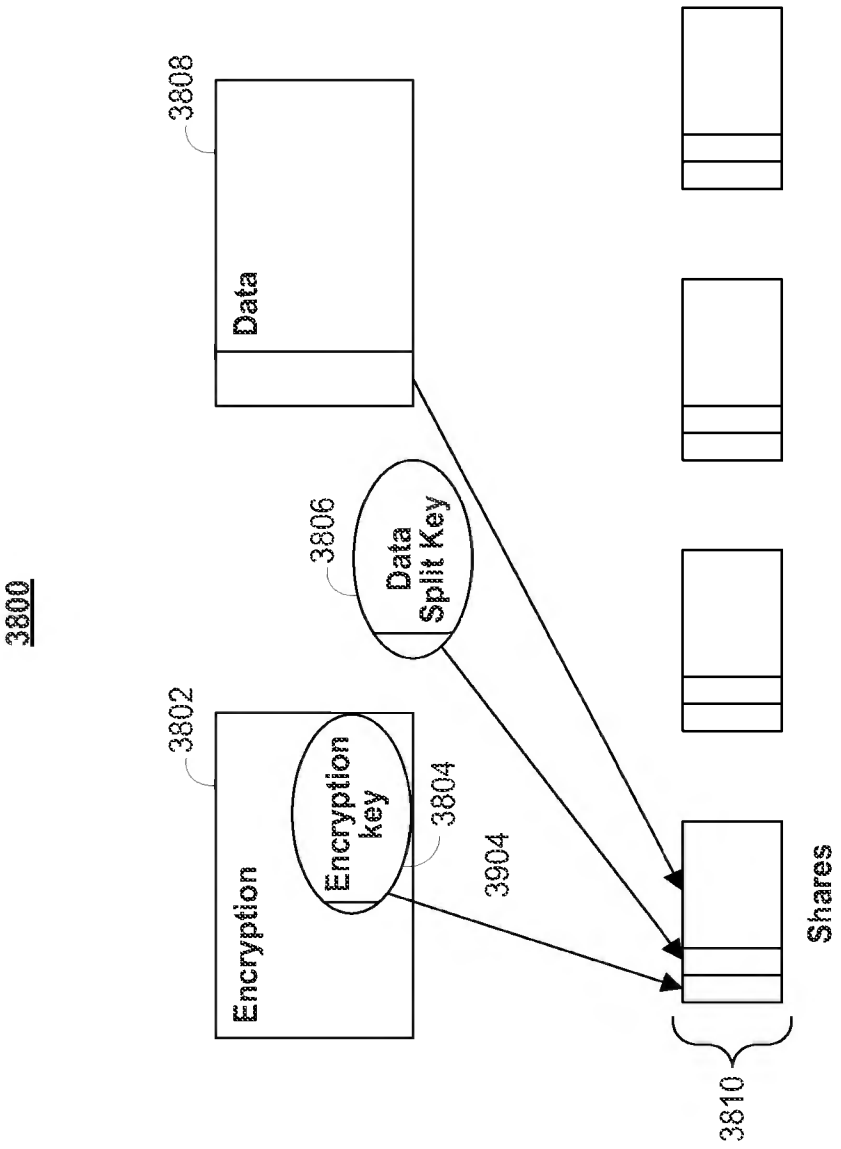

3800
3808
Data
3806
Data Split Key
3802
Encryption
Encryption key
3804
3904
Shares
3810

FIG. 38

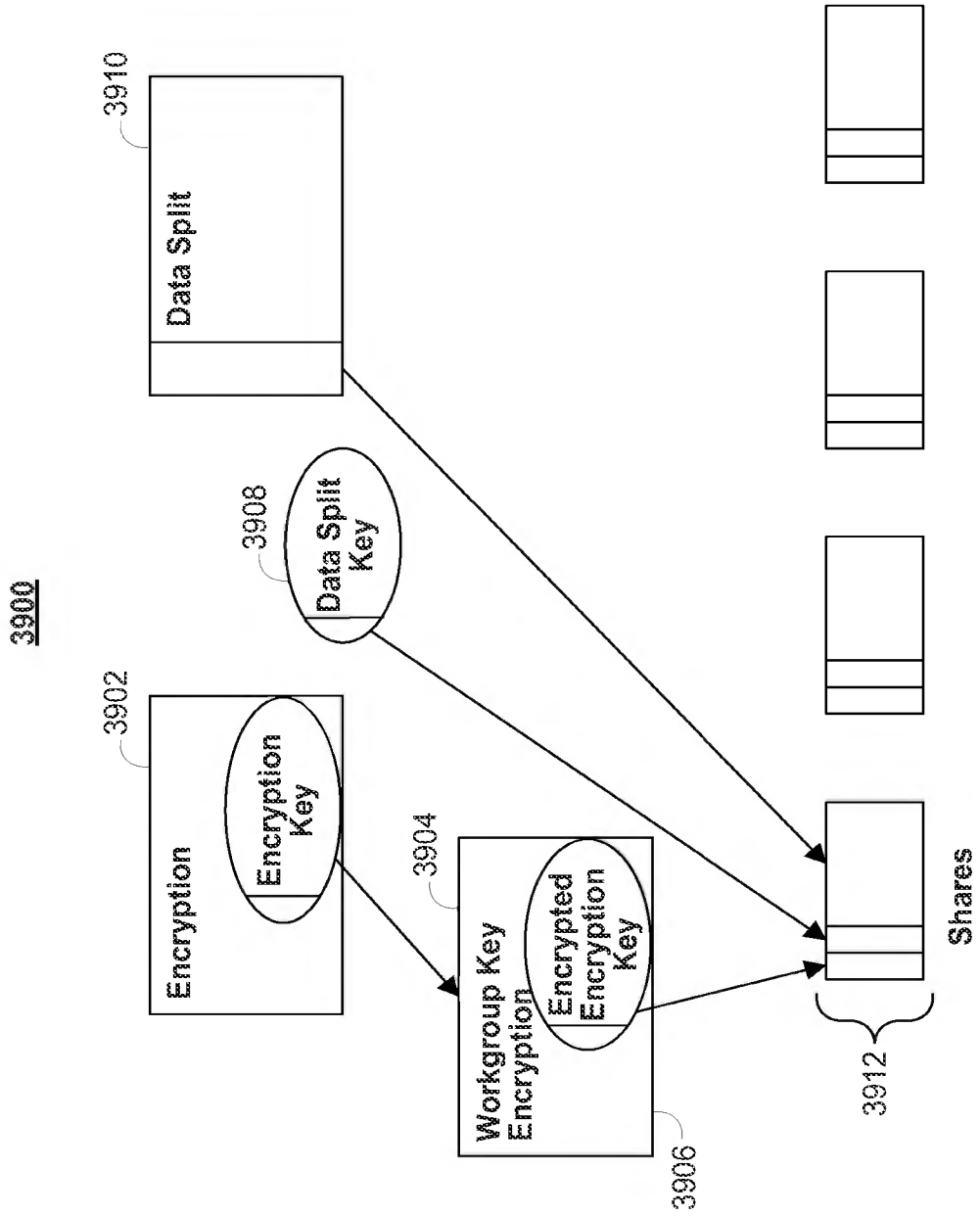

3900
3902
Encryption
Encryption Key
3904
Workgroup Key Encryption
Encrypted Encryption Key
3906
3908
Data Split Key
3910
Data Split
3912
Shares

FIG. 39

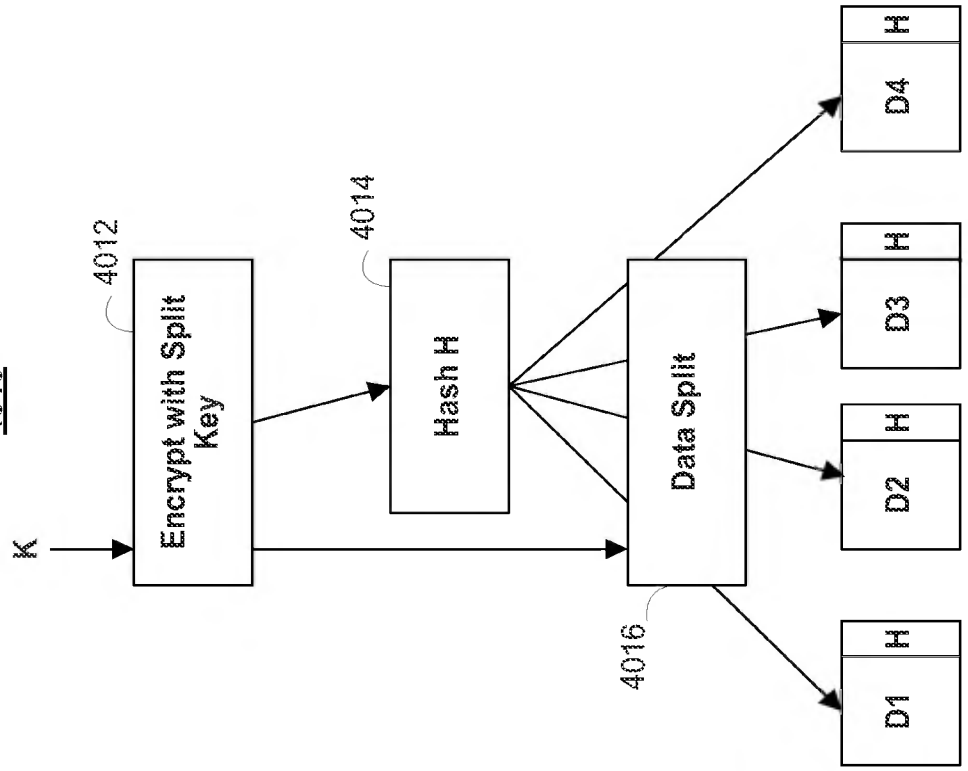


FIG. 40B

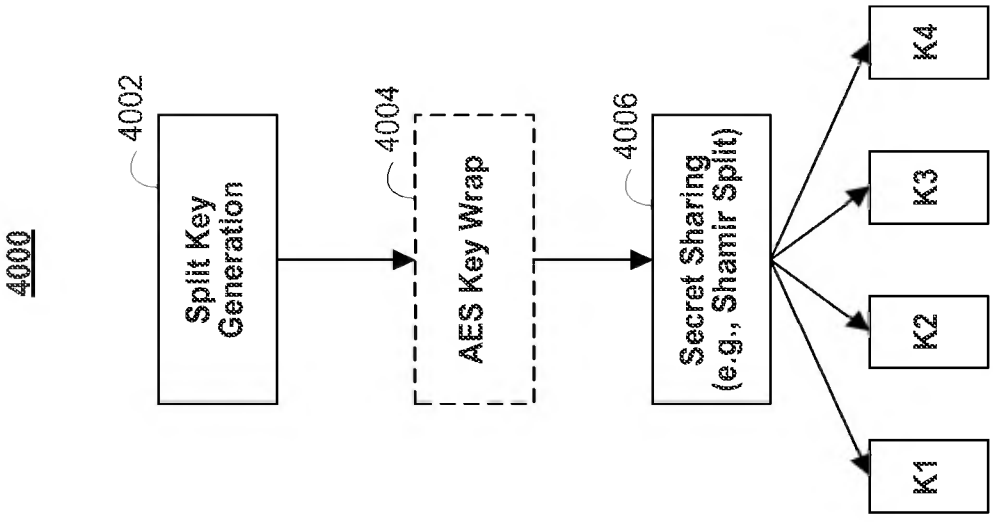


FIG. 40A

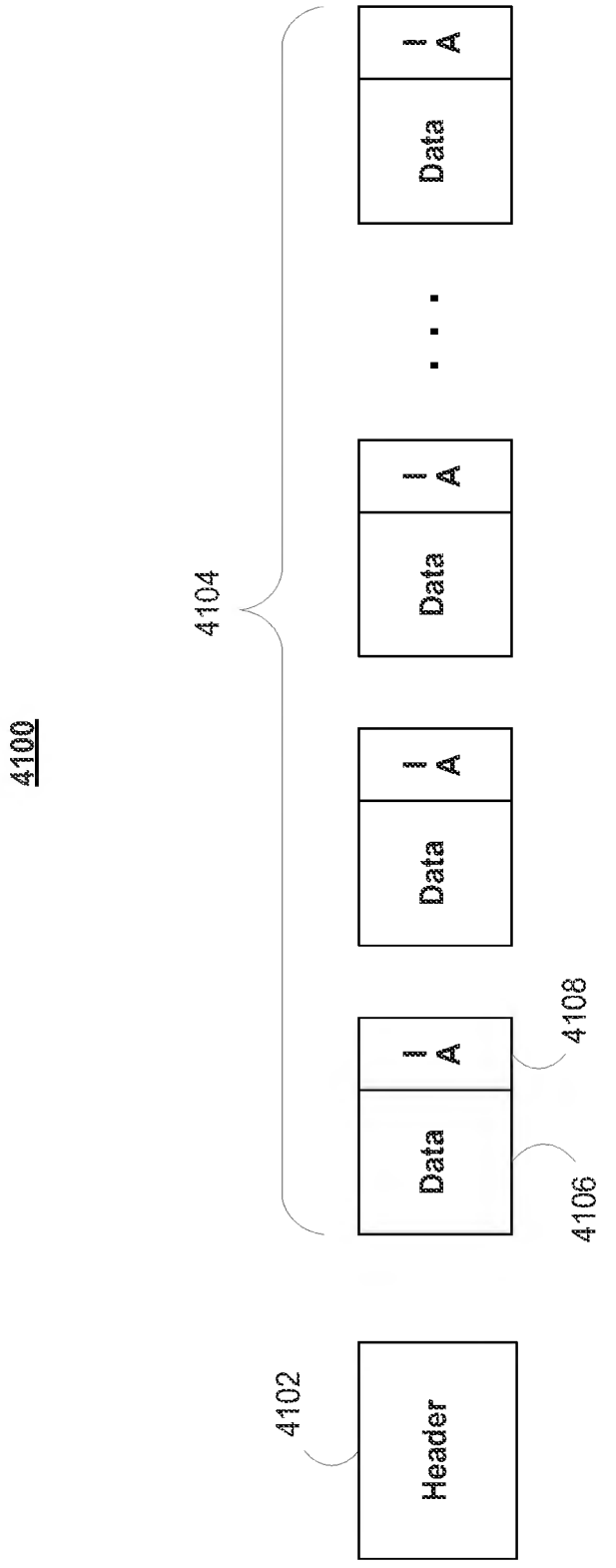


FIG. 41

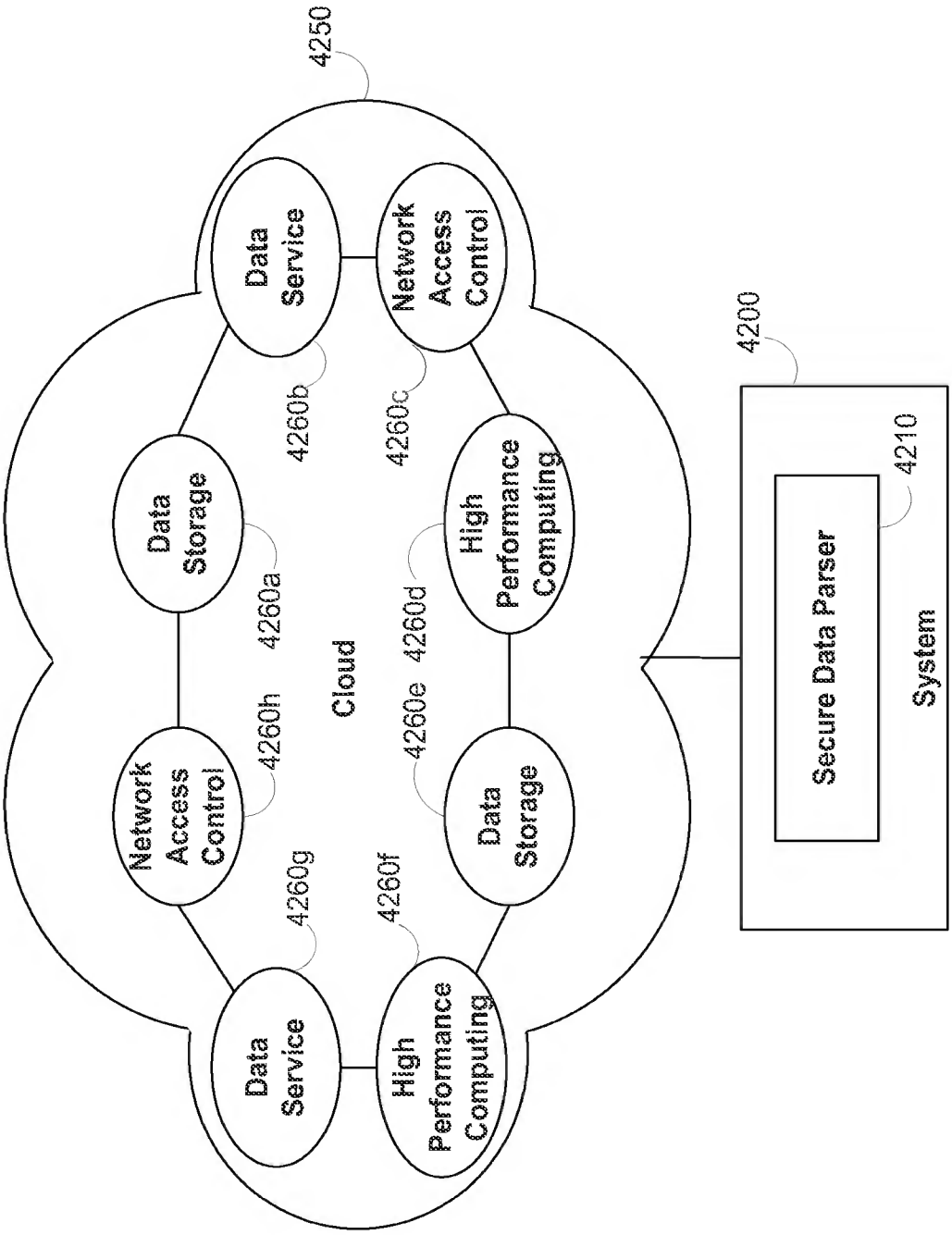

4250
Data Service
Network Access Control
4260b
4260c
Data Storage
High Performance Computing
4260a
4260d
Cloud
4260h
4260e
Network Access Control
Data Storage
4260g
4260f
Data Service
High Performance Computing
4200
Secure Data Parser
4210
System

FIG. 42

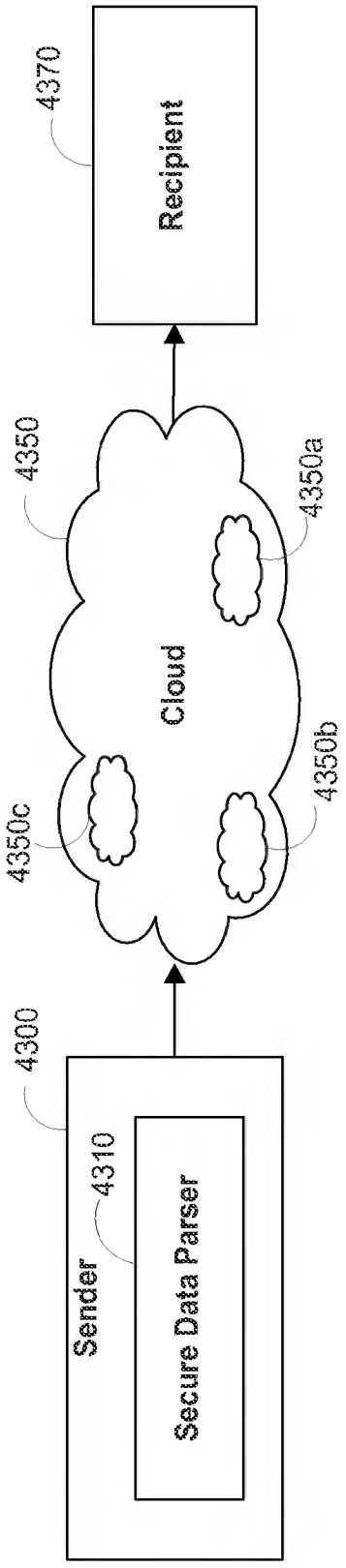

4370
Recipient
4350
4350a
Cloud
4350b
4350c
4300
4310
Sender
Secure Data Parser

FIG. 43

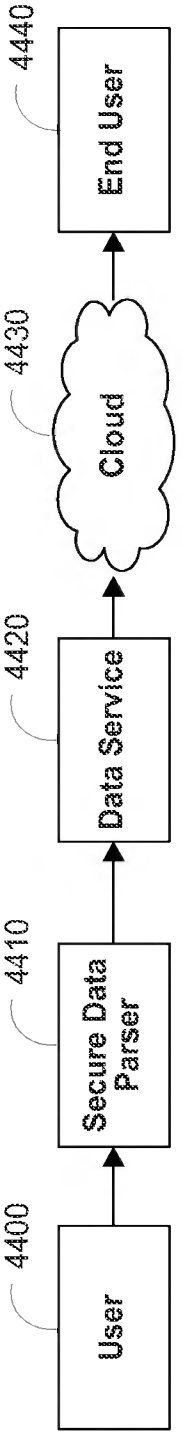

4400
User
4410
Secure Data Parser
4420
Data Service
4430
Cloud
4440
End User

FIG. 44

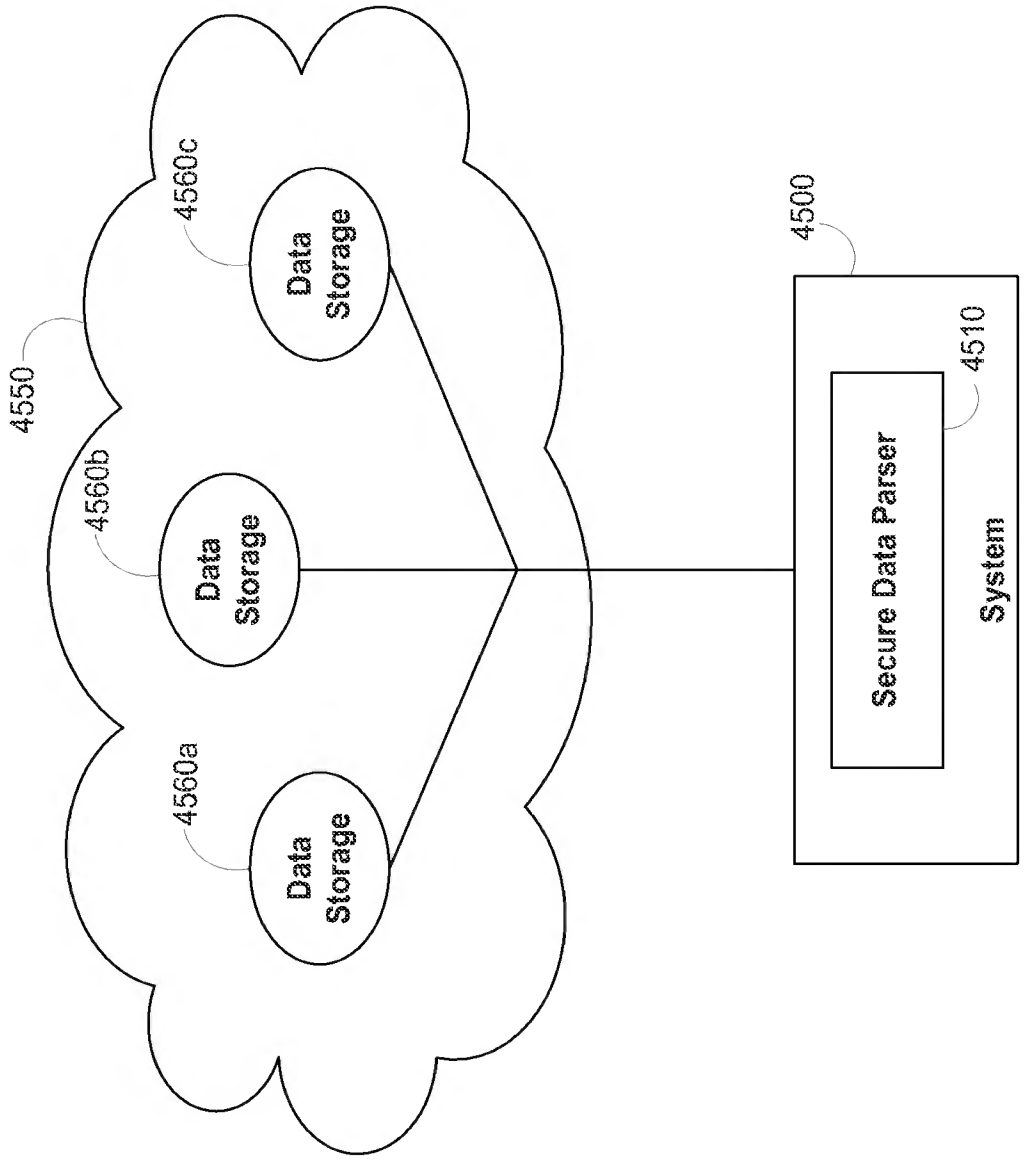

4550
4560c
Data Storage
4560b
Data Storage
4560a
Data Storage
4500
4510
Secure Data Parser
System

FIG. 45

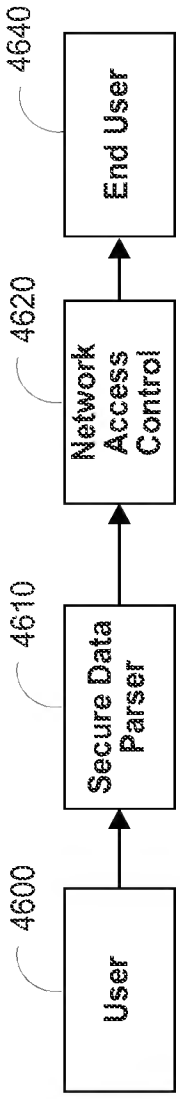

4600
User
4610
Secure Data Parser
4620
Network Access Control
4640
End User

FIG. 46

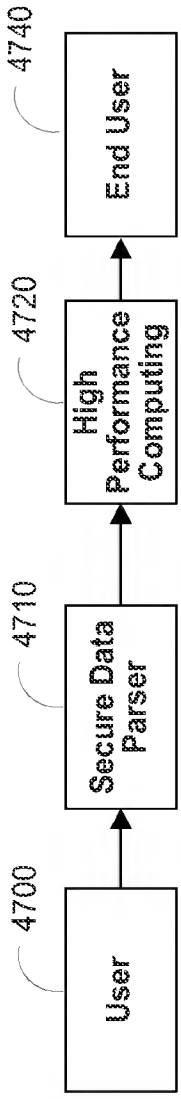

4700
User
4710
Secure Data Parser
4720
High Performance Computing
4740
End User

FIG. 47

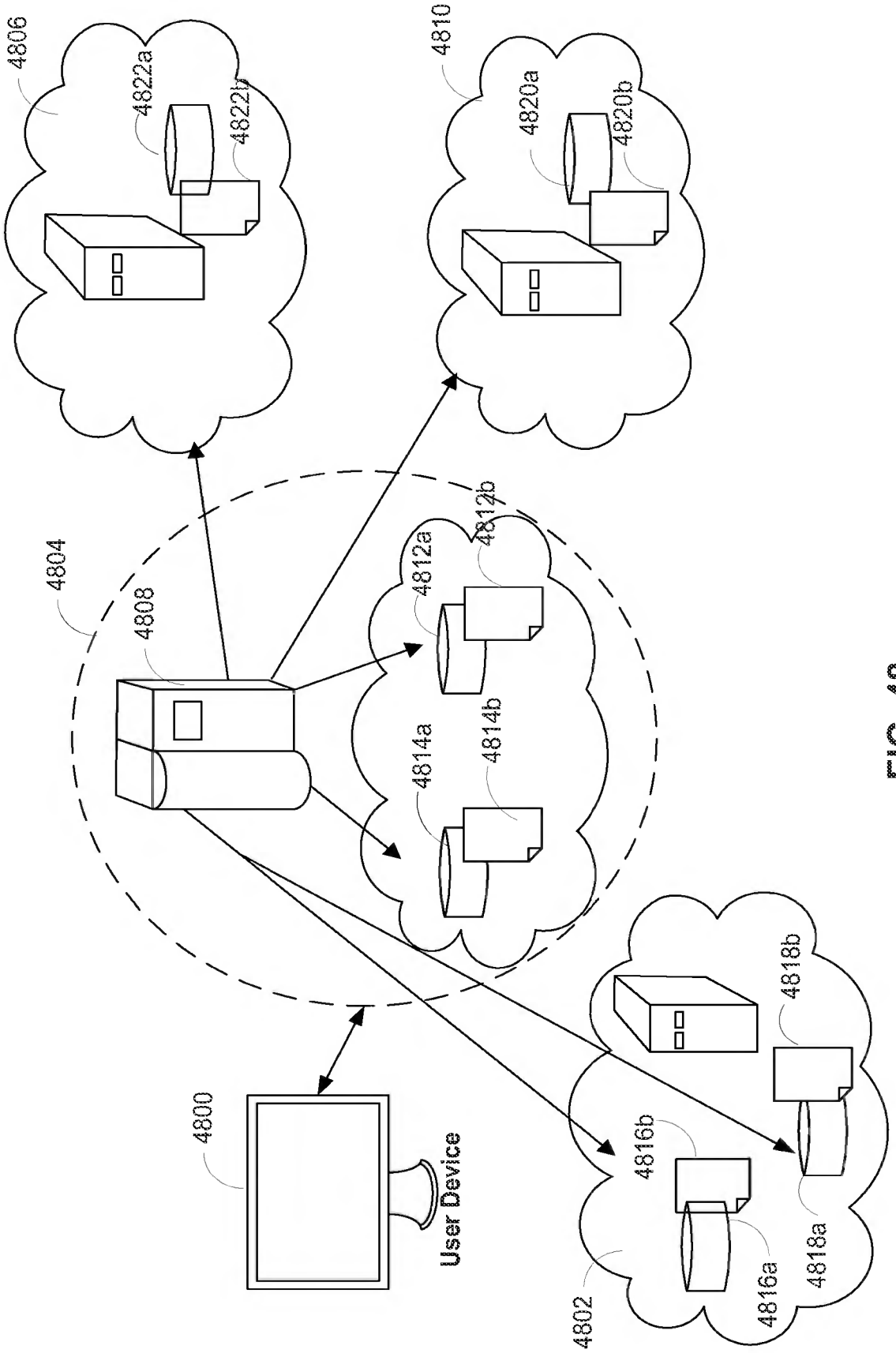

4806
4822a
4822b
4810
4820a
4820b
4804
4808
4812a
4812b
4814a
4814b
4800
User Device
4802
4816b
4816a
4818b
4818a

FIG. 48

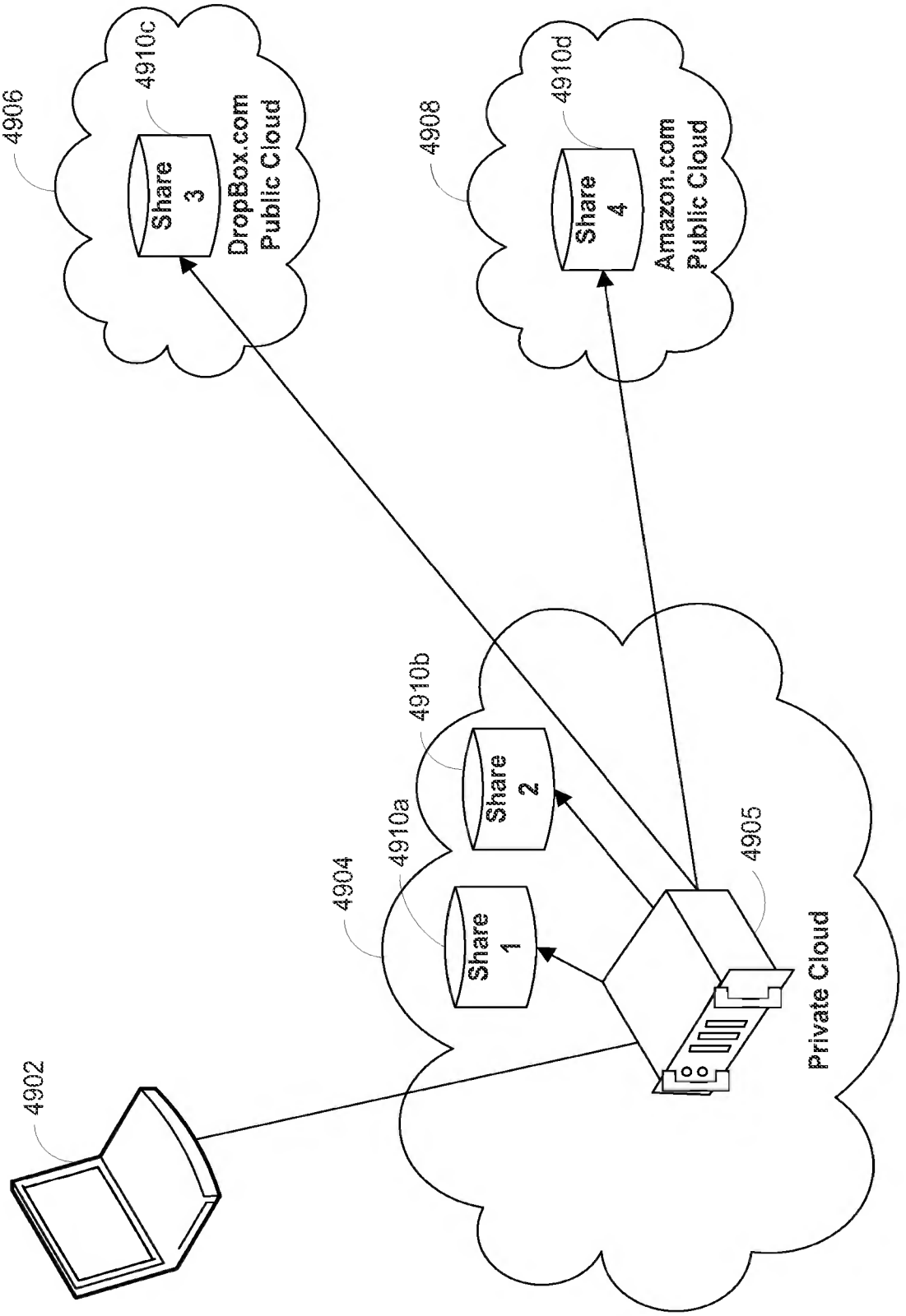

4906
4910c
Share 3
DropBox.com Public Cloud
4908
4910d
Share 4
Amazon.com Public Cloud
4910b
Share 2
4910a
Share 1
4904
4905
Private Cloud
4902

FIG. 49

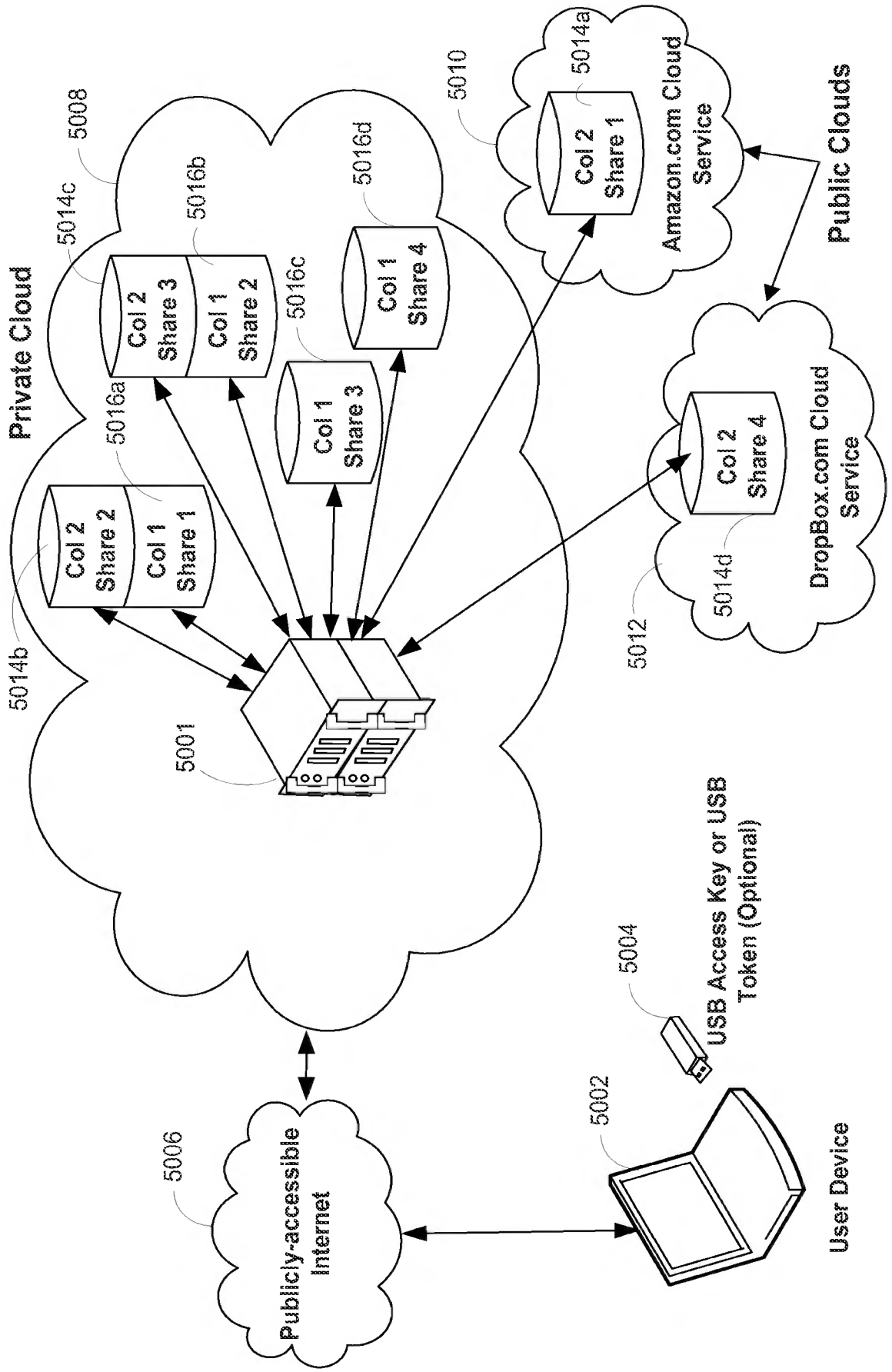

Private Cloud
5008
5014c
5016b
5016d
5010
5014a
Col 2 Share 3
Col 1 Share 2
Col 1 Share 4
Col 2 Share 1
Amazon.com Cloud Service
Public Clouds
5016a
5016c
Col 2 Share 2
Col 1 Share 1
Col 1 Share 3
Col 2 Share 4
DropBox.com Cloud Service
5014b
5001
5012
5014d
5004
USB Access Key or USB Token (Optional)
5002
5006
Publicly-accessible Internet
User Device

FIG. 50

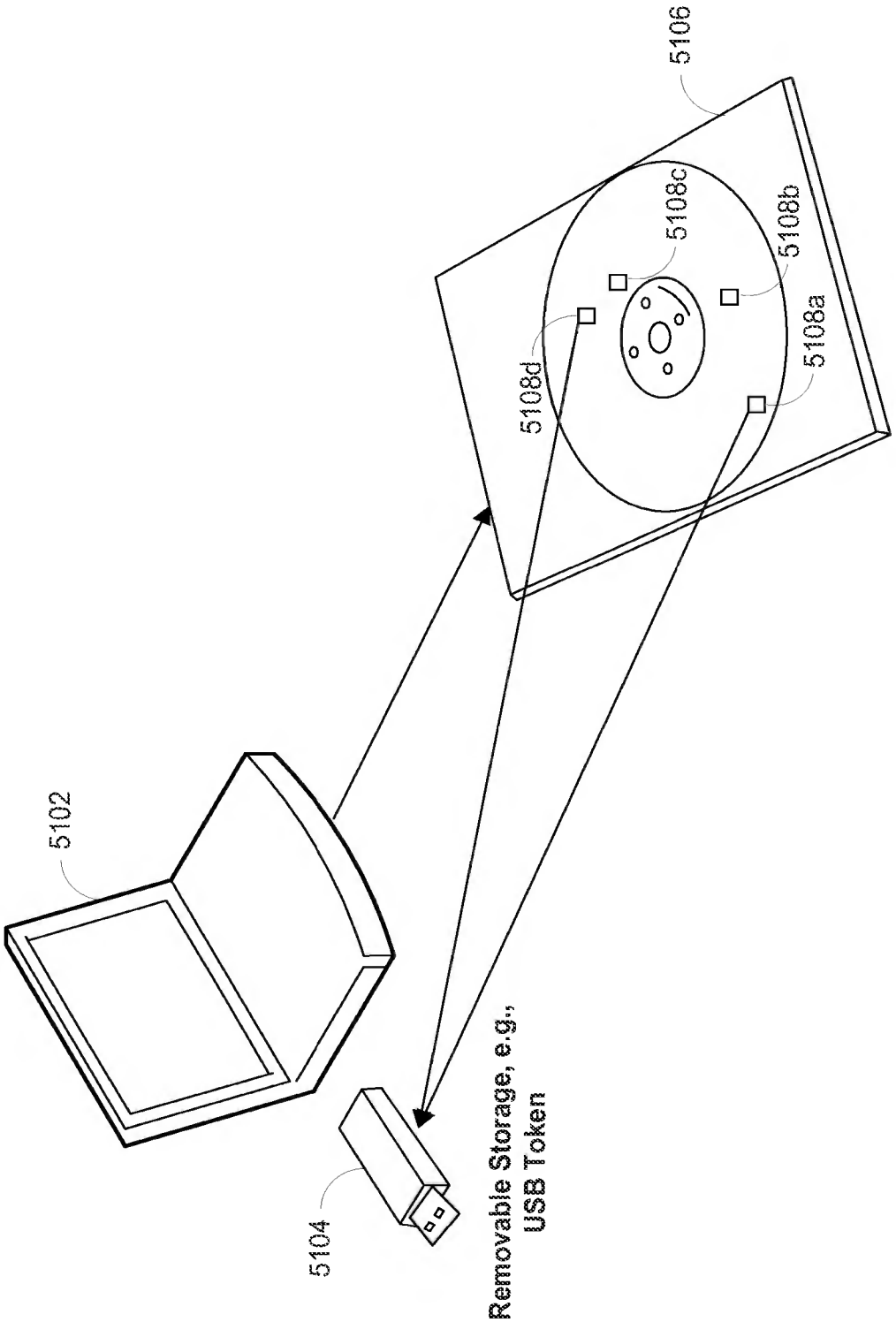

5106
5108c
5108b
5108a
5108d
5102
5104
Removable Storage, e.g.,
USB Token

FIG. 51

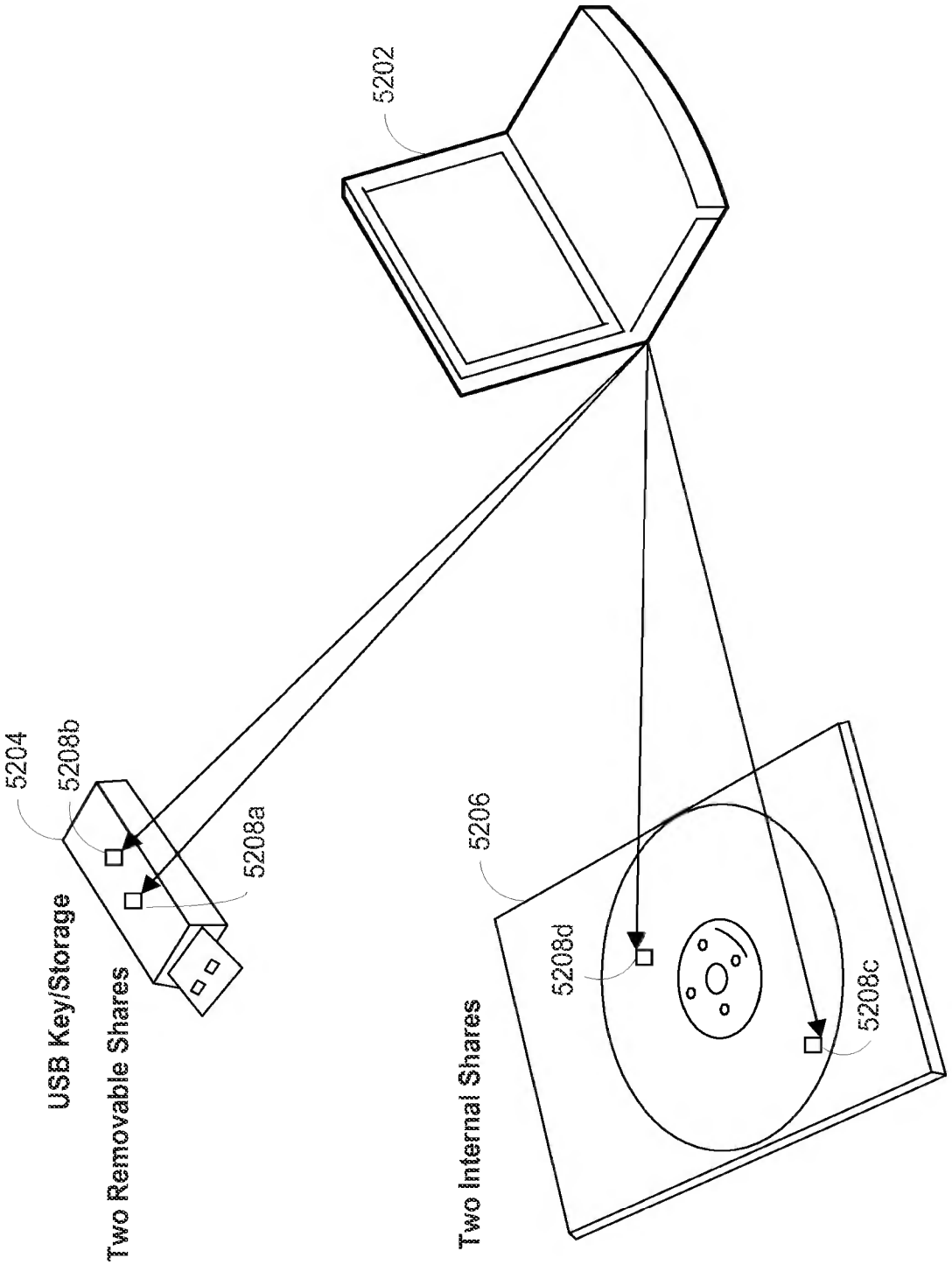

5202
5204
5208b
5208a
USB Key/Storage
Two Removable Shares
5206
5208d
5208c
Two Internal Shares

FIG. 52

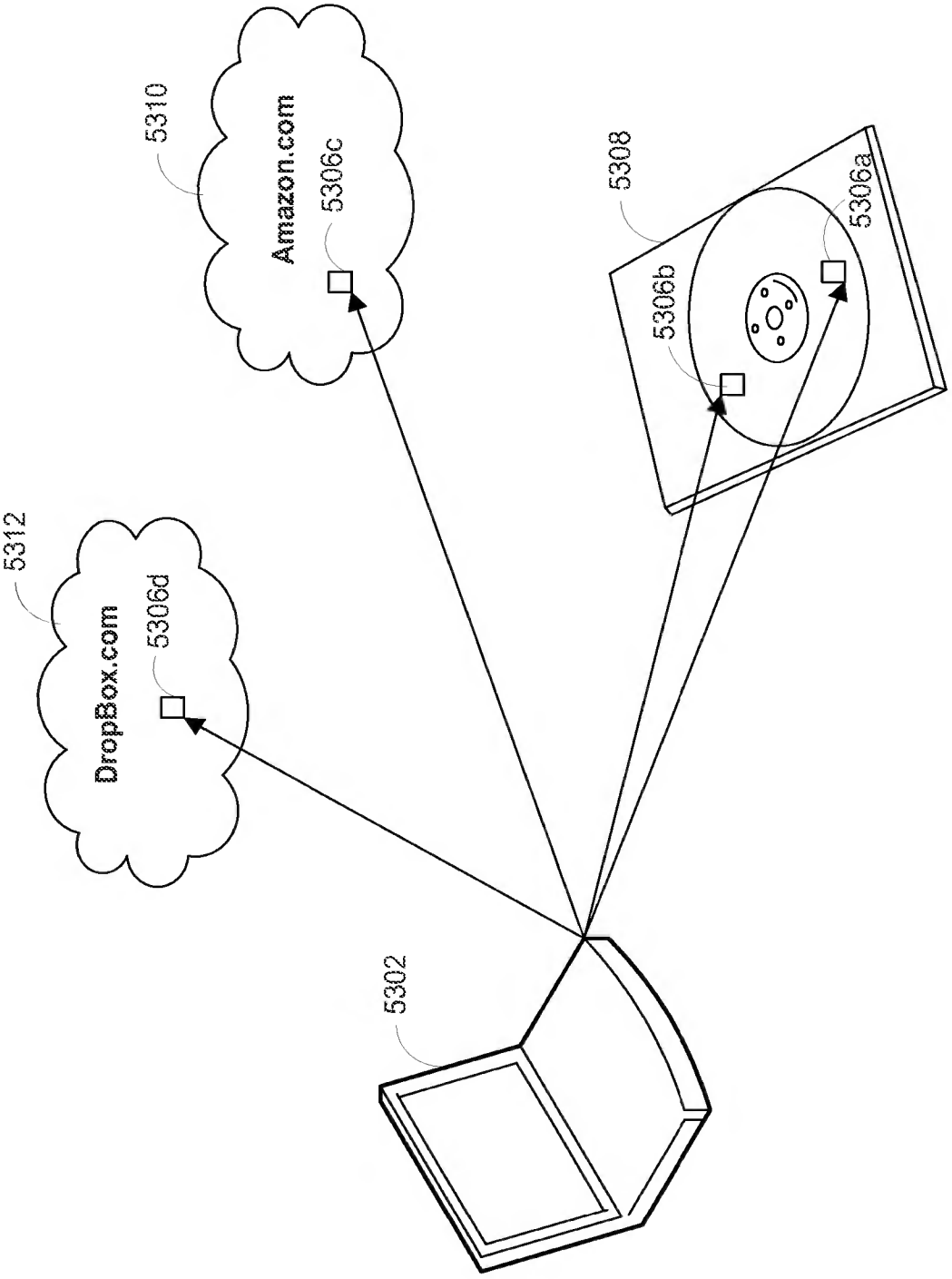

5310
Amazon.com
5306c
5308
5306a
5306b
5312
DropBox.com
5306d
5302

FIG. 53

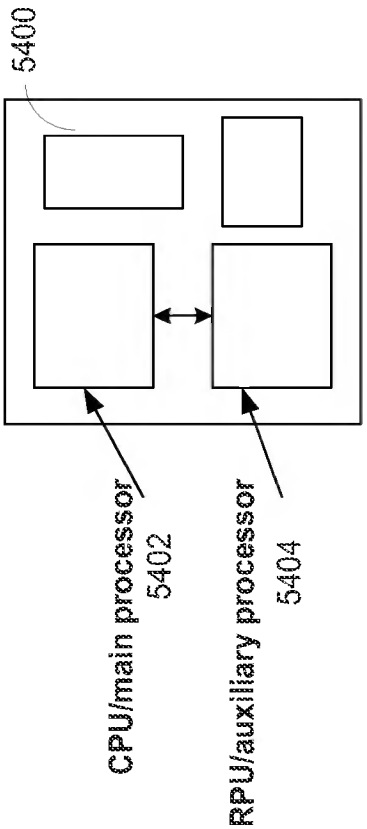


FIG. 54

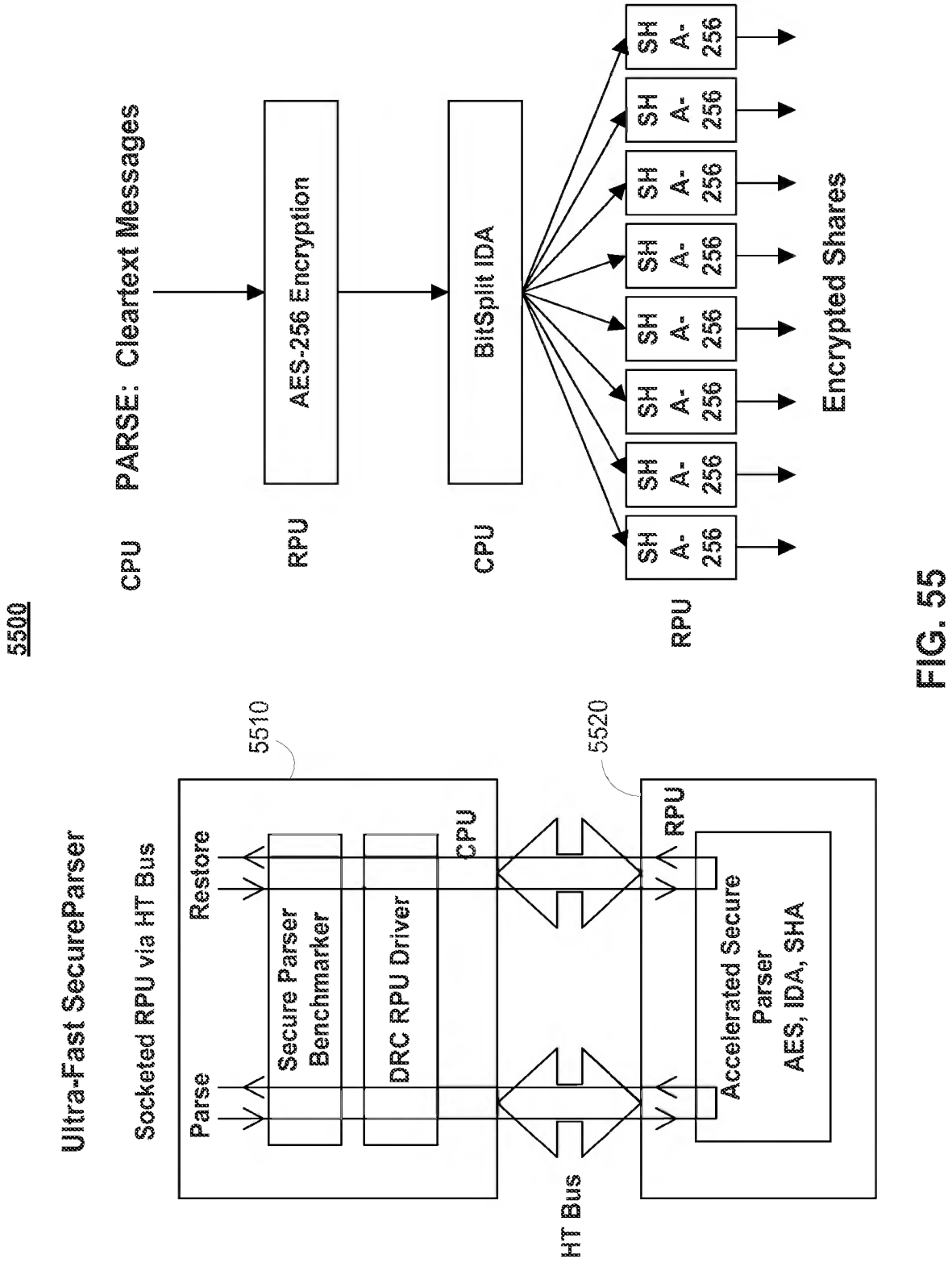

5500
Ultra-Fast SecureParser
Socketed RPU via HT Bus
5510
Restore
Parse
Secure Parser Benchmarker
DRC RPU Driver
CPU
HT Bus
5520
RPU
Accelerated Secure Parser
AES, IDA, SHA
CPU
PARSE: Cleartext Messages
RPU
AES-256 Encryption
CPU
BitSplit IDA
RPU
SHA-256
Encrypted Shares

FIG. 55

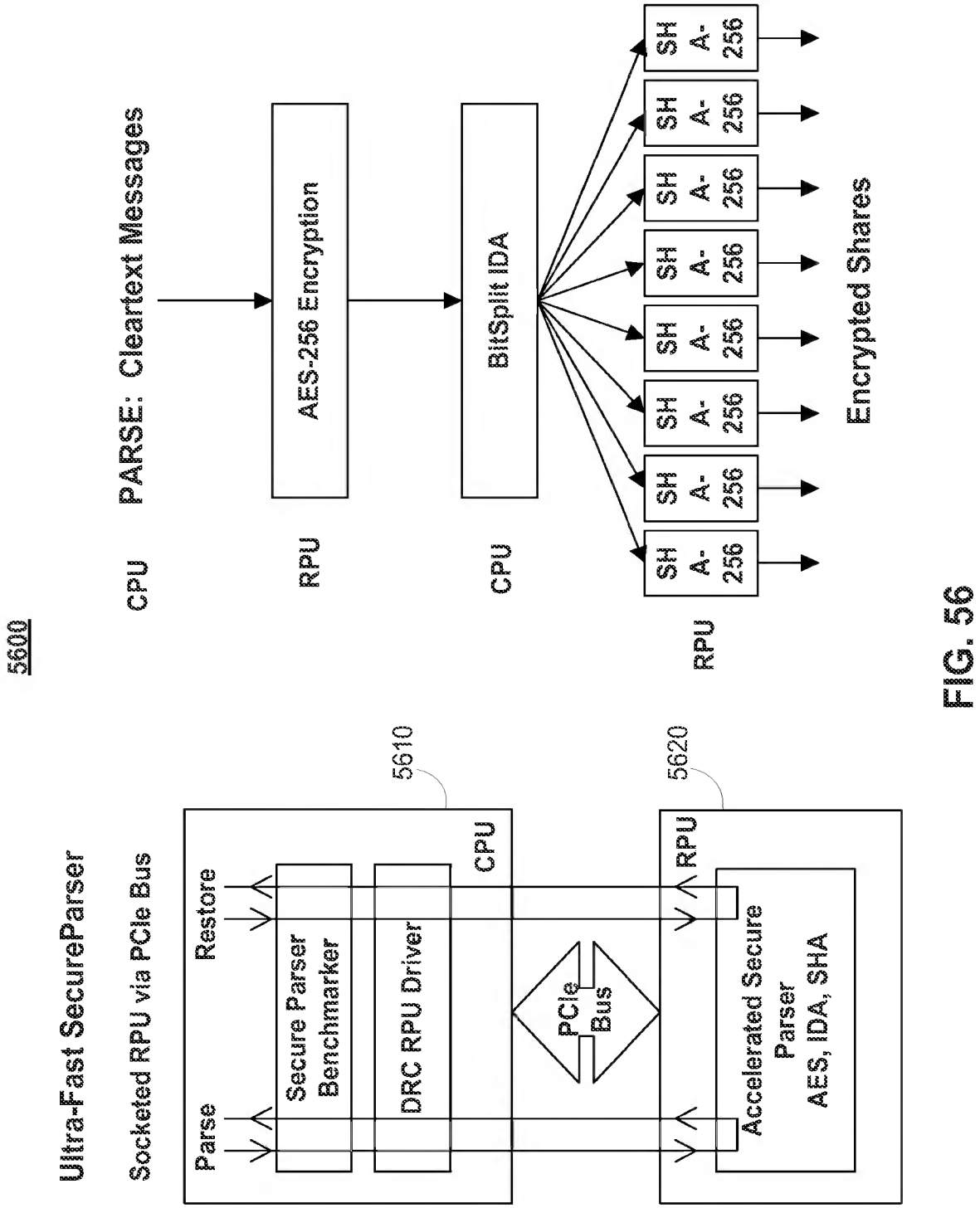

5600
CPU
PARSE: Cleartext Messages
RPU
AES-256 Encryption
CPU
BitSplit IDA
RPU
SHA-256
Encrypted Shares
Ultra-Fast SecureParser
Socketed RPU via PCIe Bus
5610
CPU
Restore
Parse
Secure Parser Benchmarker
DRC RPU Driver
PCIe Bus
5620
RPU
Accelerated Secure Parser
AES, IDA, SHA

FIG. 56

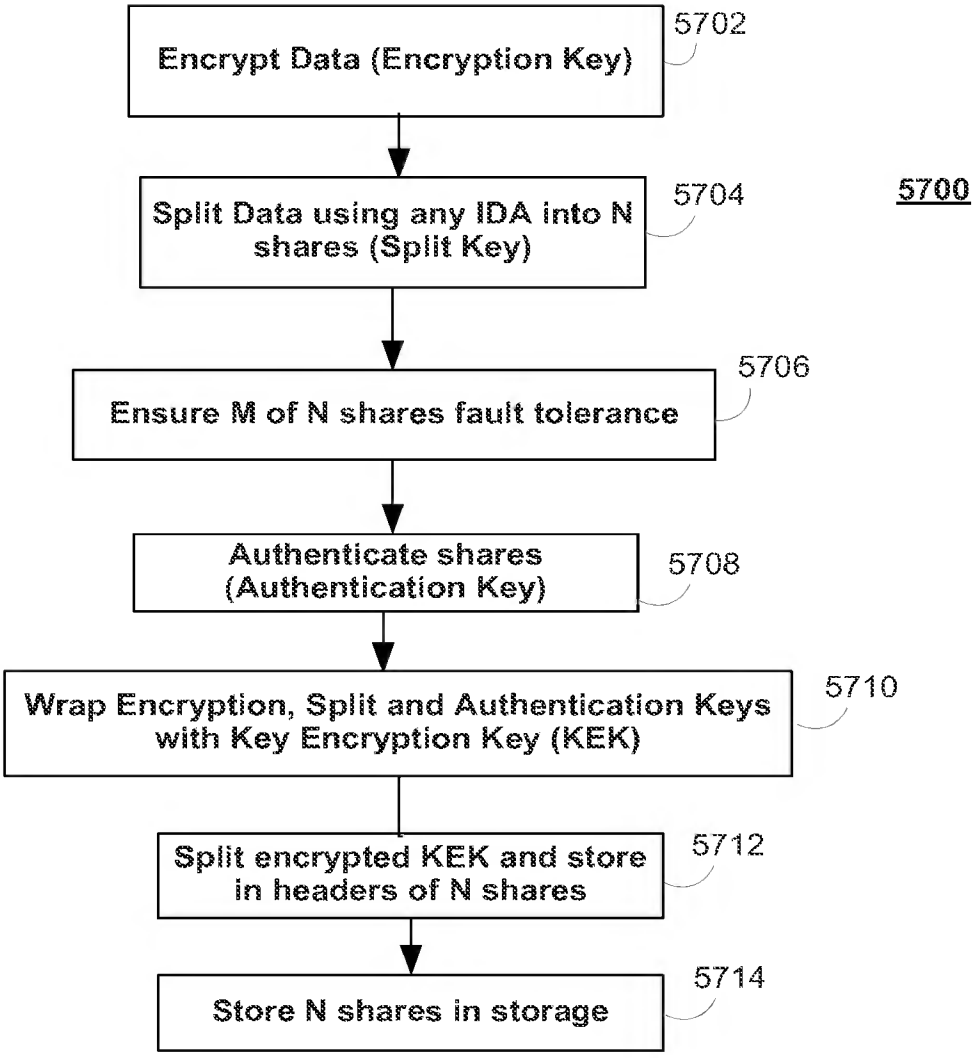

Encrypt Data (Encryption Key)
5702
Split Data using any IDA into N shares (Split Key)
5704
5700
Ensure M of N shares fault tolerance
5706
Authenticate shares (Authentication Key)
5708
Wrap Encryption, Split and Authentication Keys with Key Encryption Key (KEK)
5710
Split encrypted KEK and store in headers of N shares
5712
Store N shares in storage
5714

FIG. 57

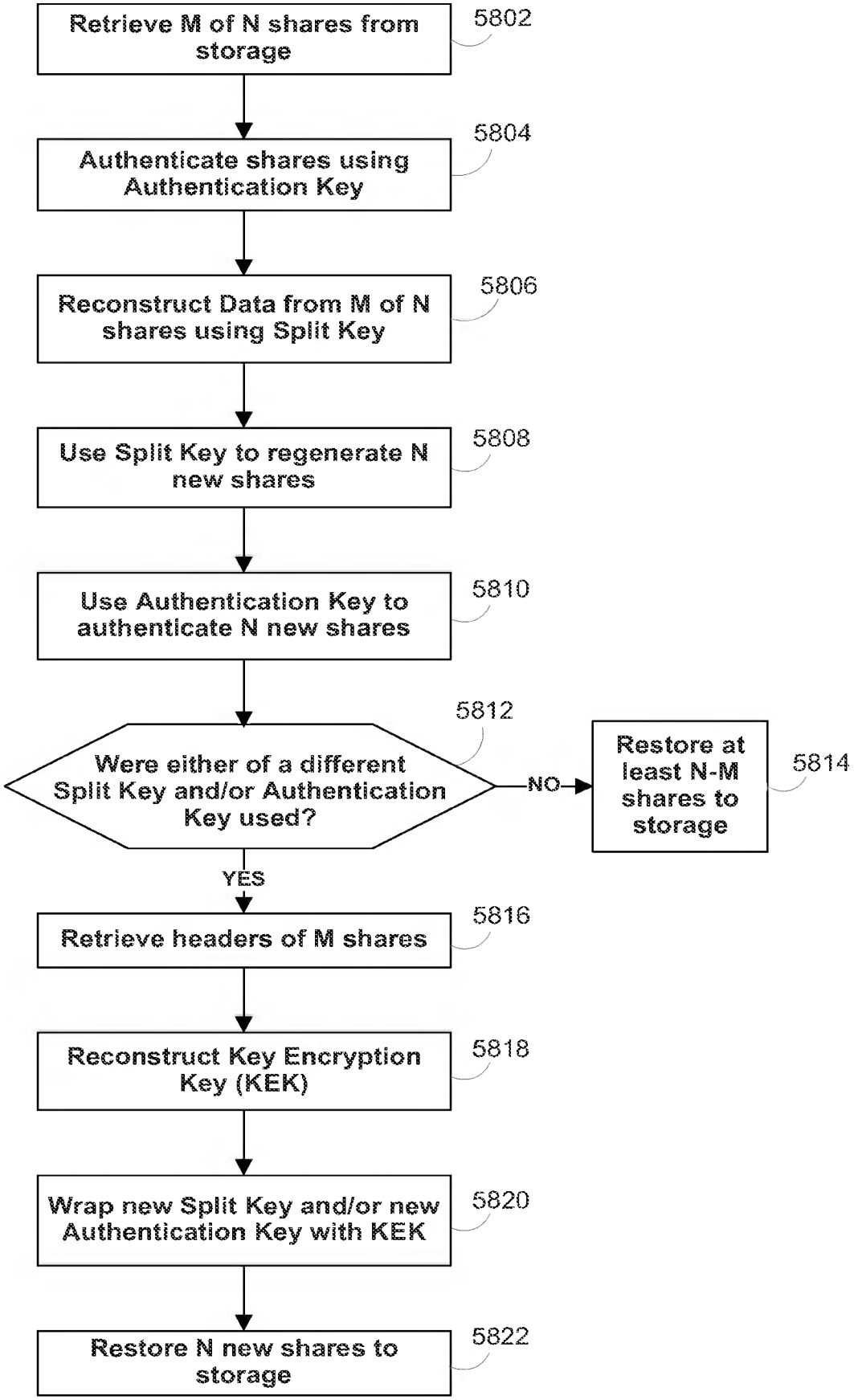

Retrieve M of N shares from storage
5802
Authenticate shares using Authentication Key
5804
Reconstruct Data from M of N shares using Split Key
5806
Use Split Key to regenerate N new shares
5808
Use Authentication Key to authenticate N new shares
5810
Were either of a different Split Key and/or Authentication Key used?
5812
NO
Restore at least N-M shares to storage
5814
YES
Retrieve headers of M shares
5816
Reconstruct Key Encryption Key (KEK)
5818
Wrap new Split Key and/or new Authentication Key with KEK
5820
Restore N new shares to storage
5822

FIG. 58

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
6 October 2011 (06.10.2011)

PCT

(10) International Publication Number
WO 2011/123692 A3

(51) International Patent Classification:
H04L 29/08 (2006.01) *G06F 11/10* (2006.01)
G06F 21/00 (2006.01) *G06F 11/20* (2006.01)
H04L 9/08 (2006.01) *G06F 3/06* (2006.01)
G06F 17/30 (2006.01)

(21) International Application Number:
PCT/US2011/030801

(22) International Filing Date:
31 March 2011 (31.03.2011)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
61/319,658 31 March 2010 (31.03.2010) US
61/320,242 1 April 2010 (01.04.2010) US

(72) Inventors; and
(71) Applicants : **ORSINI, Rick,** L. [US/US]; 2100 Kings Forest Lane, Flower Mound, TX 75028 (US). **O'HARE, Mark, S.** [US/US]; 8 Kennedy Court, Coto De Caza, CA 92679 (US).

(74) Agents: **INGERMAN, Jeffrey, H.** et al.; Ropes & Gray LLP, 1211 Avenue Of The Americas, New York, NY 10036-8704 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))*

[Continued on next page]

(54) Title: SYSTEMS AND METHODS FOR SECURING DATA IN MOTION

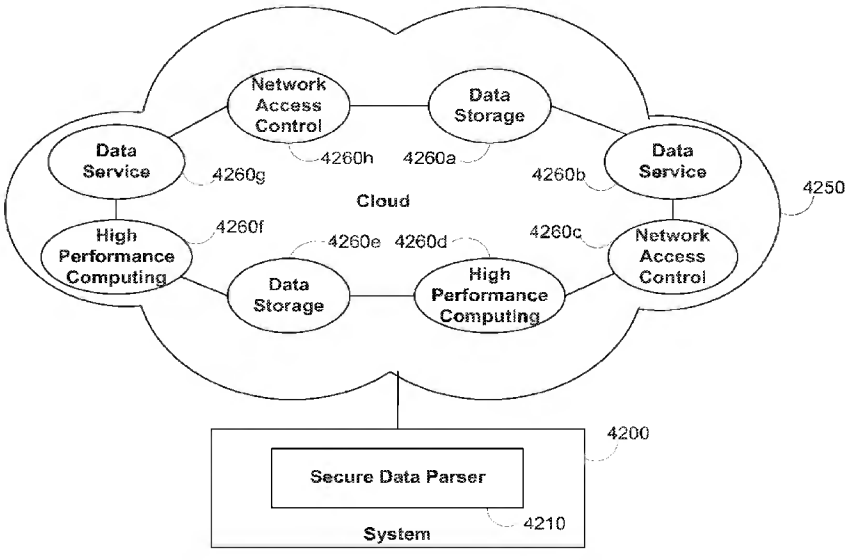


FIG. 42

(57) Abstract: The systems and methods of the present invention provide a solution that makes data provably secure and accessible - addressing data security at the bit level - thereby eliminating the need for multiple perimeter hardware and software technologies. Data security is incorporated or weaved directly into the data at the bit level. The systems and methods of the present invention enable enterprise communities of interest to leverage a common enterprise infrastructure. Because security is already woven into the data, this common infrastructure can be used without compromising data security and access control. In some applications, data is authenticated, encrypted, and parsed or split into multiple shares prior to being sent to multiple locations, e.g., a private or public cloud. The data is hidden while in transit to the storage location, and is inaccessible to users who do not have the correct credentials for access.

WO 2011/123692 A3

(88) Date of publication of the international search report:
5 January 2012

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2011/030801

A. CLASSIFICATION OF SUBJECT MATTER
INV. H04L29/08 G06F21/00 H04L9/08 G06F17/30 G06F11/10
G06F11/20 G06F3/06
ADD.
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
H04L G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)
EPO-Internal

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	RABIN M O: "EFFICIENT DISPERSAL OF INFORMATION FOR SECURITY, LOAD BALANCING, AND FAULT TOLERANCE", JOURNAL OF THE ASSOCIATION FOR COMPUTING MACHINERY, ACM, NEW YORK, NY, US, vol. 36, no. 2, 1 April 1989 (1989-04-01), pages 335-348, XP000570108, ISSN: 0004-5411, DOI: DOI:10.1145/62044.62050 the whole document ----- -/--	1

[X] Further documents are listed in the continuation of Box C.

[X] See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
15 November 2011	23/11/2011

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Mäenpää, Jari

Form PCT/ISA/210 (second sheet) (April 2005)

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	HUGO KRAWCZYK: "Secret Sharing Made Short", INTERNET CITATION, 1998, pages 1-11, XP002505270, Retrieved from the Internet: URL:http://www.cs.cornell.edu/courses/cs754/2001fa/secretshort.pdf [retrieved on 2008-11-24]	1,5
Y	the whole document	2-4
Y	US 2008/183975 A1 (FOSTER LYNN [US] ET AL) 31 July 2008 (2008-07-31) paragraph [0041] the whole document	2-4
A	SHAMIR ET AL: "HOW TO SHARE A SECRET", IP.COM JOURNAL, IP.COM INC., WEST HENRIETTA, NY, US, 30 March 2007 (2007-03-30), XP013119902, ISSN: 1533-0001 the whole document	1-5
A	GANGER G R ET AL: "Survivable storage systems", DARPA INFORMATION SURVIVABILITY CONFERENCE & EXPOSITION II, 2001. DISC EX '01. PROCEEDINGS 12-14 JUNE 2001, PISCATAWAY, NJ, USA,IEEE, vol. 2, 12 June 2001 (2001-06-12), pages 184-195, XP010548746, ISBN: 978-0-7695-1212-9 the whole document	1-5
X	US 2008/147821 A1 (DIETRICH BRADLEY W [US] ET AL) 19 June 2008 (2008-06-19) paragraph [0042] - paragraph [0056] paragraph [0084] - paragraph [0089] paragraphs [0139] - [0144]	6-13
X	Gregory R Ganger ET AL: "PASIS: A Distributed Framework for Perpetually Available and Secure Information Systems, Final technical rept. Jun 1999-Dec 2003", , 1 July 2005 (2005-07-01), pages 1-203, XP55011444, Retrieved from the Internet: URL:http://www.dtic.mil/cgi-bin/GetTRDoc?AD=ADA436245&Location=U2&doc=GetTRDoc.pdf [retrieved on 2011-11-07] page 1 - page 18 paragraph [0003]	14-22

-/--

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 2009/254572 A1 (REDLICH RON M [US] ET AL) 8 October 2009 (2009-10-08) paragraph [0073] paragraph [3155] - paragraph [3160] figures D-10	23-31
Y	EASTER R J ET AL: "S/390 parallel enterprise server CMOS cryptographic coprocessor", IBM JOURNAL OF RESEARCH AND DEVELOPMENT, INTERNATIONAL BUSINESS MACHINES CORPORATION, NEW YORK, NY, US, vol. 43, no. 5, 1 January 1999 (1999-01-01), pages 761-776, XP002335589, ISSN: 0018-8646 the whole document	23-31
X	US 2009/177894 A1 (ORSINI RICK L [US] ET AL) 9 July 2009 (2009-07-09) figure 36 the whole document paragraph [0434]	32-43
Y	US 5 485 474 A (RABIN MICHAEL O [US]) 16 January 1996 (1996-01-16) the whole document	44-54
Y	US 2004/267832 A1 (WONG THOMAS K [US] ET AL) 30 December 2004 (2004-12-30) paragraph [0050] - paragraph [0051]	44-54

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2011/030801

Box No. II Observations where certain claims were found unsearchable (Continuation of item 2 of first sheet)

This international search report has not been established in respect of certain claims under Article 17(2)(a) for the following reasons:

1. ☐ Claims Nos.:
because they relate to subject matter not required to be searched by this Authority, namely:

2. ☐ Claims Nos.:
because they relate to parts of the international application that do not comply with the prescribed requirements to such an extent that no meaningful international search can be carried out, specifically:

3. ☐ Claims Nos.:
because they are dependent claims and are not drafted in accordance with the second and third sentences of Rule 6.4(a).

Box No. III Observations where unity of invention is lacking (Continuation of item 3 of first sheet)

This International Searching Authority found multiple inventions in this international application, as follows:

see additional sheet

1. ☒ As all required additional search fees were timely paid by the applicant, this international search report covers all searchable claims.

2. ☐ As all searchable claims could be searched without effort justifying an additional fees, this Authority did not invite payment of additional fees.

3. ☐ As only some of the required additional search fees were timely paid by the applicant, this international search report covers only those claims for which fees were paid, specifically claims Nos.:

4. ☐ No required additional search fees were timely paid by the applicant. Consequently, this international search report is restricted to the invention first mentioned in the claims; it is covered by claims Nos.:

Remark on Protest

☐ The additional search fees were accompanied by the applicant's protest and, where applicable, the payment of a protest fee.

☐ The additional search fees were accompanied by the applicant's protest but the applicable protest fee was not paid within the time limit specified in the invitation.

☒ No protest accompanied the payment of additional search fees.

FURTHER INFORMATION CONTINUED FROM PCT/ISA/ 210

This International Searching Authority found multiple (groups of) inventions in this international application, as follows:

1. claims: 1-5

   A method for rebuilding dispersed data wherein rebuilding is performed in response to a determination that one or more of the data shares have been compromised. The problem to be solved is to achieve better integrity protection and reliability of the storing method.

   ---

2. claims: 6-13

   A method for rekeying a set of data shares dispersed using a key.The problem to be solved is to achieve better confidentiality protection of the stored data.

   ---

3. claims: 14-22

   A method for associating stubs with a set of data shares on the file system of a storage network. The problem to be solved is to implement an efficient storage network.

   ---

4. claims: 23-31

   A co-processor acceleration device for secure parsing functions. The problem to be solved is to implement an efficient hardware based secure parser.

   ---

5. claims: 32-43

   A method for securing data using a portable device. The problem to be solved is to implement a data security system adapted to portable devices.

   ---

6. claims: 44-54

   A method for securing the file name of a file to be split. The problem to be solved is to implement an efficient file name security.

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/US2011/030801

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2008183975 A1	31-07-2008	EP 2260387 A2	15-12-2010
		US 2008183975 A1	31-07-2008
		US 2010161916 A1	24-06-2010
		WO 2009123865 A2	08-10-2009
US 2008147821 A1	19-06-2008	NONE	
US 2009254572 A1	08-10-2009	NONE	
US 2009177894 A1	09-07-2009	AU 2009204512 A1	16-07-2009
		CA 2710868 A1	16-07-2009
		CN 101939946 A	05-01-2011
		EP 2106642 A1	07-10-2009
		US 2009177894 A1	09-07-2009
		WO 2009089015 A1	16-07-2009
US 5485474 A	16-01-1996	NONE	
US 2004267832 A1	30-12-2004	NONE	

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
16 July 2009 (16.07.2009)

PCT

(10) International Publication Number
WO 2009/089015 A1

(51) International Patent Classification:
H04L 9/00 (2006.01)

(21) International Application Number:
PCT/US2009/000083

(22) International Filing Date: 7 January 2009 (07.01.2009)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
61/010,313 7 January 2008 (07.01.2008) US

(71) Applicant *(for all designated States except US)*: **SECURITY FIRST CORPORATION** [US/US]; 22362 Gilberto #130, Rancho Santa Margarita, California 92688 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **ORSINI, Rick L.** [US/US]; 2100 Kings Forest Lane, Flower Mound, Texas 75028 (US). **O'HARE, Mark S.** [US/US]; 8 Kennedy Court, Coto De Caza, California 92679 (US). **BELLARE, Mihir** [US/US]; 11338 East San Raphael Driveway, San Diego, California 92130 (US). **ROGAWAY, Phillip** [US/US]; 1212 Purdue Drive, Davis, California 95616 (US).

(74) Agents: **INGERMAN, Jeffrey H.** et al.; Ropes & Gray LLP, 1211 Avenue of the Americas, New York, New York 10036-8704 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:
— *with international search report*

(54) Title: SYSTEMS AND METHODS FOR SECURING DATA USING MULTI-FACTOR OR KEYED DISPERSAL

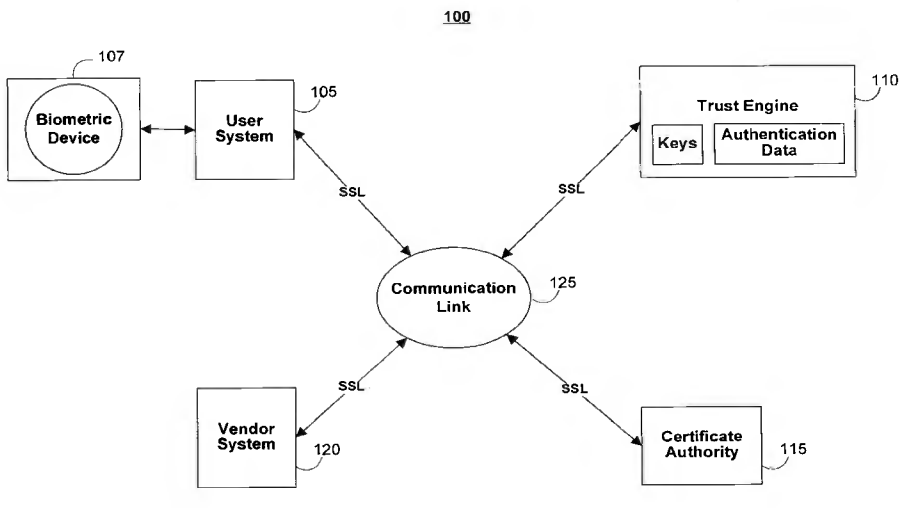


FIG. 1

(57) Abstract: A secure data parser is provided that may be integrated into any suitable system for securely storing and communicating data. The secure data parser parses data and then splits the data into multiple portions that are stored or communicated distinctly. Encryption of the original data, the portions of data, or both may be employed for additional security. The secure data parser may be used to protect data in motion by splitting original data into portions of data, that may be communicated using multiple communications paths. A keyed information dispersal algorithm (keyed IDA) may also be used. The key for the keyed IDA may additionally be protected by an external work group key, resulting in a multi-factor secret sharing scheme.

WO 2009/089015 A1

SYSTEMS AND METHODS FOR SECURING DATA USING MULTI-FACTOR OR KEYED DISPERSAL

Cross-Reference to Related Applications

[0001] This claims the benefit of United States Provisional Patent Application Nos. 61/010,313, entitled "Systems and Methods for Securing Data" filed January 7, 2008, which is hereby incorporated by reference herein in its entirety.

Field of the Invention

[0002] The present invention relates in general to improved systems and methods for securing data. Some of the systems and methods described herein utilize keyed information dispersal algorithms (called keyed IDAs) and multi-factor secret sharing techniques. The systems and methods described herein may be used in conjunction with other systems and methods described in commonly-owned U.S. Patent No. 7,391,865 and commonly-owned U.S. Patent Application Nos. 10/458,928, filed June 11, 2003, 11/258,839, filed October 25, 2005, 11/602,667, filed November 20, 2006, 11/983,355, filed November 7, 2007, 11/999,575, filed December 5, 2007, and 12/209,703, filed September 12, 2008, all of which are hereby incorporated by reference herein in their entireties.

Background of the Invention

[0003] In today's society, individuals and businesses conduct an ever-increasing amount of activities on and over computer systems. These computer systems, including proprietary and non-proprietary computer networks, are often storing, archiving, and transmitting all types of sensitive information. Thus, an ever-increasing need exists for ensuring data stored and transmitted over these systems cannot be read or otherwise compromised.

[0004] One common solution for securing computer systems is to provide login and password functionality. However, password management has proven to be quite costly with a large percentage of help desk calls relating to password issues. Moreover, passwords provide little security in that they are generally stored in a file susceptible to inappropriate access, through, for example, brute-force attacks.

[0005] Another solution for securing computer systems is to provide cryptographic infrastructures. Cryptography, in general, refers to protecting data by transforming, or encrypting, it into an unreadable format. Only those who possess the key(s) to the encryption can decrypt the data into a useable format. Cryptography is used to identify users, e.g., authentication, to allow access privileges, e.g., authorization, to create digital certificates and signatures, and the like. One popular cryptography system is a public key system that uses two keys, a public key known to everyone and a private key known only to the individual or business owner thereof.

Generally, the data encrypted with one key is decrypted with the other and neither key is recreatable from the other.

[0006] Unfortunately, even the foregoing typical public-key cryptographic systems are still highly reliant on the user for security. For example, cryptographic systems issue the private key to the user, for example, through the user's browser. Unsophisticated users then generally store the private key on a hard drive accessible to others through an open computer system, such as, for example, the Internet. On the other hand, users may choose poor names for files containing their private key, such as, for example, "key." The result of the foregoing and other acts is to allow the key or keys to be susceptible to compromise.

[0007] In addition to the foregoing compromises, a user may save his or her private key on a computer system configured with an archiving or backup system, potentially resulting in copies of the private key traveling through multiple computer storage devices or other systems. This security breach is often referred to as "key migration." Similar to key migration, many applications provide access to a user's private key through, at most, simple login and password access. As mentioned in the foregoing, login and password access often does not provide adequate security.

[0008] One solution for increasing the security of the foregoing cryptographic systems is to include biometrics as part of the authentication or authorization. Biometrics generally include measurable physical characteristics, such as, for example, finger prints or speech that can be checked by an automated

system, such as, for example, pattern matching or recognition of finger print patterns or speech patterns. In such systems, a user's biometric and/or keys may be stored on mobile computing devices, such as, for example, a smartcard, laptop, personal digital assistant, or mobile phone, thereby allowing the biometric or keys to be usable in a mobile environment.

[0009] The foregoing mobile biometric cryptographic system still suffers from a variety of drawbacks. For example, the mobile user may lose or break the smartcard or portable computing device, thereby having his or her access to potentially important data entirely cut-off. Alternatively, a malicious person may steal the mobile user's smartcard or portable computing device and use it to effectively steal the mobile user's digital credentials. On the other hand, the portable-computing device may be connected to an open system, such as the Internet, and, like passwords, the file where the biometric is stored may be susceptible to compromise through user inattentiveness to security or malicious intruders.

Summary of the Invention

[0010] Based on the foregoing, a need exists to provide a cryptographic system whose security is user-independent while still supporting mobile users.

[0011] Accordingly, one aspect of the present invention is to provide a method for securing virtually any type of data from unauthorized access or use. The method comprises one or more steps of parsing, splitting and/or separating the data to be secured into two or more parts or portions. The method also comprises encrypting the data to be secured.

Encryption of the data may be performed prior to or after the first parsing, splitting and/or separating of the data. In addition, the encrypting step may be repeated for one or more portions of the data. Similarly, the parsing, splitting and/or separating steps may be repeated for one or more portions of the data. The method also optionally comprises storing the parsed, split and/or separated data that has been encrypted in one location or in multiple locations. This method also optionally comprises reconstituting or re-assembling the secured data into its original form for authorized access or use. This method may be incorporated into the operations of any computer, server, engine or the like, that is capable of executing the desired steps of the method.

[0012] Another aspect of the present invention provides a system for securing virtually any type of data from unauthorized access or use. This system comprises a data splitting module, a cryptographic handling module, and, optionally, a data assembly module. The system may, in one embodiment, further comprise one or more data storage facilities where secure data may be stored.

[0013] Accordingly, one aspect of the invention is to provide a secure server, or trust engine, having server-centric keys, or in other words, storing cryptographic keys and user authentication data on a server. According to this embodiment, a user accesses the trust engine in order to perform authentication and cryptographic functions, such as, but not limited to, for example, authentication, authorization, digital signing and generation, storage, and retrieval of

certificates, encryption, notary-like and power-of-attorney-like actions, and the like.

[0014] Another aspect of the invention is to provide a reliable, or trusted, authentication process. Moreover, subsequent to a trustworthy positive authentication, a wide number of differing actions may be taken, from providing cryptographic technology, to system or device authorization and access, to permitting use or control of one or a wide number of electronic devices.

[0015] Another aspect of the invention is to provide cryptographic keys and authentication data in an environment where they are not lost, stolen, or compromised, thereby advantageously avoiding a need to continually reissue and manage new keys and authentication data. According to another aspect of the invention, the trust engine allows a user to use one key pair for multiple activities, vendors, and/or authentication requests. According to yet another aspect of the invention, the trust engine performs at least one step of cryptographic processing, such as, but not limited to, encrypting, authenticating, or signing, on the server side, thereby allowing clients or users to possess only minimal computing resources.

[0016] According to yet another aspect of the invention, the trust engine includes one or multiple depositories for storing portions of each cryptographic key and authentication data. The portions are created through a data splitting process that prohibits reconstruction without a predetermined portion from more than one location in one depository or from multiple depositories. According to another embodiment, the multiple depositories may be

geographically remote such that a rogue employee or otherwise compromised system at one depository will not provide access to a user's key or authentication data.

[0017] According to yet another embodiment, the authentication process advantageously allows the trust engine to process multiple authentication activities in parallel. According to yet another embodiment, the trust engine may advantageously track failed access attempts and thereby limit the number of times malicious intruders may attempt to subvert the system.

[0018] According to yet another embodiment, the trust engine may include multiple instantiations where each trust engine may predict and share processing loads with the others. According to yet another embodiment, the trust engine may include a redundancy module for polling a plurality of authentication results to ensure that more than one system authenticates the user.

[0019] Therefore, one aspect of the invention includes a secure cryptographic system, which may be remotely accessible, for storing data of any type, including, but not limited to, a plurality of private cryptographic keys to be associated with a plurality of users. The cryptographic system associates each of the plurality of users with one or more different keys from the plurality of private cryptographic keys and performs cryptographic functions for each user using the associated one or more different keys without releasing the plurality of private cryptographic keys to the users. The cryptographic system comprises a depository system having at least one server which stores the data to be secured, such as a plurality of private cryptographic keys and a plurality of

enrollment authentication data. Each enrollment authentication data identifies one of multiple users and each of the multiple users is associated with one or more different keys from the plurality of private cryptographic keys. The cryptographic system also may comprise an authentication engine which compares authentication data received by one of the multiple users to enrollment authentication data corresponding to the one of multiple users and received from the depository system, thereby producing an authentication result. The cryptographic system also may comprise a cryptographic engine which, when the authentication result indicates proper identification of the one of the multiple users, performs cryptographic functions on behalf of the one of the multiple users using the associated one or more different keys received from the depository system. The cryptographic system also may comprise a transaction engine connected to route data from the multiple users to the depository server system, the authentication engine, and the cryptographic engine.

[0020] Another aspect of the invention includes a secure cryptographic system that is optionally remotely accessible. The cryptographic system comprises a depository system having at least one server which stores at least one private key and any other data, such as, but not limited to, a plurality of enrollment authentication data, wherein each enrollment authentication data identifies one of possibly multiple users. The cryptographic system may also optionally comprise an authentication engine which compares authentication data received by users to enrollment authentication data corresponding to the user and

received from the depository system, thereby producing an authentication result. The cryptographic system also comprises a cryptographic engine which, when the authentication result indicates proper identification of the user, performs cryptographic functions on behalf of the user using at least said private key, which may be received from the depository system. The cryptographic system may also optionally comprise a transaction engine connected to route data from the users to other engines or systems such as, but not limited to, the depository server system, the authentication engine, and the cryptographic engine.

[0021] Another aspect of the invention includes a method of facilitating cryptographic functions. The method comprises associating a user from multiple users with one or more keys from a plurality of private cryptographic keys stored on a secure location, such as a secure server. The method also comprises receiving authentication data from the user, and comparing the authentication data to authentication data corresponding to the user, thereby verifying the identity of the user. The method also comprises utilizing the one or more keys to perform cryptographic functions without releasing the one or more keys to the user.

[0022] Another aspect of the invention includes an authentication system for uniquely identifying a user through secure storage of the user's enrollment authentication data. The authentication system comprises one or more data storage facilities, wherein each data storage facility includes a computer accessible storage medium which stores at least one of portions of enrollment authentication data. The

authentication system also comprises an authentication engine which communicates with the data storage facility or facilities. The authentication engine comprises a data splitting module which operates on the enrollment authentication data to create portions, a data assembling module which processes the portions from at least one of the data storage facilities to assemble the enrollment authentication data, and a data comparator module which receives current authentication data from a user and compares the current authentication data with the assembled enrollment authentication data to determine whether the user has been uniquely identified.

[0023] Another aspect of the invention includes a cryptographic system. The cryptographic system comprises one or more data storage facilities, wherein each data storage facility includes a computer accessible storage medium which stores at least one portion of one ore more cryptographic keys. The cryptographic system also comprises a cryptographic engine which communicates with the data storage facilities. The cryptographic engine also comprises a data splitting module which operate on the cryptographic keys to create portions, a data assembling module which processes the portions from at least one of the data storage facilities to assemble the cryptographic keys, and a cryptographic handling module which receives the assembled cryptographic keys and performs cryptographic functions therewith.

[0024] Another aspect of the invention includes a method of storing any type of data, including, but not limited to, authentication data in geographically remote secure data storage facilities thereby

protecting the data against composition of any individual data storage facility. The method comprises receiving data at a trust engine, combining at the trust engine the data with a first substantially random value to form a first combined value, and combining the data with a second substantially random value to form a second combined value. The method comprises creating a first pairing of the first substantially random value with the second combined value, creating a second pairing of the first substantially random value with the second substantially random value, and storing the first pairing in a first secure data storage facility. The method comprises storing the second pairing in a second secure data storage facility remote from the first secure data storage facility.

[0025] Another aspect of the invention includes a method of storing any type of data, including, but not limited to, authentication data comprising receiving data, combining the data with a first set of bits to form a second set of bits, and combining the data with a third set of bits to form a fourth set of bits. The method also comprises creating a first pairing of the first set of bits with the third set of bits. The method also comprises creating a second pairing of the first set of bits with the fourth set of bits, and storing one of the first and second pairings in a first computer accessible storage medium. The method also comprises storing the other of the first and second pairings in a second computer accessible storage medium.

[0026] Another aspect of the invention includes a method of storing cryptographic data in geographically remote secure data storage facilities thereby

protecting the cryptographic data against comprise of any individual data storage facility. The method comprises receiving cryptographic data at a trust engine, combining at the trust engine the cryptographic data with a first substantially random value to form a first combined value, and combining the cryptographic data with a second substantially random value to form a second combined value. The method also comprises creating a first pairing of the first substantially random value with the second combined value, creating a second pairing of the first substantially random value with the second substantially random value, and storing the first pairing in a first secure data storage facility. The method also comprises storing the second pairing in a secure second data storage facility remote from the first secure data storage facility.

[0027] Another aspect of the invention includes a method of storing cryptographic data comprising receiving authentication data and combining the cryptographic data with a first set of bits to form a second set of bits. The method also comprises combining the cryptographic data with a third set of bits to form a fourth set of bits, creating a first pairing of the first set of bits with the third set of bits, and creating a second pairing of the first set of bits with the fourth set of bits. The method also comprises storing one of the first and second pairings in a first computer accessible storage medium, and storing the other of the first and second pairings in a second computer accessible storage medium.

[0028] Another aspect of the invention includes a method of handling sensitive data of any type or form in a cryptographic system, wherein the sensitive data

exists in a useable form only during actions by authorized users, employing the sensitive data. The method also comprises receiving in a software module, substantially randomized or encrypted sensitive data from a first computer accessible storage medium, and receiving in the software module, substantially randomized or encrypted data which may or may not be sensitive data, from one or more other computer accessible storage medium. The method also comprises processing the substantially randomized pre-encrypted sensitive data and the substantially randomized or encrypted data which may or may not be sensitive data, in the software module to assemble the sensitive data and employing the sensitive data in a software engine to perform an action. The action includes, but is not limited to, one of authenticating a user and performing a cryptographic function.

[0029] Another aspect of the invention includes a secure authentication system. The secure authentication system comprises a plurality of authentication engines. Each authentication engine receives enrollment authentication data designed to uniquely identify a user to a degree of certainty. Each authentication engine receives current authentication data to compare to the enrollment authentication data, and each authentication engine determines an authentication result. The secure authentication system also comprises a redundancy system which receives the authentication result of at least two of the authentication engines and determines whether the user has been uniquely identified.

[0030] Another aspect of the invention includes a secure data in motion system whereby data may be

transmitted in different portions that are secured in accordance with the present invention such that any one portion becoming compromised shall not provide sufficient data to restore the original data. This may be applied to any transmission of data, whether it be wired, wireless, or physical.

[0031] Another aspect of the invention includes integration of the secure data parser of the present invention into any suitable system where data is stored or communicated. For example, email system, RAID systems, video broadcasting systems, database systems, or any other suitable system may have the secure data parser integrated at any suitable level.

[0032] Another aspect of the invention includes using any suitable parsing and splitting algorithm to generate shares of data. Either random, pseudo-random, deterministic, or any combination thereof may be employed for parsing and splitting data.

Brief Description of the Drawings

[0033] The present invention is described in more detail below in connection with the attached drawings, which are meant to illustrate and not to limit the invention, and in which:

[0034] FIGURE 1 illustrates a block diagram of a cryptographic system, according to aspects of an embodiment of the invention;

[0035] FIGURE 2 illustrates a block diagram of the trust engine of FIGURE 1, according to aspects of an embodiment of the invention;

[0036] FIGURE 3 illustrates a block diagram of the transaction engine of FIGURE 2, according to aspects of an embodiment of the invention;

[0037] FIGURE 4 illustrates a block diagram of the depository of FIGURE 2, according to aspects of an embodiment of the invention;

[0038] FIGURE 5 illustrates a block diagram of the authentication engine of FIGURE 2, according to aspects of an embodiment of the invention;

[0039] FIGURE 6 illustrates a block diagram of the cryptographic engine of FIGURE 2, according to aspects of an embodiment of the invention;

[0040] FIGURE 7 illustrates a block diagram of a depository system, according to aspects of another embodiment of the invention;

[0041] FIGURE 8 illustrates a flow chart of a data splitting process according to aspects of an embodiment of the invention;

[0042] FIGURE 9, Panel A illustrates a data flow of an enrollment process according to aspects of an embodiment of the invention;

[0043] FIGURE 9, Panel B illustrates a flow chart of an interoperability process according to aspects of an embodiment of the invention;

[0044] FIGURE 10 illustrates a data flow of an authentication process according to aspects of an embodiment of the invention;

[0045] FIGURE 11 illustrates a data flow of a signing process according to aspects of an embodiment of the invention;

[0046] FIGURE 12 illustrates a data flow and an encryption/decryption process according to aspects and yet another embodiment of the invention;

[0047] FIGURE 13 illustrates a simplified block diagram of a trust engine system according to aspects of another embodiment of the invention;

[0048] FIGURE 14 illustrates a simplified block diagram of a trust engine system according to aspects of another embodiment of the invention;

[0049] FIGURE 15 illustrates a block diagram of the redundancy module of FIGURE 14, according to aspects of an embodiment of the invention;

[0050] FIGURE 16 illustrates a process for evaluating authentications according to one aspect of the invention;

[0051] FIGURE 17 illustrates a process for assigning a value to an authentication according to one aspect as shown in FIGURE 16 of the invention;

[0052] FIGURE 18 illustrates a process for performing trust arbitrage in an aspect of the invention as shown in FIGURE 17; and

[0053] FIGURE 19 illustrates a sample transaction between a user and a vendor according to aspects of an embodiment of the invention where an initial web based contact leads to a sales contract signed by both parties.

[0054] FIGURE 20 illustrates a sample user system with a cryptographic service provider module which provides security functions to a user system.

[0055] FIGURE 21 illustrates a process for parsing, splitting and/or separating data with encryption and storage of the encryption master key with the data.

[0056] FIGURE 22 illustrates a process for parsing, splitting and/or separating data with encryption and storing the encryption master key separately from the data.

[0057] FIGURE 23 illustrates the intermediary key process for parsing, splitting and/or separating data

with encryption and storage of the encryption master key with the data.

[0058] FIGURE 24 illustrates the intermediary key process for parsing, splitting and/or separating data with encryption and storing the encryption master key separately from the data.

[0059] FIGURE 25 illustrates utilization of the cryptographic methods and systems of the present invention with a small working group.

[0060] FIGURE 26 is a block diagram of an illustrative physical token security system employing the secure data parser in accordance with one embodiment of the present invention.

[0061] FIGURE 27 is a block diagram of an illustrative arrangement in which the secure data parser is integrated into a system in accordance with one embodiment of the present invention.

[0062] FIGURE 28 is a block diagram of an illustrative data in motion system in accordance with one embodiment of the present invention.

[0063] FIGURE 29 is a block diamgram of another illustrative data in motion system in accordance with one embodiment of the present invention.

[0064] FIGURE 30-32 are block diagrams of an illustrative system having the secure data parser integrated in accordance with one embodiment of the present invention.

[0065] FIGURE 33 is a process flow diagram of an illustrative process for parsing and splitting data in accordance with one embodiment of the present invention.

[0066] FIGURE 34 is a process flow diagram of an illustrative process for restoring portions of data

into original data in accordance with one embodiment of the present invention.

[0067] FIGURE 35 is a process flow diagram of an illustrative process for splitting data at the bit level in accordance with one embodiment of the present invention.

[0068] FIGURE 36 is a process flow diagram of illustrative steps and features, that may be used in any suitable combination, with any suitable additions, deletions, or modifications in accordance with one embodiment of the present invention.

[0069] FIGURE 37 is a process flow diagram of illustrative steps and features, that may be used in any suitable combination, with any suitable additions, deletions, or modifications in accordance with one embodiment of the present invention.

[0070] FIGURE 38 is a simplified block diagram of the storage of key and data components within shares, that may be used in any suitable combination, with any suitable additions, deletions, or modifications in accordance with one embodiment of the present invention.

[0071] FIGURE 39 is a simplified block diagram of the storage of key and data components within shares using a workgroup key, that may be used in any suitable combination, with any suitable additions, deletions, or modifications in accordance with one embodiment of the present invention.

[0072] FIGURES 40A and 40B are simplified and illustrative process flow diagrams for header generation and data splitting for data in motion, that may be used in any suitable combination, with any suitable additions, deletions, or modifications in

accordance with one embodiment of the present invention.

[0073] FIGURE 41 is a simplified block diagram of an illustrative share format, that may be used in any suitable combination, with any suitable additions, deletions, or modifications in accordance with one embodiment of the present invention.

Detailed Description of the Invention

[0074] One aspect of the present invention is to provide a cryptographic system where one or more secure servers, or a trust engine, stores cryptographic keys and user authentication data. Users access the functionality of conventional cryptographic systems through network access to the trust engine, however, the trust engine does not release actual keys and other authentication data and therefore, the keys and data remain secure. This server-centric storage of keys and authentication data provides for user-independent security, portability, availability, and straightforwardness.

[0075] Because users can be confident in, or trust, the cryptographic system to perform user and document authentication and other cryptographic functions, a wide variety of functionality may be incorporated into the system. For example, the trust engine provider can ensure against agreement repudiation by, for example, authenticating the agreement participants, digitally signing the agreement on behalf of or for the participants, and storing a record of the agreement digitally signed by each participant. In addition, the cryptographic system may monitor agreements and

determine to apply varying degrees of authentication, based on, for example, price, user, vendor, geographic location, place of use, or the like.

[0076] To facilitate a complete understanding of the invention, the remainder of the detailed description describes the invention with reference to the figures, wherein like elements are referenced with like numerals throughout.

[0077] FIGURE 1 illustrates a block diagram of a cryptographic system 100, according to aspects of an embodiment of the invention. As shown in FIGURE 1, the cryptographic system 100 includes a user system 105, a trust engine 110, a certificate authority 115, and a vendor system 120, communicating through a communication link 125.

[0078] According to one embodiment of the invention, the user system 105 comprises a conventional general-purpose computer having one or more microprocessors, such as, for example, an Intel-based processor. Moreover, the user system 105 includes an appropriate operating system, such as, for example, an operating system capable of including graphics or windows, such as Windows, Unix, Linux, or the like. As shown in FIGURE 1, the user system 105 may include a biometric device 107. The biometric device 107 may advantageously capture a user's biometric and transfer the captured biometric to the trust engine 110. According to one embodiment of the invention, the biometric device may advantageously comprise a device having attributes and features similar to those disclosed in U.S. Patent Application No. 08/926,277, filed on September 5, 1997, entitled *"RELIEF OBJECT IMAGE GENERATOR,"* U.S. Patent Application

No. 09/558,634, filed on April 26, 2000, entitled *"IMAGING DEVICE FOR A RELIEF OBJECT AND SYSTEM AND METHOD OF USING THE IMAGE DEVICE,"* U.S. Patent Application No. 09/435,011, filed on November 5, 1999, entitled *"RELIEF OBJECT SENSOR ADAPTOR,"* and U.S. Patent Application No. 09/477,943, filed on January 5, 2000, entitled *"PLANAR OPTICAL IMAGE SENSOR AND SYSTEM FOR GENERATING AN ELECTRONIC IMAGE OF A RELIEF OBJECT FOR FINGERPRINT READING,"* all of which are owned by the instant assignee, and all of which are hereby incorporated by reference herein.

[0079] In addition, the user system 105 may connect to the communication link 125 through a conventional service provider, such as, for example, a dial up, digital subscriber line (DSL), cable modem, fiber connection, or the like. According to another embodiment, the user system 105 connects the communication link 125 through network connectivity such as, for example, a local or wide area network. According to one embodiment, the operating system includes a TCP/IP stack that handles all incoming and outgoing message traffic passed over the communication link 125.

[0080] Although the user system 105 is disclosed with reference to the foregoing embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize from the disclosure herein, a wide number of alternatives embodiments of the user system 105, including almost any computing device capable of sending or receiving information from another computer system. For example, the user system 105 may include, but is not limited to, a computer workstation, an interactive television, an

interactive kiosk, a personal mobile computing device, such as a digital assistant, mobile phone, laptop, or the like, a wireless communications device, a smartcard, an embedded computing device, or the like, which can interact with the communication link 125. In such alternative systems, the operating systems will likely differ and be adapted for the particular device. However, according to one embodiment, the operating systems advantageously continue to provide the appropriate communications protocols needed to establish communication with the communication link 125.

[0081] FIGURE 1 illustrates the trust engine 110. According to one embodiment, the trust engine 110 comprises one or more secure servers for accessing and storing sensitive information, which may be any type or form of data, such as, but not limited to text, audio, video, user authentication data and public and private cryptographic keys. According to one embodiment, the authentication data includes data designed to uniquely identify a user of the cryptographic system 100. For example, the authentication data may include a user identification number, one or more biometrics, and a series of questions and answers generated by the trust engine 110 or the user, but answered initially by the user at enrollment. The foregoing questions may include demographic data, such as place of birth, address, anniversary, or the like, personal data, such as mother's maiden name, favorite ice cream, or the like, or other data designed to uniquely identify the user. The trust engine 110 compares a user's authentication data associated with a current transaction, to the authentication data provided at an

earlier time, such as, for example, during enrollment. The trust engine 110 may advantageously require the user to produce the authentication data at the time of each transaction, or, the trust engine 110 may advantageously allow the user to periodically produce authentication data, such as at the beginning of a string of transactions or the logging onto a particular vendor website.

[0082] According to the embodiment where the user produces biometric data, the user provides a physical characteristic, such as, but not limited to, facial scan, hand scan, ear scan, iris scan, retinal scan, vascular pattern, DNA, a fingerprint, writing or speech, to the biometric device 107. The biometric device advantageously produces an electronic pattern, or biometric, of the physical characteristic. The electronic pattern is transferred through the user system 105 to the trust engine 110 for either enrollment or authentication purposes.

[0083] Once the user produces the appropriate authentication data and the trust engine 110 determines a positive match between that authentication data (current authentication data) and the authentication data provided at the time of enrollment (enrollment authentication data), the trust engine 110 provides the user with complete cryptographic functionality. For example, the properly authenticated user may advantageously employ the trust engine 110 to perform hashing, digitally signing, encrypting and decrypting (often together referred to only as encrypting), creating or distributing digital certificates, and the like. However, the private cryptographic keys used in the cryptographic functions will not be available

outside the trust engine 110, thereby ensuring the integrity of the cryptographic keys.

[0084] According to one embodiment, the trust engine 110 generates and stores cryptographic keys. According to another embodiment, at least one cryptographic key is associated with each user. Moreover, when the cryptographic keys include public-key technology, each private key associated with a user is generated within, and not released from, the trust engine 110. Thus, so long as the user has access to the trust engine 110, the user may perform cryptographic functions using his or her private or public key. Such remote access advantageously allows users to remain completely mobile and access cryptographic functionality through practically any Internet connection, such as cellular and satellite phones, kiosks, laptops, hotel rooms and the like.

[0085] According to another embodiment, the trust engine 110 performs the cryptographic functionality using a key pair generated for the trust engine 110. According to this embodiment, the trust engine 110 first authenticates the user, and after the user has properly produced authentication data matching the enrollment authentication data, the trust engine 110 uses its own cryptographic key pair to perform cryptographic functions on behalf of the authenticated user.

[0086] A skilled artisan will recognize from the disclosure herein that the cryptographic keys may advantageously include some or all of symmetric keys, public keys, and private keys. In addition, a skilled artisan will recognize from the disclosure herein that the foregoing keys may be implemented with a wide

number of algorithms available from commercial technologies, such as, for example, RSA, ELGAMAL, or the like.

[0087] FIGURE 1 also illustrates the certificate authority 115. According to one embodiment, the certificate authority 115 may advantageously comprise a trusted third-party organization or company that issues digital certificates, such as, for example, VeriSign, Baltimore, Entrust, or the like. The trust engine 110 may advantageously transmit requests for digital certificates, through one or more conventional digital certificate protocols, such as, for example, PKCS10, to the certificate authority 115. In response, the certificate authority 115 will issue a digital certificate in one or more of a number of differing protocols, such as, for example, PKCS7. According to one embodiment of the invention, the trust engine 110 requests digital certificates from several or all of the prominent certificate authorities 115 such that the trust engine 110 has access to a digital certificate corresponding to the certificate standard of any requesting party.

[0088] According to another embodiment, the trust engine 110 internally performs certificate issuances. In this embodiment, the trust engine 110 may access a certificate system for generating certificates and/or may internally generate certificates when they are requested, such as, for example, at the time of key generation or in the certificate standard requested at the time of the request. The trust engine 110 will be disclosed in greater detail below.

[0089] FIGURE 1 also illustrates the vendor system 120. According to one embodiment, the vendor system

120 advantageously comprises a Web server. Typical Web servers generally serve content over the Internet using one of several internet markup languages or document format standards, such as the Hyper-Text Markup Language (HTML) or the Extensible Markup Language (XML). The Web server accepts requests from browsers like Netscape and Internet Explorer and then returns the appropriate electronic documents. A number of server or client-side technologies can be used to increase the power of the Web server beyond its ability to deliver standard electronic documents. For example, these technologies include Common Gateway Interface (CGI) scripts, Secure Sockets Layer (SSL) security, and Active Server Pages (ASPs). The vendor system 120 may advantageously provide electronic content relating to commercial, personal, educational, or other transactions.

[0090] Although the vendor system 120 is disclosed with reference to the foregoing embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize from the disclosure herein that the vendor system 120 may advantageously comprise any of the devices described with reference to the user system 105 or combination thereof.

[0091] FIGURE 1 also illustrates the communication link 125 connecting the user system 105, the trust engine 110, the certificate authority 115, and the vendor system 120. According to one embodiment, the communication link 125 preferably comprises the Internet. The Internet, as used throughout this disclosure is a global network of computers. The structure of the Internet, which is well known to those

of ordinary skill in the art, includes a network backbone with networks branching from the backbone. These branches, in turn, have networks branching from them, and so on. Routers move information packets between network levels, and then from network to network, until the packet reaches the neighborhood of its destination. From the destination, the destination network's host directs the information packet to the appropriate terminal, or node. In one advantageous embodiment, the Internet routing hubs comprise domain name system (DNS) servers using Transmission Control Protocol/Internet Protocol (TCP/IP) as is well known in the art. The routing hubs connect to one or more other routing hubs via high-speed communication links.

[0092] One popular part of the Internet is the World Wide Web. The World Wide Web contains different computers, which store documents capable of displaying graphical and textual information. The computers that provide information on the World Wide Web are typically called "websites." A website is defined by an Internet address that has an associated electronic page. The electronic page can be identified by a Uniform Resource Locator (URL). Generally, an electronic page is a document that organizes the presentation of text, graphical images, audio, video, and so forth.

[0093] Although the communication link 125 is disclosed in terms of its preferred embodiment, one of ordinary skill in the art will recognize from the disclosure herein that the communication link 125 may include a wide range of interactive communications links. For example, the communication link 125 may include interactive television networks, telephone networks, wireless data transmission systems, two-way

cable systems, customized private or public computer networks, interactive kiosk networks, automatic teller machine networks, direct links, satellite or cellular networks, and the like.

[0094] FIGURE 2 illustrates a block diagram of the trust engine 110 of FIGURE 1 according to aspects of an embodiment of the invention. As shown in FIGURE 2, the trust engine 110 includes a transaction engine 205, a depository 210, an authentication engine 215, and a cryptographic engine 220. According to one embodiment of the invention, the trust engine 110 also includes mass storage 225. As further shown in FIGURE 2, the transaction engine 205 communicates with the depository 210, the authentication engine 215, and the cryptographic engine 220, along with the mass storage 225. In addition, the depository 210 communicates with the authentication engine 215, the cryptographic engine 220, and the mass storage 225. Moreover, the authentication engine 215 communicates with the cryptographic engine 220. According to one embodiment of the invention, some or all of the foregoing communications may advantageously comprise the transmission of XML documents to IP addresses that correspond to the receiving device. As mentioned in the foregoing, XML documents advantageously allow designers to create their own customized document tags, enabling the definition, transmission, validation, and interpretation of data between applications and between organizations. Moreover, some or all of the foregoing communications may include conventional SSL technologies.

[0095] According to one embodiment, the transaction engine 205 comprises a data routing device, such as a

conventional Web server available from Netscape, Microsoft, Apache, or the like. For example, the Web server may advantageously receive incoming data from the communication link 125. According to one embodiment of the invention, the incoming data is addressed to a front-end security system for the trust engine 110. For example, the front-end security system may advantageously include a firewall, an intrusion detection system searching for known attack profiles, and/or a virus scanner. After clearing the front-end security system, the data is received by the transaction engine 205 and routed to one of the depository 210, the authentication engine 215, the cryptographic engine 220, and the mass storage 225. In addition, the transaction engine 205 monitors incoming data from the authentication engine 215 and cryptographic engine 220, and routes the data to particular systems through the communication link 125. For example, the transaction engine 205 may advantageously route data to the user system 105, the certificate authority 115, or the vendor system 120.

[0096] According to one embodiment, the data is routed using conventional HTTP routing techniques, such as, for example, employing URLs or Uniform Resource Indicators (URIs). URIs are similar to URLs, however, URIs typically indicate the source of files or actions, such as, for example, executables, scripts, and the like. Therefore, according to the one embodiment, the user system 105, the certificate authority 115, the vendor system 120, and the components of the trust engine 210, advantageously include sufficient data within communication URLs or URIs for the transaction

engine 205 to properly route data throughout the cryptographic system.

[0097] Although the data routing is disclosed with reference to its preferred embodiment, a skilled artisan will recognize a wide number of possible data routing solutions or strategies. For example, XML or other data packets may advantageously be unpacked and recognized by their format, content, or the like, such that the transaction engine 205 may properly route data throughout the trust engine 110. Moreover, a skilled artisan will recognize that the data routing may advantageously be adapted to the data transfer protocols conforming to particular network systems, such as, for example, when the communication link 125 comprises a local network.

[0098] According to yet another embodiment of the invention, the transaction engine 205 includes conventional SSL encryption technologies, such that the foregoing systems may authenticate themselves, and vise-versa, with transaction engine 205, during particular communications. As will be used throughout this disclosure, the term "½ SSL" refers to communications where a server but not necessarily the client, is SSL authenticated, and the term "FULL SSL" refers to communications where the client and the server are SSL authenticated. When the instant disclosure uses the term "SSL", the communication may comprise ½ or FULL SSL.

[0099] As the transaction engine 205 routes data to the various components of the cryptographic system 100, the transaction engine 205 may advantageously create an audit trail. According to one embodiment, the audit trail includes a record of at least the type and format

of data routed by the transaction engine 205 throughout the cryptographic system 100. Such audit data may advantageously be stored in the mass storage 225.

[0100] FIGURE 2 also illustrates the depository 210. According to one embodiment, the depository 210 comprises one or more data storage facilities, such as, for example, a directory server, a database server, or the like. As shown in FIGURE 2, the depository 210 stores cryptographic keys and enrollment authentication data. The cryptographic keys may advantageously correspond to the trust engine 110 or to users of the cryptographic system 100, such as the user or vendor. The enrollment authentication data may advantageously include data designed to uniquely identify a user, such as, user ID, passwords, answers to questions, biometric data, or the like. This enrollment authentication data may advantageously be acquired at enrollment of a user or another alternative later time. For example, the trust engine 110 may include periodic or other renewal or reissue of enrollment authentication data.

[0101] According to one embodiment, the communication from the transaction engine 205 to and from the authentication engine 215 and the cryptographic engine 220 comprises secure communication, such as, for example conventional SSL technology. In addition, as mentioned in the foregoing, the data of the communications to and from the depository 210 may be transferred using URLs, URIs, HTTP or XML documents, with any of the foregoing advantageously having data requests and formats embedded therein.

[0102] As mentioned above, the depository 210 may advantageously comprises a plurality of secure data

storage facilities. In such an embodiment, the secure data storage facilities may be configured such that a compromise of the security in one individual data storage facility will not compromise the cryptographic keys or the authentication data stored therein. For example, according to this embodiment, the cryptographic keys and the authentication data are mathematically operated on so as to statistically and substantially randomize the data stored in each data storage facility. According to one embodiment, the randomization of the data of an individual data storage facility renders that data undecipherable. Thus, compromise of an individual data storage facility produces only a randomized undecipherable number and does not compromise the security of any cryptographic keys or the authentication data as a whole.

[0103] FIGURE 2 also illustrates the trust engine 110 including the authentication engine 215. According to one embodiment, the authentication engine 215 comprises a data comparator configured to compare data from the transaction engine 205 with data from the depository 210. For example, during authentication, a user supplies current authentication data to the trust engine 110 such that the transaction engine 205 receives the current authentication data. As mentioned in the foregoing, the transaction engine 205 recognizes the data requests, preferably in the URL or URI, and routes the authentication data to the authentication engine 215. Moreover, upon request, the depository 210 forwards enrollment authentication data corresponding to the user to the authentication engine 215. Thus, the authentication engine 215 has both the current

authentication data and the enrollment authentication data for comparison.

[0104] According to one embodiment, the communications to the authentication engine comprise secure communications, such as, for example, SSL technology. Additionally, security can be provided within the trust engine 110 components, such as, for example, super-encryption using public key technologies. For example, according to one embodiment, the user encrypts the current authentication data with the public key of the authentication engine 215. In addition, the depository 210 also encrypts the enrollment authentication data with the public key of the authentication engine 215. In this way, only the authentication engine's private key can be used to decrypt the transmissions.

[0105] As shown in FIGURE 2, the trust engine 110 also includes the cryptographic engine 220. According to one embodiment, the cryptographic engine comprises a cryptographic handling module, configured to advantageously provide conventional cryptographic functions, such as, for example, public-key infrastructure (PKI) functionality. For example, the cryptographic engine 220 may advantageously issue public and private keys for users of the cryptographic system 100. In this manner, the cryptographic keys are generated at the cryptographic engine 220 and forwarded to the depository 210 such that at least the private cryptographic keys are not available outside of the trust engine 110. According to another embodiment, the cryptographic engine 220 randomizes and splits at least the private cryptographic key data, thereby storing only the randomized split data. Similar to the

splitting of the enrollment authentication data, the splitting process ensures the stored keys are not available outside the cryptographic engine 220. According to another embodiment, the functions of the cryptographic engine can be combined with and performed by the authentication engine 215.

[0106] According to one embodiment, communications to and from the cryptographic engine include secure communications, such as SSL technology. In addition, XML documents may advantageously be employed to transfer data and/or make cryptographic function requests.

[0107] FIGURE 2 also illustrates the trust engine 110 having the mass storage 225. As mentioned in the foregoing, the transaction engine 205 keeps data corresponding to an audit trail and stores such data in the mass storage 225. Similarly, according to one embodiment of the invention, the depository 210 keeps data corresponding to an audit trail and stores such data in the mass storage device 225. The depository audit trail data is similar to that of the transaction engine 205 in that the audit trail data comprises a record of the requests received by the depository 210 and the response thereof. In addition, the mass storage 225 may be used to store digital certificates having the public key of a user contained therein.

[0108] Although the trust engine 110 is disclosed with reference to its preferred and alternative embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize in the disclosure herein, a wide number of alternatives for the trust engine 110. For example, the trust engine 110, may advantageously perform only

authentication, or alternatively, only some or all of the cryptographic functions, such as data encryption and decryption. According to such embodiments, one of the authentication engine 215 and the cryptographic engine 220 may advantageously be removed, thereby creating a more straightforward design for the trust engine 110. In addition, the cryptographic engine 220 may also communicate with a certificate authority such that the certificate authority is embodied within the trust engine 110. According to yet another embodiment, the trust engine 110 may advantageously perform authentication and one or more cryptographic functions, such as, for example, digital signing.

[0109] FIGURE 3 illustrates a block diagram of the transaction engine 205 of FIGURE 2, according to aspects of an embodiment of the invention. According to this embodiment, the transaction engine 205 comprises an operating system 305 having a handling thread and a listening thread. The operating system 305 may advantageously be similar to those found in conventional high volume servers, such as, for example, Web servers available from Apache. The listening thread monitors the incoming communication from one of the communication link 125, the authentication engine 215, and the cryptographic engine 220 for incoming data flow. The handling thread recognizes particular data structures of the incoming data flow, such as, for example, the foregoing data structures, thereby routing the incoming data to one of the communication link 125, the depository 210, the authentication engine 215, the cryptographic engine 220, or the mass storage 225. As shown in FIGURE 3, the incoming and outgoing data may

advantageously be secured through, for example, SSL technology.

[0110] FIGURE 4 illustrates a block diagram of the depository 210 of FIGURE 2 according to aspects of an embodiment of the invention. According to this embodiment, the depository 210 comprises one or more lightweight directory access protocol (LDAP) servers. LDAP directory servers are available from a wide variety of manufacturers such as Netscape, ISO, and others. FIGURE 4 also shows that the directory server preferably stores data 405 corresponding to the cryptographic keys and data 410 corresponding to the enrollment authentication data. According to one embodiment, the depository 210 comprises a single logical memory structure indexing authentication data and cryptographic key data to a unique user ID. The single logical memory structure preferably includes mechanisms to ensure a high degree of trust, or security, in the data stored therein. For example, the physical location of the depository 210 may advantageously include a wide number of conventional security measures, such as limited employee access, modern surveillance systems, and the like. In addition to, or in lieu of, the physical securities, the computer system or server may advantageously include software solutions to protect the stored data. For example, the depository 210 may advantageously create and store data 415 corresponding to an audit trail of actions taken. In addition, the incoming and outgoing communications may advantageously be encrypted with public key encryption coupled with conventional SSL technologies.

[0111] According to another embodiment, the depository 210 may comprise distinct and physically separated data storage facilities, as disclosed further with reference to FIGURE 7.

[0112] FIGURE 5 illustrates a block diagram of the authentication engine 215 of FIGURE 2 according to aspects of an embodiment of the invention. Similar to the transaction engine 205 of FIGURE 3, the authentication engine 215 comprises an operating system 505 having at least a listening and a handling thread of a modified version of a conventional Web server, such as, for example, Web servers available from Apache. As shown in FIGURE 5, the authentication engine 215 includes access to at least one private key 510. The private key 510 may advantageously be used for example, to decrypt data from the transaction engine 205 or the depository 210, which was encrypted with a corresponding public key of the authentication engine 215.

[0113] FIGURE 5 also illustrates the authentication engine 215 comprising a comparator 515, a data splitting module 520, and a data assembling module 525. According to the preferred embodiment of the invention, the comparator 515 includes technology capable of comparing potentially complex patterns related to the foregoing biometric authentication data. The technology may include hardware, software, or combined solutions for pattern comparisons, such as, for example, those representing finger print patterns or voice patterns. In addition, according to one embodiment, the comparator 515 of the authentication engine 215 may advantageously compare conventional hashes of documents in order to render a comparison

result. According to one embodiment of the invention, the comparator 515 includes the application of heuristics 530 to the comparison. The heuristics 530 may advantageously address circumstances surrounding an authentication attempt, such as, for example, the time of day, IP address or subnet mask, purchasing profile, email address, processor serial number or ID, or the like.

[0114] Moreover, the nature of biometric data comparisons may result in varying degrees of confidence being produced from the matching of current biometric authentication data to enrollment data. For example, unlike a traditional password which may only return a positive or negative match, a fingerprint may be determined to be a partial match, e.g. a 90% match, a 75% match, or a 10% match, rather than simply being correct or incorrect. Other biometric identifiers such as voice print analysis or face recognition may share this property of probabilistic authentication, rather than absolute authentication.

[0115] When working with such probabilistic authentication or in other cases where an authentication is considered less than absolutely reliable, it is desirable to apply the heuristics 530 to determine whether the level of confidence in the authentication provided is sufficiently high to authenticate the transaction which is being made.

[0116] It will sometimes be the case that the transaction at issue is a relatively low value transaction where it is acceptable to be authenticated to a lower level of confidence. This could include a transaction which has a low dollar value associated

with it (e.g., a $10 purchase) or a transaction with low risk (e.g., admission to a members-only web site).

[0117] Conversely, for authenticating other transactions, it may be desirable to require a high degree of confidence in the authentication before allowing the transaction to proceed. Such transactions may include transactions of large dollar value (e.g., signing a multi-million dollar supply contract) or transaction with a high risk if an improper authentication occurs (e.g., remotely logging onto a government computer).

[0118] The use of the heuristics 530 in combination with confidence levels and transactions values may be used as will be described below to allow the comparator to provide a dynamic context-sensitive authentication system.

[0119] According to another embodiment of the invention, the comparator 515 may advantageously track authentication attempts for a particular transaction. For example, when a transaction fails, the trust engine 110 may request the user to re-enter his or her current authentication data. The comparator 515 of the authentication engine 215 may advantageously employ an attempt limiter 535 to limit the number of authentication attempts, thereby prohibiting brute-force attempts to impersonate a user's authentication data. According to one embodiment, the attempt limiter 535 comprises a software module monitoring transactions for repeating authentication attempts and, for example, limiting the authentication attempts for a given transaction to three. Thus, the attempt limiter 535 will limit an automated attempt to impersonate an individual's authentication data to, for

example, simply three "guesses." Upon three failures, the attempt limiter 535 may advantageously deny additional authentication attempts. Such denial may advantageously be implemented through, for example, the comparator 515 returning a negative result regardless of the current authentication data being transmitted. On the other hand, the transaction engine 205 may advantageously block any additional authentication attempts pertaining to a transaction in which three attempts have previously failed.

[0120] The authentication engine 215 also includes the data splitting module 520 and the data assembling module 525. The data splitting module 520 advantageously comprises a software, hardware, or combination module having the ability to mathematically operate on various data so as to substantially randomize and split the data into portions. According to one embodiment, original data is not recreatable from an individual portion. The data assembling module 525 advantageously comprises a software, hardware, or combination module configured to mathematically operate on the foregoing substantially randomized portions, such that the combination thereof provides the original deciphered data. According to one embodiment, the authentication engine 215 employs the data splitting module 520 to randomize and split enrollment authentication data into portions, and employs the data assembling module 525 to reassemble the portions into usable enrollment authentication data.

[0121] FIGURE 6 illustrates a block diagram of the cryptographic engine 220 of the trust engine 200 of FIGURE 2 according to aspects of one embodiment of the invention. Similar to the transaction engine 205 of

FIGURE 3, the cryptographic engine 220 comprises an operating system 605 having at least a listening and a handling thread of a modified version of a conventional Web server, such as, for example, Web servers available from Apache. As shown in FIGURE 6, the cryptographic engine 220 comprises a data splitting module 610 and a data assembling module 620 that function similar to those of FIGURE 5. However, according to one embodiment, the data splitting module 610 and the data assembling module 620 process cryptographic key data, as opposed to the foregoing enrollment authentication data. Although, a skilled artisan will recognize from the disclosure herein that the data splitting module 910 and the data splitting module 620 may be combined with those of the authentication engine 215.

[0122] The cryptographic engine 220 also comprises a cryptographic handling module 625 configured to perform one, some or all of a wide number of cryptographic functions. According to one embodiment, the cryptographic handling module 625 may comprise software modules or programs, hardware, or both. According to another embodiment, the cryptographic handling module 625 may perform data comparisons, data parsing, data splitting, data separating, data hashing, data encryption or decryption, digital signature verification or creation, digital certificate generation, storage, or requests, cryptographic key generation, or the like. Moreover, a skilled artisan will recognize from the disclosure herein that the cryptographic handling module 825 may advantageously comprises a public-key infrastructure, such as Pretty Good Privacy (PGP), an RSA-based public-key system, or a wide number of alternative key management systems.

In addition, the cryptographic handling module 625 may perform public-key encryption, symmetric-key encryption, or both. In addition to the foregoing, the cryptographic handling module 625 may include one or more computer programs or modules, hardware, or both, for implementing seamless, transparent, interoperability functions.

[0123] A skilled artisan will also recognize from the disclosure herein that the cryptographic functionality may include a wide number or variety of functions generally relating to cryptographic key management systems.

[0124] FIGURE 7 illustrates a simplified block diagram of a depository system 700 according to aspects of an embodiment of the invention. As shown in FIGURE 7, the depository system 700 advantageously comprises multiple data storage facilities, for example, data storage facilities D1, D2, D3, and D4. However, it is readily understood by those of ordinary skill in the art that the depository system may have only one data storage facility. According to one embodiment of the invention, each of the data storage facilities D1 through D4 may advantageously comprise some or all of the elements disclosed with reference to the depository 210 of FIGURE 4. Similar to the depository 210, the data storage facilities D1 through D4 communicate with the transaction engine 205, the authentication engine 215, and the cryptographic engine 220, preferably through conventional SSL. Communication links transferring, for example, XML documents. Communications from the transaction engine 205 may advantageously include requests for data, wherein the request is advantageously broadcast to the IP address

of each data storage facility D1 through D4. On the other hand, the transaction engine 205 may broadcast requests to particular data storage facilities based on a wide number of criteria, such as, for example, response time, server loads, maintenance schedules, or the like.

[0125] In response to requests for data from the transaction engine 205, the depository system 700 advantageously forwards stored data to the authentication engine 215 and the cryptographic engine 220. The respective data assembling modules receive the forwarded data and assemble the data into useable formats. On the other hand, communications from the authentication engine 215 and the cryptographic engine 220 to the data storage facilities D1 through D4 may include the transmission of sensitive data to be stored. For example, according to one embodiment, the authentication engine 215 and the cryptographic engine 220 may advantageously employ their respective data splitting modules to divide sensitive data into undecipherable portions, and then transmit one or more undecipherable portions of the sensitive data to a particular data storage facility.

[0126] According to one embodiment, each data storage facility, D1 through D4, comprises a separate and independent storage system, such as, for example, a directory server. According to another embodiment of the invention, the depository system 700 comprises multiple geographically separated independent data storage systems. By distributing the sensitive data into distinct and independent storage facilities D1 through D4, some or all of which may be advantageously geographically separated, the depository system 700

provides redundancy along with additional security measures. For example, according to one embodiment, only data from two of the multiple data storage facilities, D1 through D4, are needed to decipher and reassemble the sensitive data. Thus, as many as two of the four data storage facilities D1 through D4 may be inoperative due to maintenance, system failure, power failure, or the like, without affecting the functionality of the trust engine 110. In addition, because, according to one embodiment, the data stored in each data storage facility is randomized and undecipherable, compromise of any individual data storage facility does not necessarily compromise the sensitive data. Moreover, in the embodiment having geographical separation of the data storage facilities, a compromise of multiple geographically remote facilities becomes increasingly difficult. In fact, even a rogue employee will be greatly challenged to subvert the needed multiple independent geographically remote data storage facilities.

[0127] Although the depository system 700 is disclosed with reference to its preferred and alternative embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize from the disclosure herein, a wide number of alternatives for the depository system 700. For example, the depository system 700 may comprise one, two or more data storage facilities. In addition, sensitive data may be mathematically operated such that portions from two or more data storage facilities are needed to reassemble and decipher the sensitive data.

[0128] As mentioned in the foregoing, the authentication engine 215 and the cryptographic engine

220 each include a data splitting module 520 and 610, respectively, for splitting any type or form of sensitive data, such as, for example, text, audio, video, the authentication data and the cryptographic key data. FIGURE 8 illustrates a flowchart of a data splitting process 800 performed by the data splitting module according to aspects of an embodiment of the invention. As shown in FIGURE 8, the data splitting process 800 begins at step 805 when sensitive data "S" is received by the data splitting module of the authentication engine 215 or the cryptographic engine 220. Preferably, in step 810, the data splitting module then generates a substantially random number, value, or string or set of bits, "A." For example, the random number A may be generated in a wide number of varying conventional techniques available to one of ordinary skill in the art, for producing high quality random numbers suitable for use in cryptographic applications. In addition, according to one embodiment, the random number A comprises a bit length which may be any suitable length, such as shorter, longer or equal to the bit length of the sensitive data, S.

[0129] In addition, in step 820 the data splitting process 800 generates another statistically random number "C." According to the preferred embodiment, the generation of the statistically random numbers A and C may advantageously be done in parallel. The data splitting module then combines the numbers A and C with the sensitive data S such that new numbers "B" and "D" are generated. For example, number B may comprise the binary combination of A XOR S and number D may comprise the binary combination of C XOR S. The XOR function,

or the "exclusive-or" function, is well known to those of ordinary skill in the art. The foregoing combinations preferably occur in steps 825 and 830, respectively, and, according to one embodiment, the foregoing combinations also occur in parallel. The data splitting process 800 then proceeds to step 835 where the random numbers A and C and the numbers B and D are paired such that none of the pairings contain sufficient data, by themselves, to reorganize and decipher the original sensitive data S. For example, the numbers may be paired as follows: AC, AD, BC, and BD. According to one embodiment, each of the foregoing pairings is distributed to one of the depositories D1 through D4 of FIGURE 7. According to another embodiment, each of the foregoing pairings is randomly distributed to one of the depositories D1 through D4. For example, during a first data splitting process 800, the pairing AC may be sent to depository D2, through, for example, a random selection of D2's IP address. Then, during a second data splitting process 800, the pairing AC may be sent to depository D4, through, for example, a random selection of D4's IP address. In addition, the pairings may all be stored on one depository, and may be stored in separate locations on said depository.

[0130] Based on the foregoing, the data splitting process 800 advantageously places portions of the sensitive data in each of the four data storage facilities D1 through D4, such that no single data storage facility D1 through D4 includes sufficient encrypted data to recreate the original sensitive data S. As mentioned in the foregoing, such randomization of the data into individually unusable encrypted

portions increases security and provides for maintained trust in the data even if one of the data storage facilities, D1 through D4, is compromised.

[0131] Although the data splitting process 800 is disclosed with reference to its preferred embodiment, the invention is not intended to be limited thereby. Rather a skilled artisan will recognize from the disclosure herein, a wide number of alternatives for the data splitting process 800. For example, the data splitting process may advantageously split the data into two numbers, for example, random number A and number B and, randomly distribute A and B through two data storage facilities. Moreover, the data splitting process 800 may advantageously split the data among a wide number of data storage facilities through generation of additional random numbers. The data may be split into any desired, selected, predetermined, or randomly assigned size unit, including but not limited to, a bit, bits, bytes, kilobytes, megabytes or larger, or any combination or sequence of sizes. In addition, varying the sizes of the data units resulting from the splitting process may render the data more difficult to restore to a useable form, thereby increasing security of sensitive data. It is readily apparent to those of ordinary skill in the art that the split data unit sizes may be a wide variety of data unit sizes or patterns of sizes or combinations of sizes. For example, the data unit sizes may be selected or predetermined to be all of the same size, a fixed set of different sizes, a combination of sizes, or randomly generates sizes. Similarly, the data units may be distributed into one or more shares according to a fixed or predetermined data unit size, a pattern or

combination of data unit sizes, or a randomly generated data unit size or sizes per share.

[0132] As mentioned in the foregoing, in order to recreate the sensitive data S, the data portions need to be derandomized and reorganized. This process may advantageously occur in the data assembling modules, 525 and 620, of the authentication engine 215 and the cryptographic engine 220, respectively. The data assembling module, for example, data assembly module 525, receives data portions from the data storage facilities D1 through D4, and reassembles the data into useable form. For example, according to one embodiment where the data splitting module 520 employed the data splitting process 800 of FIGURE 8, the data assembling module 525 uses data portions from at least two of the data storage facilities D1 through D4 to recreate the sensitive data S. For example, the pairings of AC, AD, BC, and BD, were distributed such that any two provide one of A and B, or, C and D. Noting that S = A XOR B or S = C XOR D indicates that when the data assembling module receives one of A and B, or, C and D, the data assembling module 525 can advantageously reassemble the sensitive data S. Thus, the data assembling module 525 may assemble the sensitive data S, when, for example, it receives data portions from at least the first two of the data storage facilities D1 through D4 to respond to an assemble request by the trust engine 110.

[0133] Based on the above data splitting and assembling processes, the sensitive data S exists in usable format only in a limited area of the trust engine 110. For example, when the sensitive data S includes enrollment authentication data, usable, nonrandomized enrollment authentication data is

available only in the authentication engine 215. Likewise, when the sensitive data S includes private cryptographic key data, usable, nonrandomized private cryptographic key data is available only in the cryptographic engine 220.

[0134] Although the data splitting and assembling processes are disclosed with reference to their preferred embodiments, the invention is not intended to be limited thereby. Rather, a skilled artisan will recognize from the disclosure herein, a wide number of alternatives for splitting and reassembling the sensitive data S. For example, public-key encryption may be used to further secure the data at the data storage facilities D1 through D4. In addition, it is readily apparent to those of ordinary skill in the art that the data splitting module described herein is also a separate and distinct embodiment of the present invention that may be incorporated into, combined with or otherwise made part of any pre-existing computer systems, software suites, database, or combinations thereof, or other embodiments of the present invention, such as the trust engine, authentication engine, and transaction engine disclosed and described herein.

[0135] FIGURE 9A illustrates a data flow of an enrollment process 900 according to aspects of an embodiment of the invention. As shown in FIGURE 9A, the enrollment process 900 begins at step 905 when a user desires to enroll with the trust engine 110 of the cryptographic system 100. According to this embodiment, the user system 105 advantageously includes a client-side applet, such as a Java-based, that queries the user to enter enrollment data, such as demographic data and enrollment authentication data.