

Filed on behalf of Google, LLC by:

Paper No. ____

Elisabeth H. Hunt, Reg. No. 67,336

Richard F. Giunta, Reg. No. 36,149

Thomas A. Franklin, Reg. No. 63,456

WOLF, GREENFIELD & SACKS, P.C.

600 Atlantic Avenue

Boston, MA 02210

(617) 646-8000 Phone

(617) 646-8646 Fax

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

GOOGLE LLC,
Petitioner,

v.

SECURITY FIRST INNOVATIONS, LLC
Patent Owner.

Case No. IPR2024-00213

Patent No. 9,338,140

**PETITION FOR INTER PARTES REVIEW
UNDER 35 U.S.C. §§ 311-319 AND 37 C.F.R. § 42.1 et seq**

TABLE OF CONTENTS

MANDATORY NOTICES.....	ix
A. Real Party-In-Interest.....	ix
B. Related Matters	ix
1. United States Patent & Trademark Office.....	ix
2. U.S. District Court for the Eastern District of Virginia.....	x
C. Counsel and Service Information - § 42.8(b)(3) and (4)	x
I. INTRODUCTION	1
II. STANDING.....	1
III. UNPATENTABILITY GROUNDS.....	2
IV. THE '140 PATENT.....	2
A. Reproduced Prior-Art Disclosure.....	2
B. Additional Disclosure.....	3
C. Prosecution History	4
V. PERSON OF ORDINARY SKILL IN THE ART (“POSA”).....	5
VI. CLAIM INTERPRETATION	5
VII. GROUND 1: OPHIR AND TAKEDA RENDER CLAIMS 1-6, 8, 10 OBVIOUS.....	6
A. Ophir.....	6
B. Takeda	10
C. Ophir-Takeda Combination	17
1. Reasons to Combine	17
a. Increased Reliability	17
b. Simple Substitution	18
c. Use of Known Technique to Improve Similar Devices in the Same Way	22
2. Reasonable Expectation of Success.....	23
3. Resulting Combination	24
D. Ophir-Takeda Meets All Limitations of Claims 1-6, 8, 10.....	25

1. Claim 1.....	25
a. 1[PRE]: A secure storage network comprising:	25
b. [1A]:	27
i. a plurality of physical storage devices.....	27
ii. storing thereon a plurality of shares...associated with at least one session key used to secure a dataset	28
c. [1B]: a secure storage system configured to:	32
i. [1B-1]: present to a client device a virtual disk, the virtual disk comprising a directory mapped to the plurality of physical storage devices such that physical locations of the shares are hidden from the client device;	32
(1) Block Services	35
(2) File Services	37
ii. [1B-2]: generate the plurality of shares for storage on the plurality of physical storage devices by performing a securing operation on the dataset received from the client device and distributing the dataset in the shares;	39
iii. [1B-3]: include with each of the plurality of shares data indicative of the at least one session key used to secure the dataset; and	40
iv. [1B-4]: reconstitute the dataset from at least a portion of the plurality of shares stored on the physical storage devices in response to a request from the client device for information in the dataset.	43
2. Claim 2: ...wherein the secure storage system is further configured to generate the plurality of shares by performing a cryptographic operation on the dataset.	43
3. Claim 3: ...further comprising a key management server configured to store workgroup keys.	44
4. Claim 4: ...wherein the secure storage system is further configured to provide access to the reconstituted dataset.	45

5. Claim 5: ...wherein the secure storage system is further configured to establish a secure connection between the client device and the secure storage system.	47
6. Claim 6: ...wherein the shares contain a substantially random distribution of the received dataset.....	48
7. Claim 8: ...wherein the secure storage system is configured to secure the dataset by encrypting the dataset using the at least one session key.	49
8. Claim 10: ...wherein the data indicative of the at least one session key comprises: a portion of the at least one session key or data indicative of a portion of the at least one session key.....	49
VIII. GROUND 2: OPHIR, TAKEDA, AND BIRRELL RENDER CLAIM 7 OBVIOUS.....	50
A. Birrell	50
B. Ophir-Takeda-Birrell Combination.....	50
C. Claim 7: ...wherein the secure storage system is configured to reconstitute the received dataset from fewer than all of the shares	51
IX. GROUND 3: OPHIR, TAKEDA, AND DICKINSON RENDER CLAIM 9 OBVIOUS.....	52
A. Dickinson	52
B. Ophir-Takeda-Dickinson Combination.....	52
C. Claim 9: ...wherein the secure storage system is configured to secure the dataset by splitting the dataset using the at least one session key.....	53
X. GROUND 4: DICKINSON AND TAKEDA RENDER CLAIMS 1-10 OBVIOUS.....	54
A. Dickinson	54
B. Takeda	59
C. Dickinson-Takeda Combination	59
1. Reasons to Modify Dickinson	60
2. Reasonable Expectation of Success.....	63
3. Resulting Combination	64

D. Dickinson-Takeda Meets All Limitations of Claims 1-10.....	67
1. Claim 1.....	67
a. 1[PRE].....	67
b. [1A]	68
c. [1B]	70
i. [1B-1].....	71
ii. [1B-2].....	74
iii. [1B-3].....	74
iv. [1B-4].....	76
2. Claim 2.....	76
3. Claim 3.....	76
4. Claim 4.....	77
5. Claim 5.....	77
6. Claim 6.....	77
7. Claim 7.....	77
8. Claim 8.....	78
9. Claim 9.....	78
10. Claim 10.....	78
XI. NO BASIS FOR DISCRETIONARY DENIAL	79
A. Section 314(a)	79
1. Stay Potential	79
2. Trial Timing.....	79
3. Litigation Investment.....	80
4. Issue Overlap	80
5. Litigation Defendants	80
6. Other Circumstances.....	80
B. Section 325(d)	81
XII. CONCLUSION.....	81
XIII. APPENDIX: CLAIM LISTING	83

TABLE OF AUTHORITIES

Cases

<i>10X Genomics, Inc. v. Bio-Rad Labs.</i> , IPR2020-00086, Paper 8 (Apr. 27, 2020).....	5
<i>Apple Inc. v. Fintiv, Inc.</i> , IPR2020-00019, Paper 11 (Mar. 20, 2020)	79, 80
<i>Dish Network LLC v. Broadband iTV, Inc.</i> , IPR2020-01332, Paper 14 (Jan. 27, 2021).....	80
<i>KSR Int’l v. Teleflex Inc.</i> , 550 U.S. 398 (2007).....	17, 19, 22, 62
<i>Markforged v. Continuous Composites</i> , IPR2022-00679, Paper 7 (Oct. 25, 2022)	80
<i>Nidec Motor Corp. v. Zhongshan Broad Ocean Motor Co.</i> , 868 F.3d 1013 (Fed. Cir. 2017)	5
<i>PEAG LLC d/b/a JLab Audio v. VARTA Microbattery GmbH</i> , IPR2020-01214, Paper 8 (Jan. 6, 2021).....	79
<i>Sand Revolution II v. Continental Intermodal Group–Trucking</i> , IPR2019-01393, Paper 24 (June 16, 2020).....	79, 80
<i>SHDS v. TruInject</i> , IPR2020-00937, Paper 11 (Nov. 17, 2020)	81
<i>ShenZhen Apaltek Co., Ltd. v. Asetek Danmark A/S</i> , IPR2022-01317, Paper 7 (Feb. 6, 2023)	81

Regulations

37 C.F.R. § 42.100(b)	5
37 C.F.R. § 42.104(a).....	1

Statutes

35 U.S.C. § 102(a)	2
35 U.S.C. § 102(e)	2
35 U.S.C. § 103	2
35 U.S.C. § 314(a)	79
35 U.S.C. § 325(d)	81

APPENDIX LISTING OF EXHIBITS

Exhibit	Description
1001	U.S. Patent No. 9,338,140 (“140 patent”)
1002	Prosecution History of U.S. Patent No. 9,338,140
1003	Declaration of Samrat Bhattacharjee, Ph.D. (“Bhattacharjee”)
1004	Curriculum Vitae of Samrat Bhattacharjee, Ph.D.
1005	International Application Publication WO 2001/022322 (“Dickinson”)
1006	RESERVED
1007	U.S. Patent Application Publication 2004/0143733 (“Ophir”)
1008	U.S. Patent Application Publication 2001/0053221 (“Takeda”)
1009	U.S. Patent No. 7,187,771
1010	U.S. Patent No. 7,571,320
1011	European Patent Application Publication EP 1 063 811
1012	U.S. Patent No. 6,151,659
1013	U.S. Patent No. 5,526,507
1014	U.S. Patent No. 6,065,096
1015	U.S. Patent No. 7,941,695
1016	U.S. Patent No. 8,352,518
1017	U.S. Patent Application Publication 2002/0174295
1018	U.S. Patent Application Publication 2002/0019908
1019	U.S. Patent No. 7,043,614
1020	U.S. Patent No. 6,973,455
1021	U.S. Patent No. 6,697,846
1022	U.S. Patent No. 6,324,648
1023	U.S. Patent Application Publication 2002/0019932
1024	U.S. Patent No. 5,825,890
1025	RESERVED
1026	RESERVED
1027	U.S. Patent Application Publication 2003/0028493
1028	U.S. Patent Application Publication 2002/0042859
1029	U.S. Patent Application Publication 2005/0216754
1030	U.S. Patent Application Publication 2003/0191938
1031	U.S. Patent Application Publication 2003/0167408
1032	RESERVED
1033	U.S. Patent No. 7,529,834 (“Birrell”)
1034	U.S. Patent No. 5,483,598
1035	RESERVED
1036	RESERVED

Exhibit	Description
1037	Merriam Webster's Collegiate Dictionary, 207, 381, 888 (10th ed. 1997)
1038	U.S. Patent No. 6,356,915
1039	U.S. Patent No. 4,067,058
1040	Redline Comparison of International Application Publication WO 2001/022322 to U.S. Patent No. 9,338,140
1041	Google LLC's Proposed Claim Terms for Construction in <i>Security First Innovations, LLC v. Google, LLC</i> , 2:23-cv-00097-JWK-RJK (Nov. 11, 2023)
1042	Letter from A. Mayron to D. Perlson re Preliminary Proposed Constructions in <i>Security First Innovations, LLC v. Google, LLC</i> , 2:23-cv-00097-JWK-RJK (Nov. 11, 2023)
1043	Microsoft Computer Dictionary, 99, 192, 362, 407 (5th ed. 2002)
1044	Security First Innovation's 2023-08-14 Infringement Contentions, Chart D in <i>Security First Innovations, LLC v. Google LLC</i> , 2-23-cv-00097 (E.D. Va.)
1045	Rule 16(b) Scheduling Order in <i>Security First Innovations, LLC v. Google, LLC</i> , 2:23-cv-00097-JWK-RJK, Dkt. 54 (July 20, 2023)

MANDATORY NOTICES

A. Real Party-In-Interest

Petitioner Google LLC is the Real Party-in-Interest.¹

B. Related Matters

1. United States Patent & Trademark Office

The application from which U.S. Patent No. 9,338,140 issued is a Continuation of U.S. Patent Application No. 11/258,839 (U.S. Patent No. 8,266,438), filed October 25, 2005, which claims priority to U.S. Provisional Application No. 60/622,146 filed on October 25, 2004, and U.S. Provisional Application No. 60/718,185 filed on September 16, 2005.

The following U.S. patent applications claim the benefit of priority to U.S. Patent No. 9,338,140:

- (i) U.S. Patent Application No. 17/111,262, filed on December 3, 2020;
- (ii) U.S. Patent Application No. 15/977,748 (U.S. Patent No. 11,178,116), filed on May 11, 2018; (iii) U.S. Patent Application No. 13/915,081 (U.S. Patent No. 9,992,170), filed June 11, 2013.

¹ Google LLC is a subsidiary of XXVI Holdings Inc., which is a subsidiary of Alphabet Inc. XXVI Holdings Inc. and Alphabet Inc. are not real parties-in-interest to this proceeding.

2. U.S. District Court for the Eastern District of Virginia

(i) *Security First Innovations, LLC v. Google LLC f/k/a Google Inc.*, Case No. 1:23-cv-00329; (ii) *Security First Innovations, LLC v. Google LLC*, Case No. 2:23-cv-00097.

C. Counsel and Service Information - § 42.8(b)(3) and (4)

Lead Counsel	Elisabeth H. Hunt, Reg. No. 67,336
Backup Counsel	Richard F. Giunta, Reg. No. 36,149 Thomas A. Franklin, Reg. No. 63,456
Service Information	<u>E-mail:</u> EHunt-PTAB@wolfgreenfield.com RGiunta-PTAB@wolfgreenfield.com TFranklin-PTAB@wolfgreenfield.com <u>Post and hand delivery:</u> Wolf, Greenfield & Sacks, P.C. 600 Atlantic Avenue Boston, MA 02210-2206 <u>Telephone:</u> 617-646-8000 <u>Facsimile:</u> 617-646-8646

A power of attorney is submitted with the Petition. Counsel for Petitioner consents to service of all documents via electronic mail.

I. INTRODUCTION

Petitioner requests *inter partes* review of claims 1-10 (the “challenged claims”) of Patent 9,338,140 (“the ’140 patent”) (EX1001).

The ’140 patent discusses a data storage system and method providing security for stored data to decrease the risk of unauthorized access. EX1001, Abstract. The patent discusses a so-called “data parser” that splits a dataset into “shares,” encrypts the data using a key, and distributes the shares of encrypted data to different storage locations. *Id.* This was a known technique. To gain allowance, the challenged claims were amended to recite storing the encrypted data shares together with data indicative of at least one key used to secure the dataset. EX1002, 1247-48. This, however, was also a known technique.

II. STANDING

Petitioner certifies that the ’140 patent is available for IPR and that Petitioner is not barred or estopped from requesting IPR as to the challenged claims. 37 C.F.R. §42.104(a).

III. UNPATENTABILITY GROUNDS

The table below identifies the references, applicable claims, and basis for each unpatentability ground.

Ground Number and Reference(s)		Claims	Basis
1	Ophir (EX1007), Takeda (EX1008)	1-6, 8, 10	§103
2	Ophir, Takeda, Birrell (EX1033)	7	§103
3	Ophir, Takeda, Dickinson (EX1005)	9	§103
4	Dickinson, Takeda	1-10	§103

Ophir—filed January 16, 2003; published July 22, 2004—is prior art under at least pre-AIA 35 U.S.C. §§102(a) and (e), even to the '140 patent's earliest claimed priority date of October 25, 2004. Takeda (published December 20, 2001) and Dickinson (published March 29, 2001) are prior art at least under §102(b). Birrell—filed September 22, 2000—is prior art under at least §102(e).

IV. THE '140 PATENT

A. Reproduced Prior-Art Disclosure

Much of the '140 patent's lengthy disclosure reproduces prior-art material from an application published years earlier by overlapping inventors. FIGs. 1-20 and corresponding specification portions are reproduced substantially identically from "Dickinson" (EX1005). *Compare* Dickinson, FIGs. 1-20, 7:34-54:2 with EX1001, FIGs. 1-20, 6:52-7:40, 9:16-52:54; EX1040 (redline comparison). EX1003 ("Bhattacharjee"), ¶¶32-33.

B. Additional Disclosure

The '140 patent expands on Dickinson by ***broadening*** the use cases for the security module and adding details on how data splitting can be implemented. EX1001, 52:55-53:18; Bhattacharjee, ¶34. Dickinson's use examples operated on authentication data in an e-commerce transaction. Dickinson, Abstract, 1:6-8; Bhattacharjee, ¶35. The '140 patent mentions additional uses of the data-splitting technology, which it renames a "secure data parser." EX1001, 52:55-53:18; Bhattacharjee, ¶¶35-36. Indeed, the '140 patent's alleged invention applies to "any application" for "any type of data set" and can use "any encryption process." EX1001, 52:66-53:9; Bhattacharjee, ¶¶35-36. Various ordering of the parsing, splitting, dispersing, and storage in multiple locations may be employed. EX1001, 53:9-18; Bhattacharjee, ¶¶35-36.

Thus, the '140 patent says that the same prior-art data parsing/splitting technology that was already disclosed in Dickinson can be applied beyond e-commerce. EX1001, 52:55-53:18; Bhattacharjee, ¶36. The '140 patent's figures and disclosure beyond those of Dickinson largely just describe either additional contexts (use cases) for the technology or specific encryption techniques, none of which appear in the challenged claims. Bhattacharjee, ¶¶32-36.

C. Prosecution History

During prosecution, in the rejection preceding allowance, the Examiner applied Dickinson as the primary reference in an obviousness rejection, finding that Dickinson disclosed nearly all limitations of the claims. EX1002, 1235-42.

The only claimed feature the Examiner found that Dickinson lacked was that the recited virtual disk comprises a directory mapped to the plurality of storage devices. For that, the Examiner relied on a secondary reference. EX1002, 1239.

Applicant did not dispute that Dickinson met most claim limitations, and instead narrowed each independent claim and argued that Dickinson does not disclose the added limitation reciting “include with each of the plurality of shares data indicative of the at least one session key used to secure the dataset.” EX1002, 1245-50. The Examiner then allowed the application. *Id.*, 1360-61. But the Examiner erred in failing to map Dickinson to the “directory” limitation. *See* §X.D.1.c.i *infra*. And the added limitation—the only claimed feature Applicant argued was not disclosed by Dickinson—is taught by Takeda (see §VII.B below), a reference of which the Examiner was apparently unaware. Bhattacharjee, ¶¶42-48 (citing EX1002, EX1009).

Ground 4 (§X) below explains how Dickinson and Takeda in combination disclose all features of claims 1-10. Additionally, other art not before the

Examiner (including Ophir and Takeda) separately renders claims 1-10 obvious, as in Grounds 1-3 (§§VII-IX) below.

V. PERSON OF ORDINARY SKILL IN THE ART (“POSA”)

A POSA as of the ’140 patent’s earliest claimed priority date would have had at least a bachelor’s degree in computer science, computer engineering, or a related field, with three years of experience in the area of securing data from unauthorized access or use. A higher level of education may substitute for less experience. *Bhattacharjee*, ¶40.

VI. CLAIM INTERPRETATION

Claim terms are construed herein according to their ordinary and customary meaning in light of the specification and prosecution history. 37 C.F.R.

§42.100(b). The Board “need only construe terms that are in controversy, and only to the extent necessary to resolve the controversy” regarding unpatentability. *Nidec Motor Corp. v. Zhongshan Broad Ocean Motor Co.*, 868 F.3d 1013, 1017 (Fed. Cir. 2017) (cleaned up). In the concurrent district-court proceeding, the parties have served competing proposed constructions of various claim terms. EX1041-EX1042. As discussed in detail where relevant within the Grounds below, this Petition presents alternative mappings of the prior-art disclosures to both parties’ proposed constructions of disputed terms, thus demonstrating unpatentability regardless of which proposed construction is correct. *See, e.g., 10X Genomics, Inc.*

v. *Bio-Rad Labs.*, IPR2020-00086, Paper 8, 18-22 (Apr. 27, 2020) (“Petitioner proposes alternative claim constructions and presents at least one ground of unpatentability for each construction.”).

VII. GROUND 1: OPHIR AND TAKEDA RENDER CLAIMS 1-6, 8, 10 OBVIOUS

A. Ophir

Ophir provides a secure network data storage mediator. Ophir, Title, Abstract; Bhattacharjee, ¶52. “The mediator connects over the network to one or more data clients and to one or more data storage devices, and provides secure storage of data for the data clients on the data storage devices.” Ophir, Abstract, *see also* [0015]; Bhattacharjee, ¶52. FIG. 4, reproduced below, shows that “mediator [401] operates as a central point for handling secure storage over a network both from the standpoint of the data clients as well as from the standpoint of the data storage devices, while not requiring the data clients to be involved with the protection of the data while in storage.” Ophir, [0027]; Bhattacharjee, ¶52.

Ophir’s data clients (e.g., 405, 407 in FIG. 4) are “any client [that] wishes to place data in storage or retrieve data from storage.” Ophir, [0004]; Bhattacharjee, ¶53. Ophir’s mediator provides services to data clients including database services, file services, and block services. Ophir, [0031], [0034], FIG. 7 (reproduced below); Bhattacharjee, ¶¶53-54.

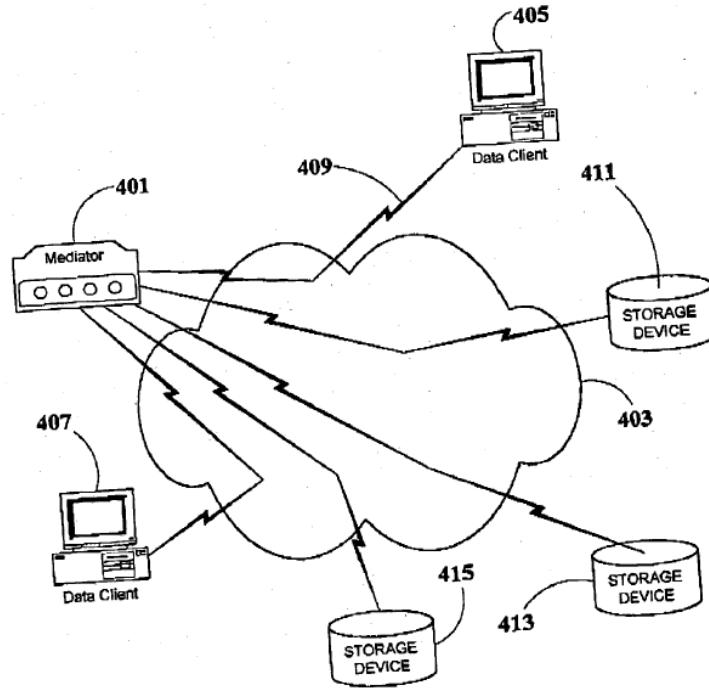


FIG. 4.

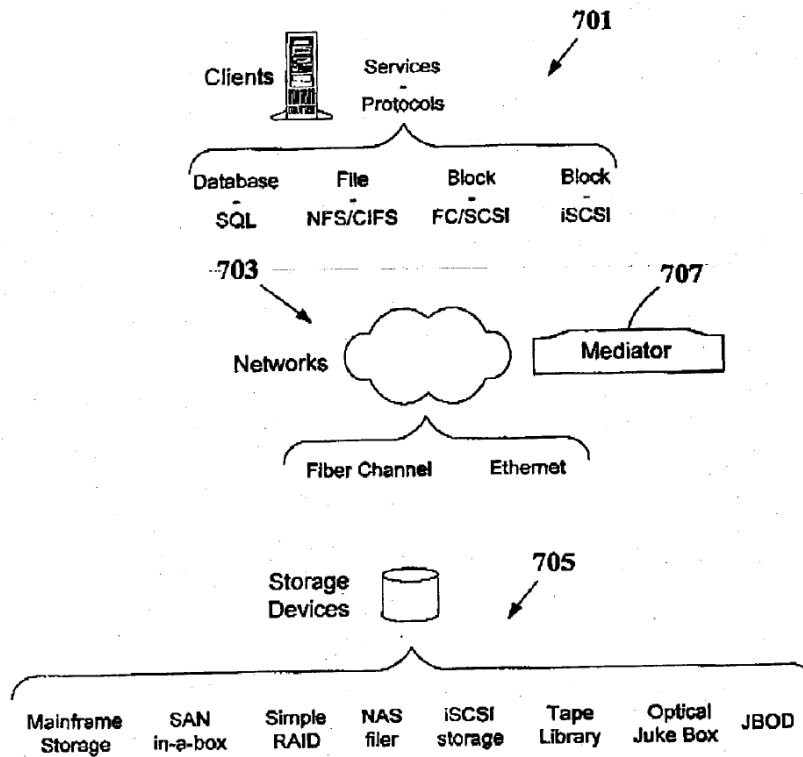


FIG. 7.

Ophir's FIG. 5 (reproduced below) illustrates an example of FIG. 4's mediator. Ophir, [0030]; Bhattacharjee, ¶55. Mediator 501 communicates with data clients and data storage devices via respective interfaces 503, 527. Ophir, [0030]; Bhattacharjee, ¶55. "Encryption/decryption unit 517" within data storage processor 519 uses encryption unit 513 to "encrypt[]" data from the data clients into encrypted data for safe storage in data storage devices," and uses decryption unit 515 to "decrypt[]" data retrieved from data storage devices into decrypted data for sending to data clients." Ophir, [0032]; Bhattacharjee, ¶55.

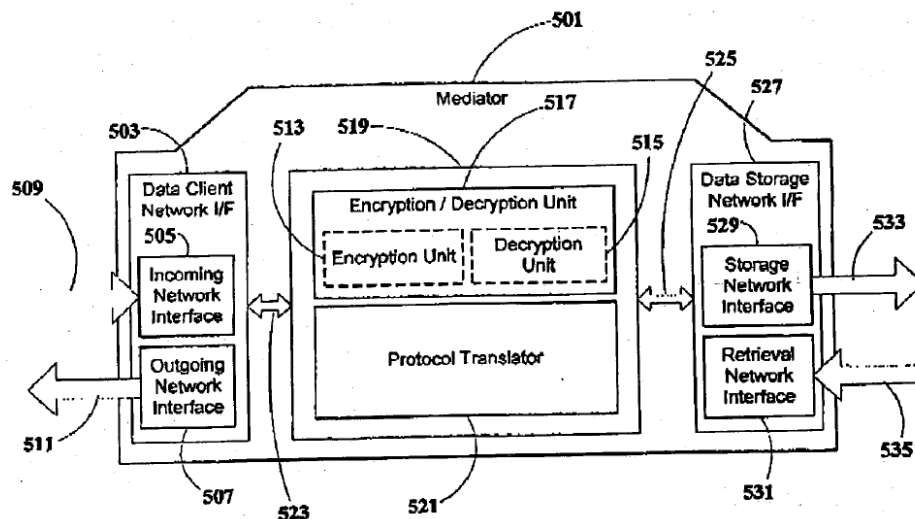


FIG. 5.

Ophir's mediator (e.g., 401 in FIG. 4, 501 in FIG. 5) thus encrypts the data to be stored and later decrypts it. Ophir, FIG. 5, [0010]-[0011], [0013], [0032]; Bhattacharjee, ¶56. Ophir provides examples of performing the encryption and decryption ("Encryption Scenarios,") but explains that they are "non-limiting examples provided for illustration, and that other scenarios are also possible within

the framework of the invention.” Ophir, [0036], [0035]-[0051]; Bhattacharjee, ¶¶57-60.

Ophir’s “**mediator provides secure virtual storage** to data clients...,” meeting the challenged claims’ “virtual disk” limitations as discussed *infra* §VII.D.1.c.i. Ophir, Abstract;² see also Ophir, FIG. 6 (reproduced below), [0013] (“the **mediator serves as a virtual secure storage device**.”), [0033] (“FIG. 6 illustrates the capacity of a **mediator 601** to **effort [sic] secure virtual data storage** for a data client 603 over a network connection 605.”); Bhattacharjee, ¶¶61-63.

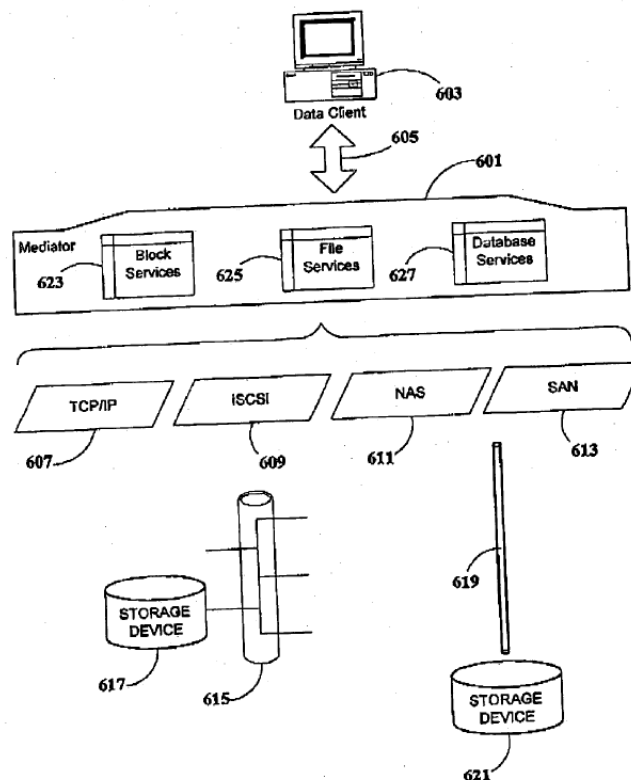


FIG. 6.

² Emphasis added throughout unless noted.

B. Takeda

Takeda discloses a ciphering apparatus and method. Takeda, Title, Abstract, [0001]; Bhattacharjee, ¶68. POSAs knew “ciphering” is synonymous with “encryption”. Bhattacharjee, ¶¶64-66; EX1037, 381 (“encrypt” means “encipher”); EX1043, 99 (“ciphertext” is “the scrambled or otherwise encoded text of an encrypted message”), 192 (dictionary entry for “encipher” says “*See encrypt*”). Takeda notes that conventional ciphering techniques divided data into blocks and then applied the same ciphering to each block, running the risk that compromise of any part of the data can lead to compromise of all the data. Takeda, [0003]-[0004]; Bhattacharjee, ¶68. Takeda’s improved ciphering technique provides that “even if a part of the data is deciphered, the rest of the data is not easily deciphered.” Takeda, [0005]-[0006]; Bhattacharjee, ¶68.

Takeda’s technique involves dividing plaintext – which is any non-encrypted (i.e., unciphered) information – into blocks, ciphering different blocks using different “ciphering attributes” (e.g., ciphering keys and/or ciphering algorithms), and outputting the ciphered blocks with the ciphering attributes. Takeda, [0007]-[0009], [0039], [0042]; Bhattacharjee, ¶¶67-69; EX1043, 407 (“plaintext” is “nonencrypted or decrypted text”); EX1037, 888, 207 (“plaintext” is “the intelligible form of an encrypted text,” as distinguished from “ciphertext” which is “the enciphered form of a text”); *see also* Takeda, [0037] (“plaintext” data can be

image data, not just text). The ciphering attributes can be stored with the ciphered blocks. Takeda, [0015], [0048] (“in FIG. 3A and FIG. 3B, the ciphertext $\{C_n\}$ and attribute $\{A_n\}$ are stored as one file in such a manner that the correspondence relation between the data can be recognized.”), FIG. 3A; Bhattacharjee, ¶¶68-70. Takeda’s ciphering process is described further below.

First, the plaintext, whether two-dimensional data or stream (linear) data, is divided into blocks. Takeda, [0007]-[0008]. FIG. 1B (reproduced below) illustrates the division of plaintext M into four blocks M_n (“ n ” is an index representing an integer), here M_1 - M_4 since there are four blocks. Takeda, [0038]; Bhattacharjee, ¶¶69-71.

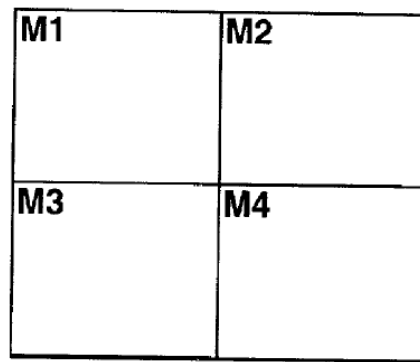


FIG.1B

Then the ciphering attributes that will be used to cipher the plaintext blocks are set. Takeda, [0007]-[0008]; Bhattacharjee, ¶72. FIG. 2A (reproduced below) illustrates a table of attributes A_n defining the ciphering that will be performed on the respective blocks M_n of the plaintext. Takeda, [0040]; Bhattacharjee, ¶72.

Ciphering attribute A1 is for plaintext block M1, ciphering attribute A2 for plaintext block M2, etc. Takeda, [0040]; Bhattacharjee, ¶72.

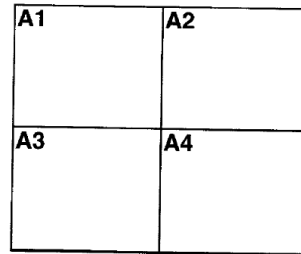


FIG.2A

Ciphering attributes A1-A4 may be ciphering keys and/or ciphering algorithms. Takeda, [0039], [0042]; Bhattacharjee, ¶73. FIG. 4 (reproduced below) is an example showing the constituents of an attribute table storing attribute data. Takeda, [0016], [0053]-[0054]; Bhattacharjee, ¶73. Each attribute table record can include a ciphering key and additional information including the key length and access privilege identifying who is permitted access to decipher the data. Takeda, [0054]; Bhattacharjee, ¶73.

RECORD1	BIT RECTANGULAR START POINT	BIT RECTANGULAR SIZE	ACCESS PRIVILEGE	KEY LENGTH	CIPHERING KEY
RECORD2	BIT RECTANGULAR START POINT	BIT RECTANGULAR SIZE	ACCESS PRIVILEGE	KEY LENGTH	CIPHERING KEY
RECORD3	BIT RECTANGULAR START POINT	BIT RECTANGULAR SIZE	ACCESS PRIVILEGE	KEY LENGTH	CIPHERING KEY
RECORD4	BIT RECTANGULAR START POINT	BIT RECTANGULAR SIZE	ACCESS PRIVILEGE	KEY LENGTH	CIPHERING KEY

...

FIG.4

Each plaintext block (e.g., M1-M4) is ciphered using its corresponding ciphering attribute (e.g., A1-A4), resulting in ciphertext blocks C1-C4, as shown below in FIG. 2B. Takeda, [0007]-[0008], [0041], Bhattacharjee, ¶74. The ciphering attributes can differ for the different blocks being ciphered. Takeda, [0009], [0054] (the records of FIG. 4's attribute table can differ by plaintext block); Bhattacharjee, ¶74.

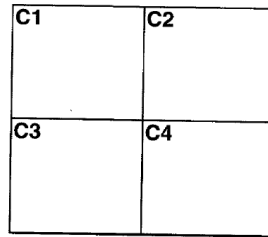


FIG.2B

Attributes (A_n) can be stored with their respective ciphertext blocks (C_n). Takeda, [0048]; Bhattacharjee, ¶75. FIG. 3A (reproduced below) illustrates the result of ciphering the blocks M1-M4 to generate ciphertext blocks C1-C4 and storing the ciphering attributes (including keys) A1-A4 with the respective ciphertext blocks. Takeda, [0048]; Bhattacharjee, ¶75.

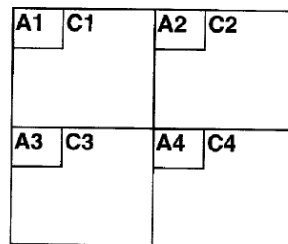


FIG.3A

Alternatively, a ciphering attribute can be stored separately from its respective ciphered block, and a pointer can be included with the ciphered block pointing to where the attribute for that block is stored. Takeda [0040]; Bhattacharjee, ¶76.

FIG. 5 (reproduced below) shows one embodiment of Takeda's above-discussed ciphering process. Takeda, [0017], [0055]-[0056]; Bhattacharjee, ¶77.

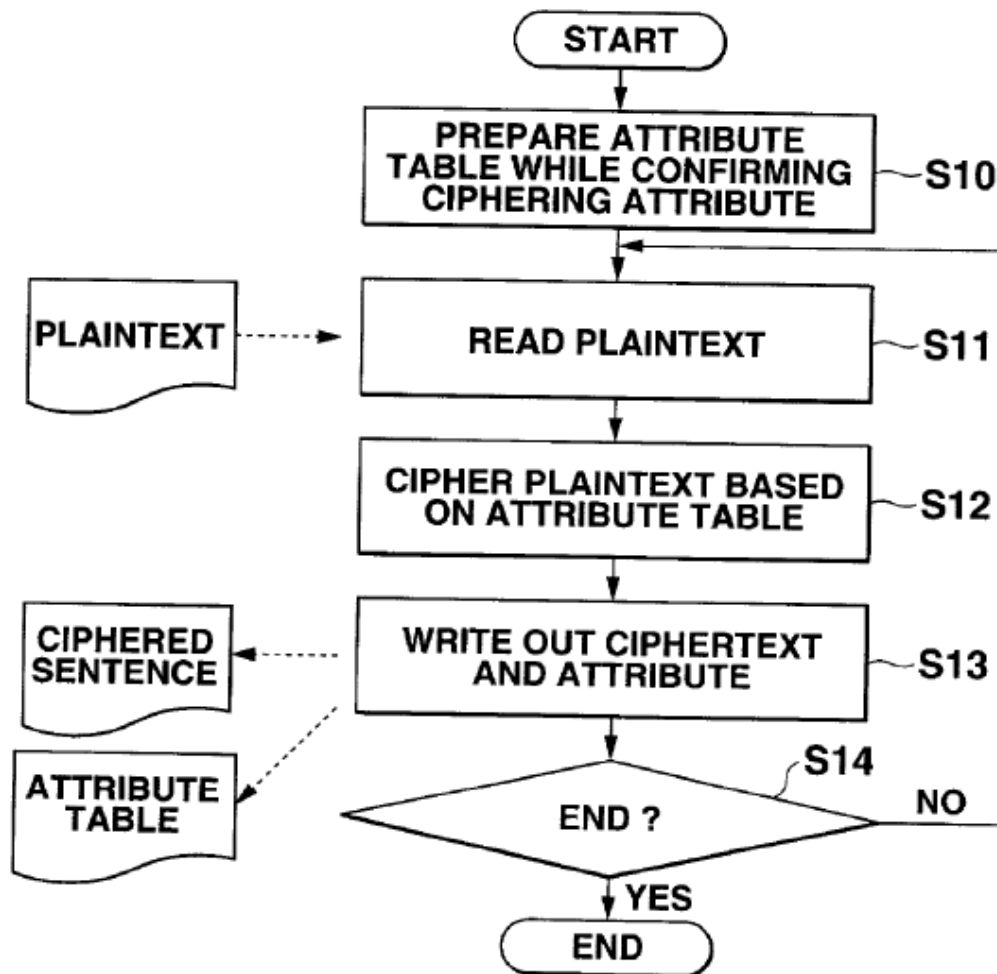


FIG.5

FIG. 6 (reproduced below) shows a deciphering process, in which stored ciphertext is retrieved and deciphered using the corresponding attribute. Takeda, [0018], [0057]-[0058]; Bhattacharjee, ¶78.

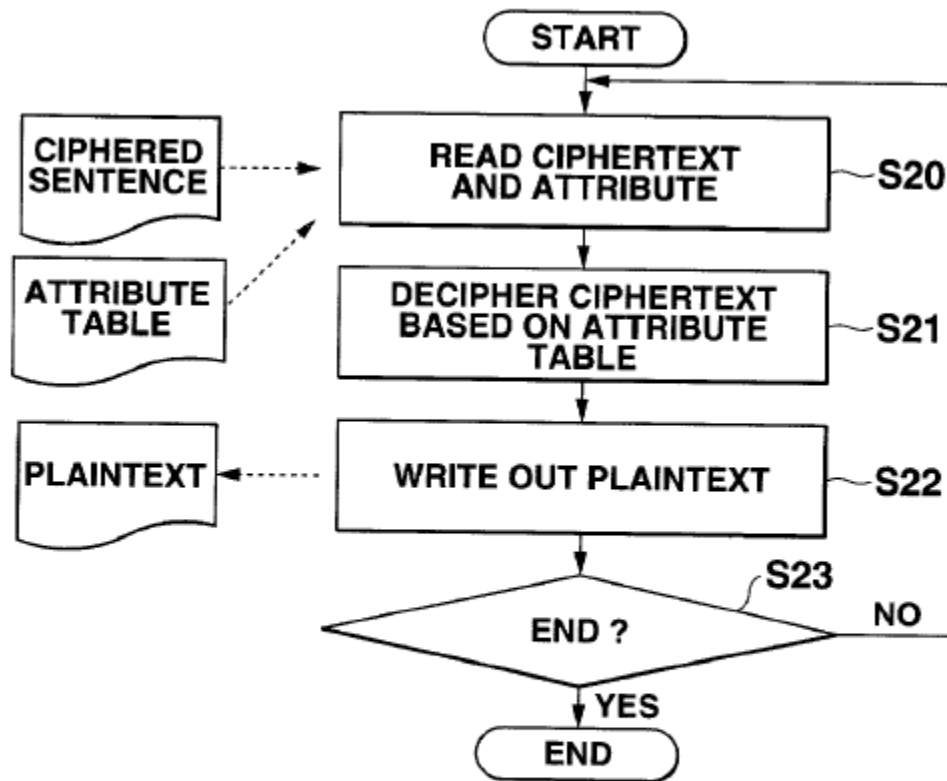


FIG.6

FIG. 19 illustrates a ciphering apparatus 10 that can perform Takeda's ciphering method. Takeda, [0031], [0087]; Bhattacharjee, ¶79. Plaintext input to plaintext input section 11 is formed into input blocks by blocking section 15. Takeda, [0088]; Bhattacharjee, ¶79. The user sets the attribute for each block using attribute setting section 12, and ciphering section 13 ciphers each plaintext block

using the corresponding attribute. Takeda, [0088]; Bhattacharjee, ¶79. The ciphertext and attribute table are output by section 14. Takeda, [0088]; Bhattacharjee, ¶79.

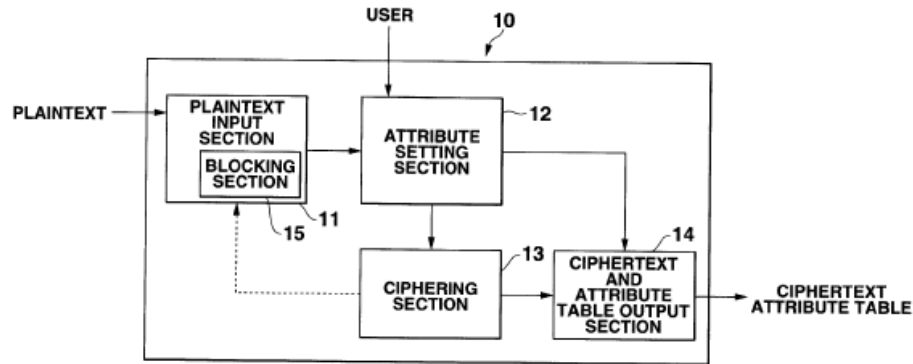


FIG.19

FIG. 20 illustrates a deciphering apparatus 20. Takeda, [0032]; Bhattacharjee, ¶80. The stored ciphertext and attribute table are retrieved and input through input section 21 to deciphering section 22, which consults the attribute table and decipheres the ciphertext to recover the plaintext, which is output through section 23. Takeda, [0090]; Bhattacharjee, ¶80.

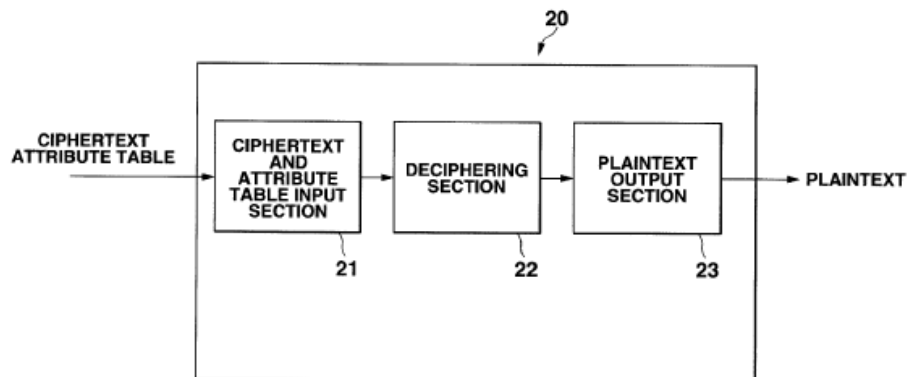


FIG.20

C. Ophir-Takeda Combination

1. Reasons to Combine

A POSA would have modified Ophir to implement Takeda's ciphering/deciphering method in Ophir's mediator for at least three independent reasons:

(1) POSAs would have expected the modification to increase Ophir's encryption/decryption reliability; (2) the modification would have represented a simple substitution of a known method and element for another method and element and would have yielded predictable results; and (3) the modification would have represented the use of a known technique to improve a similar device in the same way. *KSR Int'l v. Teleflex Inc.*, 550 U.S. 398, 416-17 (2007); Bhattacharjee, ¶81.

a. Increased Reliability

A POSA would have been motivated by Takeda to implement Takeda's ciphering/deciphering as the encryption/decryption performed by Ophir's encryption/decryption unit (e.g., encryption/decryption unit 517) to achieve "a more reliable ciphering technique" as taught by Takeda, and would have expected success in doing so, because with Takeda's technique "even if a part of the data is deciphered, the rest of the data is not easily deciphered." Takeda, [0006]; *see also* [0009]; Bhattacharjee, ¶82.

As explained in §VII.A, *supra*, Ophir's mediator (e.g., mediator 401 in FIG. 4 and mediator 501 in FIG. 5) performs encryption and decryption of the data to be

stored. Ophir, FIG. 5, [0010]-[0011], [0013], [0032]; Bhattacharjee, ¶83. Ophir is expressly not limited to the encryption scenarios Ophir describes, which are “non-limiting examples provided for illustration” because Ophir explains that “other scenarios are also possible within the framework of the invention.” Ophir, [0036]; Bhattacharjee, ¶83. Based at least on Takeda’s teachings, a POSA would have recognized that not using different encryption attributes for different parts of the data set in Ophir would create risk, because an attacker need only determine one decryption technique to gain access to Ophir’s full data set, and if any part of Ophir’s undivided encrypted data is undesirably decrypted, then all the encrypted data can be decrypted. Takeda, [0004] (“when a part of the ciphered data is deciphered, there is a risk or chance that all data will be deciphered.”); Bhattacharjee, ¶84. Thus, a POSA would have been motivated by Takeda to implement Takeda’s ciphering/deciphering technique (splitting the data into blocks and encrypting different blocks with different ciphering attributes) in Ophir to achieve a more reliable encrypting/decrypting technique that is less susceptible to unauthorized access than what Ophir alone describes. Takeda, [0006], [0009]; Bhattacharjee, ¶¶82-84.

b. Simple Substitution

As discussed below, implementing Takeda’s ciphering/deciphering as the encryption/decryption performed by Ophir’s encryption/decryption unit merely

represents the simple substitution of one known method and element for another to obtain predictable results. *KSR*, 550 U.S. at 416; *Bhattacharjee*, ¶¶85.

As explained *supra* §VII.A, Ophir's mediator (e.g., mediator 501) includes data storage processor 519 containing an encryption/decryption unit 517, optionally implemented with encryption unit 513 and decryption unit 515. Ophir, [0030], [0032]; *Bhattacharjee*, ¶¶86. "Encryption/decryption unit 517 encrypts data from the data clients into encrypted data for safe storage in data storage devices, and decrypts data retrieved from data storage devices into decrypted data for sending to data clients." Ophir, [0032]; *Bhattacharjee*, ¶¶86-87. Thus, Ophir's mediator (e.g., mediator 401 in FIG. 4, mediator 501 in FIG. 5) encrypts the data before storage and later decrypts it. Ophir, FIG. 5, [0010]-[0011], [0013], [0032]; *Bhattacharjee*, ¶¶86-87.

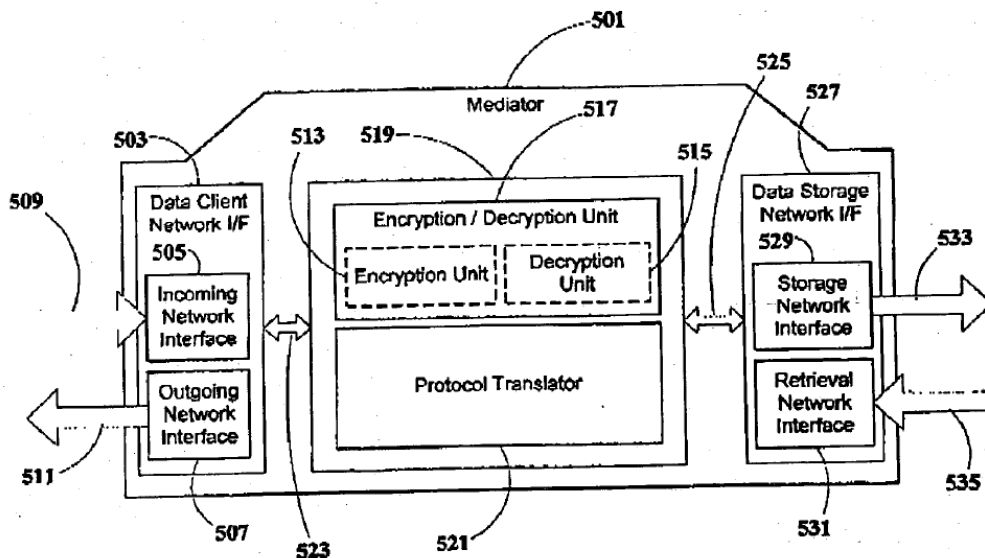


FIG. 5.

Takeda's FIG. 19 illustrates a ciphering apparatus 10, which includes plaintext input section 11, blocking section 15, attribute setting section 12, ciphering section 13, and "ciphertext and attribute table output section" 14. Takeda, [0031], [0087]-[0088]; Bhattacharjee, ¶88.

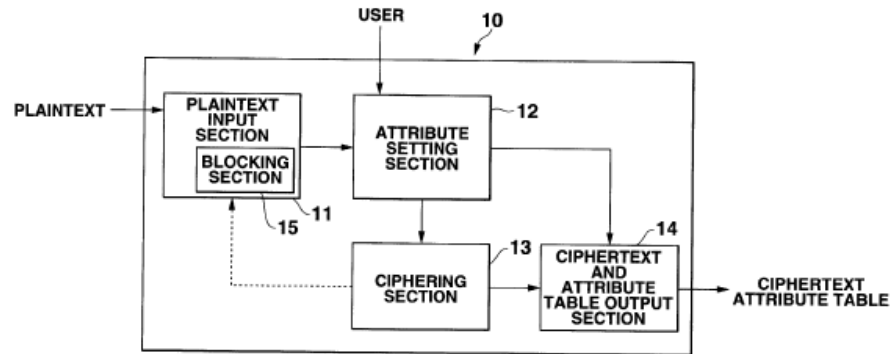


FIG.19

FIG. 20 illustrates a deciphering apparatus 20, which includes "ciphertext and attribute table input section" 21 for receiving the ciphertext and attribute table, deciphering section 22 for deciphering the ciphertext, and plaintext output section 23. Takeda, [0032], [0090]; Bhattacharjee, ¶89.

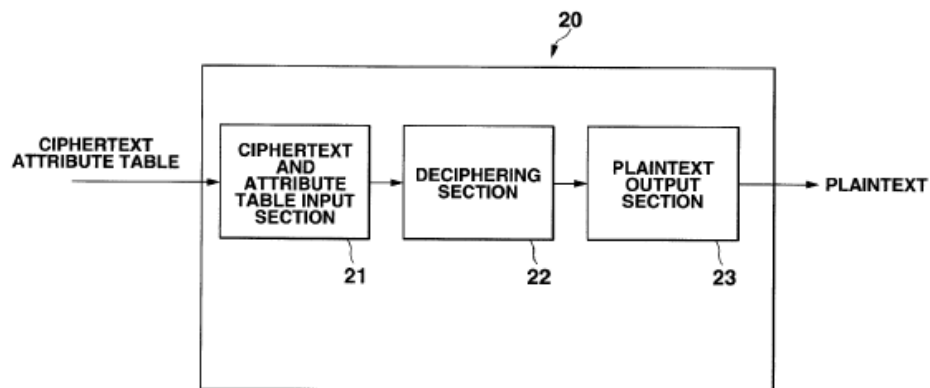


FIG.20

Implementing Takeda's ciphering/deciphering technique in Ophir's encryption/decryption unit 517 is a simple substitution of Takeda's known encryption/decryption technique for Ophir's encryption/decryption technique. Bhattacharjee, ¶90. This can be done by substituting Takeda's FIG. 19 ciphering apparatus 10 for Ophir's encryption unit 513 and Takeda's FIG. 20 deciphering apparatus 20 for Ophir's decryption unit 515, or adapting/programming Ophir's units to perform Takeda's techniques. Bhattacharjee, ¶90. For example, Takeda's ciphering apparatus 10 and Ophir's encryption unit 513 both take in data to be ciphered (i.e., encrypted) and output ciphered data. Takeda, [0088]; Ophir, [0032]; Bhattacharjee, ¶90; *see also* EX1010, 1:35-41 (ciphering engine produces encrypted data output); EX1011, [0002] ("cryptographic algorithms, also known as ciphers" are used "to encrypt data"). Similarly, Takeda's deciphering apparatus 20 and Ophir's decryption unit 515 both take in ciphered data and output deciphered (i.e., decrypted) data. Takeda, [0090] (Takeda's "ciphertext and attribute table input section" 21 receives "ciphertext" which is the ciphered data along with the attribute table and outputs "plaintext" which is deciphered data); Bhattacharjee, ¶90.

Substituting Takeda's ciphering/deciphering techniques and/or components for Ophir's encryption/decryption techniques and/or components would have predictably resulted in Ophir's data ciphering/deciphering (i.e.,

encrypting/decrypting) being performed as described by Takeda. Bhattacharjee, ¶91.

c. Use of Known Technique to Improve Similar Devices in the Same Way

Ophir's mediator is ripe for improvement using Takeda's known ciphering/deciphering technique. *KSR*, 550 U.S. at 417; Bhattacharjee, ¶92. Both Ophir and Takeda relate to methods and systems for securely storing data by using ciphering/deciphering (i.e., encryption/decryption) techniques. Takeda, Abstract; Ophir, Abstract; Bhattacharjee, ¶92. Ophir's mediator effects secure storage of clients' data on storage devices by encrypting data to be stored on the storage devices and decrypting data retrieved from the storage devices before providing the data to the clients. Ophir, Abstract, [0010]-[0011], [0013], [0015], [0032], FIGs. 4-5; Bhattacharjee, ¶92. Takeda's improved ciphering/deciphering techniques were a known technique to improve security and were directly applicable to Ophir's system. Takeda, [0005]-[0009]; Bhattacharjee, ¶92. A POSA would have recognized that applying Takeda's ciphering/deciphering in the Encryption/Decryption Unit (FIG. 5) of Ophir's mediator would have improved Ophir's secure storage system in the same way those techniques improved Takeda's system, by making the protected data more secure since "even if a part of the data is deciphered, the rest of the data is not easily deciphered." Takeda, [0006]; Bhattacharjee, ¶92.

2. Reasonable Expectation of Success

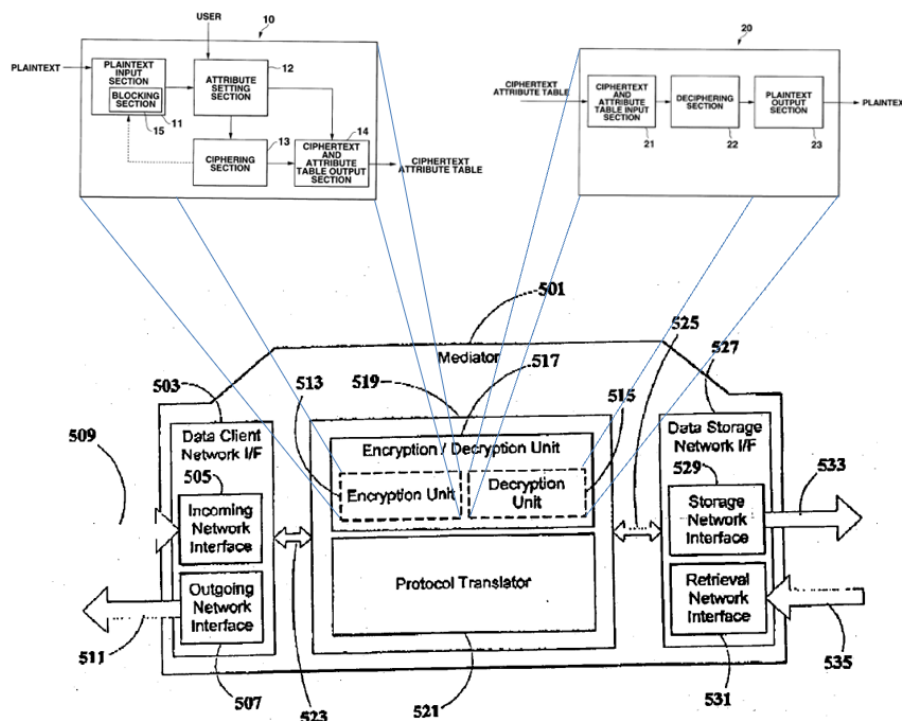
A POSA would have had a reasonable expectation of success in modifying Ophir's mediator's encryption/decryption unit to implement Takeda's ciphering/de-ciphering techniques and/or components, for the reasons explained *supra* §§VII.C.1.a-VII.C.1.c.. Bhattacharjee, ¶93. Takeda sufficiently explains its ciphering/deciphering techniques to allow a POSA to implement them in Ophir. Bhattacharjee, ¶93. A POSA would have understood that Takeda's ciphering/deciphering techniques were compatible with Ophir's system. Bhattacharjee, ¶93. For example, techniques performed by Takeda's ciphering apparatus 10 can be implemented in Ophir's encryption unit 513 as both intake data to be ciphered (i.e., encrypted) and output ciphered data, while techniques performed by Takeda's deciphering apparatus 20 can be implemented in Ophir's decryption unit 515 as both intake ciphered data and output deciphered (i.e., decrypted) data. Takeda, [0088], [0090]; Bhattacharjee, ¶93.

For the foregoing reasons, a POSA would have reasonably expected that implementing Ophir's encryption unit 513 and decryption unit 515 to use Takeda's ciphering apparatus 10 and deciphering apparatus 20, respectively, (or their techniques) would have successfully resulted in Ophir's mediator performing Takeda's ciphering/deciphering techniques and otherwise operating as Ophir describes. Bhattacharjee, ¶94.

3. Resulting Combination

A POSA would have implemented Takeda's ciphering/deciphering method in Ophir's mediator (mediator 401, 501, 601, 707) to replace Ophir's encryption/decryption performed in the "Encryption/Decryption Unit."

Bhattacharjee, ¶95. For instance, the figure below illustrates the result of implementing functionality of Takeda's ciphering apparatus 10 and deciphering apparatus 20 in Ophir's encryption unit 513 and decryption unit 515, respectively. Ophir's encryption unit 513 will perform Takeda's ciphering method and Ophir's decryption unit 515 will perform Takeda's deciphering method. The resulting system and method are referred to herein as the Ophir-Takeda Combination ("Ophir-Takeda"). Bhattacharjee, ¶95.



Result of Modifying Ophir Based on Takeda

Besides modifying Ophir's encryption/decryption techniques to use those of Takeda, Ophir-Takeda need not change Ophir's other aspects. Bhattacharjee, ¶96. In operation, a data block input to Ophir's block services 623, or a file input as plaintext to Ophir's file services 625, is divided (further divided in the case of blocks input to block services 623) into blocks to be ciphered using Takeda's ciphering method (e.g., Takeda FIG. 1B, blocks M1-M4), ciphered (i.e., encrypted), and output as ciphered blocks for storage on any of Ophir's storage devices (see Ophir, FIGs. 4, 6-7). Bhattacharjee, ¶96. Each ciphered block contains its respective ciphering attributes (e.g., the attributes of Takeda FIG. 4) or a pointer thereto as in Takeda (see Takeda, [0040], [0048]). Bhattacharjee, ¶96.

D. Ophir-Takeda Meets All Limitations of Claims 1-6, 8, 10

1. Claim 1

a. 1[PRE]: A secure storage network comprising:

Ophir-Takeda includes Ophir's data clients, mediator, and storage devices, which together represent a secure storage network. §VII.C *supra*; Ophir, FIGs. 4, 6-7, Abstract, [0001]; Bhattacharjee, ¶97. For example, Ophir's FIG. 4 (reproduced below) illustrates data clients 405 and 407, mediator 401, physical storage devices 411, 413, 415, and network 403 that together form a secure storage network. Ophir, Abstract ("The mediator connects over the **network** to one or more data clients and to one or more data storage devices, and provides **secure storage** of data for the data clients on the data storage devices."), [0011], [0015], FIGs. 6-7,

[0033] (“**FIG. 6** illustrates the capacity of a mediator 601 to effort [sic] **secure virtual data storage** for a data client 603 **over a network** connection 605.”), [0034] (networking and storage technologies and protocols implementing Ophir’s system); Bhattacharjee, ¶¶97-98. POSAs understood the claim term “network” to encompass a set of devices (e.g., Ophir’s data clients, mediator, and physical storage devices) that are connected by a communication network (e.g., Ophir’s network 403), consistent with challenged claim 1’s recitation of a “network comprising: a plurality of physical storage devices...and a secure storage system” (EX1001, 98:2-7) and the ’140 patent’s description that those devices communicate with each other “through any [communication] network” (EX1001, 97:41-51). Bhattacharjee, ¶¶97-98; EX1043, 362 (“a group of computer and associated devices that are connected by communications facilities” is a “network”).

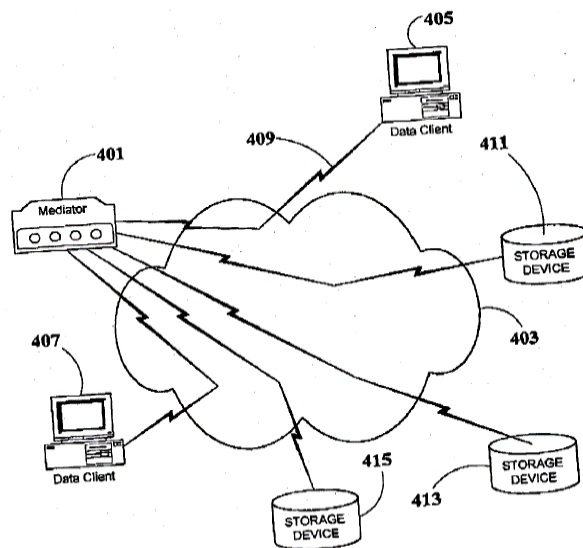


FIG. 4.

b. [1A]:

i. a plurality of physical storage devices

Ophir's "[s]torage devices...encompass various devices known in the art, including, but not limited to: mainframe storage; SAN-in-a-box; simple RAID; NAS filer; iSCSI storage; tape library; optical juke box; and JBOD ('Just a Bunch Of Disks'), which herein denotes any collection of one or more disk drives which does not necessarily include any special coordinating controller or data processing." Ophir, [0034] (discussing FIG. 7, excerpted below, illustrating "representative...technologies...which can be utilized by" Ophir's "invention"):

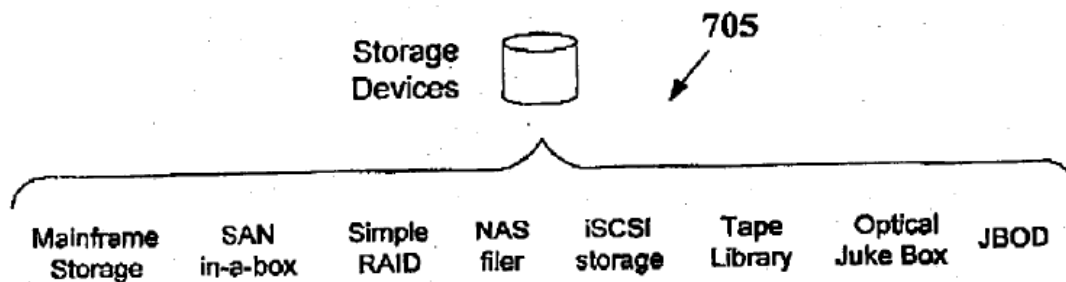


FIG. 7.

POSAs understood these are all examples of physical storage devices, at least some of which (e.g., JBOD, RAID) include a plurality of devices. Bhattacharjee, ¶¶99-105 (citing, e.g., EX1012, 1:4-6, 1:13-17; EX1013, 1:57-63; EX1014, 1:25-27 regarding RAID).

ii. storing thereon a plurality of shares...associated with at least one session key used to secure a dataset

Ophir-Takeda implements Takeda's ciphering technique, which divides received data into a plurality of blocks M_n . See §§VII.B, VII.C.3, *supra*; Takeda, FIG. 1B (illustrating four blocks M_1 - M_4), FIG. 11B, [0038]; Bhattacharjee, ¶106. In the Ophir-Takeda Combination, the received data is Ophir's data from a data client, and Ophir explains that the data is a "storage object (e.g., logical unit[, file[, director[y]])" for storage. Ophir, [0051], [0040]. POSAs understood that such an object (e.g., file, directory) is a "dataset" under both parties' constructions because it is a "collection of related information made up of separate elements [e.g., bits of a file, files in a directory] that can be treated as a unit [an "object (e.g., logical unit...)" (Ophir, [0051])] in data handling" (Google: EX1041, 9-10) and "a collection of information for storage" (SFI: EX1042, 6-7). Bhattacharjee, ¶106.

In Ophir-Takeda, each data block M_n is ciphered using a respective cipher attribute A_n (which may include a ciphering key—Takeda, [0042]) to form a ciphered block C_n . Takeda, FIGs. 2B, 11B-11D, [0040]-[0042], [0068]; §§VII.B, VII.C.3 *supra*; Bhattacharjee, ¶107.

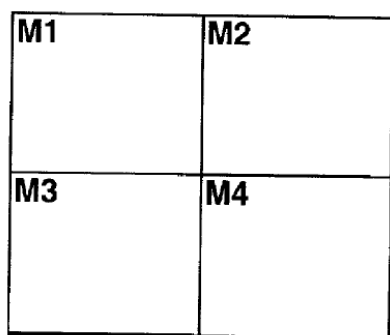


FIG.1B

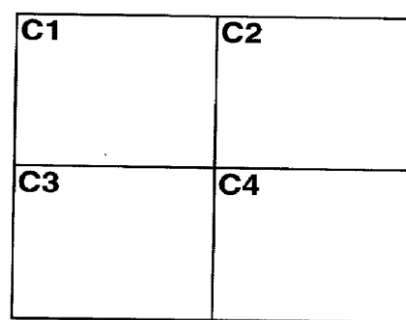


FIG.2B

Each ciphered block (Cn) is a “share” as claimed because it is a portion of the ciphered (i.e., encrypted) dataset, which the ’140 patent’s specification calls a “share.” EX1001, 53:19-28 (equating “share” and “portion”), 55:59-66 (discussing “storing the shares” after each “portion[]” is encrypted); 56:25-27 (“[s]eparating...encrypted data” results in “shares or portions of parsed data”), 56:60-62, 57:36-38, 58:11-13; *see also* EX1001, 59:15-18 (describing encrypted data “segments” as “shares”); Bhattacharjee, ¶108. Ciphertext block (Cn) together with its corresponding key (“ciphering attribute” An) (Takeda, FIG. 3A) meet a “share” as claimed if the share is construed to require storing information linking the share to the key that is used to secure the share. Bhattacharjee, ¶108. Each ciphertext block (Cn) together with a pointer to its corresponding key (“ciphering attribute” An) is a “share” as claimed if a “share” is construed to require storing information linking the share to the key used to secure it. (Takeda, [0040]); Bhattacharjee, ¶108.

The plurality of ciphered blocks (“shares”) are associated with at least one session key (attributes A1-An) used to secure the dataset, because attributes A1-An are used to cipher (i.e., encrypt) the dataset to create ciphertext blocks C1-Cn, and the ’140 patent’s specification describes “encrypt[ing]” a dataset as “securing” the dataset. EX1001, 60:52-64, 61:28-30, 64:56-59; Takeda, FIGs. 4 and 12, [0048]; Bhattacharjee, ¶109. The specification makes clear that a key used to encrypt a share – like Takeda’s attribute An used to cipher (encrypt) a corresponding plaintext block Mn to create a share Cn– is “associated with” that share as claimed. EX1001, 77:11-13, 77:31-33; 77:38-41; Bhattacharjee, ¶109. Ophir-Takeda also teaches that each ciphering attribute (A1-An), or a pointer to the attribute, may be stored with its corresponding ciphertext block (C1-Cn). Takeda, [0048], [0040], FIGs. 3A and 11E; Bhattacharjee, ¶109.

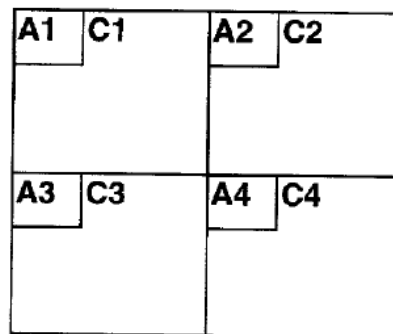


FIG.3A

The ciphering key of each attribute A_n is a “session key” under both parties’ constructions because it is a “[k]ey generated each time data is secured” (Google: EX1041, 13) and “a key created to both secure and restore a specific collection of information” (SFI: EX1042, 10). Takeda, [0042] (“attributes A_1 to A_4 are, for example, ciphering keys”). Each ciphering key A_n is generated each time data is secured and is created to secure and restore a respective block of plaintext M_n . Takeda, FIGs. 2A, 4, 5 step S10, [0056] (attribute is prepared for ciphering specific plaintext); Bhattacharjee, ¶110.

Ophir-Takeda stores the plurality of shares (whether just the ciphered blocks (C_n) or the ciphered blocks together with information linking to the corresponding ciphering keys) on the system’s plurality of storage devices. §§VII.B, VII.C.3, *supra*; Ophir, FIGs. 4, 6-7, [0030], [0032], [0034]; Bhattacharjee, ¶111. As discussed *supra* §VII.C.3, in Ophir-Takeda, Takeda’s ciphering is performed in Ophir’s “Encryption/Decryption Unit” 517 to at least support Ophir’s block services 623 and file services 625 (FIG. 6) and the encrypted shares are stored on the above-discussed plurality of storage devices such as Simple RAID and/or JBOD storage devices. Ophir, FIGs. 4, 6-7, [0030], [0032], [0034]; Bhattacharjee, ¶111.

c. [1B]: a secure storage system configured to:

Ophir-Takeda includes Ophir's mediator, which is a secure storage system as claimed. Ophir, FIGs. 4-5, Abstract (the mediator provides "secure virtual storage" to data clients); Bhattacharjee, ¶112. The claimed "secure storage system" need not include the physical storage devices on which data is stored, because claim 1 recites the secure storage system as distinct from the "plurality of physical storage devices," and the functionality claim 1 ascribes to the "secure storage system" is not functionality described in the '140 patent as being performed by the physical storage devices. Bhattacharjee, ¶112. Ophir's mediator in Ophir-Takeda (like the '140 patent's data parser) is configured to perform the functions that claim 1 recites for the secure storage system as explained below.

i. [1B-1]: present to a client device a virtual disk, the virtual disk comprising a directory mapped to the plurality of physical storage devices such that physical locations of the shares are hidden from the client device;

The term "virtual disk" does not appear in the '140 patent's specification. Ophir teaches that its mediator in Ophir-Takeda meets [1B-1] at least under SFI's construction of "virtual disk" as "a system that simulates or emulates a physical storage device" (EX1042, 8); Bhattacharjee, ¶¶113-15.

Ophir ([0033]) states:

FIG. 6 illustrates the capacity of a **mediator** 601 to effort [sic] **secure virtual data storage** for a data client 603 over a network connection 605. The **storage is considered “virtual”** because the data from data client 603 can be stored on a variety of **storage devices** using a variety of protocols, technologies, and services, as managed by mediator 601...

Ophir ([0013]) explains that, because “the mediator is able to store data in, and retrieve data from, any suitable storage device having access to the network”:

[i]n [this] manner, the mediator functions as a central coordinator for data storage between...clients requesting data storage and...storage devices providing data storage. In this central point, the mediator serves as a **virtual secure storage device**. The data clients do not have to be involved in any storage or retrieval operation with any storage devices, and **need not know the locations where the data is stored**.

Ophir further teaches that “data sent to the mediator for storage by a particular data client can be retrieved by the mediator from storage and sent back to that same data client.” Ophir, [0012]; Bhattacharjee, ¶116.

POSAs would have understood the above disclosures to teach that Ophir’s mediator system simulates or emulates a physical storage device, meeting SFI’s construction of “virtual disk,” because the client can make storage requests to the

mediator which takes care of storing and retrieving the data from the actual physical storage devices. Bhattacharjee, ¶117. Because the client can retrieve stored data through the mediator without knowing the data's stored location (Ophir, [0012]-[0013]) (i.e., "physical locations of the shares are hidden from the client device" as claimed (EX1001, 98:10-11)), POSAs would have understood that Ophir maintains a directory mapping where in the physical storage locations each piece of client data is stored, so that the mediator can retrieve the data when the client desires. Bhattacharjee, ¶117. Alternatively, maintaining such a directory mapped to the physical storage devices would have been a conventional and obvious way to implement what Ophir describes to allow the mediator to retrieve client data for the data clients when the client does "not know the locations where the data is stored." Ophir, [0013]; Bhattacharjee, ¶¶117-19 (citing EX1016, 5:39-6:16; EX1014, 1:13-18).

Additionally, Ophir ([0033]) discloses (Bhattacharjee, ¶120):

Moreover, ***mediator 601 is also able to provide block services 623, file services 625, and database services 627...***, while providing protocol translation between application protocols used with clients and application protocols used for storage devices and encrypting and decrypting the data that is stored on the storage devices.

Ophir’s “block services” further meet [1B-1] under SFI’s construction, and “file services” meet [1B-1] under both parties’ constructions, as detailed below. Bhattacharjee, ¶121.

(1) Block Services

Ophir’s mediator effects “secure virtual data storage,” and can do so by providing “block services” (Ophir, [0033]), which POSAs would have understood means that the mediator provides block-based virtual storage. Bhattacharjee, ¶122. POSAs knew that block-based storage addresses and operates on data in blocks, and that in a “virtual data storage” system like Ophir’s, the blocks present virtual disks. Bhattacharjee, ¶¶122-31, citing corroborating EX1015, FIG. 14A, 8:55-9:3 (“disk storage blocks” are “virtual volumes”), 7:32-40 (the “volume” is a “virtual disk device” that “provides a storage abstraction of multiple RAID devices” [physical storage devices]); EX1016, 2:57-3:3, 5:44-45 (“blocks represent a virtual disk”); EX1017, [0020] (“In the RAID architecture, multiple [physical] disks a[re] typically mapped to a single ‘virtual disk’” and “virtual disk blocks” are mapped to a file); EX1018, [0045]-[0048] (“The virtual disk object...represents a block-store disk”).

Given this background knowledge and Ophir’s disclosure that the mediator provides “virtual” storage using “a variety of storage devices” and that data clients need not know the physical location of the stored data (Ophir, [0033], [0013]),

POSAs would have understood that storage blocks presented to a data client are blocks of a virtual (rather than physical) disk, wherein the virtual disk comprises a directory mapped to the plurality of physical storage devices. EX1016, 5:39-58; EX1017, [0020]; EX1018, [0029], FIG. 16; Bhattacharjee, ¶132. That is, the client data stored in a block of a virtual disk is physically stored in one or more physical “locations” on one or more of Ophir’s physical storage devices (Ophir, [0013], [0033]-[0034]), and the virtual disk maintains a directory mapping each virtual block to the physical locations where its data is stored so that the data can be written to and read from the correct physical storage locations. EX1016, 5:39-58; EX1017, [0020]; EX1018, [0029], FIG. 16; Bhattacharjee, ¶132. The mediator’s block services thus present to a client device a “virtual disk” under SFI’s construction as “a system that simulates or emulates a physical storage device.” EX1042, 8. Bhattacharjee, ¶132.

To the extent that Ophir is considered to not expressly disclose the use of a virtual disk to support Ophir’s block services, or implementing a virtual disk by using a mapping directory, that would have been the conventional and obvious way to implement what Ophir describes – i.e., that the mediator provides “virtual” storage, data is accessed by a client on a block level, and the client need not know where the data is physically stored. Ophir, [0013], [0033]; *see also* EX1019, 1:39-50 (“Storage virtualization conceals the physical storage details from user/client

applications that access the physical storage.”), 2:13-24 (“Typically, when providing storage virtualization,” a “middle layer” will “creat[e] a virtual view of the storage data”); EX1015, FIG. 14A, 7:32-40, 8:55-9:3; EX1016, 2:57-3:3, 5:39-58; EX1017, [0020]; EX1018, [0029], [0045]-[0048]; Bhattacharjee, ¶133. Virtual disks, at least under SFI’s construction of that term, were a well-known technique for providing virtual block-level storage services to clients that need not know where the data is physically stored, and such virtual disks were known to use a mapping directory to map virtual storage blocks to physical storage locations, as the above-cited references corroborate. Ophir, [0013], [0033]; Bhattacharjee, ¶¶133-34 (citing EX1016, 5:39-6:16).

(2) File Services

In the co-pending litigation Google proposes construing “virtual disk comprising a directory mapped to the plurality of physical storage devices” as “a distributed file system shared among file servers that includes a directory with a clients’ view of all file assets mapped to the physical storage devices on which they are stored.” EX1041, 10; Bhattacharjee, ¶135.

POSAs would have understood that the file services 625 of Ophir’s mediator presents a file system to the data clients, because Ophir explains that (1) the storage objects can be files and (2) file services 625 communicates using file system application protocols, such as NFS (“network file system”) and CIFS

(“common internet file system”). Ophir, [0025], [0033]-[0034], [0051], FIGs. 6-7; EX1020, 1:21-27 (“NFS” means “Network File System”), 3:15-28 (“CIFS” means “Common Internet File System”); Bhattacharjee, ¶136. Like the ’140 patent’s parsed file system, Ophir’s file system provides virtual storage to the data client that hides from the client the physical location(s) where the file’s data is stored. Ophir, [0033], [0034]; Bhattacharjee, ¶136. POSAs would have understood that Ophir’s file system is a distributed file system shared among file servers because Ophir explains that the mediator’s data processing is “highly distributed” and can be performed by a number of “master devices” (Ophir, [0029], [0022], FIG. 4), which in the context of file services would be understood to be file servers. Bhattacharjee, ¶136.

A POSA would have understood that Ophir’s file system comprises a directory with a client’s view of all the client’s file assets, since the client device is able to retrieve the files which they have stored via the mediator. Ophir, [0046]-[0047]; Bhattacharjee, ¶137.

A POSA would further have understood that the directory of Ophir’s file system maps each file to the physical storage locations where the file’s data is physically stored because that is how Ophir’s mediator can retrieve stored data from the physical storage devices and, moreover, that is how file systems were known to work. Ophir, [0046]-[0047]; Bhattacharjee, ¶¶138-40 (citing EX1020,

19:61-20:4; EX1021, 1:64-2:21). Alternatively, maintaining a “directory” mapped to the physical storage locations would have been a conventional and obvious way to implement Ophir’s file system because a file system conventionally maintains such directories. Bhattacharjee, ¶141 (citing same).

POSAs would have understood that Ophir’s mediator’s presentation of a file system to the client device as part of providing file services simulates or emulates the underlying physical storage devices, and therefore presents a “virtual disk” under SFI’s proposed construction as well. EX1042, 8; Bhattacharjee, ¶142.

ii. [1B-2]: generate the plurality of shares for storage on the plurality of physical storage devices by performing a securing operation on the dataset received from the client device and distributing the dataset in the shares;

In Ophir-Takeda, either ciphertext blocks C_n , blocks C_n together with respective ciphering attributes A_n , or blocks C_n together with pointers to their respective ciphering attributes meet the claimed “shares,” as discussed *supra* §VII.C.3, VII.D.1.b.ii. These shares are generated by Ophir’s mediator (using Takeda’s ciphering techniques) by performing a securing operation (encryption/ciphering) on the dataset (plaintext M) received from the client device. See §VII.C.3 *supra*; Bhattacharjee, ¶¶143-44. Using Takeda’s ciphering techniques, the plaintext M is divided into blocks (M_n) and blocks M_n are ciphered (i.e., encrypted) to create ciphertext blocks (C_n). See §VII.B *supra*; Takeda,

[0041], [0068]; Bhattacharjee, ¶143. CIPHERING/encrypting blocks is a securing operation as claimed. EX1001, 60:56-64 (“process for securing data” includes “encrypt[ing]” “data shares”), 61:28-30 (“data securing” includes “encryption”), 64:56-59 (“encryption” is “a component of...methods...for securing...data”); Takeda, [0041]; Bhattacharjee, ¶¶143-44.

Ophir-Takeda, “distribut[es] the dataset in the shares” as claimed, under either party’s proposed construction. See EX1041, 3 (Google construction: “splitting the data set into shares”); EX1042, 2 (SFI construction: “dividing and placing the data set into a plurality of [data chunks/the shares/a plurality of shares]”). Ophir-Takeda’s dataset (M) provided from the client is split/divided into blocks M_n which are encrypted and placed into the plurality of shares C_n (or C_n plus A_n , or C_n plus a pointer to A_n). See §VII.C.3 *supra*; Bhattacharjee, ¶145. Takeda FIG. 3A shows an example with four shares into which the dataset was distributed. Takeda, FIG. 3A, FIG. 11B; Bhattacharjee, ¶145.

The shares are generated for storage on the plurality of physical storage devices in Ophir-Takeda, as discussed §VII.C.3 *supra*. Bhattacharjee, ¶146.

iii. [1B-3]: include with each of the plurality of shares data indicative of the at least one session key used to secure the dataset; and

As established *supra* §VII.D.1.b.ii, Ophir-Takeda’s mediator generates shares (ciphertext C_n) associated with at least one session key (respective

ciphering key of ciphering attribute An) used to secure a dataset (an object for storage). Ophir-Takeda includes with each of the shares the respective ciphering attribute or a pointer to the respective ciphering key, as discussed *supra* §§VII.C.3, VII.D.1.b.ii. This includes with each of the plurality of shares data indicative of the at least one session key used to secure the dataset under either party's construction, as discussed below. Bhattacharjee, ¶147. In the co-pending litigation, the parties offer competing constructions for the meaning of “data indicative of the at least one session key,” with Google proposing the term covers metadata about the key but not a key itself, and SFI proposing the term covers “information that points out, points to, makes known, shows, directs to, hints at, or suggests...” the at least one session key. EX1041, 5; EX1042, 3-4. Ophir-Takeda meets either construction. Bhattacharjee, ¶147.

As explained *supra* §VII.B, Ophir-Takeda's attributes An can include the key length, which is metadata about the ciphering key but is not the ciphering key itself. Takeda, FIG. 4, FIG. 12; Bhattacharjee, ¶148. Thus, the metadata in attribute An, such as the key length, is “data indicative of the at least one session key” under Google's proposed construction of “metadata about [the] key[]; not an encryption key.” EX1041, 5; Bhattacharjee, ¶148.

The attribute An can include the ciphering key itself, which meets SFI's construction of “data indicative of the at least one session key” because, e.g., it

“makes known” or “shows” the ciphering key. EX1042, 3-4; Bhattacharjee, ¶149; *see also* EX1044, 34 (alleging that storing data indicative of a key is met by storing the key itself).

Alternatively, Ophir-Takeda stores a pointer with the ciphered data, which also meets SFI’s construction because it “points to” or “directs to” the ciphering key. EX1042, 3-4; Bhattacharjee, ¶150.

Claim 1 does not limit the manner of including the data indicative of the at least one session key with each share.

If the “share” is just the encrypted data, then Ophir-Takeda storing the attribute An or a pointer to An in the same file as its encrypted data block Cn (per Takeda, [0040], [0048]) includes data (An or its pointer) indicative of the session key (An) with the share (Cn) as claimed. EX1001, FIG. 21 (key with encrypted data share); Bhattacharjee, ¶¶151-52.

Alternatively, if the share is the encrypted data plus the data indicative of the session key, then Ophir-Takeda storing the attribute An or a pointer to An in the same file as its encrypted data block Cn (per Takeda, [0040], [0048]) also includes data (An or its pointer) indicative of the session key (An) with (in fact, within) the share (Cn and An, or An and its pointer) as claimed. *Id.*; Bhattacharjee, ¶153.

- iv. **[1B-4]: reconstitute the dataset from at least a portion of the plurality of shares stored on the physical storage devices in response to a request from the client device for information in the dataset.**

Ophir-Takeda's mediator's Encryption/Decryption Unit 517 decrypts the stored data in response to a client device's request for the data. Ophir, FIG. 5, [0032] ("***Encryption/ decryption unit 517 ... decrypts data retrieved from data storage devices into decrypted data for sending to data clients.***"), [0046] (data is "requested" via communication from client), [0050]; Bhattacharjee, ¶154. In Ophir-Takeda, Encryption/Decryption Unit 517 implements Takeda's decryption techniques (see §VII.C.3 *supra*), which reconstitute the original dataset (plaintext M) from the plurality of shares stored on the physical storage devices. Takeda, FIGs. 6, 14, [0048] ("During deciphering, the attribute data {An} is set for the corresponding ciphertext {Cn} to decipher the data, and the original plaintext M is obtained."); Bhattacharjee, ¶¶154-56.

2. **Claim 2: ...wherein the secure storage system is further configured to generate the plurality of shares by performing a cryptographic operation on the dataset.**

The ciphering operation that Ophir-Takeda's mediator performs on the dataset (M) to generate the shares (see §VII.D.1.c.ii regarding [1B-2]) is a cryptographic operation, as claimed. Bhattacharjee, ¶¶157-58; EX1037, 381 (encrypt means encipher).

3. Claim 3: ...further comprising a key management server configured to store workgroup keys.

The '140 patent describes using “a workgroup key” to “further encrypt[]” the “encryption keys, splitting keys, or both.” EX1001, 75:63-67; Bhattacharjee, ¶159.

Ophir likewise teaches “to use ‘*master keys*’ to encrypt encryption/decryption keys, thereby making it safe to store encryption/decryption keys on external storage instead of in limited internal memory.” Ophir, [0032]; Bhattacharjee, ¶160.

Ophir’s “master keys” are thus “workgroup keys” as claimed. Bhattacharjee, ¶161. A POSA would have used Ophir’s master keys to encrypt Ophir-Takeda’s ciphering keys to make it safe to store the ciphering keys externally in those implementations where the ciphering keys are not stored with the ciphertext. Ophir, [0032]; Bhattacharjee, ¶161.

To the extent claim 3’s “workgroup keys” are interpreted to require providing different levels of access to different groups of users, Ophir-Takeda still renders claim 3 obvious. Bhattacharjee, ¶162. Ophir-Takeda designates access privileges for users as part of Takeda’s ciphering attributes. See §VII.C.3; Takeda, FIG. 4 (“access privilege”), [0054], [0060]; Bhattacharjee, ¶162. A POSA would have implemented access privileges via Ophir’s “master keys” in Ophir-Takeda to

provide enhanced security by restricting access to only designated users. Takeda, [0050], [0054], [0060]; Bhattacharjee, ¶162.

The '140 patent illustrates a “key management system” which “may provide for external key management for keys utilized in the encryption and/or splitting operations of the secure parser.” EX1001, FIG. 42, 82:31-33. Ophir likewise teaches that “encryption/decryption keys...are either generated locally...or...are *stored on an external key server*...).” Ophir, [0032]. Ophir’s external key *server* thus performs the same key-management function as the '140 patent’s key management system, and therefore is a “key management server” as claimed. Bhattacharjee, ¶163. POSAs would have known to similarly store Ophir’s “master keys” on such an external key server so as not to take up “limited internal memory” (Ophir, [0032]), thus meeting claim 3. Bhattacharjee, ¶164.

4. Claim 4: ...wherein the secure storage system is further configured to provide access to the reconstituted dataset.

The mediator in Ophir-Takeda’s secure storage network is configured to provide access to the reconstituted dataset. Ophir, [0032]; Bhattacharjee, ¶165.

Neither claim 4 nor the '140 patent’s specification explains what is meant by “the reconstituted dataset.” It should be interpreted to encompass retrieving encrypted shares of the dataset from storage and decrypting them to recover the dataset. See EX1001, 2:46-48 (mentioning “reconstituting...the secured data into its original form for authorized access”); Bhattacharjee, ¶166.

Ophir-Takeda's mediator "retrieve[s]" the stored data from the physical storage locations and "send[s] the data back to [the] data client" (Ophir, [0012]) after the data is decrypted by encryption/decryption unit 517. Ophir, [0016], [0032]; Bhattacharjee, ¶167. In Ophir-Takeda, the decryption is performed using Takeda's deciphering techniques. See §VII.C.3 *supra*; Takeda, [0048]; Bhattacharjee, ¶167. The decryption involves ordering the blocks appropriately to reconstitute the input data M in its original form. Takeda, [0048]; Bhattacharjee, ¶167.

To the extent Takeda is not determined to expressly disclose ordering the retrieved data blocks in the dataset's original order before providing them back to the requesting client, doing so would have been the obvious, common-sense way to implement Ophir-Takeda. Bhattacharjee, ¶168. POSAs knew it was desirable to have the mediator properly order the dataset blocks before returning them to the client to make the dataset understandable, and because it was the mediator that broke the dataset into blocks, it would be difficult (if not impossible) for the client to determine the correct order of the returned data blocks if they were out of order. Bhattacharjee, ¶168. The decrypted and properly ordered blocks of the dataset is a reconstituted dataset as claimed. Bhattacharjee, ¶¶168-69.

5. Claim 5: ...wherein the secure storage system is further configured to establish a secure connection between the client device and the secure storage system.

The term “secure connection” appears only in the claims, and has no limiting description in the ’140 patent. Bhattacharjee, ¶170.

Ophir teaches that data transmitted between the client device and the mediator (the claimed secure storage system – see §VII.D.1.c *supra*) is “protect[ed]” by further encrypting the data for transmission between the client and the mediator over the communication network. Ophir, [0014]-[0015]. A POSA would have understood that this establishes a secure connection between the client and mediator, as claimed. *Id.*; Bhattacharjee, ¶171.

To the extent that Ophir is considered to not explicitly disclose that its data clients and mediator are connected by a secure connection, that would have been the routine, conventional, and obvious way to implement what Ophir describes. Bhattacharjee, ¶172. Secure connections between network components were well known before the filing date of the ’140 patent. Bhattacharjee, ¶172; EX1022, FIG. 1 (secure connections 52, 54, 56); 4:19-22; EX1023, [0013], [0015]; EX1024, 5:15-28 (describing secure socket layers). A POSA would have been motivated to use such a secure connection in Ophir-Takeda because the very purpose of Ophir to ensure security in the storage of data would be enhanced by further securing the data for its transmission from the client for storage in the secure storage system.

Ophir, Title (“**Secure** Network Data Storage Mediator”), [0009]; Bhattacharjee, ¶¶172-73.

6. Claim 6: ...wherein the shares contain a substantially random distribution of the received dataset.

A POSA would have implemented Ophir-Takeda by randomizing the plaintext data M when creating the blocks Mn. Bhattacharjee, ¶174.

Randomization of data in a security context was a well-known technique to enhance data security, as evidenced by multiple prior-art references. EX1027, [0106]; EX1028, [0058]; EX1029, [0011]; EX1030, [0016]-[0018]; EX1031, Title, Abstract, [0004]; Dickinson, 15:26-27, 23:9-11; Bhattacharjee, ¶174. A POSA would thus have known to randomize the plaintext data within the blocks Mn when creating the blocks, to provide enhanced security. Bhattacharjee, ¶174. Thus, each block Mn would contain randomly arranged plaintext data from dataset M. *Id.* Ciphertext blocks Cn (claimed “shares”), formed by encrypting the plaintext blocks Mn, would likewise contain randomized distributions of the received dataset (M), meeting claim 6. *Id.*

The POSA would have reasonably expected in implementing such randomization because techniques for randomizing data, including as part of splitting data into blocks, were well known, as the above-cited references demonstrate. Bhattacharjee, ¶¶175-77.

7. Claim 8: ...wherein the secure storage system is configured to secure the dataset by encrypting the dataset using the at least one session key.

Ophir-Takeda's mediator uses Takeda's ciphering attributes A_n , which include session keys (e.g., encryption keys), to secure the dataset by securing respective portions of the dataset. See §§ VII.D.1.b.ii, VII.D.1.c.iii *supra*; Bhattacharjee, ¶178. Since the ciphering attributes A_n perform an encryption on the respective portions M_n of plaintext, Ophir-Takeda secures the dataset by encrypting the dataset using the at least one session key (ciphering key). Takeda, FIG. 3A, [0042]; Bhattacharjee, ¶¶178-79.

8. Claim 10: ...wherein the data indicative of the at least one session key comprises: a portion of the at least one session key or data indicative of a portion of the at least one session key.

In Ophir-Takeda, each key A_n used to encrypt a block C_n is a session key (see §VII.D.1.c.iii *supra*) so all the keys A_1 - A_n collectively meet the recited “**at least one** session key.” Bhattacharjee, ¶¶180-82.

Ophir-Takeda stores with the ciphertext block C_n either the key (A_n) and/or key metadata (e.g., key length), or a pointer to the key (A_n) that was used to cipher that block. See §VII.C.3 *supra*. Each alternative meets claim 10. Bhattacharjee, ¶181. For each block (C_n), the respective key A_n stored with it is “a portion of the at least one session key” (A_1 - A_n) as claimed, because a single key A_n is a portion of the collective “at least one session key” (A_1 - A_n) used to encrypt the entire

dataset. Bhattacharjee, ¶181. Similarly, for each block (C_n), the metadata (e.g., key length) or pointer to respective key A_n is “data indicative of a portion of the least one session key” (A₁-A_n) under both parties’ competing constructions as discussed *supra* §VII.D.1.c.iii. EX1041, 5-6; EX1042, 3-4; Bhattacharjee, ¶¶181-82.

VIII. GROUND 2: OPHIR, TAKEDA, AND BIRRELL RENDER CLAIM 7 OBVIOUS

A. Birrell

Birrell backs up data on a plurality of network-connected computers. Birrell, Title, Abstract; Bhattacharjee, ¶183. To protect against failure of some of the storage devices on which data is backed up, Birrell provides “fault tolerance” by achieving data redundancy by either replication or encoding. Birrell, 15:39-43, 16:7-17:32. Encoding for redundancy makes up for lost data without having to replicate the data. Birrell, 15:39-57. Either way, only a subset of the stored data need be retrieved to restore the original data. Birrell, 18:14-18; Bhattacharjee, ¶183.

B. Ophir-Takeda-Birrell Combination

A POSA would have further modified Ophir-Takeda to enhance its ciphering algorithms by providing redundancy of the ciphered data via redundancy encoding or replication. Bhattacharjee, ¶184. Ophir-Takeda’s ciphertext C_n is stored on Ophir’s storage devices as explained *supra* §VII.C.3. A POSA would

have recognized based on Birrell that the risk from a storage device failing – thus hindering the mediator’s ability to restore the original data – can be reduced by providing redundancy of the ciphertext blocks C_n . Birrell, 15:39-42, 16:7-17:32; Bhattacharjee, ¶184. Takeda does not mandate any particular ciphering algorithm; therefore a POSA implementing Takeda’s method in Ophir-Takeda would have, based on Birrell, implemented a ciphering algorithm providing redundancy by encoding and/or replication, such that reconstituting the original data involves retrieving only a subset of the ciphertext blocks C_n (without need to retrieve redundant blocks). Bhattacharjee, ¶184.

A POSA would have reasonably expected success in achieving this Ophir-Takeda-Birrell combination. Bhattacharjee, ¶185. Birrell provides multiple techniques for providing such redundancy, including redundancy encoding and mere replication. Birrell, 15:39-18:29; Bhattacharjee, ¶185. Applying either technique to the ciphertext blocks C_n (“shares”), would have been well within the POSA’s skill. Bhattacharjee, ¶185.

C. Claim 7: ...wherein the secure storage system is configured to reconstitute the received dataset from fewer than all of the shares

Ophir-Takeda-Birrell’s mediator generates ciphertext blocks C_n with redundancy. *See* §VIII.B *supra*. Therefore, when the mediator reconstitutes the

original plaintext data, it does so using fewer than all the ciphertext blocks, because it need not use the blocks that are redundant. Bhattacharjee, ¶¶186-87.

IX. GROUND 3: OPHIR, TAKEDA, AND DICKINSON RENDER CLAIM 9 OBVIOUS

A. Dickinson

In prior-art disclosure identical to portions of the ‘140 patent, Dickinson discloses that a set of sensitive data (S) can be secured by “split[ting] the data” into independently undecipherable portions by performing a conventional “XOR” operation (a well-known mathematical operation that changes the bits of binary data) on S with a random number. Dickinson, 23:9-12; *see also* FIG. 8 (steps 805, 810, 820, 825, and 830 as part of “data splitting process”), 20:22-34; EX1001, 23:26-33, 20:23-48 (same); Bhattacharjee, ¶188. The result is an encrypted version of the data (the XOR output) and the random number used in the XOR process. Dickinson, FIG. 8; Bhattacharjee, ¶188. Dickinson states that the sensitive data is therefore “split” because it cannot be recovered from the XOR result or the random number alone. Dickinson, 23:9-12; Bhattacharjee, ¶188.

B. Ophir-Takeda-Dickinson Combination

POSAs would have further modified Ophir-Takeda, based on Dickinson, to implement Takeda’s ciphering of plaintext blocks M_n by performing the mathematical operation $C_n = M_n \text{ XOR } A_n$, with the ciphering attribute (key) A_n implemented as a random number. Bhattacharjee, ¶189. Takeda does not restrict

what ciphering attributes A_n (including keys or algorithms) are used in its ciphering process. Takeda, [0041]-[0042]; Bhattacharjee, ¶190. Implementing an XOR operation as the ciphering algorithm and a random number as the ciphering key A_n would have been using a known technique from Dickinson in the resulting Ophir-Takeda-Dickinson system and method, with predictable results (ciphertext C_n would have represented the predictable result of the XOR between M_n and A_n). EX1001, 20:48-50 ('140 patent conceding XOR operation was well known); Bhattacharjee, ¶190. POSAs would have reasonably expected success, as the XOR was a well-known mathematical operation for changing the bits of binary data. Bhattacharjee, ¶190; *see also* EX1001, 20:48-50 (XOR operation was “well known”).

C. Claim 9: ...wherein the secure storage system is configured to secure the dataset by splitting the dataset using the at least one session key

Ophir-Takeda-Dickinson splits dataset M using the at least one session key (ciphering attributes A_n including a ciphering key). Bhattacharjee, ¶191; *see* EX1001, 22:6-11 (purported invention not limited to particular splitting process). Performing an XOR operation of each plaintext data block M_n with a respective ciphering key from A_n (claimed “session key”) meets claim 9 via the same XOR “splitting” operation discussed in the '140 patent. Dickinson, 23:9-12; EX1001, 23:26-33; Bhattacharjee, ¶¶191-92.

X. GROUND 4: DICKINSON AND TAKEDA RENDER CLAIMS 1-10 OBVIOUS

The Examiner in prosecution found that Dickinson disclosed all but two of claim 1's limitations. The Examiner did not find *explicit* mention in Dickinson that the virtual disk comprises a directory mapping to the physical storage devices, and found that Dickinson did not disclose "includ[ing] with each of the plurality of shares data indicative of the at least one session key used to secure the dataset." EX1002, 1236-42, 1247-50 (original claim 14), 1360-61. Dickinson does in fact disclose (or alternatively render obvious) the former; Takeda discloses the latter but was not before the Examiner. The Dickinson-Takeda combination renders obvious claims 1-10. Bhattacharjee, ¶193.

A. Dickinson

Dickinson discloses that a set of sensitive data "S" (e.g., user's "password, login, financial credentials, or the like") associated with a user system 105 can be stored in a "sensitive data vault" in "the data storage facilities of the depository 210 [of trust engine 110, FIG. 2], ***or split and stored in the depository system 700***" that includes multiple data storage facilities D1-D4 (as shown in FIG. 7). Dickinson, 29:13-23; Bhattacharjee, ¶194.

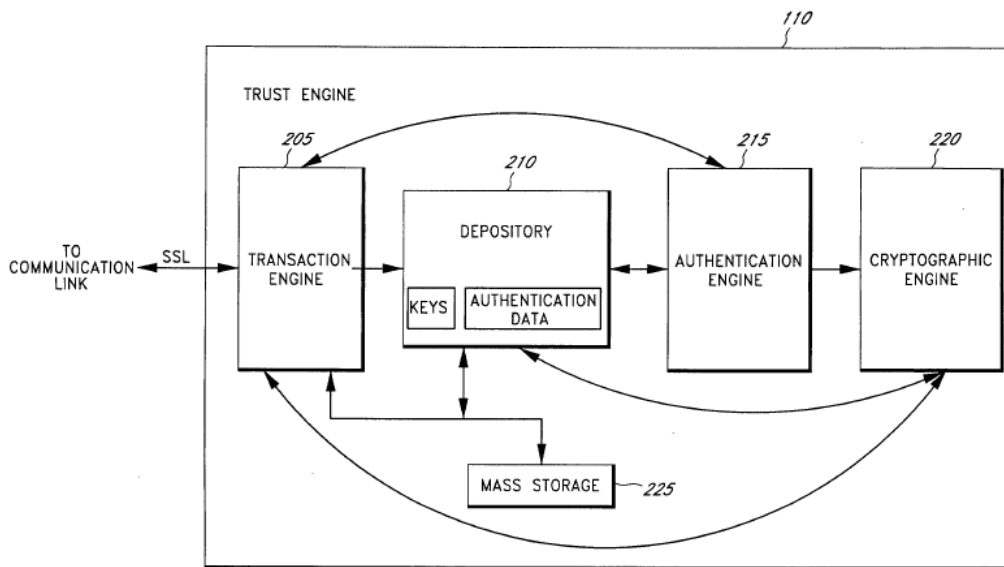


FIG. 2

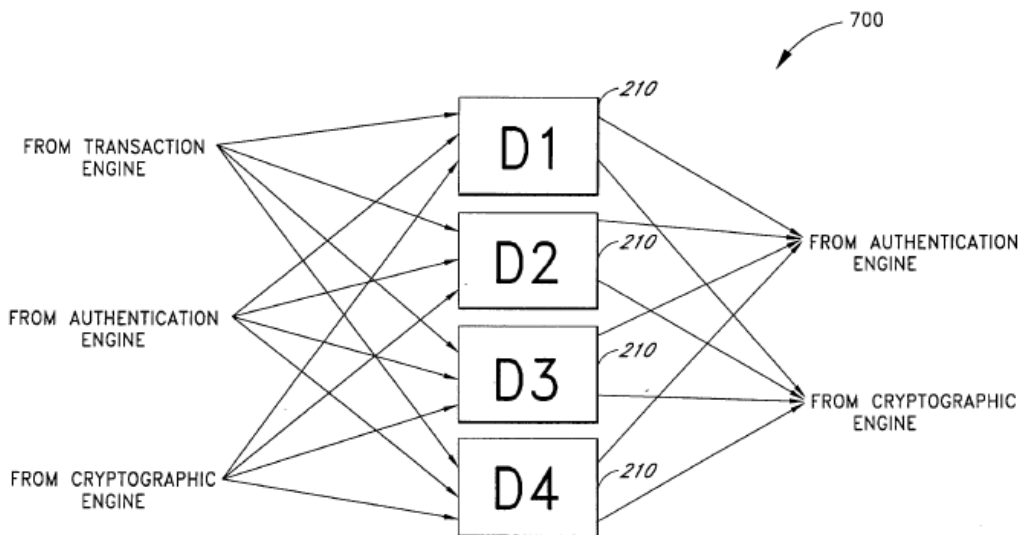


FIG. 7

“Data splitting process” 800 (FIG. 8) “splits” sensitive data S into at least two independently undecipherable portions—four portions A-D in FIG. 8—and

distributes those portions among data storage facilities D1-D4. Dickinson, 8:11-14, 19:16-20:18, 21:10-14; Bhattacharjee, ¶195. The portions are “independently undecipherable” because no one portion is sufficient, by itself, to reconstruct the original data S. Dickinson, 20:30-21:14, 23:6-15.

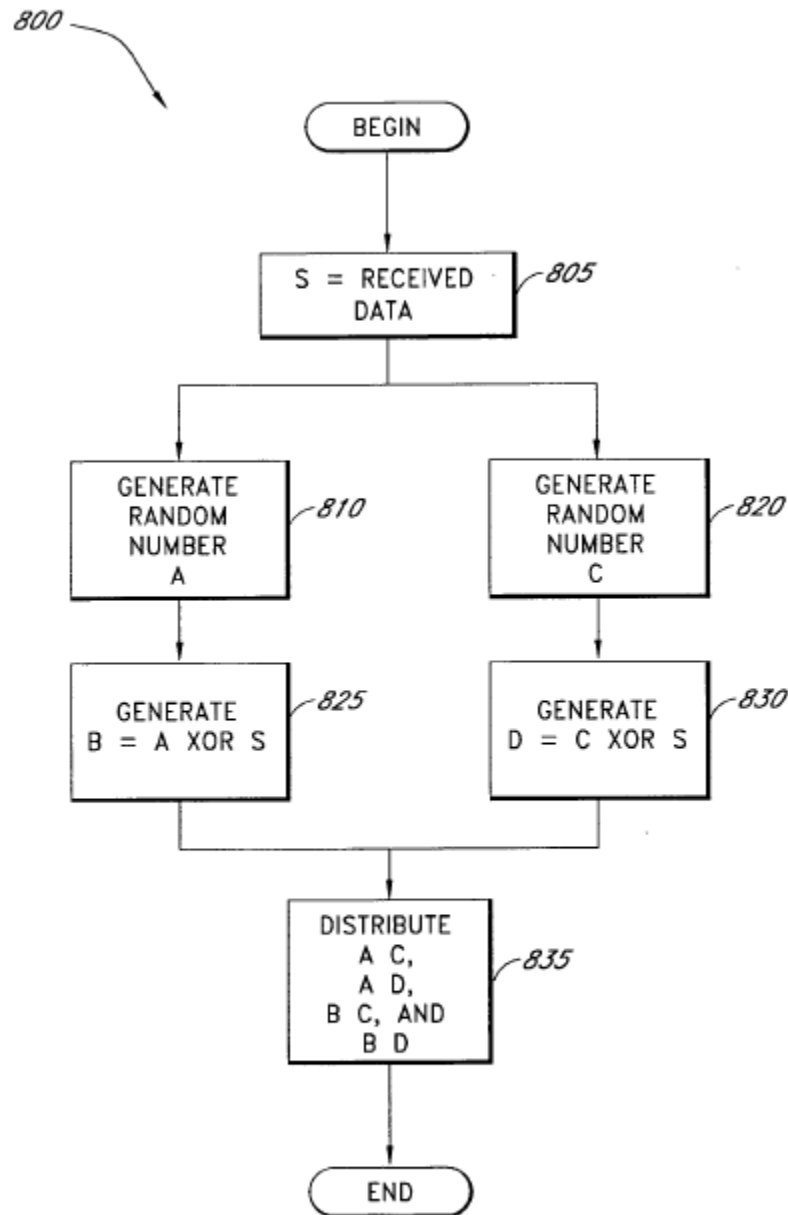


FIG. 8

The independently undecipherable portions in FIG. 8 include: (1) two “encrypted portions” (B and D) that are encrypted versions of the full data S and (2) the keys (A and C) required to decrypt them. Dickinson, FIG. 8 (steps 825 and 830), 20:25-21:14; Bhattacharjee, ¶196. Specifically, process 800 performs two parallel encryption steps 825 and 830 on sensitive data S. Dickinson, 20:32-21:1, 21:10-14; Bhattacharjee, ¶196, (citing EX1034, 4:25-29). Step 825 performs an exclusive OR (“XOR”) of S with a random number A, resulting in encrypted data B; and Step 830 XORs S with a different random number C resulting in differently encrypted data D. Dickinson, FIG. 8, 20:33-21:1; Bhattacharjee, ¶196.

In step 835, numbers A, B, C, and D are paired and distributed among data storage facilities D1-D4, in such a way that “none of the pairings contain sufficient data, by themselves, to reorganize and decipher the original sensitive data S.” Dickinson, 21:1-14; Bhattacharjee, ¶197. For example, portions A and B are never stored on the same storage system; neither are portions C and D. Dickinson, 21:3-14 (describing storing all other combinations of A-D); Bhattacharjee, ¶197. This is because the encrypted data can be decrypted by XORing it with the same number that was used to encrypt it—i.e., “when the data assembling module receives one of A and B, or, C and D, [it] can advantageously reassemble the sensitive data S.” Dickinson, 21:28-30; Bhattacharjee, ¶197.

That is what occurs when the authorized user needs access to the user's sensitive data S. Upon request from the trust engine's transaction engine (FIGs. 1-2), stored data is forwarded by the depository to a data assembling module to reassemble it into original data S for user system 105. Dickinson, 19:27-29; Bhattacharjee, ¶198. Data assembling module 525 “uses data portions from at least two of the data storage facilities D1 through D4”—since “any two” of the pairings “AC, AD, BC, and BD” will provide either both “A and B” or both “C and D”—allowing module 525 to reassemble S as “ $S = A \text{ XOR } B$ or $S = C \text{ XOR } D$.” Dickinson, 21:21-32; Bhattacharjee, ¶198.

Dickinson's trust engine includes a “stand-alone software engine implementing the [above-described] data vault functionality,” including splitting and storing data (FIG. 8) for later reassembling and serving the data to user system 105 or another requesting application. Dickinson, 29:26-30; Bhattacharjee, ¶199. The data vault software engine is shown in an annotated version of FIG. 2, below. Dickinson, 29:26-30; Bhattacharjee, ¶199.

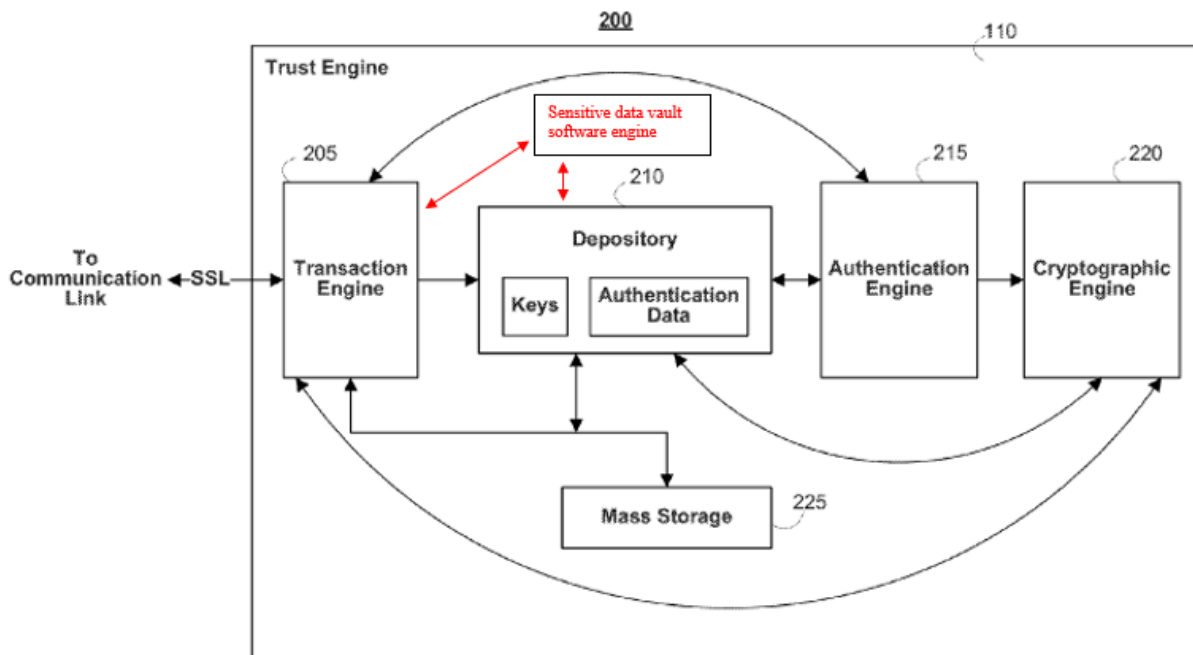


FIG. 2

B. Takeda

As explained *supra* §VII.B, Takeda teaches storing with a block of encrypted data attributes such as the encryption key and key length, or a pointer to a stored encryption key that was used to encrypt the data block. Takeda, [0040], [0054]; Bhattacharjee, ¶200.

C. Dickinson-Takeda Combination

A POSA having the teachings of Dickinson and Takeda before him/her would have readily recognized at least three alternatives for storing encrypted data portions (B and D) and the keys that encrypted them (key A that encrypted B, and key C that encrypted D):

1. Store encrypted data B with key C, and store encrypted data D with key A, as Dickinson teaches. Dickinson, 21:3-5.
2. Apply to Dickinson Takeda's teaching to store with a data portion a pointer to a key and/or metadata about the key (e.g., key length) rather than the key itself, resulting in storing encrypted data B with a pointer to and/or metadata about key C, and storing encrypted data D with a pointer to and/or metadata about key A. Takeda, [0040], [0054].
3. Further apply Takeda's teaching to store with each data portion a pointer to and/or metadata about the key that encrypted that portion, rather than a different key—resulting in storing encrypted data B with a pointer to and/or metadata about key A, and storing encrypted data D with a pointer to and/or metadata about key C. Takeda, [0048].

Bhattacharjee, ¶201.

As detailed below, POSAs had reasons to implement Dickinson's data-splitting process 800 in any of the three alternative ways above, and reasonable expectation of success in doing so. Bhattacharjee, ¶202. Any of these alternatives meets all limitations of claims 1-10, as demonstrated below. Bhattacharjee, ¶202.

1. Reasons to Modify Dickinson

Alternative (1) above requires no modification to Dickinson. Bhattacharjee, ¶203.

As to alternatives (2) and (3), Dickinson’s system must ultimately retrieve both an encrypted data portion and the key that encrypted it (e.g., either both B and A, or both D and C) to recover the original dataset S. Dickinson, 20:30-21:32. Dickinson does not specify how its system knows where to retrieve the correct key (e.g., A) for decrypting the corresponding encrypted data portion (e.g., B) to recover S. Bhattacharjee, ¶204. POSAs would have understood that Takeda’s technique of storing, with encrypted data, a pointer to and/or metadata about a key rather than the key itself (alternative (2)) (Takeda, [0040]) beneficially allows for separate management of the encrypted data and the keys for that data, for example to best use limited storage space. Takeda, [0048], [0056]; Dickinson, Bhattacharjee, ¶¶204-206; *see also* Ophir, [0032] (“...making it safe to store encryption/decryption keys on external storage instead of in limited internal memory.”). POSAs would have understood that Takeda’s further technique of storing, with each encrypted data portion, a pointer to and/or metadata about the key used to encrypt/decrypt that particular data portion (alternative (3)) provides a technique for providing the association between the encrypted data portion and its key “in such a manner that the correspondence relation between the data can be recognized” from the data/key pair being stored together. Takeda, [0048]; Bhattacharjee, ¶207. The stored metadata could also be used beneficially during decryption. Bhattacharjee, ¶207.

POSAs had numerous reasons to implement Dickinson's data-splitting process 800 by storing with an encrypted data portion (B or D) a pointer to and/or metadata about the encryption key (A or C) used to encrypt that data portion or the other data portion. Bhattacharjee, ¶208.

First, Takeda's teaching that an encryption key and encrypted data can be associated by storing a key pointer/metadata with the encrypted data would have motivated POSAs to use that technique to provide that association in Dickinson. Takeda, [0040]; Bhattacharjee, ¶209.

Second, implementing Dickinson's data-splitting process to use Takeda's pointer/metadata technique simply applies Takeda's known technique to Dickinson's device which is ready for the improvements of separate organization/storage of keys and providing an association between data-portion pairs (e.g., A and B) that together provide the original data S, with predictable results. Bhattacharjee, ¶210; *KSR*, 550 U.S. at 416.

Third, the Dickinson-Takeda alternatives are simple substitutions of one known technique (Takeda's pointer/metadata techniques) of associating encrypted data with locations of and metadata about encryption keys for another (e.g., Dickinson's undisclosed technique for forming the association). Takeda, [0040]; Bhattacharjee, ¶211; *KSR*, 550 U.S. at 416.

Fourth, Dickinson-Takeda combines prior-art elements (Takeda's pointer/metadata techniques with Dickinson's data-splitting process) according to known techniques, yielding predictable results. Bhattacharjee, ¶212; KSR, 550 U.S. at 416. Storing a pointer/metadata with the encrypted data portion provides for straightforward retrieval of the encryption key (e.g., A or C) from its storage location when needed to decrypt the encrypted data portion (e.g., B or D, respectively). Takeda, [0040], [0072]; Bhattacharjee, ¶212. The pointer's/metadata's function in Dickinson-Takeda is the same as in Takeda – to identify the location where the encryption key is stored so it can be retrieved to decrypt the encrypted portion. Bhattacharjee, ¶212.

2. Reasonable Expectation of Success

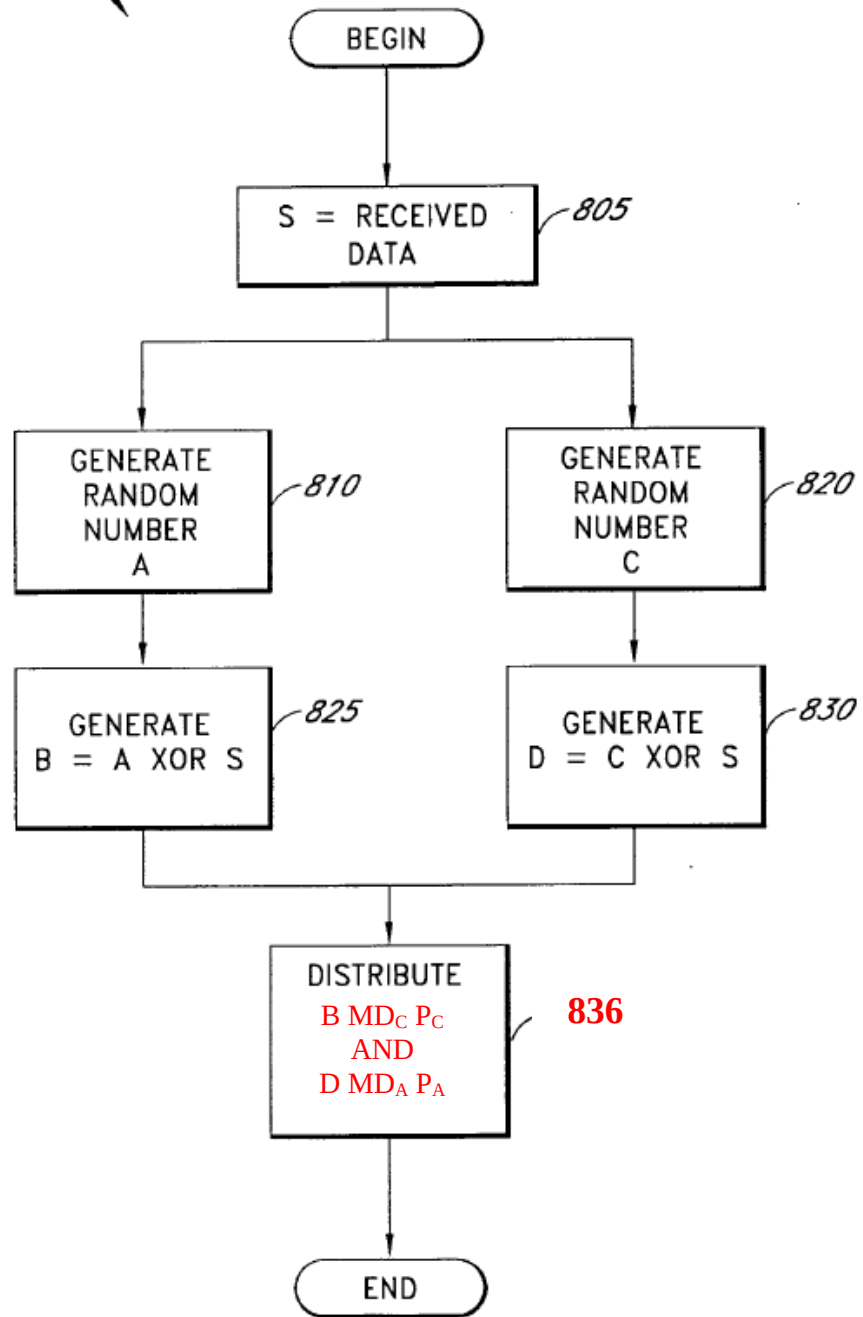
A POSA would have reasonably expected success in modifying Dickinson's FIG. 8 data-splitting process to implement Takeda's pointer/metadata techniques. Bhattacharjee, ¶¶213-15. Storing data, including metadata, was a common technique, including doing so with encrypted data. Takeda, FIGs. 3A-3B, 4, [0048]; Bhattacharjee, ¶213 (ciphering attributes include metadata such as key length). Pointers were a known software tool to identify locations of stored data. Takeda, [0044]; Bhattacharjee, ¶214 (citing EX1039, 3:35-38). POSAs readily could have implemented a pointer and/or metadata in Dickinson's FIG. 8 data-splitting process to point to the storage location of the encryption keys used in

steps 825 and 830. Takeda, [0044]; Bhattacharjee, ¶214 (citing EX1039, 3:35-38). In each alternative, Dickinson-Takeda retains the characteristic of Dickinson's system wherein pairs of values stored on the same repository (*see* step 835 – AC, AD, BC, BD) do not include encrypted data with its key (i.e., A and B are stored on different repositories, as are C and D—because the key itself is not stored with any encrypted data portions in both alternatives (2) and (3)). Dickinson, 21:1-14; *See* §X.C.3 *infra* (steps **836/837**); Bhattacharjee, ¶215.

3. Resulting Combination

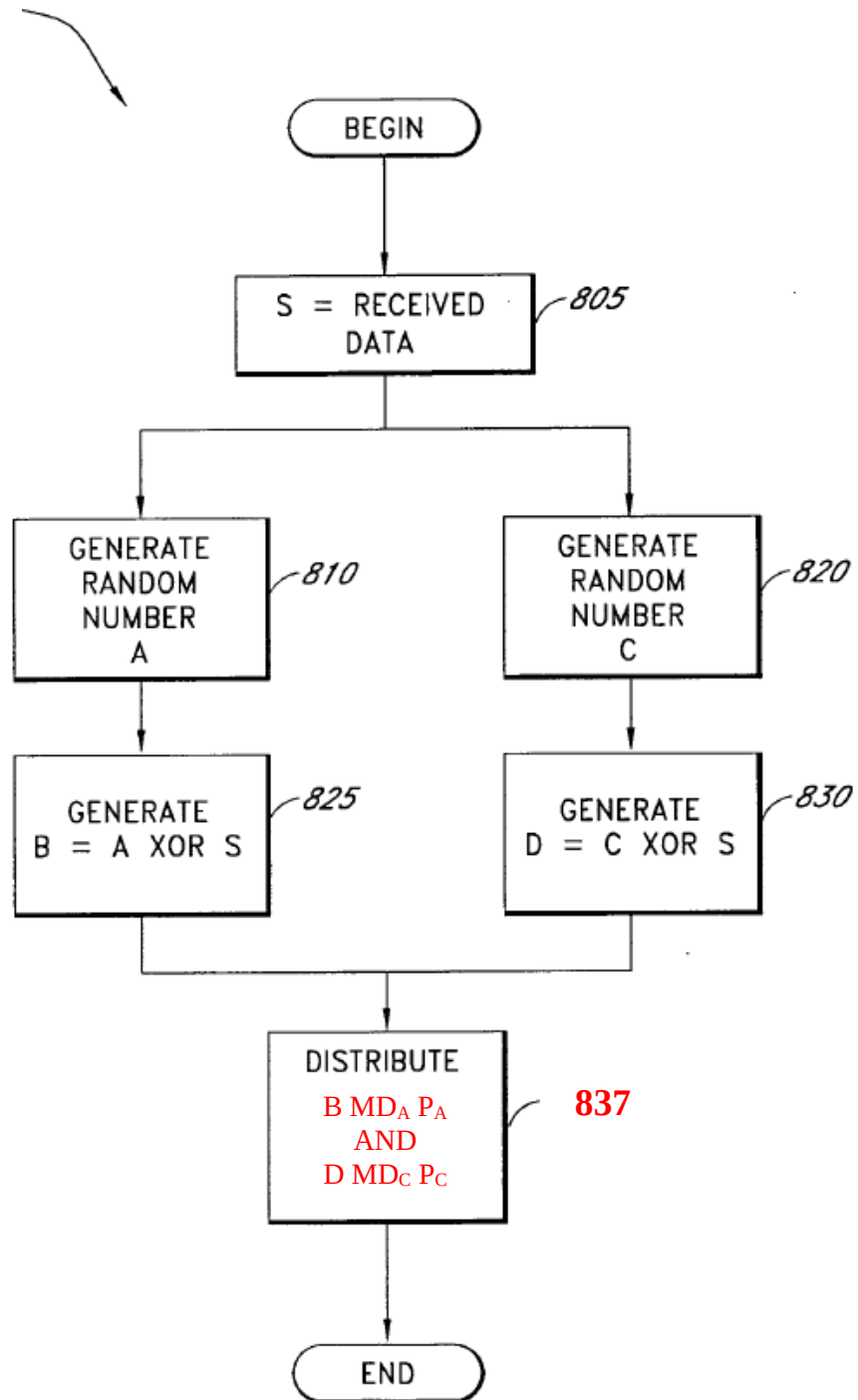
Alternative (1) does not modify Dickinson, and alternatives (2) and (3) differ from Dickinson only in implementation of a modified data-splitting process, performed by the sensitive data vault software engine, in which the encrypted data (e.g., B and D) is stored with either: (i) additional information (“MD”) about the a data string used to encrypt/decrypt a different portion of encrypted data and/or a pointer (“P”) to the data string used to encrypt/decrypt the different portion of encrypted data, represented by data splitting process 801 in the first annotated version of Dickinson's FIG. 8, below; or (ii) additional information (“MD”) about the data string used to encrypt/decrypt it and/or a pointer (“P”) to the data string used to encrypt/decrypt it (e.g., A and C, respectively), represented by data splitting process 802 in the second annotated version of Dickinson's FIG. 8 below. Bhattacharjee, ¶216.

801



First Annotated FIG. 8 of Dickinson

802



Second Annotated FIG. 8 of Dickinson

D. Dickinson-Takeda Meets All Limitations of Claims 1-10.

1. Claim 1

a. 1[PRE]

Dickinson-Takeda includes Dickinson's cryptographic system that includes user system 105, vendor system 120, trust engine 110 and certificate authority 115, all connected via communication links (e.g., networks) 125. Dickinson, 7:34-35, 10:20-22, 13:1-4, FIG. 1; Bhattacharjee, ¶¶217-19. This is the same type of cryptographic system disclosed in the '140 patent (EX1001, 6:52-53, FIG. 1), which never uses the term "secure storage network" outside of the claims, and never explains what component(s) of the cryptographic system correspond to the claimed secure storage network.

Dickinson's trust engine 110 (annotated FIG. 2, below) includes multiple components forming a network, including a sensitive data vault software engine controlling storage and retrieval of sensitive data in the depository 210. Dickinson, 10:20-22, 14:31-33. At least Dickinson's trust engine is a secure storage network as claimed. Bhattacharjee, ¶218. If patent owner argues that the secure storage network includes additional components of the cryptographic system, then Dickinson has the same components as the '140 patent because of its identically overlapping disclosure, and discloses a secure storage network in the same way. Bhattacharjee, ¶218.

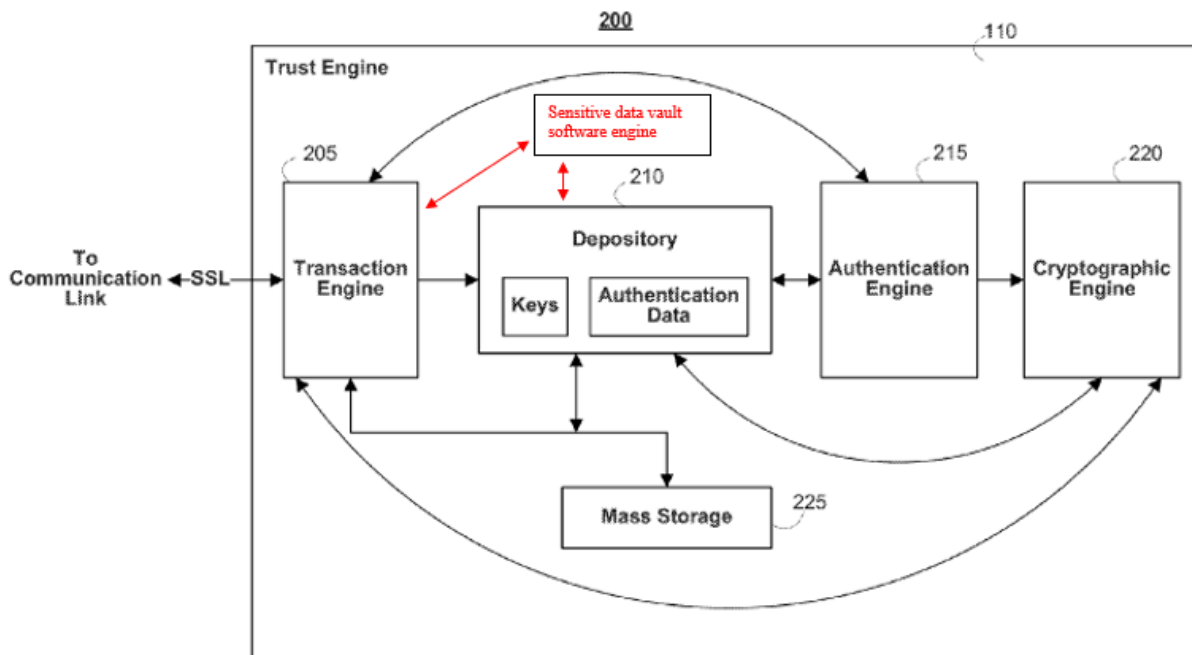


FIG. 2

b. [1A]

Dickinson-Takeda includes Dickinson’s trust engine depository system 700 having “physically separated” data storage facilities D1-D4, each being a physical storage device. Dickinson, 8:11-12, 17:4-5, 19:16-20, 20:1-3; EX1002, 1236 (Examiner cited Dickinson’s D1-D4 as physical storage devices), 1245-1246 (Applicant did not rebut); Bhattacharjee, ¶¶220-25.

Dickinson-Takeda (*supra* §X.C.3) stores encryption results (“B” and “D”) on at least two of storage devices D1-D4. Dickinson, 21:3-9; Bhattacharjee, ¶221. “B” and “D” each is a “portion” of sensitive data S, with the random number (“A” and “C,” respectively) used to do the encryption (XOR) being another “portion.”

Dickinson, 21:10-14, 18:20-28, 19:31-34. The '140 patent equates “portions” with “shares” and describes the same XOR encryption process as Dickinson’s as producing “shares.” EX1001, 58:64-59:5 (for “b = a XOR s”: “a” and “b” are “shares” or “portions” of “s”); 53:19-28 (splitting data produces portions or shares); Bhattacharjee, ¶221. Thus, Dickinson-Takeda’s data splitting process **801/802** produces shares (“B” and “D”) stored on physical storage devices D1-D4. Dickinson, 21:3-9; EX1002, 1236-37 (Examiner citing “portions” resulting from Dickinson’s data splitting process 800 as “shares”), 1245-46 (Applicant did not rebut); Bhattacharjee, ¶221.

In Dickinson-Takeda’s data splitting process **801/802**, shares “B” and “D” are associated with random numbers “A” and “C” used in the XOR processes. Dickinson, FIG. 8, steps 825 and 830; Bhattacharjee, ¶222. Each of random numbers “A” and “C” is a key used to secure dataset S since no encrypted share (B or D) is recoverable without its associated key (A or C, respectively). Dickinson, 21:27-32; Bhattacharjee, ¶222; EX1001, 58:45-59:14.

The sensitive data S can be, for example, password login, financial credentials, or the like, and therefore is a “dataset” under both parties’ proposed constructions. Dickinson, 29:15-16; EX1041, 9 (dataset is “collection of related information made up of separate elements that can be treated as a unit in data handling”); EX1042, 6 (dataset is “collection of information for storage”);

Bhattacharjee, ¶223. Any of a password login or financial credentials represents a collection of related information made up of separate elements (e.g., characters or other data elements) that can be handled as a data unit, as well as a collection of information for storage (e.g., in Dickinson’s sensitive data vault). Bhattacharjee, ¶223.

Each of random numbers “A” and “C” in Dickinson-Takeda’s data splitting process **801/802** is a session key under both parties’ constructions because it is “a key generated each time data is secured” (Google: EX1041, 13) and a “[k]ey created to both secure and restore a specific collection of information” (SFI: EX1042, 10). Bhattacharjee, ¶224. Each random number (A, C) is generated each time data S is secured and is created to secure and restore a respective dataset S. Dickinson, FIG. 8; 20:25-32; EX1001, 32:36-52, 58:45-59:14, 73:41-46; EX1002, 1236-1237 (Examiner citing to Dickinson’s FIGs. 8 and 12 and page 19, lines 31-34 and page 21, lines 10-14 as disclosing session key), 1245-1246 (Applicant did not rebut); Bhattacharjee, ¶¶224-25.

c. [1B]

Dickinson-Takeda includes a secure storage system comprising at least the transaction and data vault engines of Dickinson’s trust engine 110. *Supra* §X.D.1.a 1[PRE]; Bhattacharjee, ¶¶226-28. The ’140 patent specification does not identify a “secure storage system” but says a data splitting module as used in the

trust engine's authentication and cryptographic engines can be integrated into a "storage system." EX1001, 66:46-49 (data parser can be integrated into a storage system); 52:60-64 (equating "data splitting module" with "secure data parser"); Bhattacharjee, ¶226. Dickinson discloses a trust engine with a data vault engine that performs the same data splitting. Dickinson, FIGs. 1, 2, 5-6, 29:22-30; Bhattacharjee, ¶226. As discussed *supra* §VII.D.1.c, the claimed "secure storage system" need not include the physical storage devices because claim 1 recites the secure storage system as working with but being distinct from the "physical storage devices." Bhattacharjee, ¶226. Dickinson's transaction and data vault engines securely store data on physical storage devices D1-D4 by not storing the portions needed to recover sensitive data S on the same storage device (*supra* §X.A), and are thus a "secure storage system." Bhattacharjee, ¶¶226-27.

As explained below, Dickinson's transaction and data vault engines perform all the claimed functions of the "secure storage system." Bhattacharjee, ¶228.

i. [1B-1]

According to SFI, a virtual disk is "a system that simulates or emulates a physical storage device." EX1042, 8. Dickinson's depository 210, which may comprise FIG. 7's storage facilities D1-D4, comprises a single **logical memory** structure. Dickinson, 16:28-31, 17:4-5, 29:22-30; Bhattacharjee, ¶229. That logical memory structure is a virtual disk under SFI's construction because it is a

virtual (“logical”) representation of storage that appears to the user to be a physical disk but where the actual physical data storage facilities (e.g., D1-D4) on which the data is stored are hidden from the user. Dickinson, 16:28-31, 29:12-30:2; Bhattacharjee, ¶229; EX1002, 1237-39 (Examiner cited depository 210’s logical memory structure as a virtual disk), 1245-46 (Applicant did not rebut). Therefore, the logical memory structure simulates or emulates a physical storage device. EX1042, 8; Bhattacharjee, ¶229.

The depository’s logical memory structure comprises an index indexing a user ID against the user’s stored information. Dickinson, 16:28-31; Bhattacharjee, ¶230. Dickinson describes the index before describing the data vault engine. Dickinson, 16:28-31, 29:22-30; Bhattacharjee, ¶230. However, a POSA would have understood that in the embodiment that has a data vault engine, the data stored therein is included in the index of the user’s data. Dickinson, 54:3-4 (combinations of disclosed features “will be apparent to the skilled artisan”); Bhattacharjee, ¶230. Alternatively, that would have been the obvious way to implement what Dickinson describes so that the user can see all the user’s stored data in the index and retrieve the data the user desires to access. Bhattacharjee, ¶230.

Dickinson’s index is a directory mapped to the data storage facilities D1-D4 so that for each piece of sensitive data S in the index there is a mapping to the

particular locations among the data storage facilities D1-D4 where the data is stored. Dickinson, 16:28-31; Bhattacharjee, ¶231. To the extent Dickinson is read to not expressly disclose a directory with this mapping, that would have been the conventional and obvious way to implement Dickinson's logical memory structure so that each piece of sensitive data S included in that logical memory structure can be retrieved from data storage facilities D1-D4 when desired. Bhattacharjee, ¶231.

The logical memory structure (virtual disk) with its index (directory) is presented to a client device by Dickinson's data vault engine and transaction engine ("secure storage system"). Dickinson, 29:12-30:2; Bhattacharjee, ¶232. When the user system 105 requests the sensitive data from the sensitive data vault, the logical memory structure ("virtual disk") is presented to the user system 105 ("client device") by the transaction engine so the user can select the desired sensitive data to access. Dickinson, 29:15-16 ("provide the user access to password login, financial credentials, or the like"), 16:28-31, *generally* 29:12-30:2; Bhattacharjee, ¶232. To the extent Dickinson is read to not expressly disclose that the logical memory structure is presented to the user system 105, that would have been an obvious way to implement what Dickinson describes because, for example, the data vault stores "passwords for an almost limitless number of systems and applications" (Dickinson, 29:14-16), and a POSA would have found it desirable to provide the user with an index of the user's stored passwords so the

user can choose the password of interest to be retrieved from the data vault.

Bhattacharjee, ¶¶233-34 (citing EX1038, FIG. 9, 21:3-6).

ii. [1B-2]

Dickinson-Takeda's data vault engine performs data splitting process **801/802** (see §§X.A, X.C, *supra*) on the dataset S to generate shares B and D (see §X.A, *supra*) and distribute them for storage on data storage facilities D1-D4. Dickinson, 21:1-14; Bhattacharjee, ¶235. Data splitting process **801/802** is a securing operation on the sensitive data S ("dataset") received from the user system 105, and distributes S in shares, because it splits S into individually undecipherable portions including shares B and D. *Id.*; Bhattacharjee, ¶¶235-36; EX1002, 1238 (Examiner citing Dickinson's FIG. 8 data splitting and distribution as meeting claim limitation), 1245-46 (Applicant did not rebut).

iii. [1B-3]

Each of alternatives (1)-(3) discussed *supra* §X.C meets [1B-3], as discussed below. Bhattacharjee, ¶¶237-42.

SFI has taken the position in district court that "data indicative of" a key is met by the key itself. EX1042, 3-4 (SFI construing "data indicative of a key" as including "information that...makes known" or "shows" the key); EX1044, 34 (alleging that storing data indicative of a key is met by storing the key itself). If construed this way, Dickinson's technique in alternative (1) meets [1B-3] because

it includes with each of the shares (B and D) a key (key C is included with B, and key A is included with D) that meets the claimed “session key used to secure the dataset” (see *supra* §X.D.1.b). Bhattacharjee, ¶238. Key A secures dataset S by encrypting it to form B, and key C secures dataset S by encrypting it to form D. Dickinson, 20:32-21:3; Bhattacharjee, ¶238. To the extent SFI argued in prosecution that Dickinson does not include these keys with the shares (EX1002, 1245-46), the Examiner erred in agreeing with that mischaracterization. Bhattacharjee, ¶239.

Alternatives (2) and (3) meet both parties’ proposed constructions, including Google’s construing “data indicative of” a key as “metadata about” the key. EX1041, 5. In both alternatives (2) and (3), Dickinson-Takeda’s data vault engine performs data-splitting process **801/802** that includes with each of the shares (B and D) metadata “MD” about and/or a pointer to a key (A or C) used to encrypt the dataset S. Dickinson, Annotated FIGs. 8 (step **836/837**); Bhattacharjee, ¶240. The metadata about the key may include, for example, key length, encryption type, etc. and therefore is “data indicative of” the corresponding key under SFI’s interpretation (EX1042, 3-4) and Google’s (EX1041, 5-7). Bhattacharjee, ¶240. The pointer in step **836/837** is also data indicative of at least one session key under SFI’s interpretation because it “points to” the key (EX1042, 3). See §VII.D.1.c.iii, *supra*.

As discussed above, both A and C are “session key[s] used to secure the dataset” as claimed. In alternative (2), each is stored with a data portion other than the one it encrypted. In alternative (3), each is stored with the data portion that it encrypted. *See supra* §X.C; Bhattacharjee, ¶¶241-42.

iv. [1B-4]

Dickinson-Takeda’s data vault engine reconstitutes dataset S from at least a portion (e.g., “any two”—Dickinson, 21:23-32) of the shares stored on data storage facilities D1-D4 in response to a request from user system 105 for the sensitive data S (e.g., a password). Dickinson, 21:21-32, 29:22-30:9; Bhattacharjee, ¶¶243-44; *see supra* §X.A.

2. Claim 2

Dickinson-Takeda’s data vault engine performs Dickinson’s data splitting to generate encrypted data portions (“shares”) B and D. *See* §§X.A, X.C, *supra*; Bhattacharjee, ¶¶245-46. The XOR operations (steps 825, 830) are cryptographic operations performed on dataset S because they encrypt dataset S into undecipherable portions. Dickinson, FIG. 8, 20:32-21:1; Bhattacharjee, ¶¶245-46.

3. Claim 3

Dickinson-Takeda’s trust engine includes a key management server storing keys. Dickinson, FIG. 2 (depository 210 storing keys), 10:20-22, 13:5-7, 14:17-19; Bhattacharjee, ¶¶247-48. Based on Takeda, a POSA would have associated with the keys different access privileges for different groups of people, making them

workgroup keys meeting claim 3. Takeda, FIG. 4 (“access privilege”); Bhattacharjee, ¶¶247-48.

4. Claim 4

Dickinson-Takeda’s data vault engine reassembles (reconstitutes) dataset S from the shares in data storage facilities D1-D4 and presents it to user system 105 via the transaction engine, thus providing access to the reconstituted dataset. Dickinson, 21:21-32, 29:22-30:9; Bhattacharjee, ¶¶249-50.

5. Claim 5

Dickinson-Takeda’s secure storage system establishes via its transaction engine (*supra* §X.D.1.c) a secure connection to user system 105 using secure socket layer (SSL) communication. Dickinson, FIGs. 1-2 (communication links 125), 14:25-27 (SSL provides secure communication); Bhattacharjee, ¶¶251-52.

6. Claim 6

A POSA would have implemented Dickinson-Takeda’s data-splitting process to create the shares (B, D) to contain a substantially random distribution of dataset S, for the same reasons explained *supra* §VII.D.6 and because Dickinson teaches that randomization enhanced security. Dickinson, 18:20-28; Bhattacharjee, ¶¶253-54.

7. Claim 7

Dickinson-Takeda’s secure storage system (via its data vault) reconstitutes dataset S from *either* encrypted share B *or* D, which is fewer than all the shares (B

and D). See §§X.A, X.C *supra*; Dickinson, 21:25-32; Bhattacharjee, ¶¶255-56; EX1002, 1242 (Examiner citing ability to reconstitute sensitive data from a subset of portions created by Dickinson’s data splitting process), 1245-46 (Applicant did not rebut).

8. Claim 8

Dickinson-Takeda’s data vault engine secures dataset S by encrypting it using the at least one session key (A, C—see §§X.A, X.C *supra*). Dickinson, FIG. 8; 29:22-30; Bhattacharjee, ¶¶257-58; EX1002, 1239 (Examiner asserting Dickinson discloses claim 8 limitations), 1245-46 (Applicant did not rebut).

9. Claim 9

Dickinson-Takeda’s data vault engine secures dataset S by “splitting” it (Dickinson, 21:10-20) using at least one session key (A, C—see §§X.A, X.C, X.D.1.b *supra*). Dickinson, FIG. 8, 8:13-14, 20:22-23 (FIG. 8 is a “data splitting process”); Bhattacharjee, ¶¶259-60; EX1002 (Examiner asserting Dickinson meets claim 9), 1245-46 (Applicant did not rebut).

10. Claim 10

In Dickinson-Takeda, the claimed “at least one session key” comprises session keys A and C (§§X.A, X.D.1.b *supra*), and the data indicative of the at least one session key includes, e.g., A, pointer P_A, or metadata MD_A (step 835’ in §X.C *supra*), which is indicative of only session key A and thus indicative of a

portion of the “at least one session key” (**A and C**). Takeda, [0040]; Dickinson, FIG. 8; Bhattacharjee, ¶¶261-62.

XI. NO BASIS FOR DISCRETIONARY DENIAL

A. Section 314(a)

“[T]he PTAB will not deny institution...under *Fintiv*” if, as here, there is “compelling evidence of unpatentability.” Director’s Interim Procedure for Discretionary Denials, 5 (June 21, 2022) (“Interim Procedure”) (discussing *Apple Inc. v. Fintiv, Inc.*, IPR2020-00019, Paper 11 (Mar. 20, 2020) (precedential)). That should conclude the *Fintiv* analysis.

If considered, the *Fintiv* factors also weigh against denial.

1. Stay Potential

Petitioner will seek a pre-institution stay. Without “specific evidence” of how the court will rule, this factor is at worst neutral. *Sand Revolution II v. Continental Intermodal Group–Trucking*, IPR2019-01393, Paper 24, 7 (June 16, 2020) (informative).

2. Trial Timing

The district-court trial is presently scheduled for July 2024. EX1045, 1. This factor is not dispositive especially where, as here, the petition presents strong merits. *See, e.g., PEAG LLC d/b/a JLab Audio v. VARTA Microbattery GmbH*, IPR2020-01214, Paper 8, 19-20 (Jan. 6, 2021).

3. Litigation Investment

Though there has been some activity in the district-court litigation, Petitioner's diligence in filing this Petition about eight months from being served with a complaint and about three months after SFI identified which claims would be asserted weighs against discretionary denial. *Fintiv*, 11-12 (expeditious filing weighs against denial); *Dish Network LLC v. Broadband iTV, Inc.*, IPR2020-01332, Paper 14, 28-29 (Jan. 27, 2021) (finding petition diligently filed within three months of receiving patent owner's full set of asserted claims); EX1044 (SFI's 2023-08-14 interrogatory response identifying asserted claims).

4. Issue Overlap

Given its early stage, the validity issues in the litigation are unknown. However, this Petition challenges ten claims, whereas only six claims are yet asserted in the litigation. See EX1044. This weighs against denial. *Markforged Inc. v. Continuous Composites*, IPR2022-00679, Paper 7, 32-33 (Oct. 25, 2022).

5. Litigation Defendants

That Petitioner is the defendant in the district-court proceeding "is far from an unusual circumstance." *Sand Revolution II*, 12-13. This factor is neutral.

6. Other Circumstances

The Petition's "strong evidence on the merits" weighs against denial. Interim Procedure, 3-5.

B. Section 325(d)

Grounds 1-2 present new art (Ophir, Takeda, and Birrell) not considered during prosecution. This fact alone warrants consideration of the Petition’s merits. *ShenZhen Apaltek Co., Ltd. v. Asetek Danmark A/S*, IPR2022-01317, Paper 7, 8-10 (Feb. 6, 2023).

Grounds 3-4 combine new art (Ophir, Takeda) with Dickinson which was applied during prosecution. EX1002, 1235-42. Takeda discloses “the very [limitation] that, in the Examiner’s view, placed the claims...into a condition for issuance” over Dickinson, and the Examiner “did not have the benefit of the teachings of [Takeda]”—in such cases, the Board “do[es] not regard a discretionary denial of institution under...§ 325(d) as appropriate.” *SHDS v. TruInject*, IPR2020-00937, Paper 11, 11-12 (Nov. 17, 2020).

XII. CONCLUSION

The Board should institute review and cancel claims 1-10.

Dated: 2023-11-27

Respectfully submitted,
Google LLC,

/Elisabeth Hunt/

Elisabeth H. Hunt, Reg. No. 67,336
WOLF, GREENFIELD & SACKS, P.C

XIII. APPENDIX: CLAIM LISTING

The following claim listing assigns element labels (*e.g.*, 1[PRE], [1A], etc.) to certain claims for convenience of reference.

Claim 1
1[PRE]. A secure storage network comprising
[1A] a plurality of physical storage devices storing thereon a plurality of shares, the plurality of shares being associated with at least one session key used to secure a dataset; and
[1B] a secure storage system configured to:
[1B-1] present to a client device a virtual disk, the virtual disk comprising a directory mapped to the plurality of physical storage devices such that physical locations of the shares are hidden from the client device;
[1B-2] generate the plurality of shares for storage on the plurality of physical storage devices by performing a securing operation on the dataset received from the client device and distributing the dataset in the shares;
[1B-3] include with each of the plurality of shares data indicative of the at least one session key used to secure the dataset; and
[1B-4] reconstitute the dataset from at least a portion of the plurality of shares stored on the physical storage devices in response to a request from the client device for information in the dataset.
Claim 2
2. The secure storage network of claim 1, wherein the secure storage system is further configured to generate the plurality of shares by performing a cryptographic operation on the dataset.
Claim 3
3. The secure storage network of claim 1, further comprising a key management server configured to store workgroup keys.
Claim 4
4. The secure storage network of claim 1, wherein the secure storage system is further configured to provide access to the reconstituted dataset.
Claim 5
5. The secure storage network of claim 1, wherein the secure storage system is further configured to establish a secure connection between the client device and the secure storage system.

Claim 6
6. The secure storage network of claim 1, wherein the shares contain a substantially random distribution of the received dataset.
Claim 7
7. The secure storage network of claim 1, wherein the secure storage system is configured to reconstitute the received dataset from fewer than all of the shares.
Claim 8
8. The secure storage network of claim 1, wherein the secure storage system is configured to secure the dataset by encrypting the dataset using the at least one session key.
Claim 9
9. The secure storage network of claim 1, wherein the secure storage system is configured to secure the dataset by splitting the dataset using the at least one session key.
Claim 10
10. The secure storage network of claim 1, wherein the data indicative of the at least one session key comprises: a portion of the at least one session key or data indicative of a portion of the at least one session key.

CERTIFICATE OF SERVICE UNDER 37 C.F.R. § 42.6 (E)(4)

I certify that on November 27, 2023, I will cause a copy of the foregoing document, including any exhibits or appendices filed therewith, to be served via Priority Express Mail at the following correspondence address of record for the patent:

Security First Innovations, LLC
C/O Farjami & Farjami, LLP
26522 La Alameda Ave., Suite 360
Mission Viejo, CA 92691

Date: November 27, 2023

/MacAulay Rush/
MacAulay Rush
Paralegal
WOLF, GREENFIELD & SACKS, P.C.

CERTIFICATE OF WORD COUNT

Pursuant to 37 C.F.R. § 42.24, the undersigned certifies that the foregoing Petition for *Inter Partes* Review contains 13,984 words excluding a table of contents, a table of authorities, Mandatory Notices under § 42.8, a certificate of service or word count, or appendix of exhibits or claim listing. Petitioner has relied on the word count feature of the word processing system used to create this paper in making this certification.

Date: November 27, 2023

/MacAulay Rush/
MacAulay Rush
Paralegal
WOLF, GREENFIELD & SACKS, P.C.