

AT UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

NETSKOPE, INC.,

Petitioner,

v.

FORTINET, INC.,

Patent Owner.

PTAB Case No. IPR2023-00458

Patent No. 9,280,678 B2

**PETITION FOR *INTER PARTES* REVIEW
OF U.S. PATENT NO. 9,280,678 B2**

TABLE OF CONTENTS

Page

EXHIBIT LIST	III
I. INTRODUCTION	5
II. MANDATORY NOTICES	5
A. Real party in interest.....	5
B. Related matters	5
C. Counsel and service information.....	5
III. IPR REQUIREMENTS	6
A. Standing.....	6
B. Challenged claims and relief requested.....	6
1. Prior art	6
2. Challenged claims and grounds	7
IV. TECHNOLOGY DESCRIPTION	7
A. Background	7
1. Cloud Storage.....	7
2. Searchable Encryption	8
3. Namespace	9
B. '678 Patent.....	11
1. Alleged invention.....	11
2. Prosecution history.....	14
V. SECTION 325(D).....	15
VI. LEVEL OF ORDINARY SKILL	15
VII. CLAIM CONSTRUCTION	15
VIII. GROUNDS	15
A. Ground 1: Cidon, Shikfa, and Herrmann render claims 1-3, 8-14, and 16-27 obvious.	15
B. Ground 2: Cidon, Shikfa, Herrmann, and Fellenstein render claim 6 obvious.	69

TABLE OF CONTENTS
(continued)

	Page
C. Ground 3: Cidon, Shikfa, Herrmann, and Hirai render claim 15 obvious.	71
D. Ground 4: Auradkar, Chiueh, and Chambers render claims 1-3, 8-19, and 21-27 obvious.	72
E. Ground 5: Auradkar, Chiueh, Chambers, and Inoue render Claim 6 obvious.	135
IX. CONCLUSION.....	138
LISTING OF CHALLENGED CLAIMS.....	140

EXHIBIT LIST

Exhibit	Description
Ex. 1001	U.S. Patent No. 9,280,678 to Redberg (“the ’678 Patent”)
Ex. 1002	Supporting Declaration of Dr. Wenke Lee (“Lee”)
Ex. 1003	File History of U.S. Patent No. 9,280,678
Ex. 1004	U.S. Patent Application Publication No. 2014/0013112 (“Cidon”)
Ex. 1005	U.S. Patent Application Publication No. 2003/0055994 (“Herrmann”)
Ex. 1006	U.S. Patent Application Publication No. 2014/0359282 (“Shikfa”)
Ex. 1007	U.S. Patent Application Publication No. 2005/0010593 (“Fellenstein”)
Ex. 1008	U.S. Patent No. 6,622,248 (“Hirai”)
Ex. 1009	U.S. Patent Application Publication No. 2011/0119481 (“Auradkar”)
Ex. 1010	U.S. Patent Application Publication No. 2014/0068030 (“Chambers”)
Ex. 1011	U.S. Patent Application Publication No. 2013/0159694 (“Chiueh”)
Ex. 1012	U.S. Patent Application Publication No. 2003/0063321 (“Inoue”)
Ex. 1013	U.S. Patent Application Publication No. 2002/0099666 (Dryer)
Ex. 1014	Dictionary of Computer and Internet Terms, Tenth Edition (2009)
Ex. 1015	U.S. Patent Application Publication No. 2011/0145594 (“Ex. 1015”)
Ex. 1016	U.S. Patent Application Publication No. 2009/0300351 (“Ex. 1016”)
Ex. 1017	U.S. Patent Application Publication No. 2008/0069480 (“Ex. 1017”)
Ex. 1018	Song, D. X., Wagner, D., & Perrig, A. (2000, May). Practical techniques for searches on encrypted data. In Proceeding 2000 IEEE symposium on security and privacy. S&P 2000 (pp.44-55) (“Song”)

Ex. 1019	Agrawal, R., Kiernan, J., Srikant, R., & Xu, Y. (2004, June). Order preserving encryption for numeric data. In Proceedings of the 2004 ACM SIGMOD international conference on Management of data (pp.563-574) (“Agrawal”)
Ex. 1020	Adya, A., Bolosky, W. J., Castro, M., Cermak, G., Chaiken, R., Douceur, J. R., Howell, J., Lorch, J. R., Theimer, M., and Wattenhofer, R. P. 2002. Farsite: federated, available, and reliable storage for an incompletely trusted environment. SIGOPS Oper. Syst. Rev. 36, SI (Dec. 2002), 1-14 (“Ex. 1020”)
Ex. 1021	Mika, P., & Potter, T. (2012). Metadata Statistics for a Large Web Corpus. LDOW, 937 (“Ex. 1021”)
Ex. 1022	Microsoft Computer Dictionary, Fifth Edition (2002)
Ex. 1023	Balakrishnan, H., Lakshminarayanan, K., Ratnasamy, S., Shenker, S., Stoica, I., & Walfish, M. (2004). A layered naming architecture for the Internet. ACM SIGCOMM Computer Communication Review, 34(4), 343-352 (“Ex. 1023”)

I. INTRODUCTION

U.S. Patent No. 9,280,678 (“the ’678 Patent”) relates to cloud storage that uses a naming convention (i.e., “namespace”) to search encrypted files. But cloud storage is decades older, and searching encrypted files and using namespaces were commonplace. Accordingly, for the reasons below, Petitioner requests that the Board institute IPR and cancel ’678 claims 1-3, 6, 8-17, and 21-27.

II. MANDATORY NOTICES

A. Real party in interest

Netskope, Inc. (“Petitioner”) is the sole real party-in-interest.

B. Related matters

Petitioner has filed a civil action seeking a declaratory judgment that it does not infringe the ’678 patent in the United States District Court for the Northern District of California, Case No. 3:22-CV-01852-TLT. That case is currently stayed. Petitioner has challenged the validity of other patents allegedly owned by the Patent Owner in co-pending *inter partes* review proceedings (*See* IPR2023-00456, IPR2023-00457, IPR2023-00459, IPR2022-01587, IPR2023-00175, IPR2023-00030).

C. Counsel and service information

Lead counsel: Thomas N. Millikan, Reg. No. 72,316.

Back-up counsel: Babak Tehrani (Reg. No. 55,937), Wei Yuan (Reg. No. 71,772), Andrew N. Klein (*pro hac* vice forthcoming).

These attorneys can be reached by mail at Perkins Coie LLP, 11452 El Camino Real, Suite 300, San Diego, CA 92130-2080; by phone at (858) 720-5700; and by fax at (858) 720-5799.

Petitioners consent to electronic service. All services and communications to the attorneys listed above may be sent to:

Netskope-FortinetIPRTeam@perkinscoie.com.

A Power of Attorney is being filed concurrently.

III. IPR REQUIREMENTS

A. Standing

The '678 Patent is available for IPR and Petitioner is not barred or estopped from challenging it on the grounds herein.

B. Challenged claims and relief requested

Petitioner respectfully requests cancellation of the challenged claims under §103.

1. Prior art

Petitioner relies on Exhibits 1004 to 1013 in the Table of Exhibits, all of which are prior art under §102(a)(1) and/or §102(a)(2).

Petitioner also relies on the declaration of Dr. Wenke Lee ("Lee" (Ex. 1002)).

2. Challenged claims and grounds

Ground	Basis	Reference(s)	Challenged Claims
1	§103	Cidon, Shikfa, Herrmann	1-3, 8-14, 16-19, 21-27
2	§103	Cidon, Shikfa, Herrmann, Fellenstein	6
3	§103	Cidon, Shikfa, Herrmann, Hirai	15
4	§103	Auradkar, Chiueh, Chambers	1-3, 8-19, 21-27
5	§103	Auradkar, Chiueh, Chambers, Inoue	6

IV. TECHNOLOGY DESCRIPTION

A. Background

1. Cloud Storage

Cloud storage has existed since at least 1983, when CompuServe allowed users to upload files to online disk space. Cloud storage became popular with consumers in the late 2000s. For example, Dropbox had over 1 million users by April 2009. By 2010, Dropbox offered the ability to search encrypted files. (Ex. 1024; Lee, ¶¶34-35.)

The Wayback Machine - <https://web.archive.org/web/20100730120646/http://www.dropbox.com:80/features>

[Log in](#)



[Back to the homepage](#)

Dropbox Features

Web Access

A copy of your files are stored on Dropbox's secure servers. This lets you access them from any computer or mobile device.

Manipulate files as you would on your desktop - add, edit, delete, rename etc.

Search your entire Dropbox for files.

A "Recent Events" feed that shows you a summary of activity in your Dropbox.

Create shared folders and invite people to them.

Recover previous versions of any file or undelete deleted files.

Security & Privacy

Dropbox takes the security and privacy of your files very seriously.

Shared folders are viewable only by people you invite.

All transmission of file data and metadata occurs over an encrypted channel (SSL).

All files stored on Dropbox servers are encrypted (AES-256) and are inaccessible without your account password.

Dropbox website and client software have been hardened against attacks from hackers.

Dropbox employees are not able to view any user's files.

Online access to your files requires your username and password.

Public files are only viewable by people who have a link to the file(s). Public folders are not browsable or searchable.

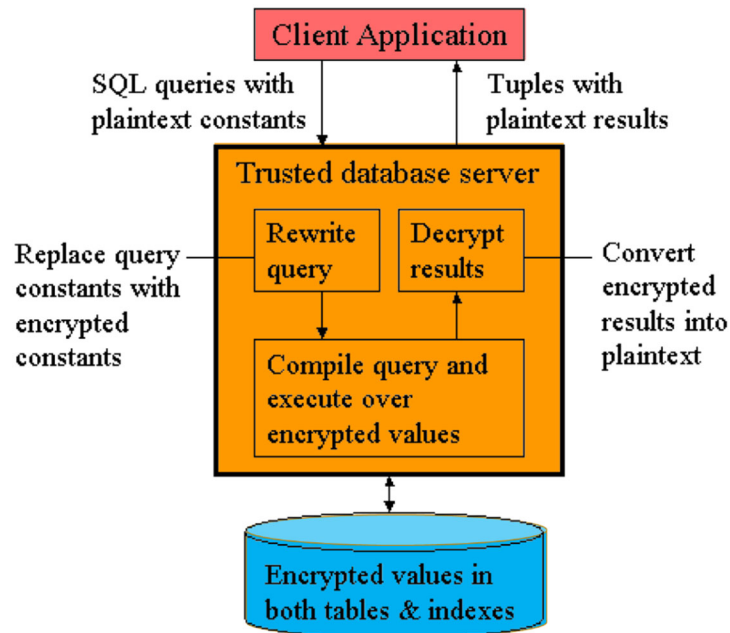
Ex. 1024 (excerpt; annotated)

2. Searchable Encryption

Users have been able to search encrypted data since the 1980s. In the 1990s, researchers developed more advanced searchable encryption schemes, including “keyword search scheme.” (Song; Lee, ¶¶36-37.) This scheme allowed users to search remotely-stored documents using an encrypted index. That index contained encrypted keywords and corresponding pointers to documents without revealing plaintext keywords to the searcher.

In 2004, researchers proposed a searchable encryption scheme for records

stored in remote, untrusted storage (**database**). The scheme used a trusted **database server** as a proxy between a **client device** and the storage. (Agrawal, §3, FIG. 1.)



Agrawal, FIG. 1 (annotated)

The database server received plaintext queries from a client application, encrypted them, then searched the database without revealing the search content to the database. (*Id.*, §3.)

Since the early 2000s, searchable encryption has continued to evolve and has been applied to various applications, including secure file storage, secure email, and secure search engines. (Lee, ¶40.)

3. Namespace

Using naming conventions to identify objects is a conceptual framework for

organizing and managing data and resources introduced decades ago. For example, in 2002 Adya described a file storage system using a hierarchical directory namespace. (Ex. 1020.) Similarly, Mika tabulated the most commonly used URL namespaces (below), illustrating that a URL namespace can have different sections, such as a domain (e.g., www.facebook.com) and a subdirectory (e.g., 2008). (Ex. 1021.)

Namespace	URLs
ogp.me/ns#	493,443,016
www.facebook.com/2008/	150,246,016
www.w3.org/1999/02/22-rdf-syntax-ns#	26,402,165
rdf.data-vocabulary.org/#	19,413,470
purl.org/dc/terms/	16,424,800
https:// www.facebook.com/2008/	7,472,815
mixi-platform.com/ns#	6,323,861
ogp.me/ns/fb#	4,636,260
creativecommons.org/ns#	4,622,272
www.w3.org/2006/vcard/ns#	4,205,037
http://	3,881,321
http://www.facebook.com/	3,126,045
http://www.w3.org/2000/01/rdf-schema#	3,042,839
http://developers.facebook.com/schema/	2,720,567
http://search.yahoo.com/searchmonkey/commerce/	2,664,743
http://purl.org/dc/elements/1.1/	2,642,796
http://opengraphprotocol.org/schema/	2,293,024
http://search.yahoo.com/searchmonkey/media/	2,095,577
http://oexchange.org/spec/0.8/rel/	2,034,467
http://xmlns.com/foaf/0.1/	1,837,749

Table 5: Top namespaces in RDFa as measured by the number of URLs

Although a namespace ensures that every object within it can be identified uniquely by its name(s), namespaces need not be hierarchical or complex. For

example, in the table above, one namespace was simply “http://.” Namespaces can have a flat structure forming a single collection of unique names. (Lee, ¶42; Ex. 1022.) As Balakrishnan (2004) explained, a flat namespace can simply be a number chosen from within a range (e.g., 0 to $2^{128}-1$). (Ex. 1023, p.344.) Alternatively, a namespace can provide the context of identifiers in a program, e.g., variables, functions, such that an identifier is referred to by its unique short, local name within the namespace and by its unique fully-qualified name outside the namespace. Other namespace examples include assigning names and groupings to resources.

B. '678 Patent

1. Alleged invention

The '678 is directed to the well-known concept of searchable encryption schemes using encrypted keyword indices. (Lee, ¶49.)

'678 Figure 1 illustrates a **cloud gateway** that “acts as an interface between **clients 102** and **stores 114**” and handles “different file/data read/write requests.”¹ ('678 Patent, FIG. 1, 7:9-14.)

¹ All **bold** or *italic* emphases are added by Petitioner unless otherwise specified.

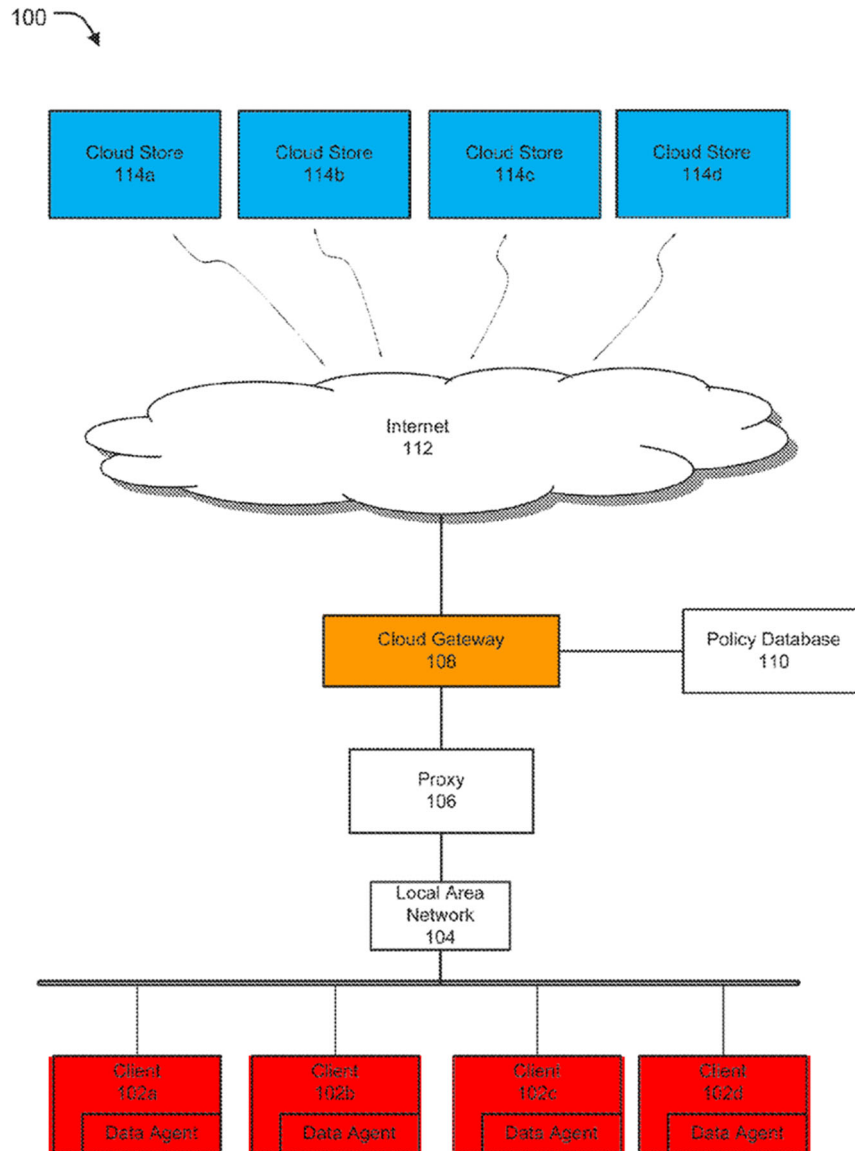


Fig. 1(Annotated)

Here, the gateway receives a file, partitions the file into chunks, and encrypts each chunk “such that [n]amespace is [c]reated for each [f]ile [c]hunk.” (*Id.*, FIG. 3A, 11:49-64.) The gateway also encrypts “keywords” (“search terms”) into “searchable indices” and stores the encrypted keywords as filenames for the chunks. (*Id.*, 12:7-18.) The gateway stores the encrypted chunks to a cloud store.

(*Id.*, 12:24-26.)

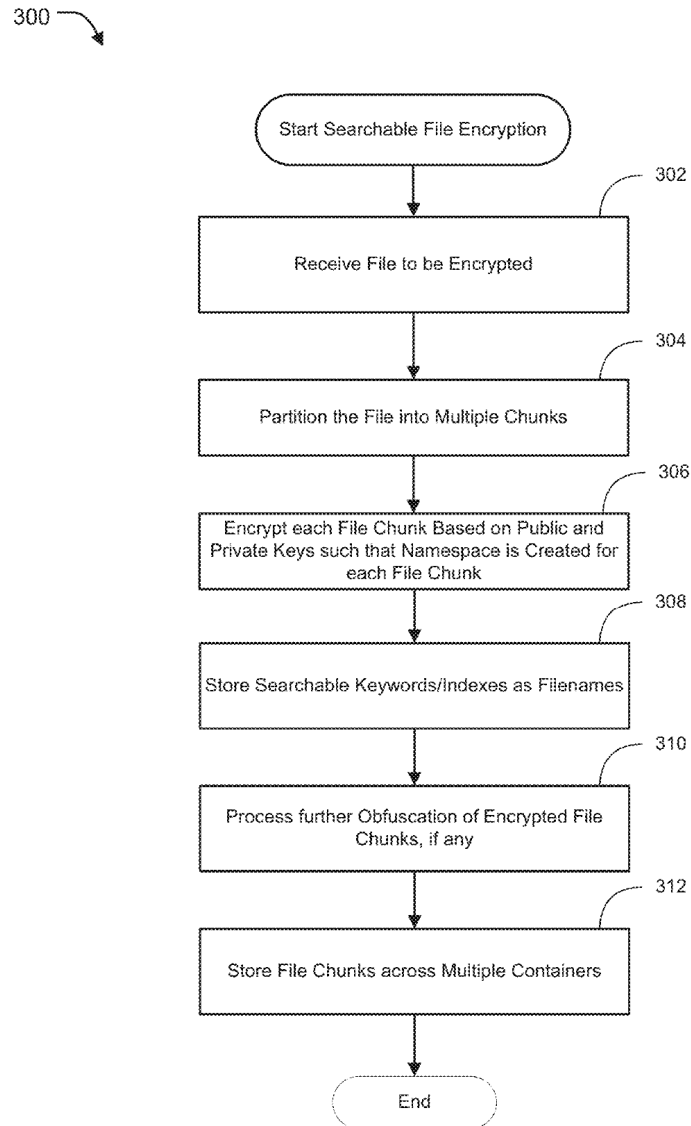


Fig. 3A

To search encrypted files, the gateway receives a query, extracts keywords, converts them into searchable indices (i.e., encrypted keywords), and identifies chunks matching the searchable indices. (*Id.*, 12:32-55, FIG. 3B.)

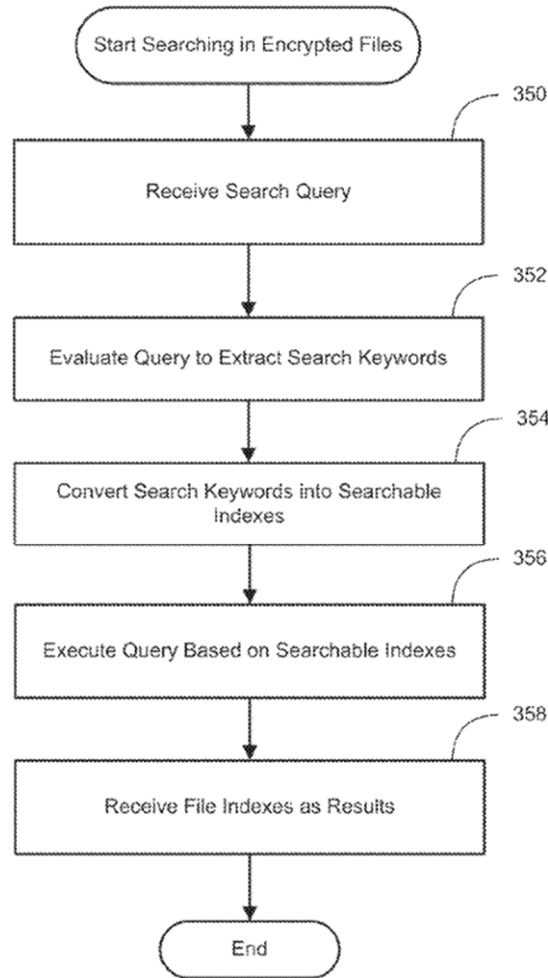


Fig. 3B

2. Prosecution history

The '678 Patent issued after one office action, which indicated that original claims 8-11 and 23-25 contained allowable subject matter. To obtain allowance, Applicant amended the independent claims by incorporating claim 8's subject matter, which relates to creating and configuring namespaces to make encrypted files searchable. (*See* Ex. 1003, 119, 138, 143, 153-154.)

V. SECTION 325(D)

The Board should review the '678 Patent notwithstanding its §325(d) discretion. *Becton, Dickinson & Co. v. B. Braun Melsungen AG*, IPR2017-01586, Paper 8 (P.T.A.B. Dec. 15, 2017); *Advanced Bionics, LLC v. Med-El Electromedizinische Geräte GMBH*, IPR2019-01469, Paper 6 at 10 (P.T.A.B. Feb. 13, 2020) (precedential).

None of the references herein were before the examiner during original examination. They present new and distinct invalidity challenges that better reflect the state of the art than what was originally considered.

VI. LEVEL OF ORDINARY SKILL

As of the '678 Patent's filing date, a POSITA would have had a B.S. in computer science, computer engineering, or electrical engineering, with at least two years' experience in computer networking/security. (Lee, ¶28.)

VII. CLAIM CONSTRUCTION

The challenged claims should be construed according to their ordinary and customary meaning. 37 C.F.R. §42.200(b); *see Phillips v. AWH Corp.*, 415 F.3d 1303 (Fed. Cir. 2005). Petitioner believes no constructions are necessary.

VIII. GROUNDS

A. Ground 1: Cidon, Shikfa, and Herrmann render claims 1-3, 8-14, and 16-27 obvious.

Cidon: Cidon concerned storing and accessing encrypted files in cloud-

based storage. (Abstract, ¶¶83, 100.)

Figure 2 (below) depicted a **management server 100** that accessed enterprise file systems, was connected to consumer file repositories (e.g., **Dropbox 13**, **Gmail 11**), and “encrypt[ed] files in the cloud and enforce[ed] secure access to these cloud files.” (*Id.*, ¶¶100, 102, FIG. 2.)

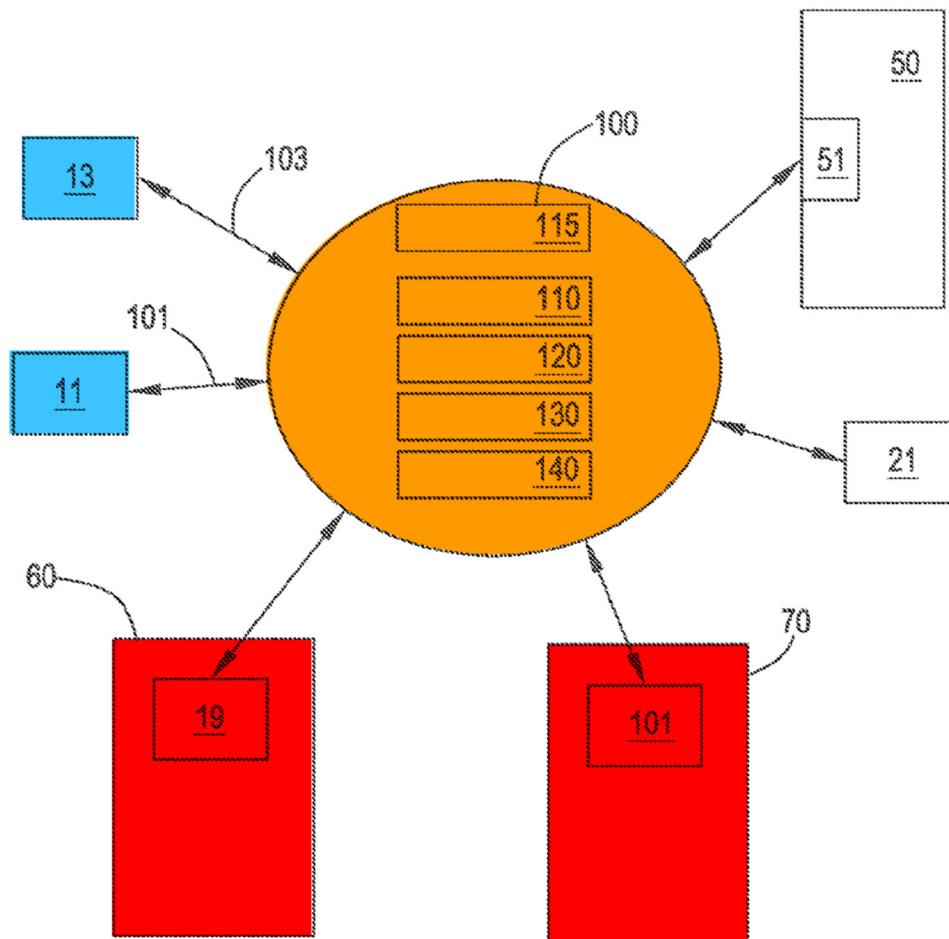


FIG. 2 (Annotated)

Figures 13 and 15 show the management server divided a **file** into multiple **file segments** and encrypted them. (*Id.*, FIGS. 13, 15, ¶¶305, 309.) The

management server sent the encrypted file to a cloud storage service. (*Id.*, ¶309.)

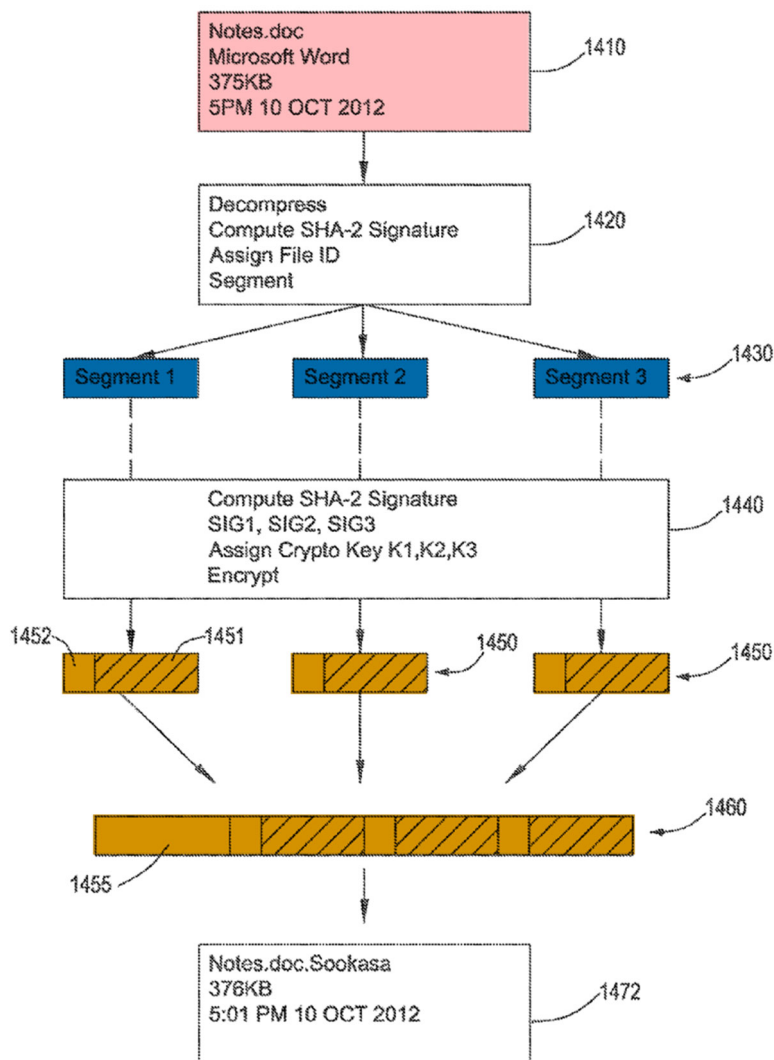


FIG. 13 (Annotated)

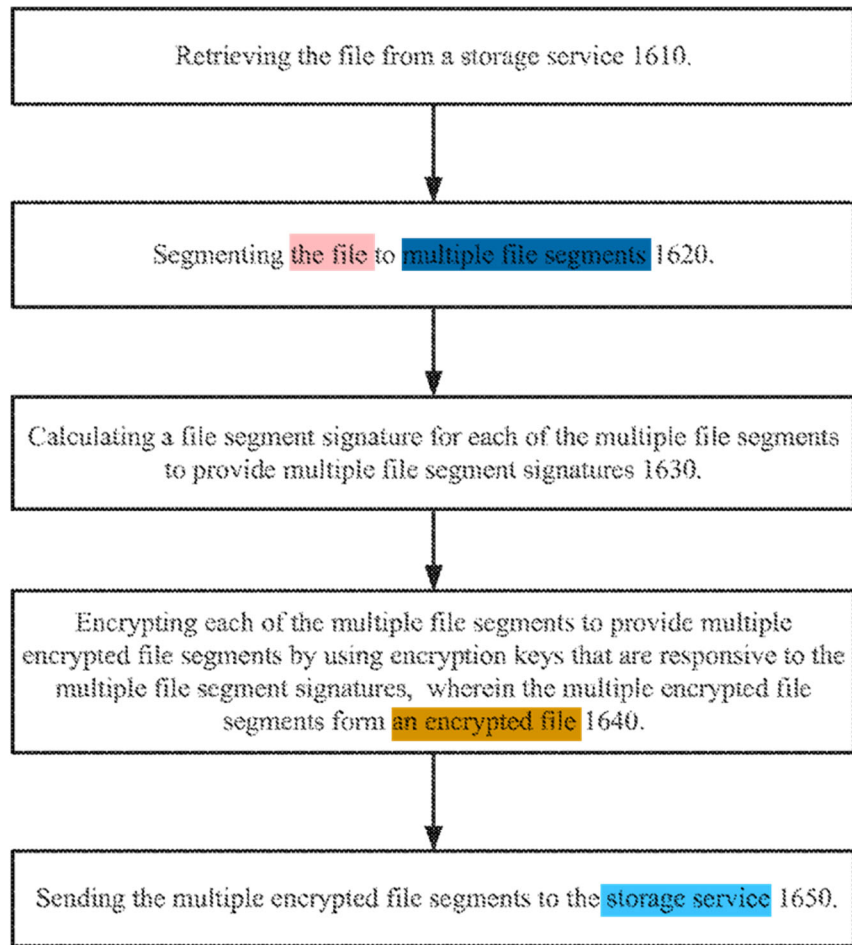


FIG. 15 (Annotated)

The management server also “provide[d] access policy configuration for secure file access,” e.g., for **user devices 60** and **70**, and “enforce[d] access policy for secure file access.” (*Id.*, ¶102, FIG. 2.)

Herrmann: Herrmann concerned implementing policies for accessing data stored in protected resources by client devices. (Abstract.)

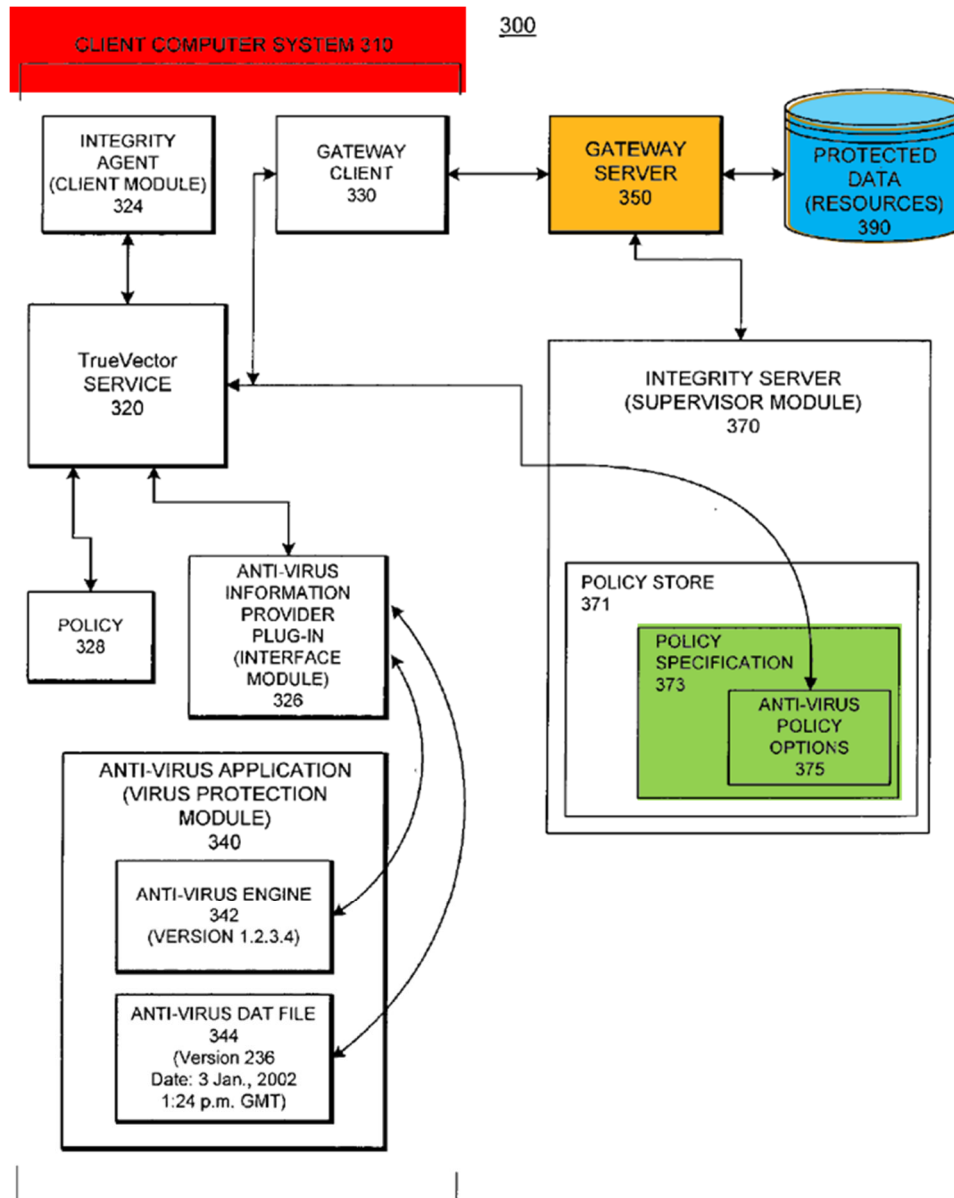


FIG. 3 (Annotated)

Herrmann disclosed a **gateway server 350** that “accept[ed] requests for access to” **protected data or resources 390** from **client machines**. (*Id.*, ¶60, FIG. 3.) Herrmann disclosed a policy store 371, from which “appropriate policy specification 373 applicable to the client computer 310” could be retrieved. (*Id.*,

¶67.) Herrmann’s gateway server “assign[ed] an **access policy** to the connection/session between the gateway server 350 and the client computer 310).” (*Id.*, ¶77.)

Shikfa: Shikfa concerned encrypting files and storing them in the cloud. (Shikfa, ¶3.) Shikfa disclosed a “broker system [that] transfer[red] the **encrypted documents** between the **client** and the **storage providers**.” (*Id.*, ¶3, FIG. 2.)

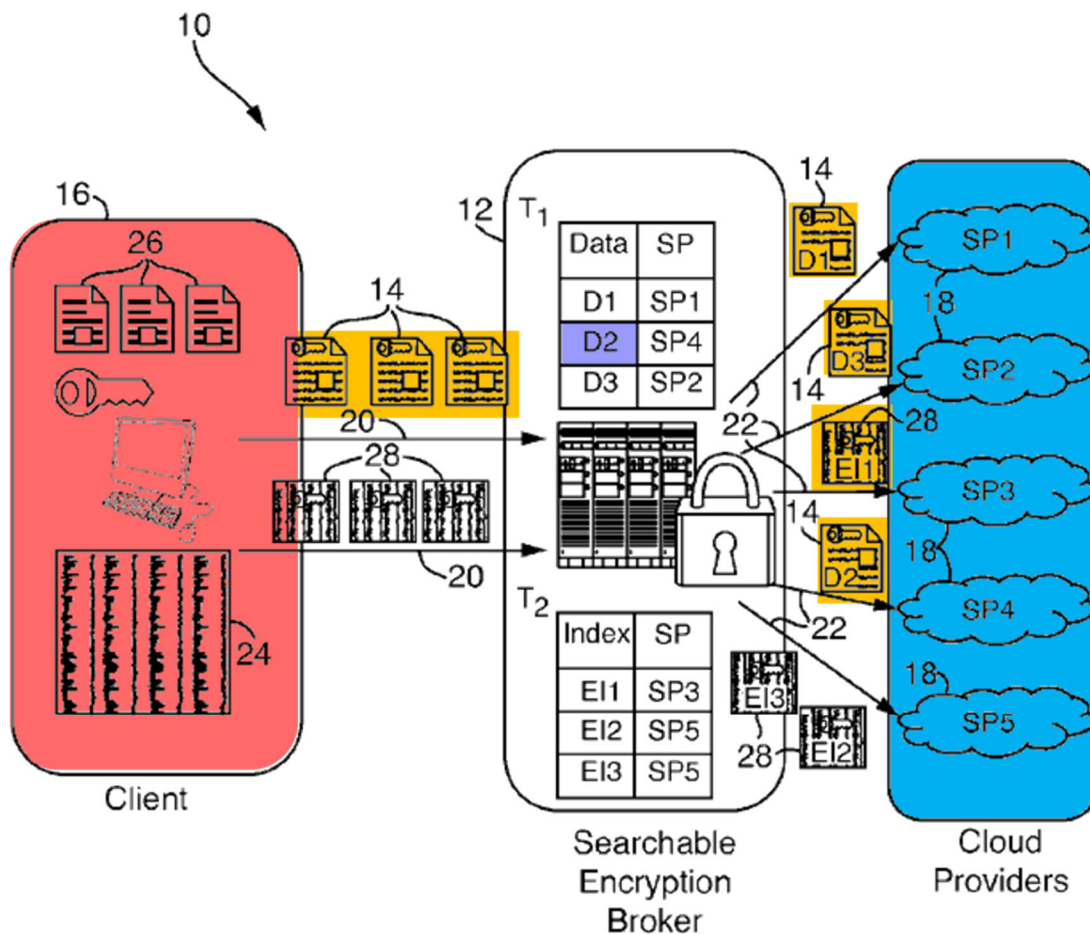


FIG. 2 (Annotated)

Like the '678 Patent, when searching encrypted documents, Shikfa used an

encrypted query to search indexes containing encrypted keywords and returned the identified encrypted documents to the client. (*Id.*, ¶12, FIG. 3.)

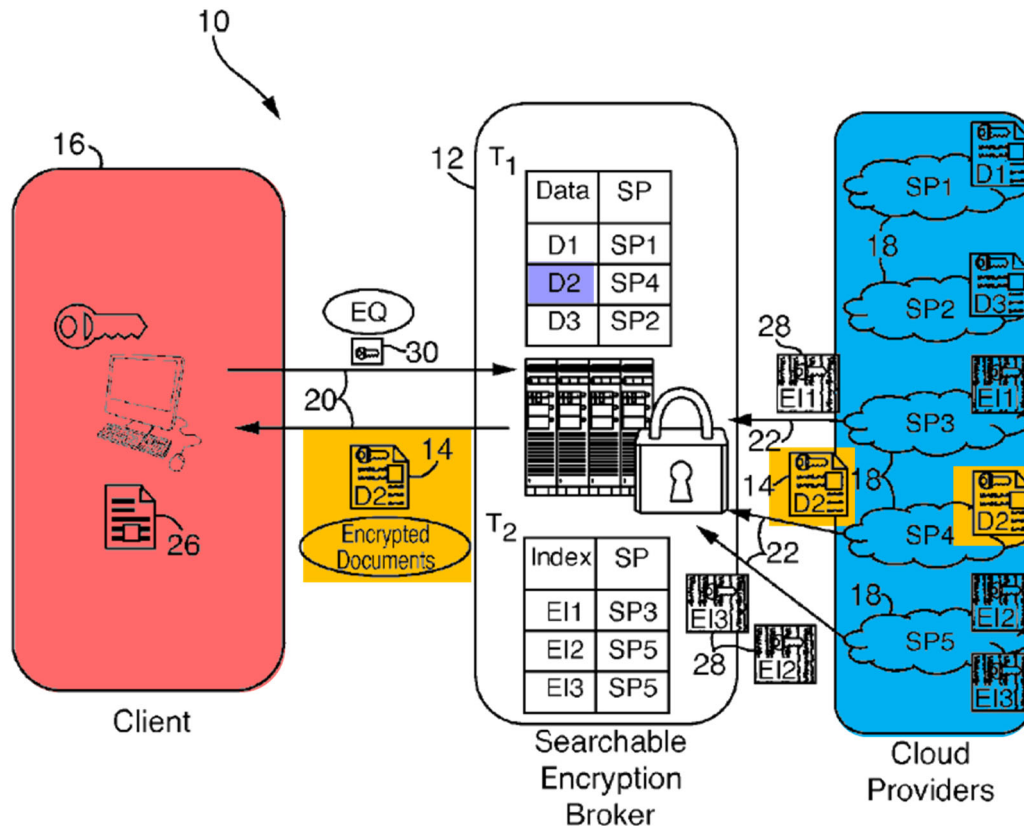


FIG. 3 (Annotated)

Notably, Shikfa's system configured identifiers (namespaces) to map encrypted documents to potential search keywords, which allowed encrypted documents to be searched. (*Id.*, ¶¶29, 31, 36, FIG. 2.)

1. Motivation to combine

The Cidon-Herrmann-Shikfa combination would have entailed implementing Cidon's searchable encryption procedure on Herrmann's gateway

device, and modifying the search methodology with Shikfa's teachings. A POSITA would have been motivated to make this combination. (Lee, ¶67.)

First, Cidon, Shikfa, and Herrmann concerned improvements to client devices securely accessing files stored in a remote location. (Cidon, ¶100; Shikfa, ¶3; Herrmann, Abstract.) A POSITA would have found it obvious that Cidon's system benefited from Shikfa and Herrmann's complementary techniques with the similar goal. (Lee, ¶68.)

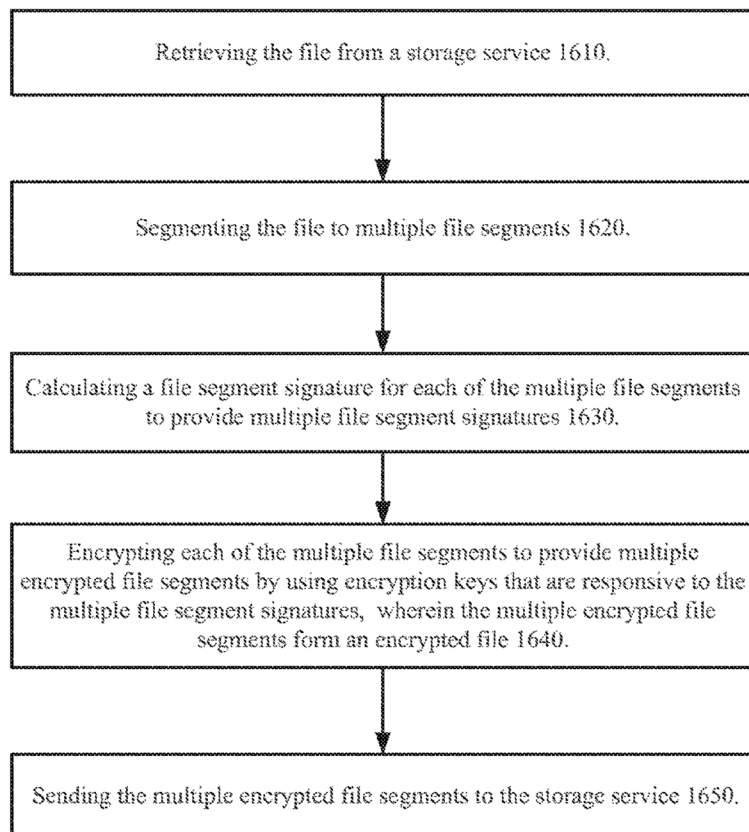
Second, the Cidon-Shikfa-Herrmann combination would have produced a better system. For example, adding Shikfa's mechanism would have enhance Cidon by, e.g., allowing a chunk-by-chunk search of encrypted documents stored on cloud storage that had no search capability. (Cidon, ¶¶91, 166, 171; Shikfa, ¶2.) Additionally, both Cidon and Herrmann aimed to increase flexibility through different access policies for different users. (Cidon, ¶106; Herrmann, ¶50.) A POSITA would have understood that incorporating Herrmann's techniques to Cidon's system would have allowed automatic policy assignment and implementation, thereby improving system efficiency and reducing human errors. (Herrmann, ¶77; Lee, ¶69.)

Additional motivations are discussed below.

2. Claim 1

a. 1[pre]: “A method comprising:”

Cidon disclosed a method for “encrypting a file,” including “segmenting the file into multiple file segments...encrypting each of the multiple file segments to provide multiple encrypted file segments...and sending the multiple encrypted file segments to the storage service.” (Cidon, ¶¶10, 309, FIG. 15.)



1600

FIG. 15

- b. 1[a][i]: “assigning to one or more users, by a gateway device, a policy for managing access to and processing a file to be stored on one or more cloud platforms,”

Cidon disclosed a **management server** that provided access policy configuration for **users**, implemented policies, and processed a file to be stored on **cloud platforms**, e.g., “Dropbox, Box.” (Cidon, ¶¶99.) Cidon’s management server was “connected to consumer-based file repositories...accessing stored files.” (*Id.*, ¶100; *see also id.*)

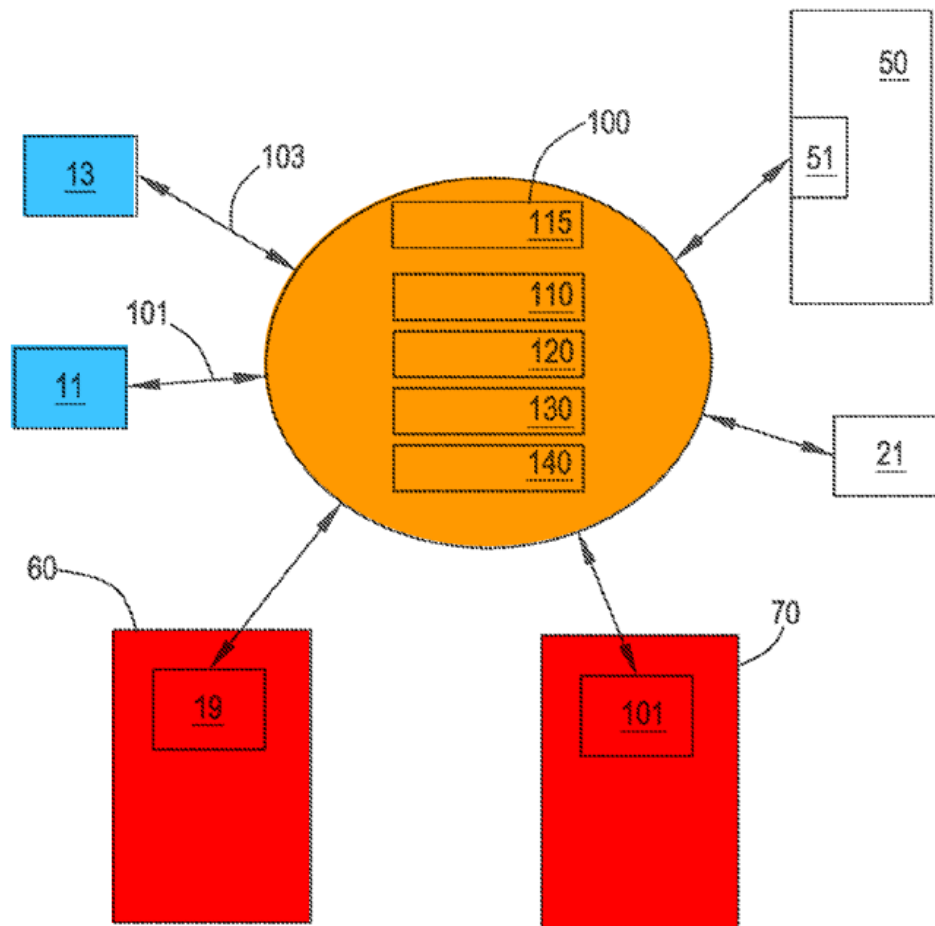


FIG. 2 (Annotated)

Assigning the Policy: While Cidon defined an access policy for a user (e.g., ¶108), provided access policy configuration, and enforced the access policy, it did not explicitly disclose assigning such a policy “*by a gateway device*.” Herrmann taught or suggested this limitation. (Lee, ¶¶74-79.)

Specifically, Cidon described its management server was responsible for, “provid[ing] access policy configuration for secure file access...[and]...enforce[ing] access policy for secure file access.” (Cidon, ¶¶102, 108.) Cidon’s policy also “dictate[d]” whether certain files were to be encrypted (policy for processing a file). (*Id.*, ¶¶224, 227.) Cidon further recognized that “[d]ifferent IT departments [would] *want different policies* for access to these secure files” and “[u]sers...may be organized in further groups and *each group may have different policies*.” (*Id.*, ¶108.) To achieve these goals, Cidon’s management server “allow[ed] for an administrator to define access policies...through a web browser interface.” (*Id.*)

Herrmann disclosed a **gateway server 350** that “act[ed] as a gateway...to...**protected data or resources 390**...and accept[ed] requests for access to such resources from **client machines**.” (Herrmann, ¶60, FIG. 3 (below).)

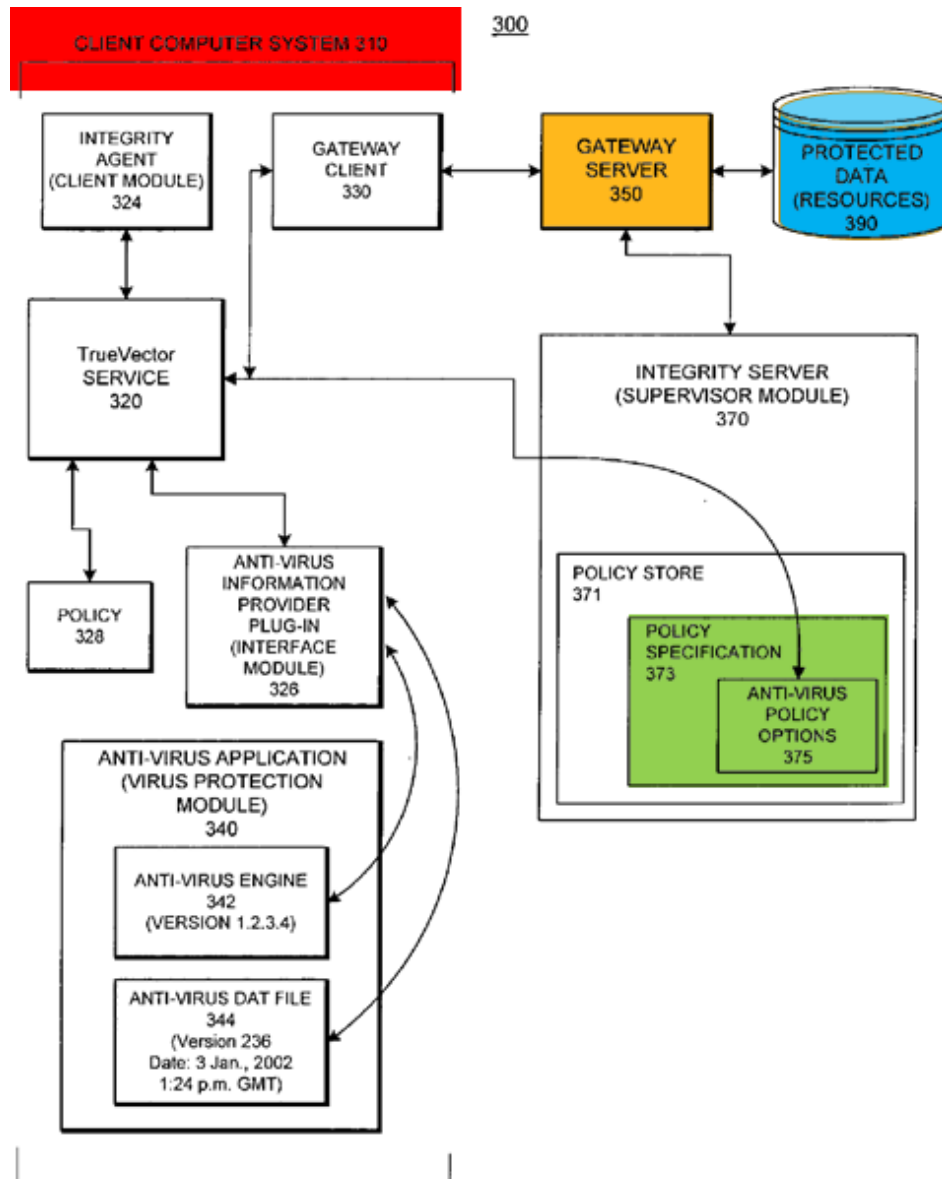
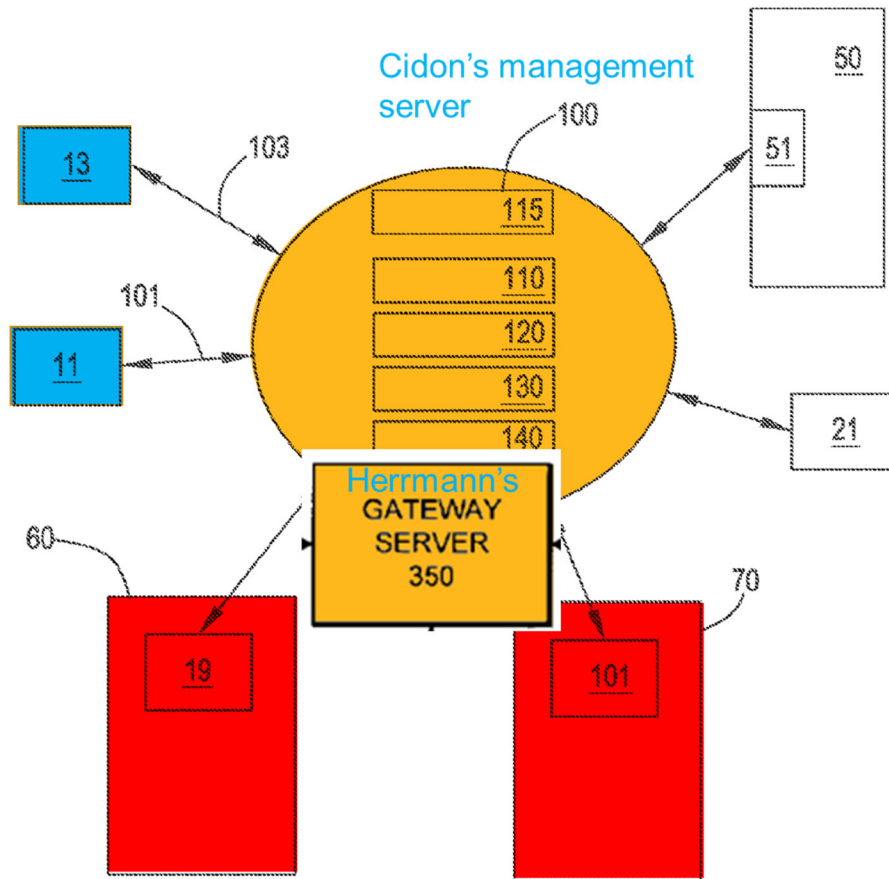


FIG. 3 (Annotated)

Herrmann’s gateway server 350 verified that “client computer 310 [wa]s authorized to access the protected data 390” and also “*assign[ed]* an **access policy** to the connection/session between the gateway server 350 and the client computer 310.” (Herrmann, ¶¶60, 77, claim 16.) Herrmann’s policy store 371 maintained

the “appropriate **policy specification 373** applicable to the client computer 310” that could be retrieved in different ways “including...by **user**, by **group**, or by computer.” (Herrmann, ¶67; Lee, ¶79.)

In the Cidon-Herrmann combination, a POSITA would have added Herrmann’s gateway server and its access policy assignment functionality to Cidon’s management server (as illustrated in modified Figure 2 below). The gateway would have performed Cidon’s and Herrmann’s functions, including Herrmann’s “assign[ing] an access policy” and “accept[ing] requests for access to such resources from client machines” (e.g., Cidon’s user device 60). (Cidon, FIG. 2; Herrmann, ¶¶77, 60; Lee, ¶80.)



Modified FIG. 2

A POSITA would have been motivated to make this combination because instead of waiting for an administrator to assign a policy manually, the modified system would have allowed automatic policy assignment and implementation, improving system efficiencies. (Herrmann, claim 21 (“[U]pdated policy [was] automatically applied to a device.”); Lee, ¶81.) Moreover, the modified system would have allowed better real-time protection. (*Id.*)

This combination would have been straightforward because gateway devices were well-known for network security-related functionalities. (*See, e.g.,* Herrmann,

¶¶18, 77; Chambers, ¶¶66, 82 (describing network access devices, operated as gateway devices); Lee, ¶82.) Additionally, Herrmann contemplated its gateway server performing other functions—e.g., those performed by its integrity server. (Herrmann, ¶65.) Accordingly, Herrmann’s gateway server could have accommodated Cidon’s management server operations and yielded predictable results. (Lee, ¶82.)

c. 1[a][ii]: “wherein the policy defines access rights of the one or more users,”

Cidon disclosed this feature. Cidon’s management server was “responsible for....**enforcing secure access** to these cloud files,” including “provid[ing] access policy configuration for secure file access...[and]...**enforc[ing] access policy for secure file access.**” (Cidon, ¶102; *see also id.*, ¶108 (explaining that different users had different policies to access “file repositories[,] e.g.[,] Dropbox”), ¶¶112, 115.) Herrmann corroborated Cidon by explaining that its policy could be assigned “by user or by group.” (*See* Element 1[a][i]; Herrmann, ¶67; Lee, ¶83.)

d. 1[b]: “encrypting, by the gateway device, using cryptographic key information defined by the policy, content of the file to produce a searchable encrypted file by:”

As explained in Element 1[a][i], a POSITA would have combined Herrmann’s gateway server with Cidon’s management server, such that the claimed operations would be conducted by the gateway. (*See supra* VIII.A.2.b;

Lee, ¶¶84-91.)

“encrypting...content of the file to produce a searchable encrypted file”:

In Cidon+Herrmann, the gateway device included the functionality of Cidon’s management server that encrypted file content to produce a searchable encrypted file.

Specifically, Cidon described that its management server was “responsible for encrypting files in the cloud.” (Cidon, ¶102; *see also id.*, ¶222 (“encrypting the file to provide an encrypted file”).) In Figure 13 (below), the management server encrypted the content of **file 1410** to produce an **encrypted file 1460**. (*Id.*, ¶¶303-305, 337 (explaining that the “segmenting, the calculating and the encrypting may be executed by a management server”), FIG. 13.)

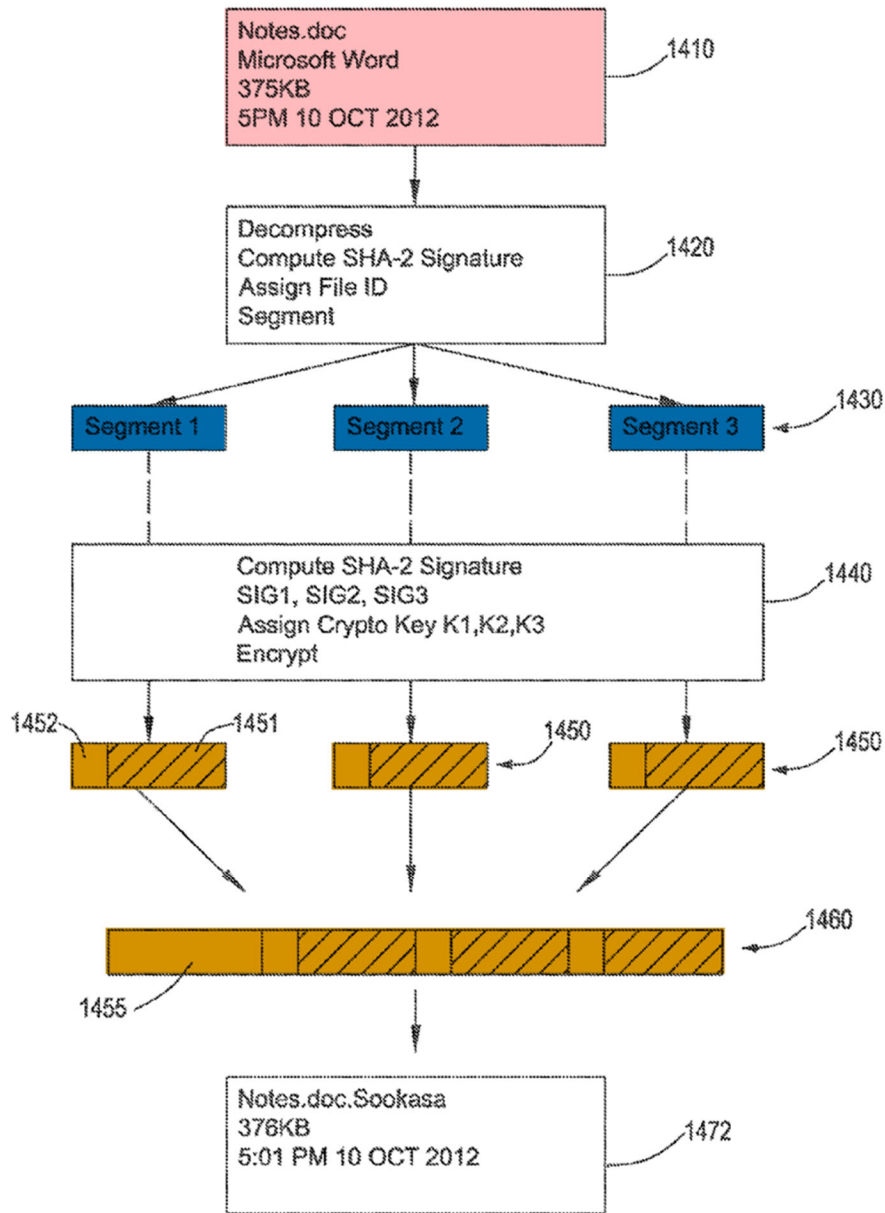
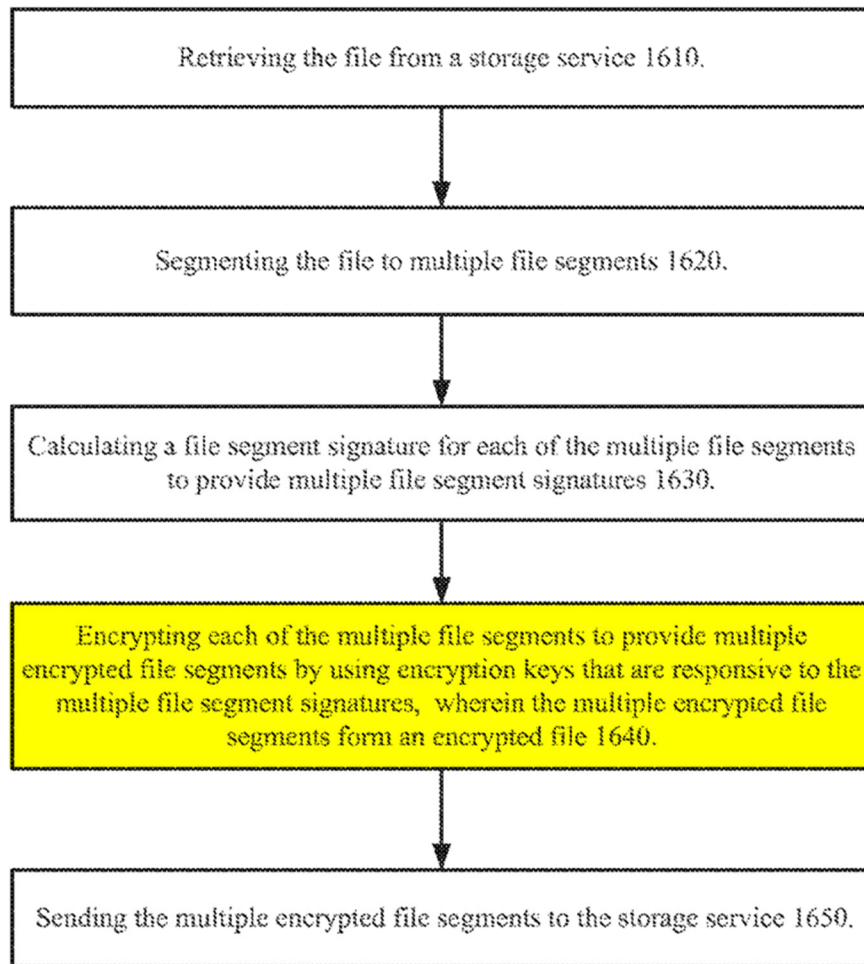


FIG. 13 (Annotated)

(See also *id.*, FIG. 15, ¶309 (“Method 1600 may include:...encrypting each of the multiple file segments to provide multiple encrypted file segments..., wherein the multiple encrypted file segments form an encrypted file....”).)



1600

FIG. 15 (Annotated)

Moreover, Cidon taught or suggested that the encrypted files were searchable because Cidon's system allowed a user to "access[] the complete user file and folder structure by browsing the web service account using...an Application Interface (API)...[which] also allow[ed] the IT to program additional functionalities such as **document search**." (Cidon, ¶91; Lee, ¶88.) Cidon also

described that its management server extracted metadata from files uploaded to the management server, and these metadata “may help in **file search operation, using text/tags based search.**” (Cidon, ¶¶166, 171.)

“encrypting ... using cryptographic key information defined by the policy, content of the file”:

While the '678 Patent does not explicitly define “cryptographic key information,” it describes that “a selected policy can be used to define encryption keys, decryption keys, and encryption type, among **other attributes for carrying out the encryption** of data.” ('678 Patent, 4:4-7.) Consistent with this description, Cidon taught or suggested this limitation. (Lee, ¶89.)

Cidon explained “[u]sing **encryption parameters** for encrypting a file associated with a specific user and **using different encryption parameters** for encrypting another file that is associated with another user that differs from the specific user.” (Cidon, ¶324; *see also* ¶326.) A POSITA would have found it obvious that Cidon’s encryption parameters would include “cryptographic key information” because an encryption key was a common and well-known encryption parameter necessary for encrypting a file. (Lee, ¶90.) (Compare to '678 Patent, 4:4-7, 10:1-3 (“[E]ncryption module 210 can encrypt each file and/or part thereof using cryptographic key information....”).)

A POSITA would have also found it obvious that Cidon’s encryption parameters (cryptographic key information) would be defined by the policy

because Cidon’s policy already included encryption information (Cidon, ¶¶207-208, 212), Cidon described that different users/user groups were assigned different policies (Cidon, ¶108), and Cidon further explained that encryption parameters could differ based on the specific user/groups of users. (Cidon, ¶¶324, 326; Lee, ¶91.)

e. 1[b][i]: “dividing the file into a plurality of chunks;”

Cidon disclosed “segmenting the file into multiple file segments” (chunks). (Cidon, Abstract.) For example, Cidon’s Figure 13 illustrates dividing **file 1410** into three **file segments** 1430. (Cidon, FIG. 13, ¶305.)

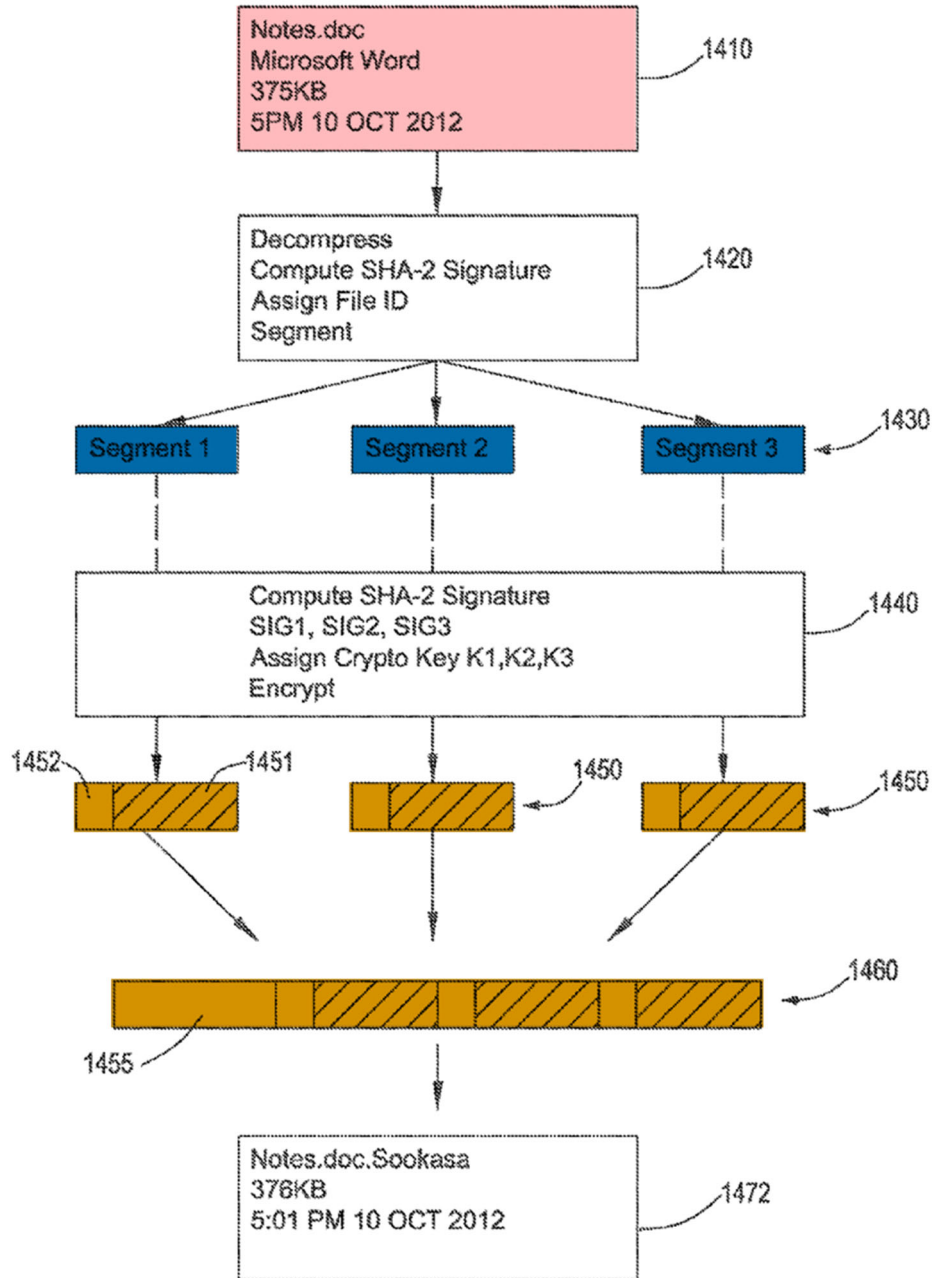
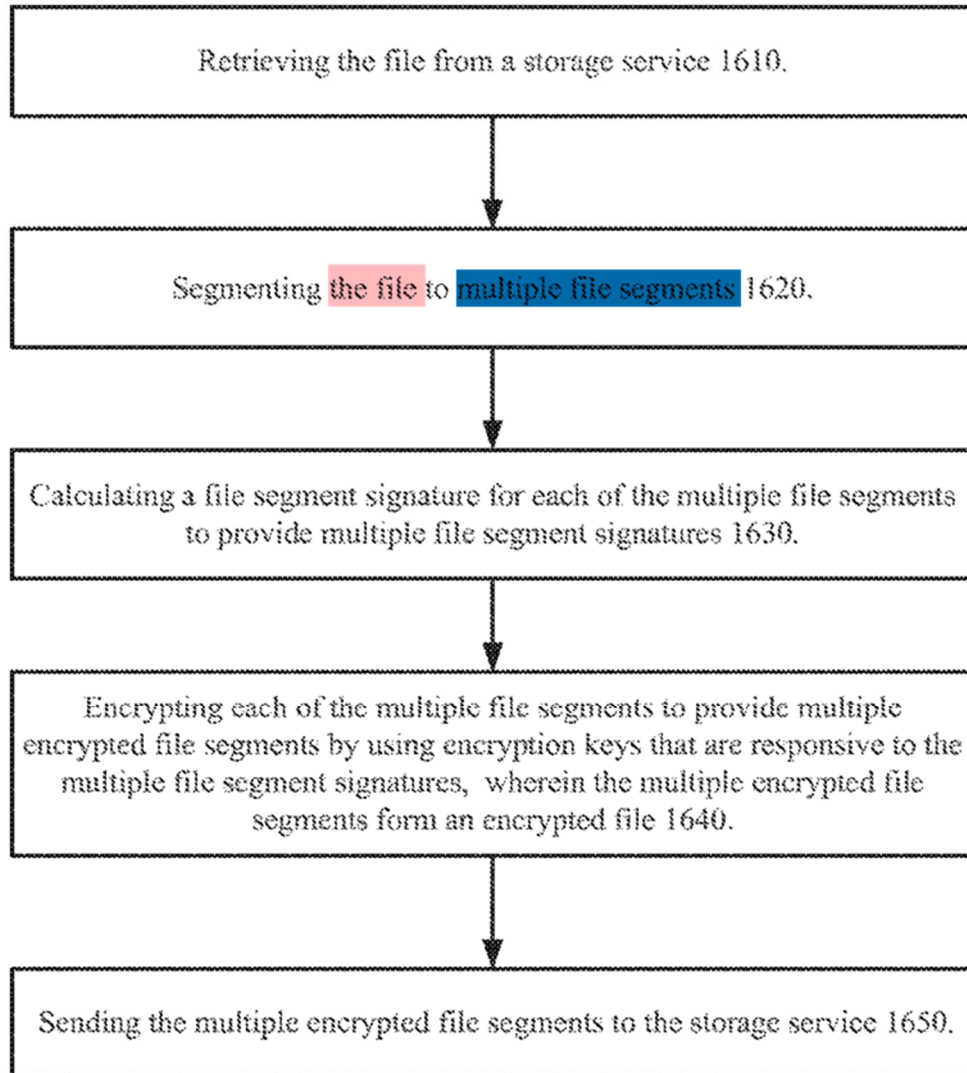


FIG. 13 (Annotated)

(See also *id.*, ¶35, FIG. 15, ¶309.)



1600

FIG. 15 (Annotated)

- f. 1[b][ii]: “creating namespaces for one or more of the plurality of chunks; and”

Cidon disclosed this limitation because it described calculating (and

creating) a “unique” file **segment identifier** (or file segment signature) for each of **file segments** (chunks) based on the content of a file segment using, e.g., SHA2 signature computation. (Cidon, ¶¶305, 383, FIG. 13.)

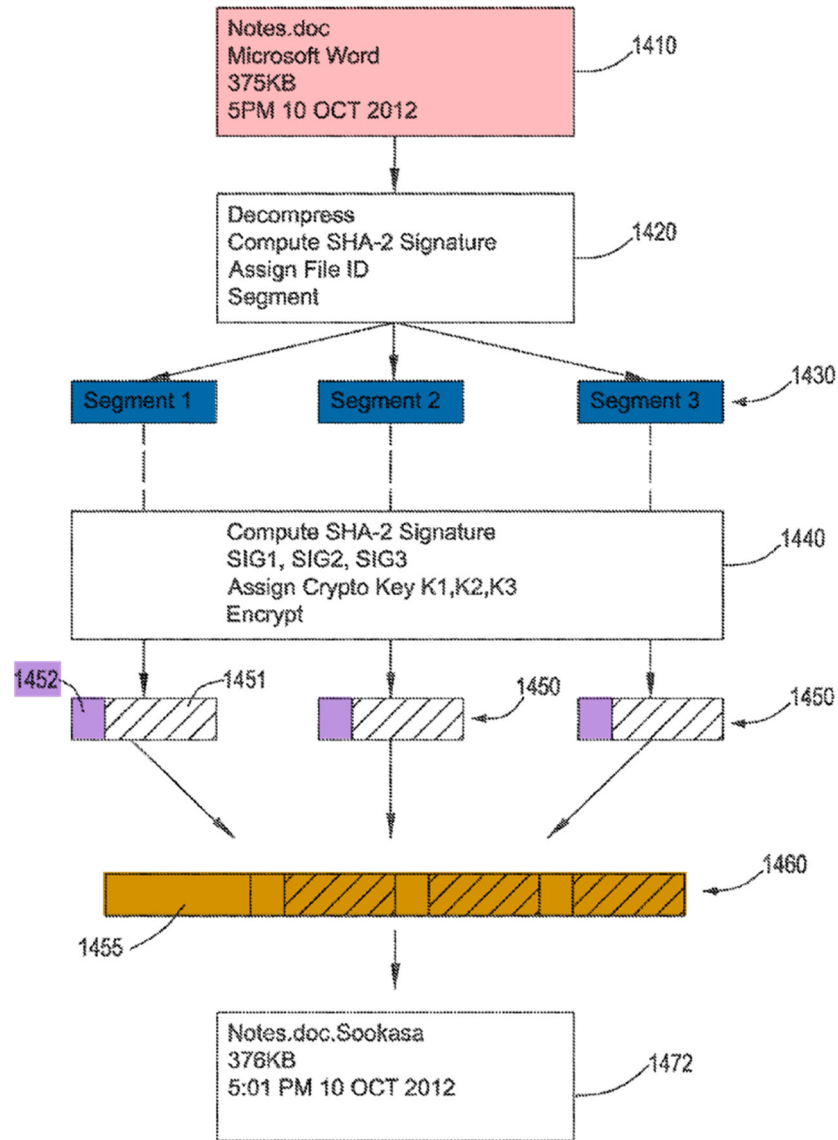


FIG. 13(Annotated)

(See also *id.* FIG. 15, ¶309 (“Method 1600 may include:...calculating a file segment signature for each of the file segments to provide multiple file segment

signatures 1630....”).)

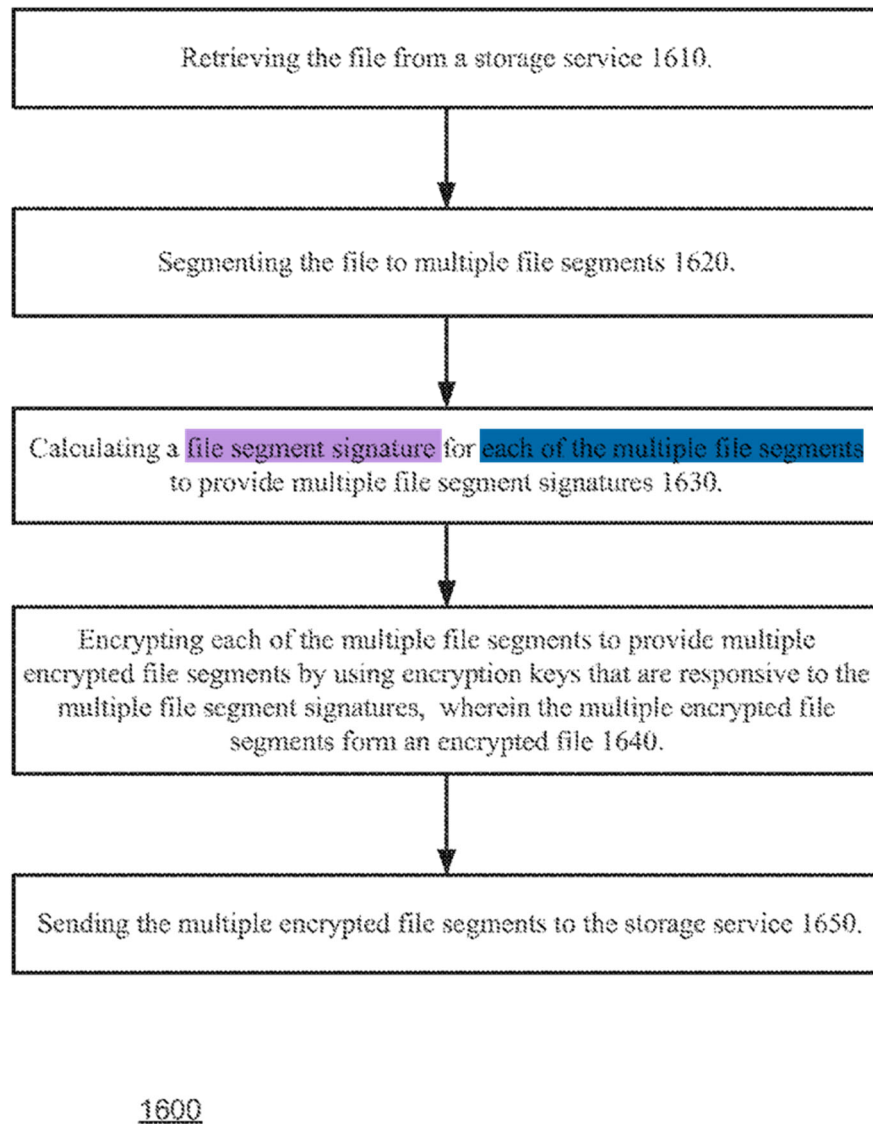


FIG. 15 (Annotated)

A POSITA would have understood that these segment identifiers, which could be calculated using SHA2 signature algorithm, identified the corresponding segments. (Cidon, ¶106; Lee, ¶96.)

Cidon also described assigning the calculated segment identifiers to the file segments. (Cidon, ¶305, FIG. 13 (illustrating that a **segment identifier 1452** was assigned to each of the segments), ¶309, FIG. 15.)

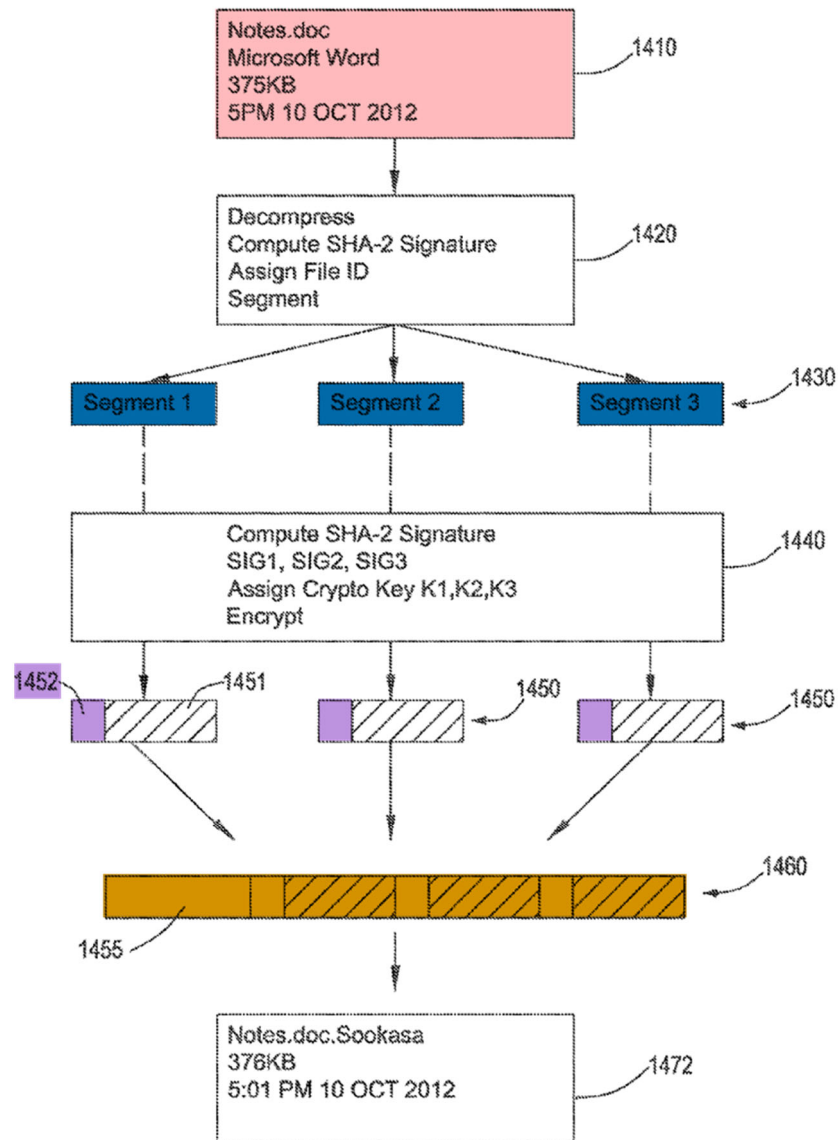


FIG. 13(Annotated)

A POSITA would have found it obvious that Cidon's segment identifiers for the file segments constituted "namespaces" because these identifiers were "the set

of names available for naming” corresponding file segments consistent with the plain and ordinary meaning of this term. (Ex. 1014; Lee, ¶98.)

- g. 1[b][iii]: “configuring the namespaces of the one or more chunks such that content of the file is encrypted in a manner that makes it searchable;”**

While Cidon taught encrypting file content to produce a searchable encrypted file (*see supra* VIII.A.2.d) and producing segment identifiers, it did not explicitly disclose configuring the segment identifiers (namespaces) so the file content was encrypted in a searchable manner. Shikfa, however, taught this limitation by describing pointers/identifiers configured to allow search and retrieval of encrypted file segments. (Lee, ¶¶99-107.)

Specifically, Shikfa disclosed “generat[ing] one or more indexes 24 for one or more documents 26 that [were] to be encrypted as **encrypted documents 14** and stored on the **storage providers 18**.” (Shikfa, ¶29, FIG. 2.)

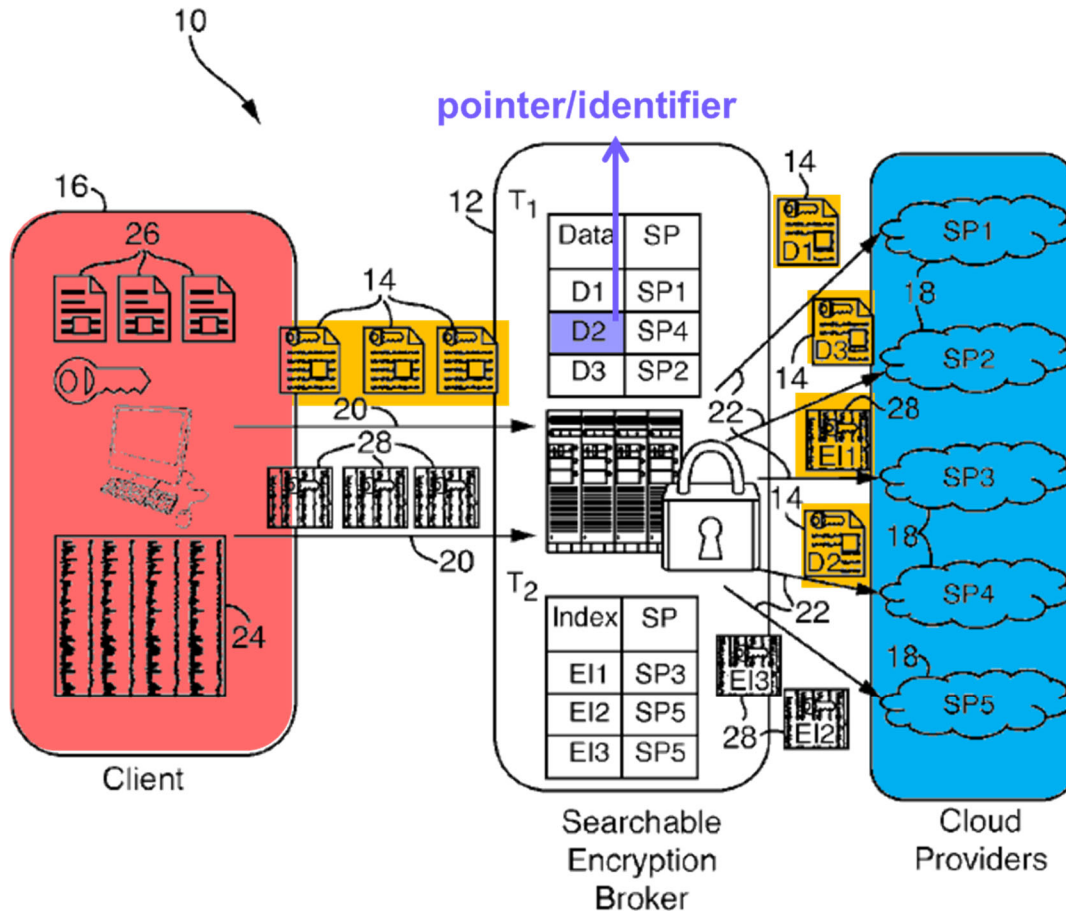


FIG. 2 (Annotated)

“Each index 24 include[d] one or more keywords for the encrypted documents 14 and **pointers** to the encrypted documents 14 containing the keywords.” (*Id.*, ¶29.) Indexes 24 were then encrypted using a searchable encryption algorithm to produce encrypted indexes 28. (*Id.*) Shikfa, alternatively referred to those “**pointers**” as “**identifiers**” (e.g., D1, D2, D3 in FIG. 2). For example, while Shikfa initially described that an index consisted of (1) search keywords and (2) pointers to the documents, it later explained that a search of the

encrypted indexes that matched the keywords returned “**identifiers**” (e.g., **D1**, **D2**, **D3**) that identified the encrypted documents. (*Id.*, ¶¶29, 31.) A POSITA reading Shikfa would have understood that the pointers and identifiers were the same, and both constituted a field within the index that identified the encrypted document. (Ex. 1014; Lee, ¶101.) A POSITA would have also found it obvious that Shikfa’s pointers/identifiers were namespaces because they allowed identification of encrypted documents. (*Id.*)

These identifiers, stored in translation table T1 (Shikfa, ¶31, FIG. 2), were configured to allow the encrypted documents to be searched, as shown in Figure 3 (annotated below). Specifically, upon receiving an encrypted search query 30, the broker server 12 used encrypted indexes 28 to conduct the search, which “return[ed] **identifiers** (e.g. [,] **D1**, **D2**, **D3**, etc.)...that satisfy[ied the] encrypted query 30” (Shikfa, ¶36, FIG. 2.) Translation table (T1) was then used to “look up the **identifier(s)** returned...(e.g.[,] **D2**) and then [to] retrieve[] the corresponding encrypted document(s) 14 from the storage providers 18.” (*Id.*, ¶36, FIG. 3.)

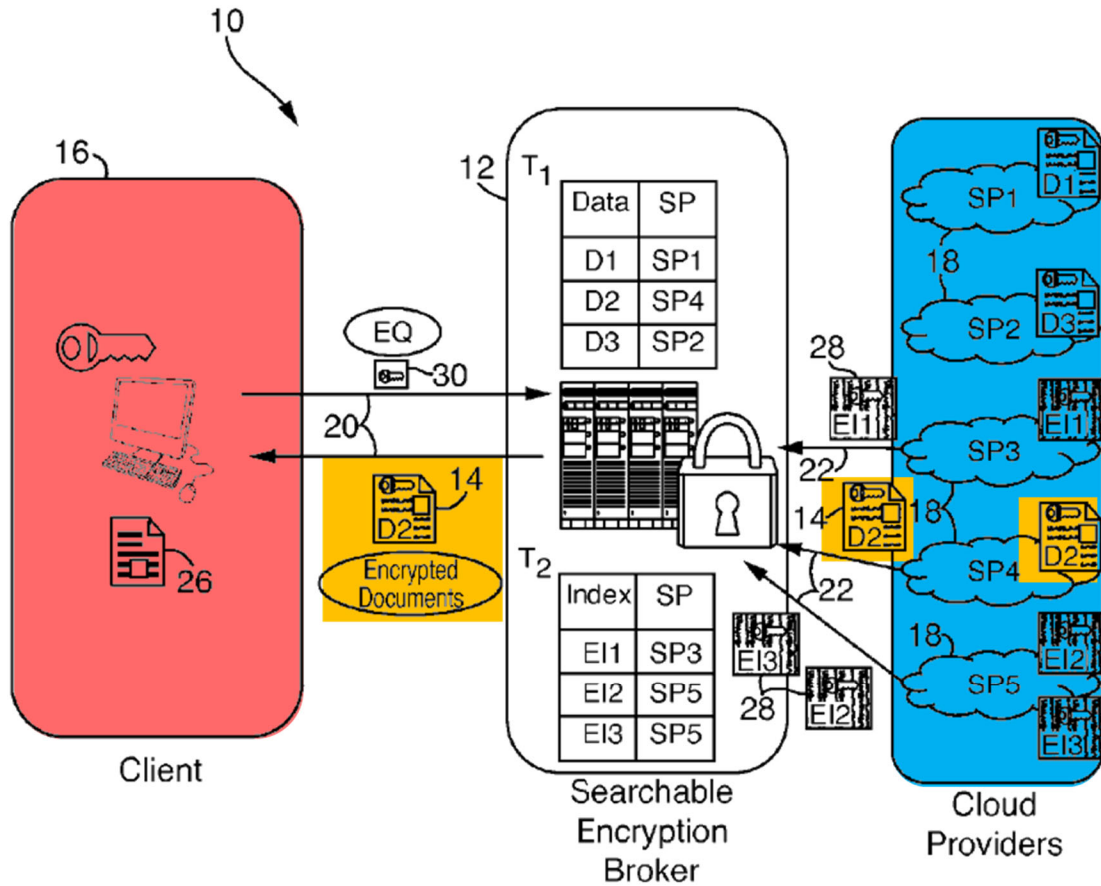


FIG. 3 (Annotated)

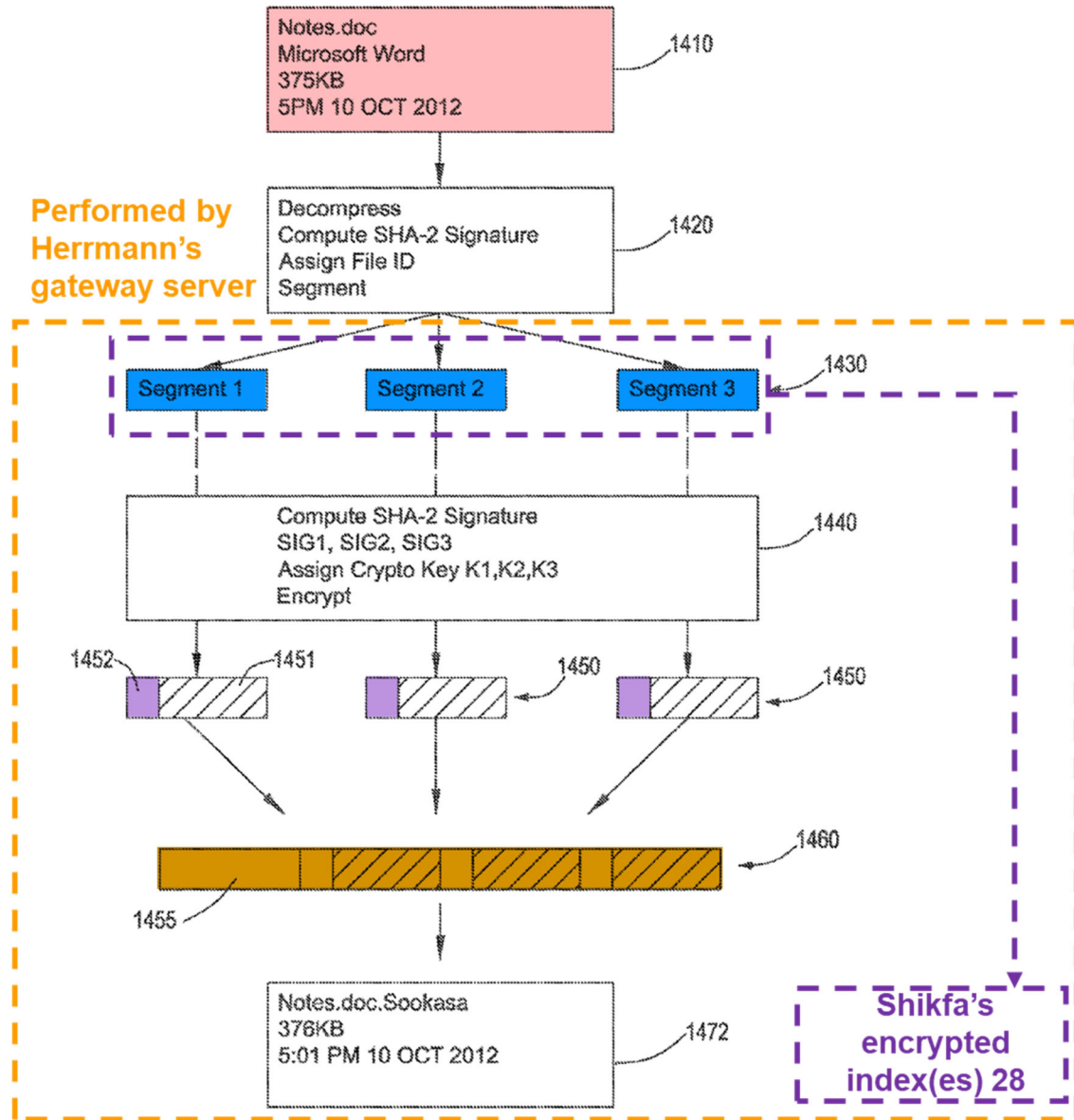
A POSITA would have understood that Shikfa’s identifiers (namespaces) were configured to allow a search of the encrypted content because they mapped search keywords and encrypted documents. (Lee, ¶103.)

Combination of Shikfa with Cidon and Herrmann: While Shikfa primarily concerned searching and retrieving full documents, it also explained that a “*part* of *a* particular encrypted *document*” may be stored to cloud storage providers to “guarantee availability of the stored data.” (Shikfa, ¶39.) A POSITA would have been motivated to combine Shikfa’s searchable encryption mechanism with

Cidon's encryption technique that divided the content into segments (chunks) and created a segment identifier for each chunk to enable search of encrypted chunks.

The modified system would have been benefitted from various advantages that Shikfa described. For example, the system would have allowed file content to be encrypted and searched "without requiring any such capability" at a storage provider, and thus would have provided more cloud storage options (e.g., files on cloud storages that did not have search capabilities). (Shikfa, ¶50; Lee, ¶105.) Additionally, the modified system would have "accommodate[d] document searchability" without "compromis[ing the]...privacy needs," thereby protecting confidential information. (Shikfa, ¶50; Lee, ¶105.)

In the combination, the configuration of Shikfa's **identifiers** (e.g. [,] **D1**, **D2**, **D3**), which identified the full document, would have been implemented as part of configuring Cidon's segment identifiers. This would allow Cidon's system to search and locate files on a segment-by-segment basis; these operations would have been implemented in Herrmann's gateway server (illustrative diagram of a modified Cidon Figure 13 shown below). (Lee, ¶106.)



Modified FIG. 13

A POSITA would have reasonably expected success in combining Cidon, Shikfa, and Herrmann because searchable encryption mechanisms had been well-known and would have been compatible with Cidon and Herrmann's systems, and the combination would have yielded expected results. (*See* Chiueh, Exs. 1015-1016; Lee, ¶107.)

h. 1[c]: “storing, by the gateway device, the searchable encrypted file on the one or more cloud platforms based on the policy; and”

In the Cidon-Herrmann-Shikfa combination, the gateway device including the functionality of Cidon’s management server would store the encrypted files on a cloud platform based on the policy. Specifically, Cidon disclosed storing the searchable encrypted file on cloud platforms through its server’s “connect[ion] to **consumer-based file repositories** (e.g., Dropbox 13, Gmail 11, or others such as Evernote) through Application Interfaces (API) 101 and 103 or through web access.” (Cidon, ¶¶99-100, FIG. 2.)

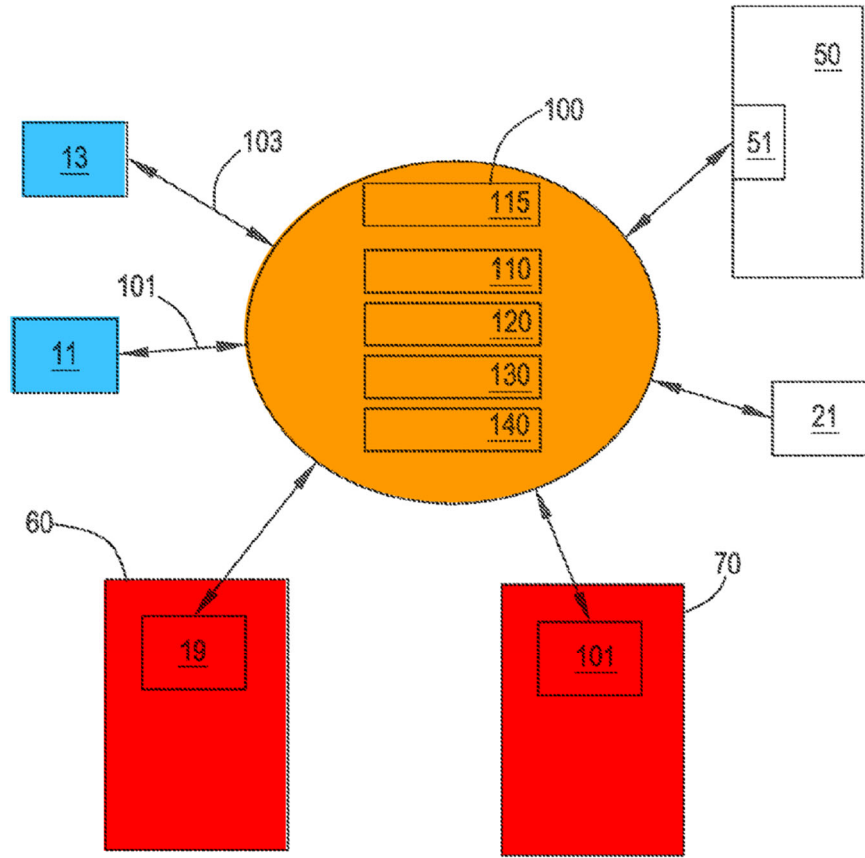
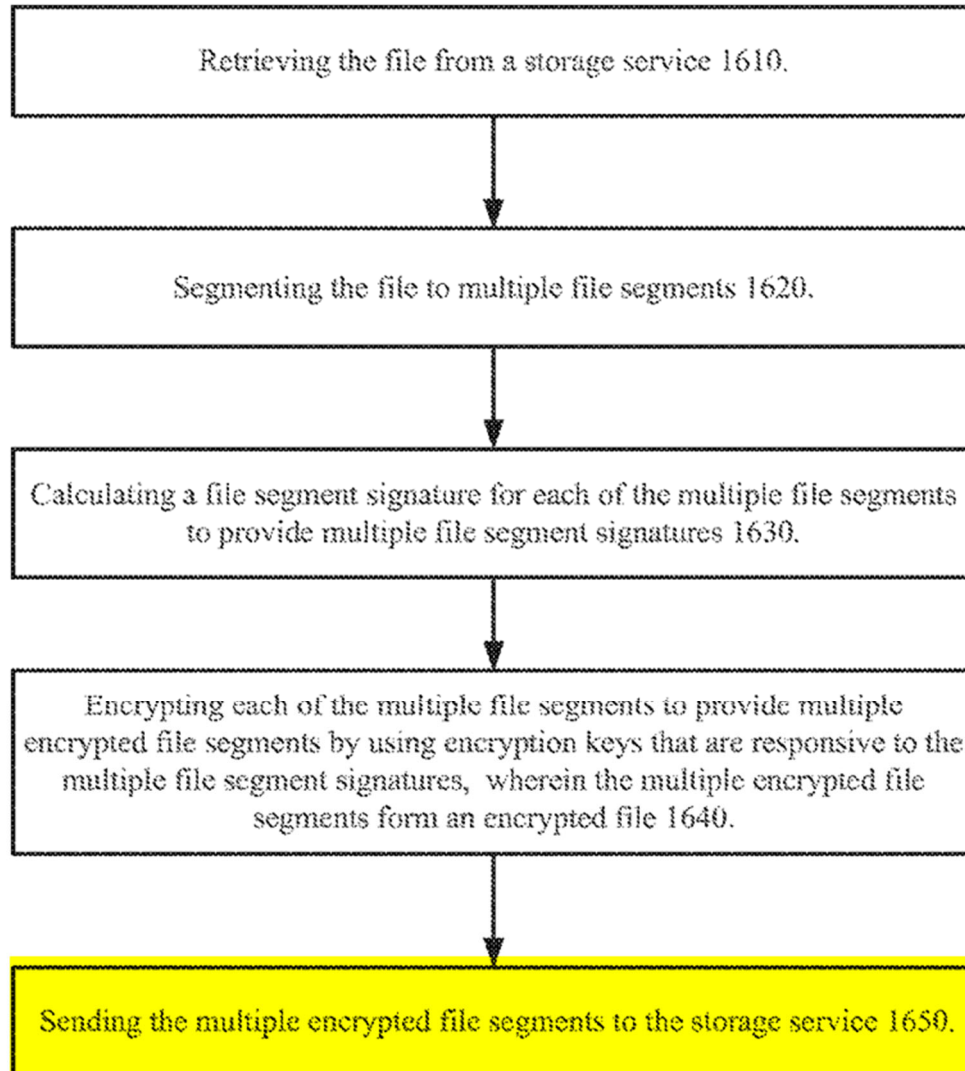


FIG. 2 (Annotated)

Cidon's management server sent encrypted file segments (which formed an encrypted file) to the cloud storage. (*Id.*, ¶309, FIG. 15; *see also id.*, Abstract, ¶¶10, 43, 229, claims 1.)



1600

FIG. 15 (Annotated)

A POSITA would have understood that storing the searchable encrypted file on a cloud storage was based on the policy because Cidon's policies included "[p]lacement [of files] in specific directories or shares" (Cidon, ¶¶208-209), and

placement policy violations could have resulted in moving the file to reverse the violation. (Cidon, ¶¶214-215; Lee, ¶110.)

i. 1[d]: “managing access to the searchable encrypted file by the one or more users based on the policy.”

Cidon contemplated different users would have different access policies for files in cloud storages. (Cidon, ¶108.) Cidon’s management server was responsible for “enforcing secure access to these cloud files,” including “provid[ing] access policy configuration for secure file access” and “enforc[ing] access policy for secure file access.” (*Id.*, ¶102.) For example, the management server “provide[d] user management, API access (as well as API feed) to cloud storage services such as Dropbox and Box.” (*Id.*, ¶113.) Also, the management server’s IT Policy Enforcement Module 120 provided “an admin interface [that] allow[ed] for an administrator to define access policies...through a web browser interface to the module,” and “[o]nce this policy [was] defined it [would] be enforced through application interfaces (API).” (*Id.*, ¶108; Herrmann, ¶67 (described assigning policies by “user [or] group”).)

A POSITA would have understood that Cidon’s management server controlled user access to the searchable encrypted file based on the policy by, e.g., “provid[ing] access policy configuration for secure file access” and “enforc[ing] access policy for secure file access.” (Cidon, ¶102; Lee, ¶112.)

3. Claim 16

Claim 16 does not require (or prohibit) that the recited functions be performed by a single gateway device. As explained for claim 1, a single gateway device (e.g., Hermman) could be used to perform similarly-recited functions. (*Supra* VIII.A.2.)

a. 16[pre]: “A secure cloud storage system comprising:”

Cidon disclosed a **management server** (secure cloud storage system) for “encrypting a file,” including “segmenting the file into multiple file segments;...encrypting each of the multiple file segments to provide multiple encrypted file segments...;...and sending the multiple encrypted file segments to the **storage service**.” (Cidon, FIG. 2, ¶57; *see also id.*, ¶¶99-100; *supra* VIII.A.2.a.)

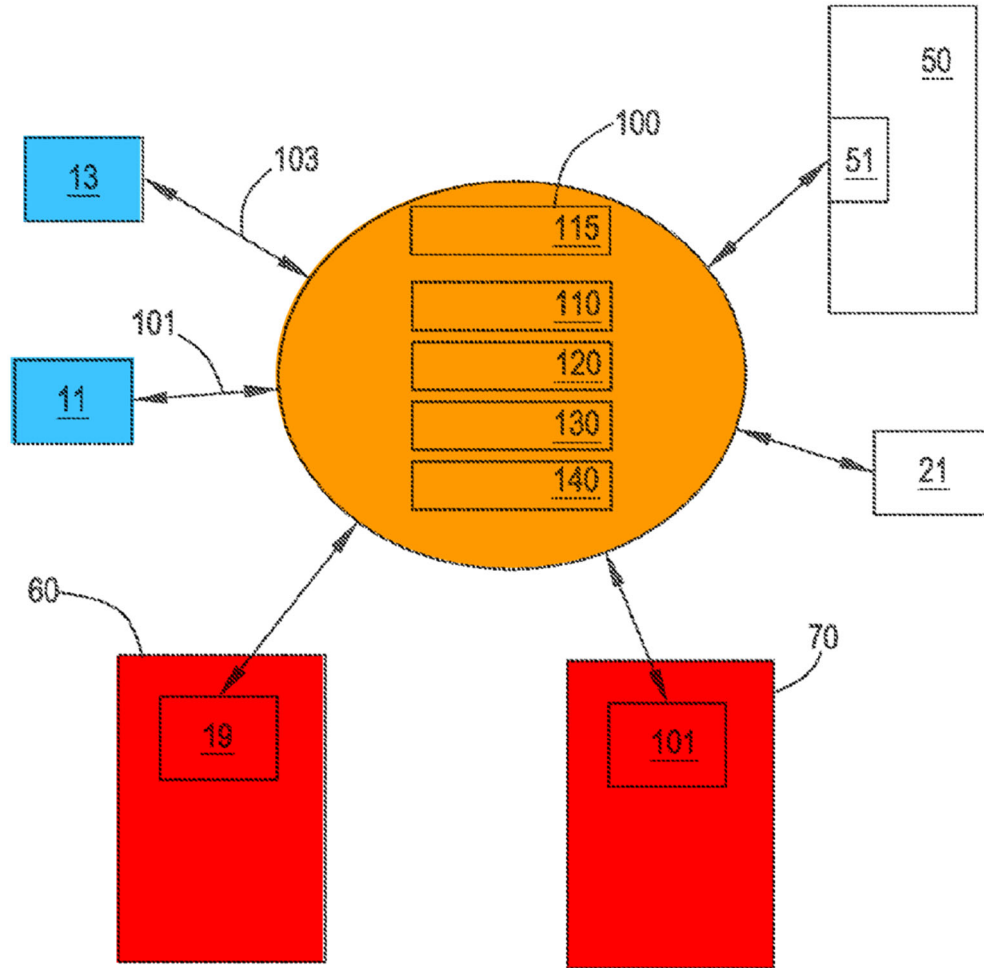


FIG. 2 (Annotated)

b. 16[a]: “one or more processors;”

Cidon explained its invention program running on a computer, like management server 100, and such computer system included one “processing unit” (processor). (Cidon, ¶¶307, 407, 416.)

c. 16[b]: “a communication interface mechanism;”

Cidon disclosed that its management server “communicate[d] with user devices 60 and 70,” e.g., receiving requests for decrypting files or sending

encryption keys to user devices. (Cidon, ¶¶101, 340, 342.) Cidon also described that the management server was connected to “file repositories,” e.g., Dropbox 13, Gmail 11, Evernote, “through Application Interfaces (API) 101 and 103 or through web access..., accessing stored files through interfaces such as FTP, Secured FTP (SFTP) or WebDAV” and sent encrypted file segments to cloud storage service. (*Id.*, ¶¶100, 309.)

A POSITA would have understood that Cidon’s management server would include a communication interface mechanism for communicating with user devices and cloud storage services described in Cidon. (Lee, ¶118.)

d. 16[c]: “one or more internal data storage devices operatively coupled to the one or more processors and storing:”

As explained above, Cidon disclosed that a computer system, such as management server 100, included memory (internal data storage) associated with (operatively coupled to) a processing unit (processor) that stored software instructions or computer program that, when executed by the processing unit, caused the computer system to “execute any or all of the methods described” therein, including various modules to carry out certain functions. (Cidon, ¶¶50, 101-102, 307, 407, 412, 416; Lee, ¶119.)

- e. **16[c][i]: “a policy assignment module, which, when executed by the one or more processors, implements a policy selected from a group of policies for a user,”**

Cidon’s computer system, including its management server 100, included a processing unit (processor) and software instructions that “execute[d] any or all of the methods described” therein. (Cidon, ¶50.) Such computer programs included an IT policy enforcement module that provided “an admin interface allow[ing] for an administrator to define access *policies*” for users and implemented these policies. (*Id.*, ¶¶101, 108.)

Cidon described that different users/groups were assigned different policies. (*Id.*, ¶108). Therefore, it would have been obvious to a POSITA that Cidon’s IT policy enforcement module would select a policy for the appropriate user/group. (Lee, ¶121.)

- f. **16[c][i][A]: “wherein the policy defines and manages access to and processing of a file to be uploaded on a cloud,”**

Cidon’s policy defined and managed access to and processing of a file to be uploaded on a cloud. (*See* VIII.A.2.b.)

- g. **16[c][i][B]: “wherein the policy defines access rights of the user;”**

Cidon’s policy defined access rights of the user. (*See* VIII.A.2.c.)

h. 16[c][ii]: “an encryption module, which, when executed by the one or more processors:”

Cidon explained its “methods...including any stages—and any combinations” can be implemented using executable instructions or a computer program. (Cidon, ¶50.) Therefore, a POSITA would have found it obvious that Cidon’s encryption would be performed by an encryption module using instructions executable by the processor. (Lee, ¶124; *see also* Cidon, ¶102 (describing its management server as a broad term to describe the complete software system).)

i. 16[c][ii][A]: “using cryptographic key information defined by the policy, produc[ing] a searchable encrypted file by encrypting content of the file to be stored across one or more cloud platforms;”

Cidon disclosed “using cryptographic key information defined by the policy, produc[ing] a searchable encrypted file by encrypting content of the file.” (*See supra* VIII.A.2.d.) Cidon also disclosed that the file’s encrypted content was stored across more than one cloud platform because its system “archiv[ed] and backup[ed a file] to other storage services, and [performed] file synchronization between services” (**two or more** cloud platforms). (Cidon, ¶174; Lee, ¶125.)

j. 16[c][ii][B]: “divid[ing] the file into a plurality of chunks;”

Cidon disclosed this limitation. (*See supra* VIII.A.2.e.)

k. 16[c][ii][C]: “creat[ing] namespaces for one or more of the plurality of chunks; and”

Cidon disclosed this limitation. (*See supra* VIII.A.2.f.)

l. 16[c][ii][D]: “configure[ing] the namespaces of the one or more chunks such that content of the file is encrypted in a manner that makes it searchable;”

Cidon and Shikfa rendered this limitation obvious. (*See supra* VIII.A.2.g)

m. 16[c][iii]: “a storage module, which, when executed by the one or more processors, stores the searchable encrypted file on the one or more cloud platforms based on the policy; and”

Cidon disclosed that its management server sent (and stored) the searchable encrypted file on the cloud storage, and a POSITA would have found it obvious that this was done based on the policy. (*See supra* VIII.A.2.h.) Cidon described “any or all of the methods described above, including any stages—and any combinations” can be implemented using executable instruction or computer program. (Cidon, ¶50; *supra* VIII.A.3.b, VIII.A.3.d; Cidon, ¶¶50, 102; Lee, ¶130.)

n. 16[c][iv]: “a management module, which, when executed by the one or more processors manages and controls access to the searchable encrypted file by the user based on the policy.”

Cidon disclosed that its management server managed access to the searchable encrypted file by the users based on the policy, (*see supra* VIII.A.2.h), and implementing “any or all of the methods described above, including any

stages—and any combinations” using executable instruction or computer program.

(Cidon, ¶50; *supra* VIII.A.3.b, VIII.A.3.d; Cidon, ¶102; Lee, ¶131.)

4. Claims 2/17: “...wherein the step of managing comprises allowing[/the management module further allows] the one or more users to download the searchable encrypted file to a local device”

Cidon’s management server allowed the user to download the searchable encrypted file to a user device (local device) by “enabling user devices to connect to their web storage services” (cloud platforms) “through a standard web file access such as WebDAV, FTP and secure FTP supported on...[the] server...[,]...serv[ing] as a proxy to these file services.” (Cidon, ¶221, *see also id.*, ¶231 (“[T]he file may be...downloaded by some other application.”), ¶299 (“The app may...allow the download of the file.”); Lee, ¶132.)

5. Claim 3: “wherein the local device comprises a mobile computing device.”

Cidon’s user devices (local devices) could be mobile computing devices, such as “mobile devices” (Cidon, ¶116), “iPhone/iPad or Windows [l]aptop” (*id.*, ¶¶108-109, 129). (*See also id.*, ¶95 (“When these files are accessed by the user they will be encrypted and read by the user using a decryption mechanism available on his device (PC, **laptop, notebook, tablet, smart phone** etc.).”); Lee, ¶133.)

6. Claim 8: “wherein the step of encrypting the file is carried out independent of the service provider of the one or more cloud platforms.”

Cidon disclosed that, rather than relying on encryption services from the cloud platform service providers (Cidon, ¶99), Cidon’s management server *itself* “provide[d] file and folder encryption services to the user.” (Cidon, ¶¶113, 102 (the management server provided “encryption of...files”); Lee, ¶134.) Indeed, Cidon’s management server allowed uploading *encrypted* files to different web storage services and its single connection “served *multiple* web storage instances from the same or *multiple vendors*.”) (Cidon, ¶221; Lee, 134.)

7. Claims 9/23: “...wherein the policy is implemented for a group of users.”

Cidon explained that “[d]ifferent IT **departments** [would] want different policies for access to these secure files” and “[u]sers in a company may be organized in further groups and **each group may have different policies**.” (Cidon, ¶108; Lee, ¶135.) Cidon’s system also allowed administrators to define these policies, which could be for user groups. (*Id.*, ¶108; *see also* Herrmann, ¶67 (describing that a policy could be assigned by “group”).)

8. Claims 10/24: “...wherein the policy defines the manner in which the file is stored, accessed, and processed.”

Cidon’s policy defined the manner in which the file was stored (e.g., “[p]lacement in specific directories or shares”), accessed (e.g., “[a]ccess and

sharing permissions,” “[c]opy control”), and processed (e.g., “[e]ncryption,” “[c]opy control”). (Cidon, ¶¶207-212; *see also id.*, ¶¶102, 108, 401-402, FIG. 23.)

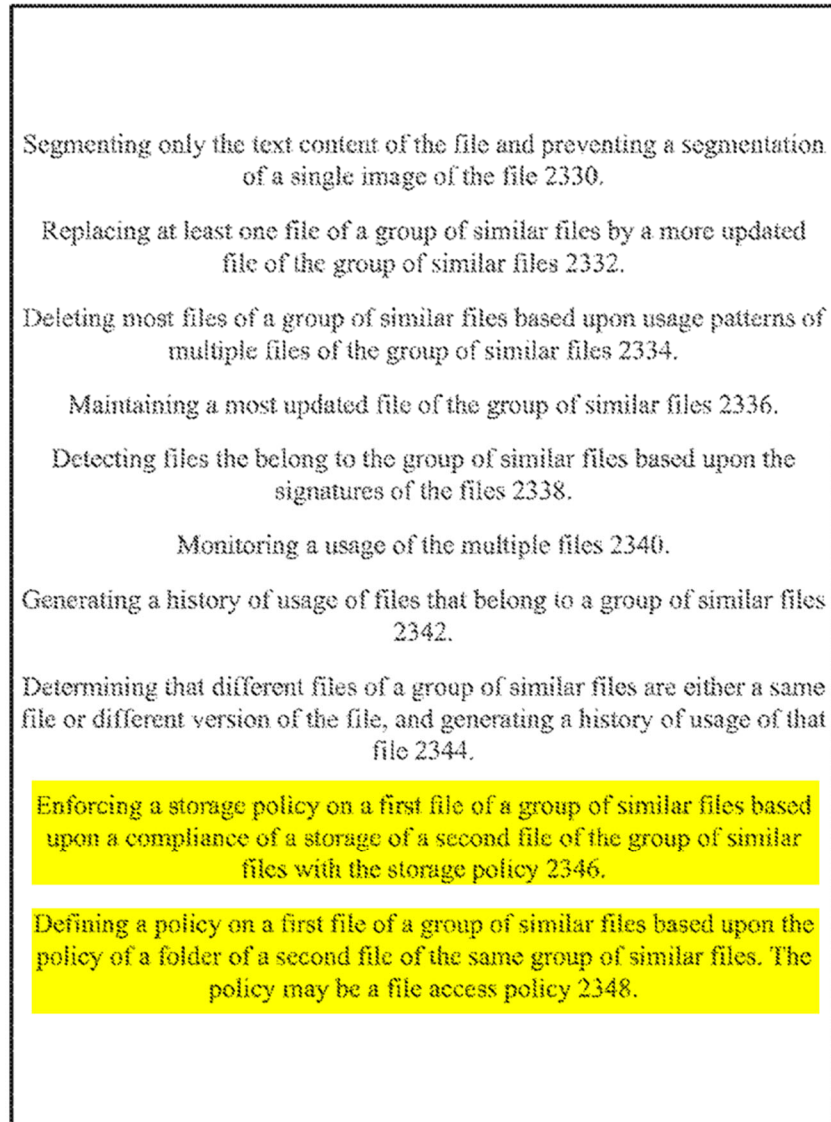


FIG. 23 (Annotated)

9. Claims 11/25: “...wherein the file is stored across two or more cloud platforms.”

Cidon’s system performed file operations including “archiving and backup

to other storage services, and file synchronization between services.” (Cidon, ¶174.) A POSITA would have understood that Cidon’s backup and/or synchronization of the file on or between storage services constituted storing the file across two or more cloud platforms. (See, e.g., *id.*, ¶221 (describing that Cidon’s system allowed uploading encrypted files to different web storage services); Lee, ¶137.)

10. Claim 12: “...wherein said storing the searchable encrypted file on the one or more cloud platforms is carried out independent of the service provider of the one or more cloud platforms.”

Cidon rendered obvious claim 12 because, consistent with the ’678 Patent, Cidon’s searchable encrypted file stored on storage services (cloud platforms) could “be moved to any cloud platform at any moment, thereby avoiding vendor lock-in.” (’678 Patent, 13:54-59; Lee, ¶138.)

Cidon’s management server allowed “archiving and backup [of files] to other storage services, and file synchronization between services” (Cidon, ¶174), such as “consumer-based file repositories (e.g., Dropbox 13, Gmail 11, or...Evernote)” (*id.*, ¶100), “BOX 24 service and in an enterprise file system 50” (*id.*, ¶173). Cidon’s management server also allowed storing encrypted files to different web storage services, and its single connection “served *multiple* web storage instances from the same or *multiple vendors*.” (*Id.*, ¶221, FIG. 2.)

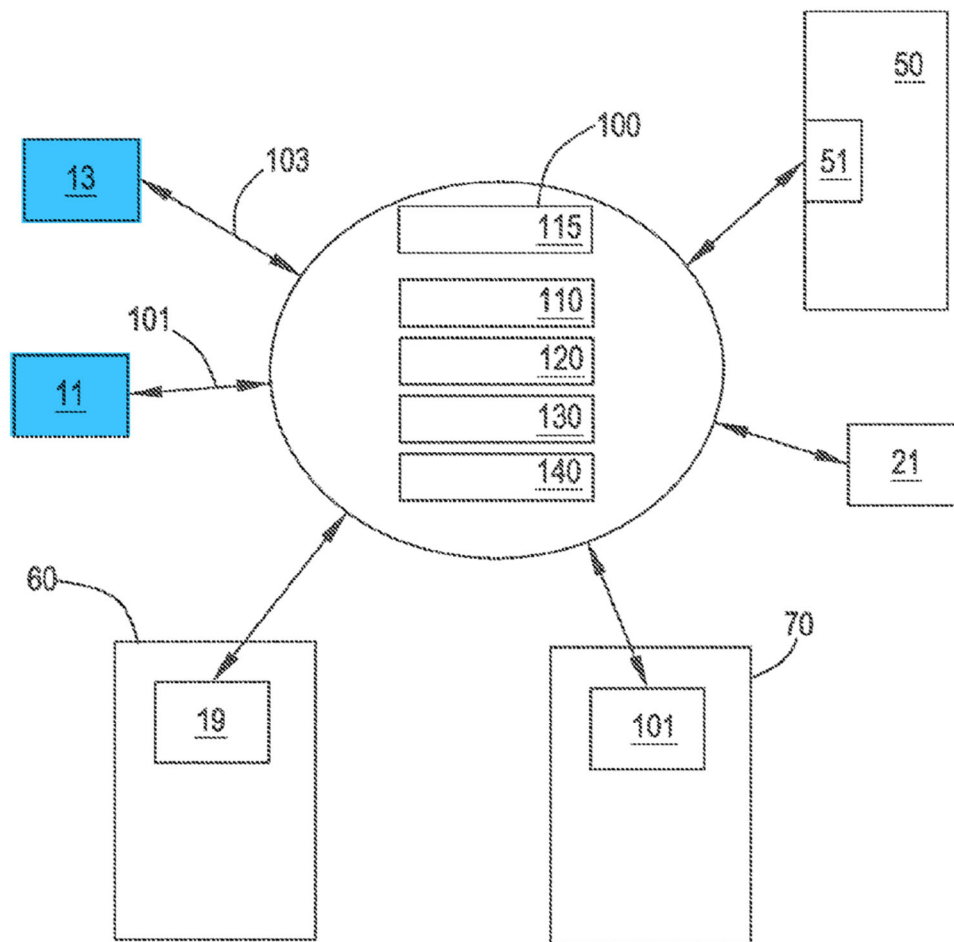


FIG. 2 (Annotated)

Further, as explained for claim 8, the files were processed using a searchable encryption independent of the storage service provider. (*See supra* VIII.A.6.) Therefore, a POSITA would have found it obvious that Cidon’s technique was not “lock[ed]-in” (’678 Patent, 13:54-59) to a specific service, and Cidon’s searchable encrypted file could “be moved to any cloud platform.” (Lee, ¶140.)

11. Claims 13/26: “...wherein the policy is selected from a plurality of policies stored in a database, wherein the database is operatively coupled with the gateway device.”

Cidon and Herrmann disclosed these limitations. (Lee, ¶¶141-146.)

Cidon explained that “[d]ifferent IT departments will want different policies for access to these secure files...[and u]sers in a company may be organized in further groups and each group may have different policies.” (Cidon, ¶108.)

Cidon’s IT Policy Enforcement Module (part of the management server) also provided an administrator with “an admin interface allow[ing] for an administrator to define access *policies*” for users and implemented these policies. (*Id.*, ¶¶101, 108.) Cidon further described a “management web application” on user’s devices communicated with the management server to “manage team *policies* and permissions, and to track team members’ usage of the cloud storage services.” (*Id.*, ¶115.)

Herrmann similarly recognized the need for assigning different policies to different users and disclosed a policy store (database) for storing policy specification. (Herrmann, ¶¶50, 67.)

While Cidon was not explicit about storing the policies in a “database,” a POSITA would have been motivated to store Cidon’s policies in a database coupled to the management server (with Herrmann’s gateway). The modified system would have enabled selecting a policy from the stored policies for the same

user and applying the selected policy instead of repeatedly creating a new policy, thereby improving system efficiency and policy implementation consistency. (Lee, ¶145.)

A POSITA would have reasonably expected success in making this modification because Cidon already disclosed storing data, such as “file metadata, keys, access tables and user profiles,” “the user and team information,” in a database (Cidon, ¶¶103, 110, 119) and storing policies into a database was well-known. (Chambers, claim 1 (“an access control policy (ACP) database to store ACPs”); Herrmann, ¶67 (“retriev[ing] the appropriate policy specification 373 applicable to the client computer 310 from the policy store 371”); Lee, ¶146.)

12. Claims 14/27: “...wherein the policy defines [/and manages] one or more of a set of encryption keys, a set of decryption keys, search indices, a type of encryption and one or more authorized locations to which the file may be stored.”

The Cidon-Shikfa-Herrmann combination disclosed claims 14/27.

The recited features would have been obvious features of a security policy, because Herrmann described that “[i]n general terms, a security policy [was] an organization’s statement defining the rules and practices that regulate[d] how it [would] provide Security, handle intrusions, and recover from damage caused by Security breaches. An explicit and well-defined Security policy include[d] a set of rules that [were] used to determine whether a given Subject [could] be permitted to gain access to a specific object.” (Herrmann, ¶24, Lee, ¶148.)

the policy defines [/and manages] one or more of a set of encryption keys, a set of decryption keys...[/and] a type of encryption:

Cidon disclosed using a 256-bit AES encryption key to encrypt a file. (Cidon, ¶110; Shikfa, ¶29 (describing encrypting documents and indexes using different encryption algorithms); Herrmann, ¶21 (describing two types of encryption—transport and tunnel modes); Lee, ¶149.) Cidon also disclosed generating, storing, and retrieving encryption keys. (Cidon, ¶¶102, 110.) Cidon’s keys were used for decrypting the encrypted files and were, thus, also decryption keys. (*Id.*, ¶43; Lee, ¶150.) Cidon also explained that different users could have different policies configured based on administrator input. (Cidon, ¶108.)

A POSITA would have been motivated to have a policy define (and manage) an encryption type, and encryption/decryption keys as taught by Cidon. For example, instead of awaiting administrator input, the modified system could have retrieved a corresponding policy to obtain the keys and encryption type previously configured by the administrator. (Cidon, ¶108.) Doing so would have improved system efficiency and ensured files were encrypted in the administrator-prescribed manner. (Lee, ¶151.) For instance, a policy could have defined stronger encryption/decryption for users with more sensitive information (e.g., employees in the finance or HR departments). (*Id.*)

the policy defines [/and manages]...one or more authorized locations to which the file may be stored:

Cidon disclosed storing a modified file to a new location to prevent a user from directly accessing the location where the original file was stored. (Cidon, ¶¶364, 367.) A POSITA would have found it obvious that the file locations would be part of Cidon’s security policy that could also define (and manage) such location information. To this end, when encrypting and storing a file (e.g., a modified file), Cidon’s system would obtain the authorized location for storing the encrypted modified file from the policy, and store the encrypted modified file there. Doing so would have “prevented [the user] from directly accessing” other unauthorized storages, such as the location where the original file was stored, thereby improving data security. (*Id.*, ¶367, Lee, ¶152.)

the policy defines [/and manages]...search indices:

Shikfa disclosed generating a single encrypted search index or multiple encrypted indexes for an encrypted document. (Shikfa, ¶29.) Shikfa also described that the number of encrypted indexes generated depended on the specific searchable encryption mechanism used. (*Id.*) In implementing the Cidon-Shikfa combination’s searchable encryption, a POSITA would have found it obvious to define (and manage) whether to generate multiple indexes or one index for an encrypted document in its policy. (Lee, ¶153.) Doing so would have ensured the number of generated indexes would be consistent with the searchable encryption mechanism, instead of having the system to make arbitrary decisions or wait for

input from the administrator. (*Id.*)

13. **Claim 21: “...wherein one or more of the policy assignment module, the encryption module, the storage module, and the management module are implemented on different devices, wherein the devices are selected from one or more of access management devices, proxy devices, gateway devices, and network controllers.”**

Herrmann taught claim 21’s requirement that various recited modules (or functions) are implemented on different devices.

A POSITA would have found it obvious to use Herrmann’s gateway server to perform the recited functions in claims 1 and 16. (*Supra* VIII.A.2.) Herrmann disclosed an **integrity server** that performed similar functions, such as managing policies and access requests based on policies. (Herrmann, ¶¶60, 65, FIG. 3.)

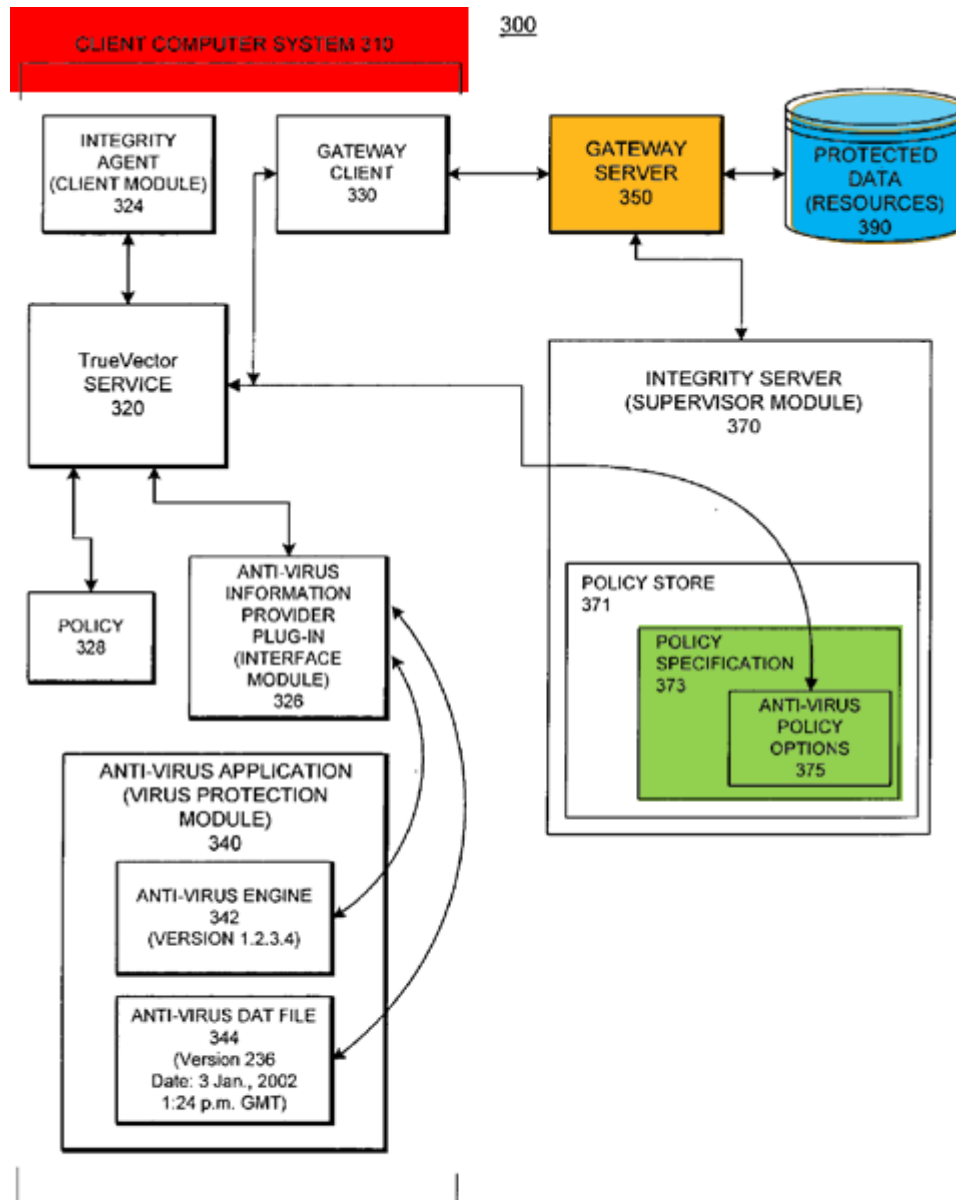


FIG. 3 (Annotated)

Herrmann also described that its integrity server and gateway server could be “installed on the same machine” or different machines. (Herrmann, ¶65.) Thus, a POSITA would have been motivated to implement these recited modules on different machines. (Lee, ¶157.) For example, encryption and storage modules

could have been implemented on Hermman’s gateway service, while policy assignment and management modules were implemented on Hermman’s integrity server (or Cidon’s management server). Doing so would have allowed these functions to be performed in parallel, increasing processing power and efficiency and helping balance task load. (*Id.*)

A POSITA would have reasonably expected success in making such modifications because distributed computing, i.e., using different devices to implement related tasks, had been well-known. Herrmann and Cidon themselves contemplated that their functions could be configured as different modules. Thus, the combination would have yielded expected results. (Herrmann, ¶¶65; Cidon, ¶¶101-102); Lee, ¶158.)

- 14. Claim 22: “...further comprising a mediation module, which, when executed by the one or more processors makes implementation of one or more of the policy assignment module, the encryption module, the storage module, and the policy management module independent of the service provider of the one or more cloud platforms.”**

Cidon rendered claim 22 obvious.

The ’678 Patent’s mediation module “mediate[s] vendor specific protocol/APIs thereby facilitating geographical and provider diversity in data placement, making the system agnostic with respect to specific cloud vendor APIs.” (’678 Patent, 4:46-49; Lee, ¶159.) Cidon described its management server similarly as being “connected to consumer-based file repositories (e.g., Dropbox

13, Gmail 11, or others such as Evernote) through Application Interfaces (API) 101 and 103” and “provid[ing] user management, API access (as well as API feed) to cloud storage services such as Dropbox and Box.” (Cidon, ¶¶100, 113, FIG. 2.)

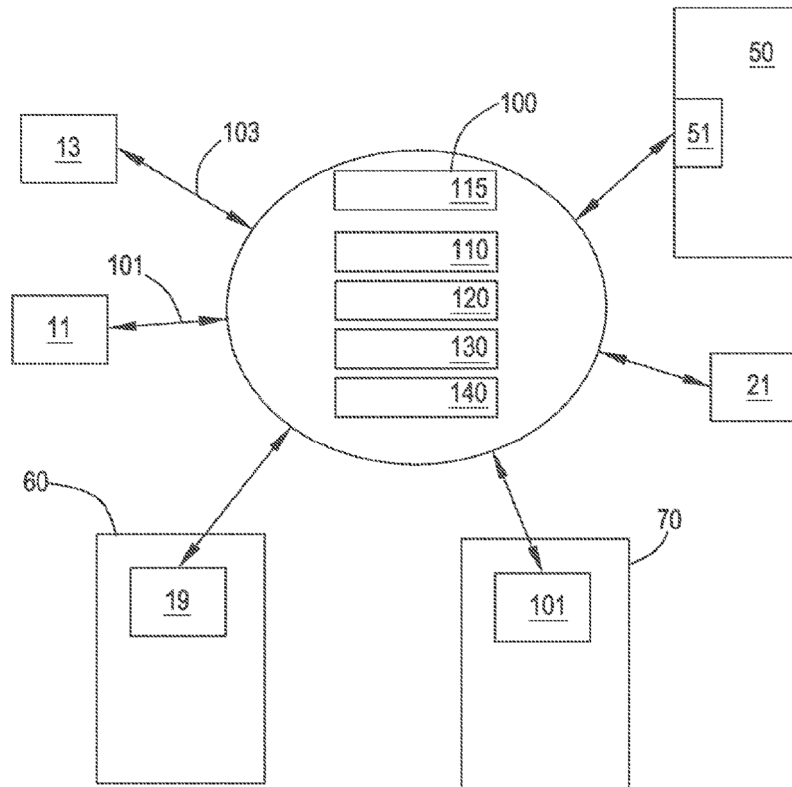


FIG. 2

Additionally, as explained above, Cidon described that its encryption and storage were carried out independent of the cloud platform service provider. (*See supra* VIII.A.6, VIII.A.10.) Cidon’s management server also “serve[d as] a proxy” between the cloud services (connected to which via vendor-specific APIs, e.g., APIs 101, 103) and a user device. (Cidon, ¶221; FIG. 2.)

Further, Cidon described its management server as “a broad term to describe

the complete software system which [was] responsible for encrypting files in the cloud and enforcing secure access to these cloud files” and “include[d] various software...components” that when executed by its processing unit (processor), to execute the functions described therein, including the function performed by this so-called “mediation module.” (Cidon, ¶102; *see also supra* VIII.A.3.b, VIII.A.3.d.; Lee, ¶162.)

From these disclosures, a POSITA would have found it obvious that Cidon’s management server already included the so-called mediation module because it included the aforementioned functionalities for serving as a “proxy” between the user device and various cloud platforms. (Lee, ¶163.) As noted earlier, Cidon also described implementing “any or all of the methods described above, including any stages—and any combinations” using executable instructions or a computer program. (Cidon, ¶50; *see also supra* VIII.A.3.b, VIII.A.3.d; Cidon, ¶102; Lee, ¶163.)

15. Claim 26: “...wherein the group of policies are stored in a database operatively coupled with the system.”

Cidon rendered claim 26 obvious. (*See supra* VIII.A.3.e; Lee, ¶164.)

B. Ground 2: Cidon, Shikfa, Herrmann, and Fellenstein render claim 6 obvious.

1. Claim 6: “wherein the step of configuring the namespaces comprises storing content of the file as filenames.”

Cidon and Fellenstein rendered claim 6 obvious.

Fellenstein described key words (content of the file) that were added to filenames, which could then be used as keywords to retrieve the file. (Fellenstein, ¶8 (“Another **widely used** method [was] one in which one or more keywords for retrieval [were] added to each file name in advance and the file name having the keywords one of which matches an inputted retrieval word [was] regarded as a result of retrieval.”).)

A POSITA would have been motivated to add this well-known functionality into the Cidon-Herrmann-Shikfa combination because the modified system would provide another search option. Searching only the filenames could be done with a simple string-matching operation that would yield quicker results. (*Id.*) In the modified system, one or more keywords (disclosed in Fellenstein) and file segment identifier (disclosed in Cidon; *see supra* VIII.A.2.f) would have been included as part of a file segment’s filename. (Lee, ¶167.)

A POSITA would have reasonably expected success in combining Cidon with Fellenstein because assigning keywords to a filename was well-known, Fellenstein’s technique was compatible with Cidon’s system, and the combination would have yielded expected results. (Lee, ¶168.)

C. Ground 3: Cidon, Shikfa, Herrmann, and Hirai render claim 15 obvious.

1. Claim 15: “...wherein said encrypting further comprises encrypting one or more of filename and search indices.”

Cidon, Shikfa, and Hirai rendered claim 15 obvious.

Shikfa disclosed generating search indices containing “one or more keywords for the encrypted documents” that were used for searching the encrypted documents. (Shikfa, ¶29, Lee, ¶170.). Shikfa also disclosed encrypting those indices. (Shikfa, ¶29, FIG. 2.)

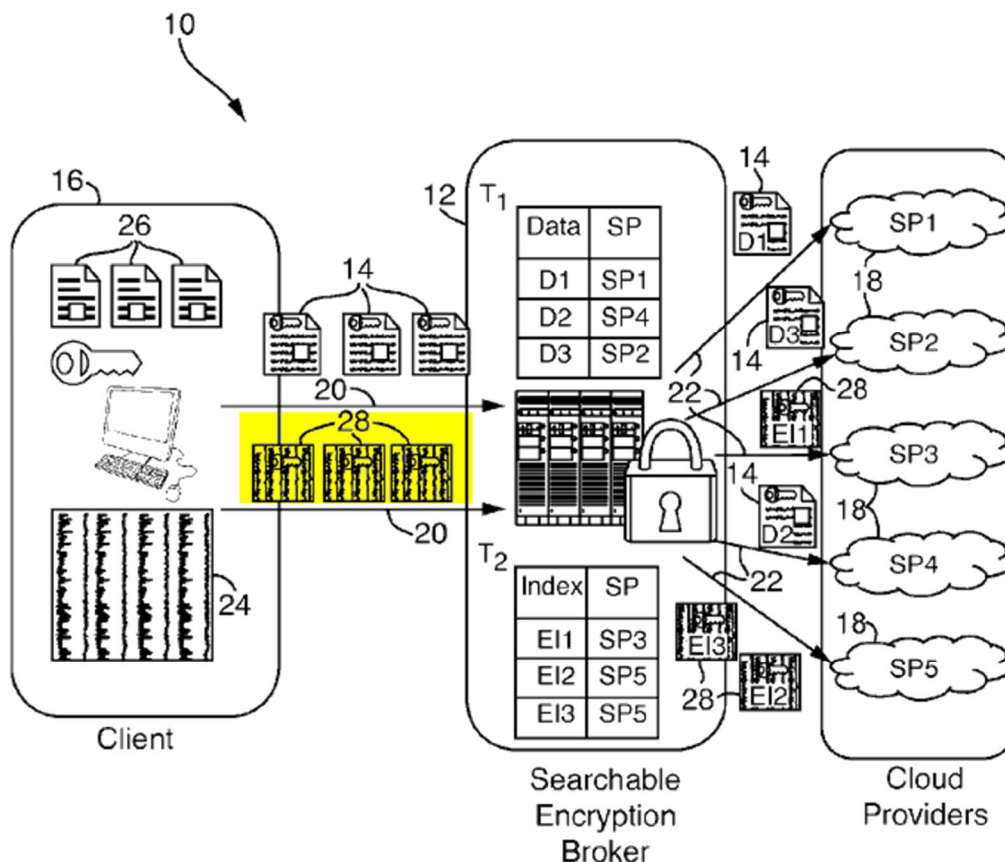


FIG. 2 (Annotated)

Shikfa did not explicitly disclose encrypting filenames, but Hirai described a file data retrieving system that generated “a file name and a plurality of key words related to the file data” and cryptographically encoded “the key words and the file name” using, for example, PGP encryption. (Hirai, 5:64-6:13.)

Filenames and search indexes were often descriptive of the file contents and could reveal confidential information. (Lee, ¶172.) Thus, a POSITA would have been motivated to add Shikfa’s and Hirai’s functionalities to Cidon’s system. (Lee, ¶172.) Doing so would help prevent confidential information in the filenames and search indexes from being revealed to the cloud storage provider. (Shikfa, ¶50; Lee, ¶172.)

A POSITA would have reasonably expected success in making the combination because encryption techniques were well-known, Shikfa and Hirai’s methods would have been compatible with Cidon’s system (given that Cidon’s system could already encrypt data), and the combination would have yielded expected results. (Lee, ¶173.)

D. Ground 4: Auradkar, Chiueh, and Chambers render claims 1-3, 8-19, and 21-27 obvious.

Auradkar: Auradkar concerned storing encrypted data in cloud storages and managing policy-based user access to the encrypted data. (See, e.g., Auradkar, Abstract, FIG. 1.)

For example, Auradkar disclosed assigning to enterprise staff a **policy** for

managing access to and processing a file to be stored on **cloud container 500**.

(Auradkar, ¶¶112-113, 116-117, FIGS. 1-2, 5-6.)

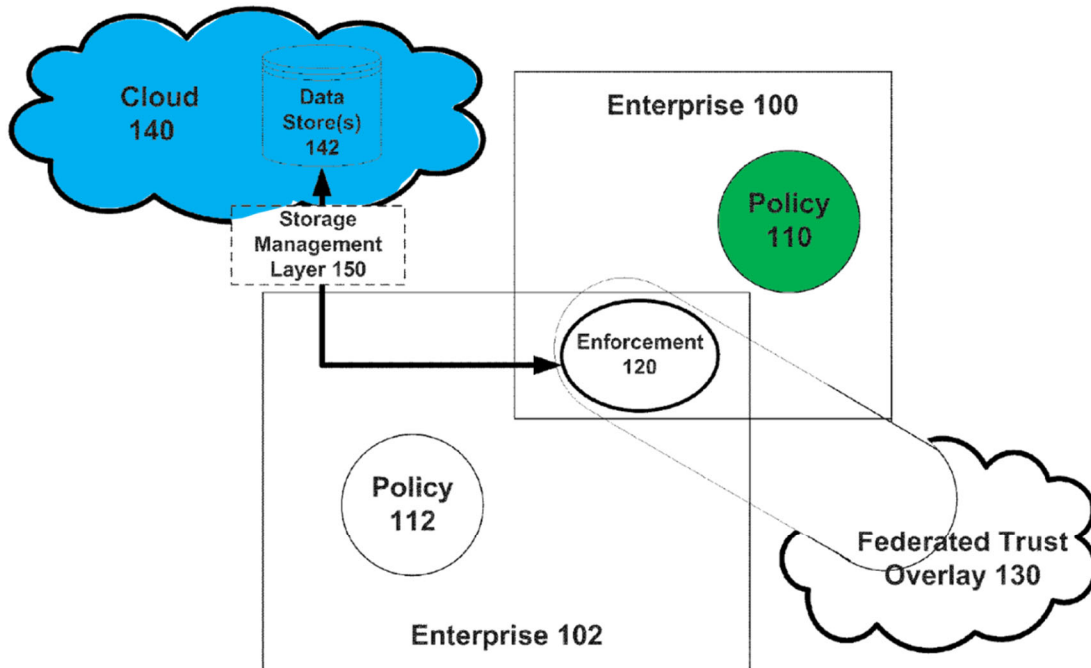


FIG. 1 (Annotated)

Like the '678 Patent's policy, Auradkar defined how data stored in the cloud “[could] be accessed, manipulated, retrieved, searched” and whether/how to encrypt the data (e.g., AES 256 encryption). (Auradkar, ¶¶113, 96.)

In Figure 15, Auradkar described encrypting documents (e.g., **document 1500**) and associated keywords (1510) and uploading the **encrypted document** and encrypted keywords to “the **cloud**.” (*Id.*, ¶¶130-131, FIG. 15.)

Technology for Confidentiality: Searchable Encryption

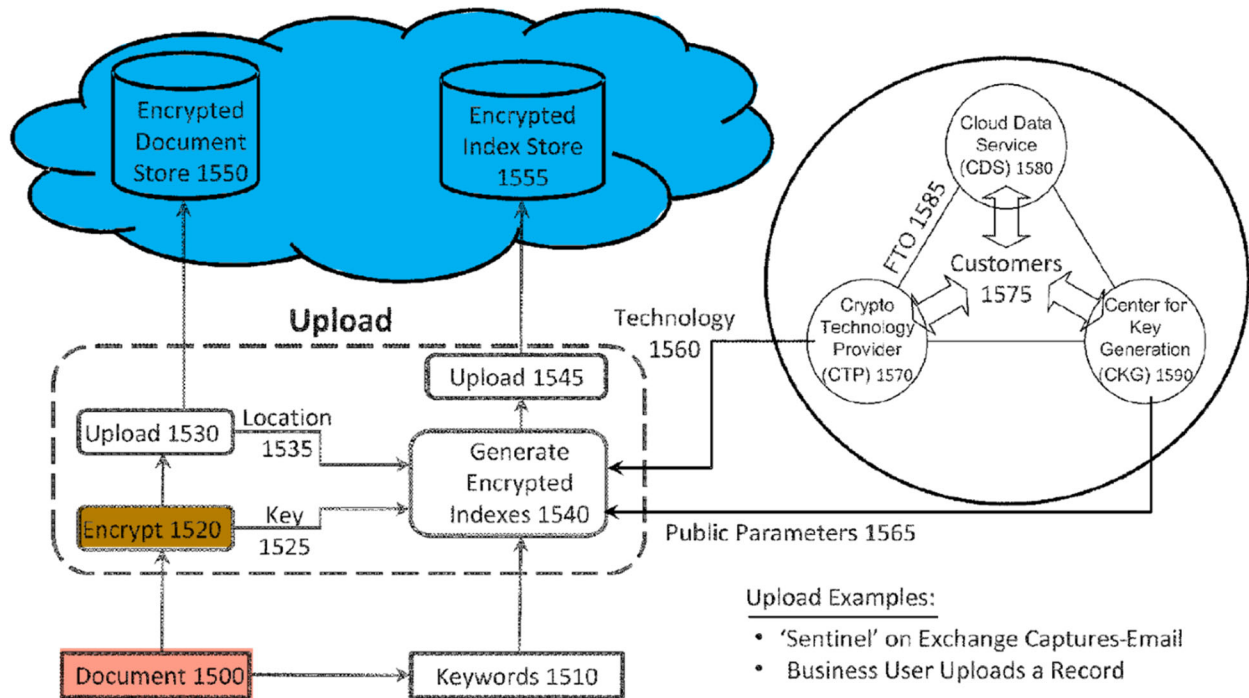
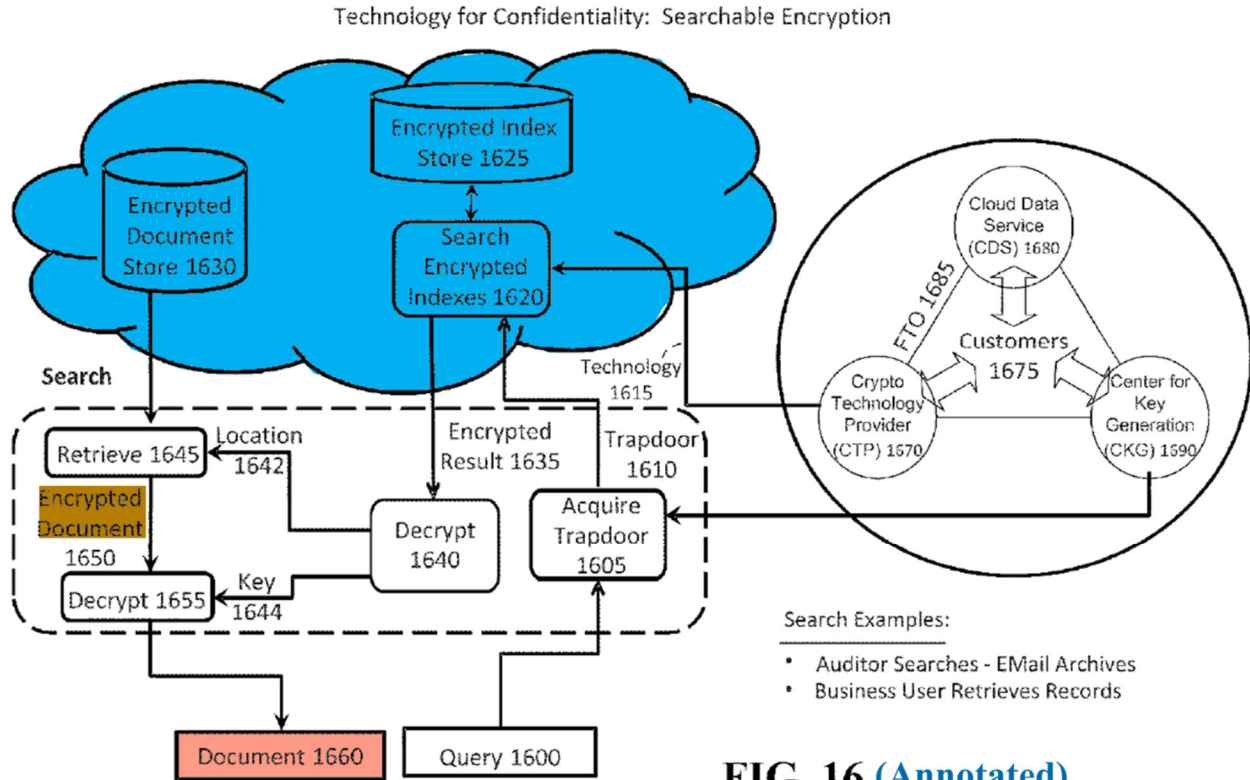


FIG. 15 (Annotated)

As Figure 15 shows, when searching encrypted documents, “customer 1675 form[ed] a query 1600, and then acquire[d] a trapdoor 1610,” which was “sent along with query 1600 to the cloud.” (*Id.*, ¶133, FIG. 16.)



“[B]ased on a search of the indexes, various content or records [could] be identified matching a given index or indexes and then the user [could] either access the matching content or records based on privileges or not.” (*Id.*, ¶127.) Specifically, “[i]n the cloud, the encrypted indexes in encrypted index store 1625 [were] searched...[, and]results 1635 [were] then returned still encrypted and decrypted at 1640.” (*Id.*, ¶133.)

Chiueh: Chiueh also concerned storing searchable encrypted documents to a remote site. Chiueh disclosed dividing a document into individual pages, encrypting them, assigning each encrypted page a document identification (ID), and uploading the encrypted pages to a remote server. (Chiueh, ¶¶20, 40, FIG. 2.)

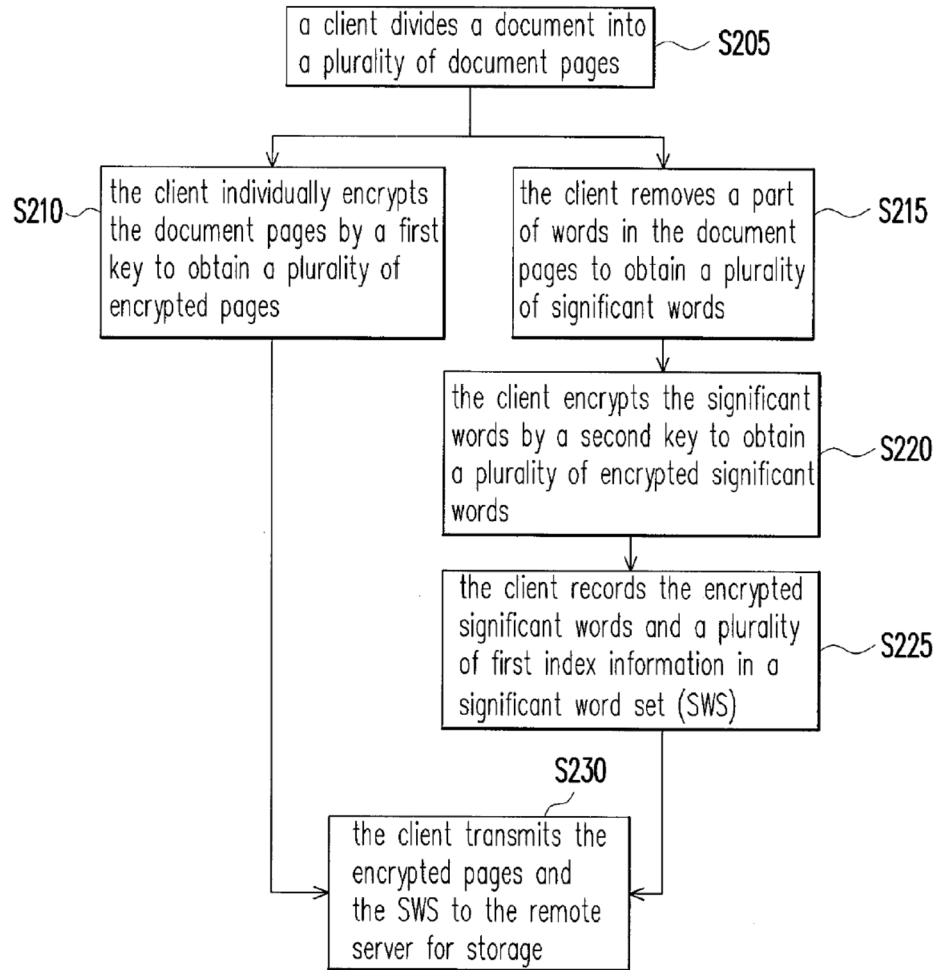


FIG. 2

In Figure 2, Chiueh disclosed creating an index (e.g., a significant word set (SWS)) mapping encrypted keywords to corresponding document ID(s) of the encrypted pages, so that the encrypted pages' content could be keyword searchable. (Chiueh, ¶¶21-23, FIG. 2.)

Chiueh's Figure 3 described a search process like the '678 Patent and Auradkar. Specifically, a user's search keyword was encrypted, and Chiueh's system searched a global search index (GSI) containing previously-uploaded

encrypted keywords to identify any document ID that matched the keyword. (*Id.*,

¶¶23-25, 40, FIG. 3.)

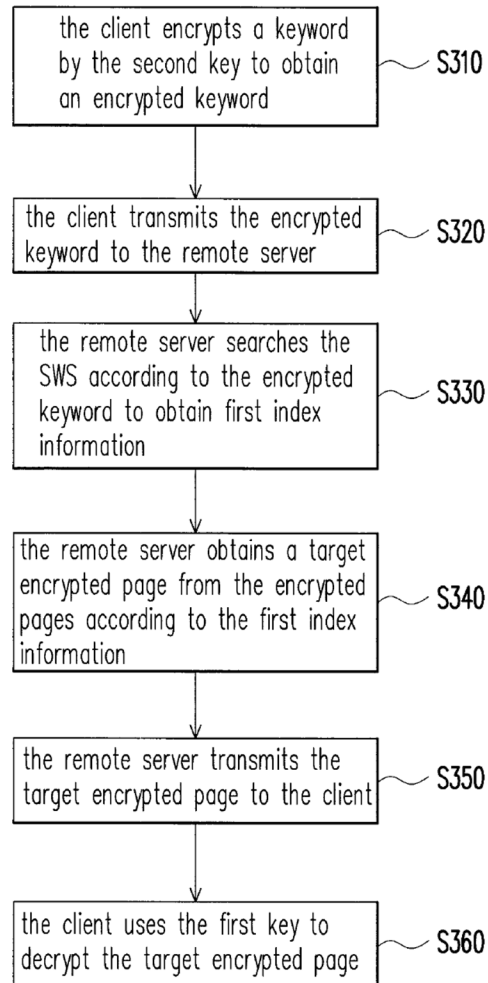
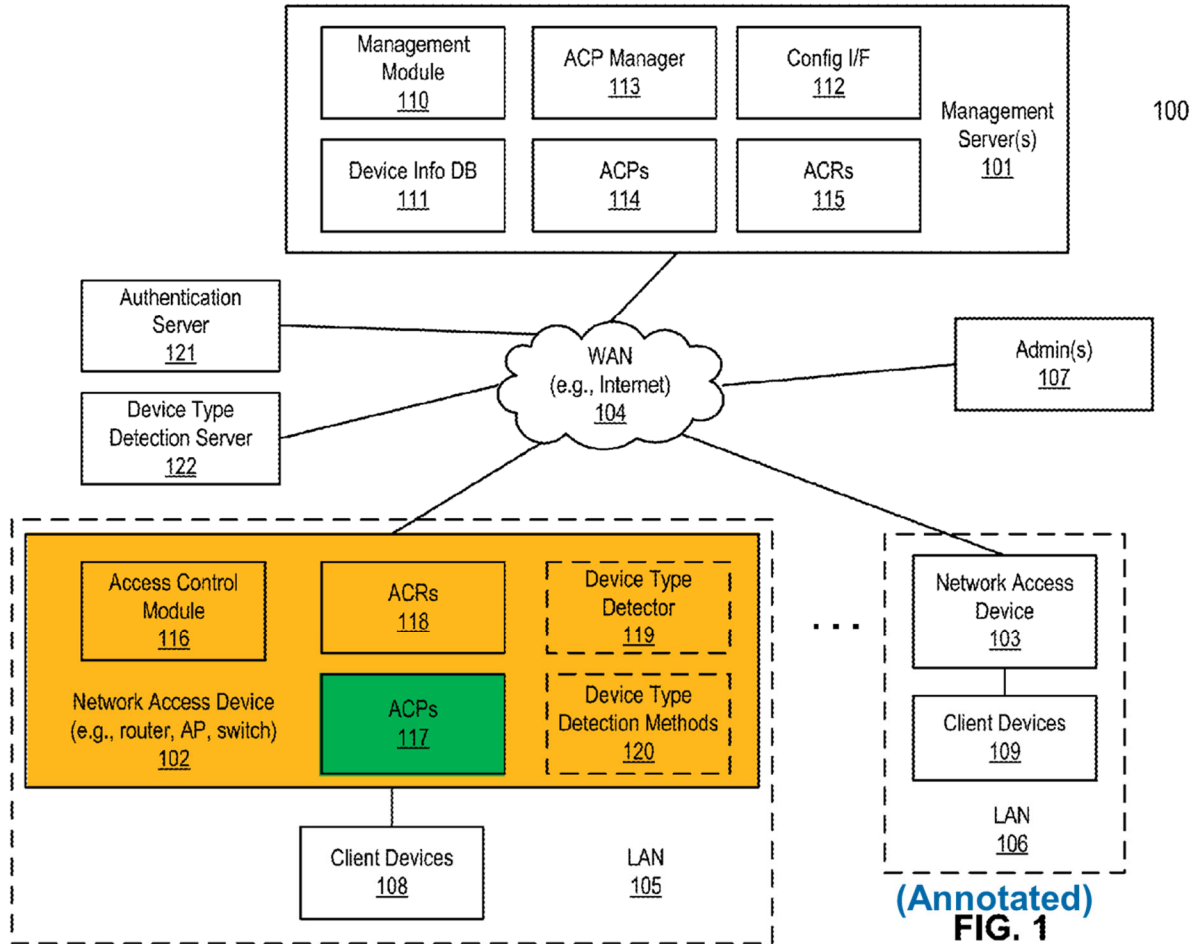


FIG. 3

Chambers: Chambers disclosed a network access device (**NAD**) 102 that “operate[d] as a gateway device...to LAN[] 105...where various client devices 108[]...[could] be communicatively coupled to LAN[] 105.” (Chambers, ¶24, FIG. 1.)



Chambers’ NAD 102 included an access control module (ACM) 116 that “select[ed] and applie[d] appropriate...[access control policies (]**ACPs**)] 117” to client devices 108. (*Id.*, ¶46; *see also id.*, ¶21.) Chambers is used below to illustrate that implementing the claim limitations in a gateway device was obvious.

1. Motivation to Combine

A POSITA would have been motivated to combine Auradkar, Chiueh, and Chambers with a reasonable expectation of success. (Lee, ¶¶185-187.)

All three references concern improving computer networking and security for accessing files stored remotely from client devices. (Auradka, Abstract;

Chiueh, ¶3; Chambers, Abstract.) Further, the Auradkar-Chiueh-Chambers combination would have produced a better system. For example, while Auradkar’s system allowed searching encrypted data, a POSITA would have been motivated to modify this technique by Chiueh’s mechanism that allowed searching individual document pages. Doing so would have enabled a more efficient search and retrieval of individual pages (chunks) instead of the whole document. (Auradkar, ¶¶127, 133, FIG. 16; Chiueh, ¶¶20, 40, FIGs. 2-3; Lee, ¶186.) Additionally, both Auradkar and Chambers aimed to implement policies for accessing protected enterprise data, and a POSITA would have understood that incorporating Chambers’ dedicated gateway as an interface between the client and protected data sources would have enhanced network security by, e.g., shielding these sources from the client. (Lee, ¶187.)

Additional motivations are discussed below.

2. Claim 1

a. 1[pre]

Auradkar disclosed a method for encrypting files and storing them. (*See, e.g.,* Auradkar, Abstract (“A digital escrow pattern and trustworthy platform is provided for data services including mathematical transformation techniques, such as searchable encryption techniques, for obscuring data stored at remote site or in a cloud service, distributing trust across multiple entities to avoid a single point of

data compromise.”), FIG. 15; Lee, ¶190.)

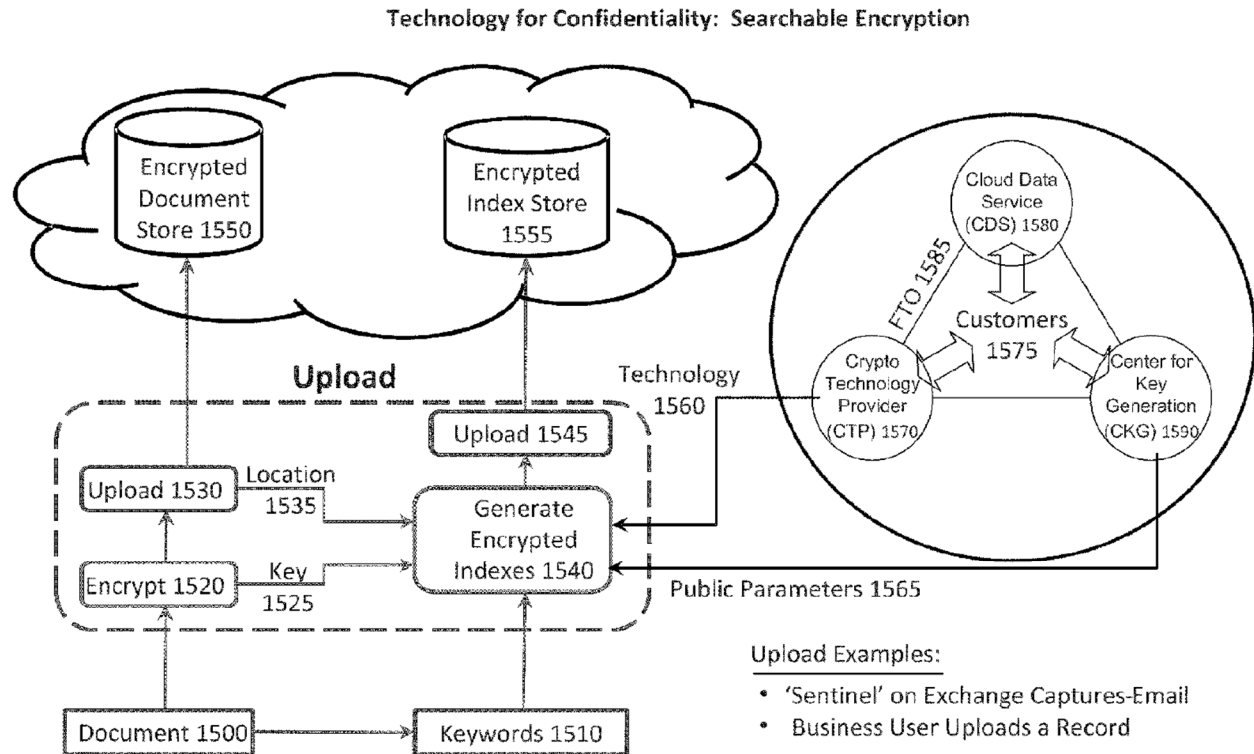


FIG. 15

Auradkar, FIG. 15

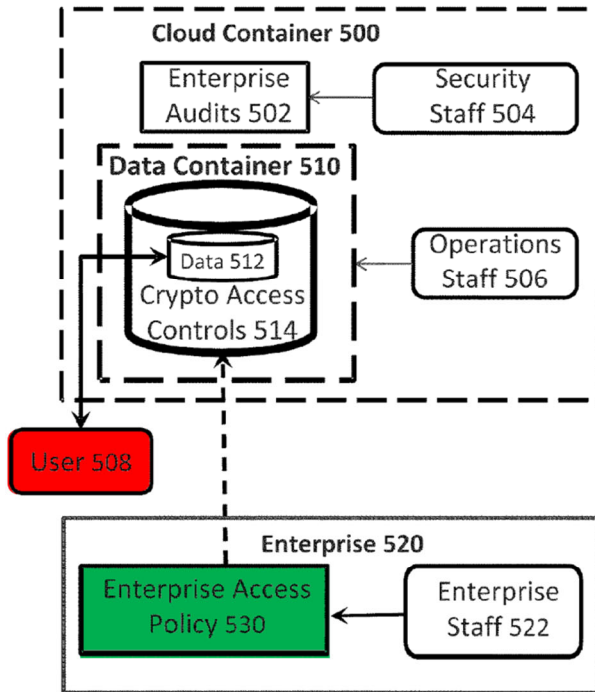
b. 1[a][i]

Auradkar disclosed assigning to **enterprise staff** (users) a **policy** for managing access to a file to be stored on **cloud container 500** (cloud platform). (*Id.*, ¶¶112-113, 116-117, FIGS. 1-2, 5-6.)

Similar to the '678 Patent's policy, which "can be selected, by an administrator...based on the role, responsibility, enterprise practices" ('678 Patent, 9:11-19), Auradkar's policy (e.g., policy 530 in annotated Figure 5 below) was set by "some of...enterprise staff 522" and different policies were applied to "different

accessing entities or tasks...to ensure that visibility is restricted to those to whom access should be enabled.” (Auradkar, ¶117, FIGS. 5-6, ¶119 (“[E]nterprise 620 define[d] its enterprise **access policy 630 applicable to enterprise staff 622....**”).)

Moreover, Auradkar’s policy managed access to and processed files to be stored in cloud (cloud platform). For example, as Figure 5 shows, Auradkar’s policy could “affect how data 512 stored in a data container 510 of a **cloud container 500** [could] be **accessed, manipulated**, retrieved, searched,” and “when users 508 of data 512 attempt[ed] to access such data 512, various **crypto access controls 514 guided by**, but separated from, **enterprise access policy 530** protect[ed] the data 512 from unwarranted access by users 508.” (*Id.*, ¶117, FIG. 5.)

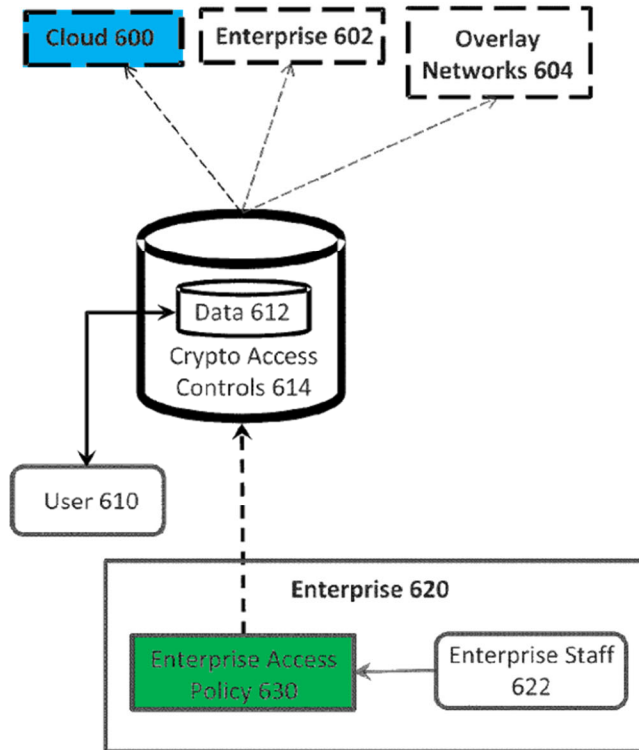


Data as Custodian

- Containers are Redundant for Security
- Access is Enforced by Crypto Wrapper
- Policy is Set by Owner and guaranteed by Crypto Wrapper

FIG. 5 (Annotated)

Similar operations were discussed in connection with Figure 6, annotated below. (*See id.*, ¶119, FIG. 6.)



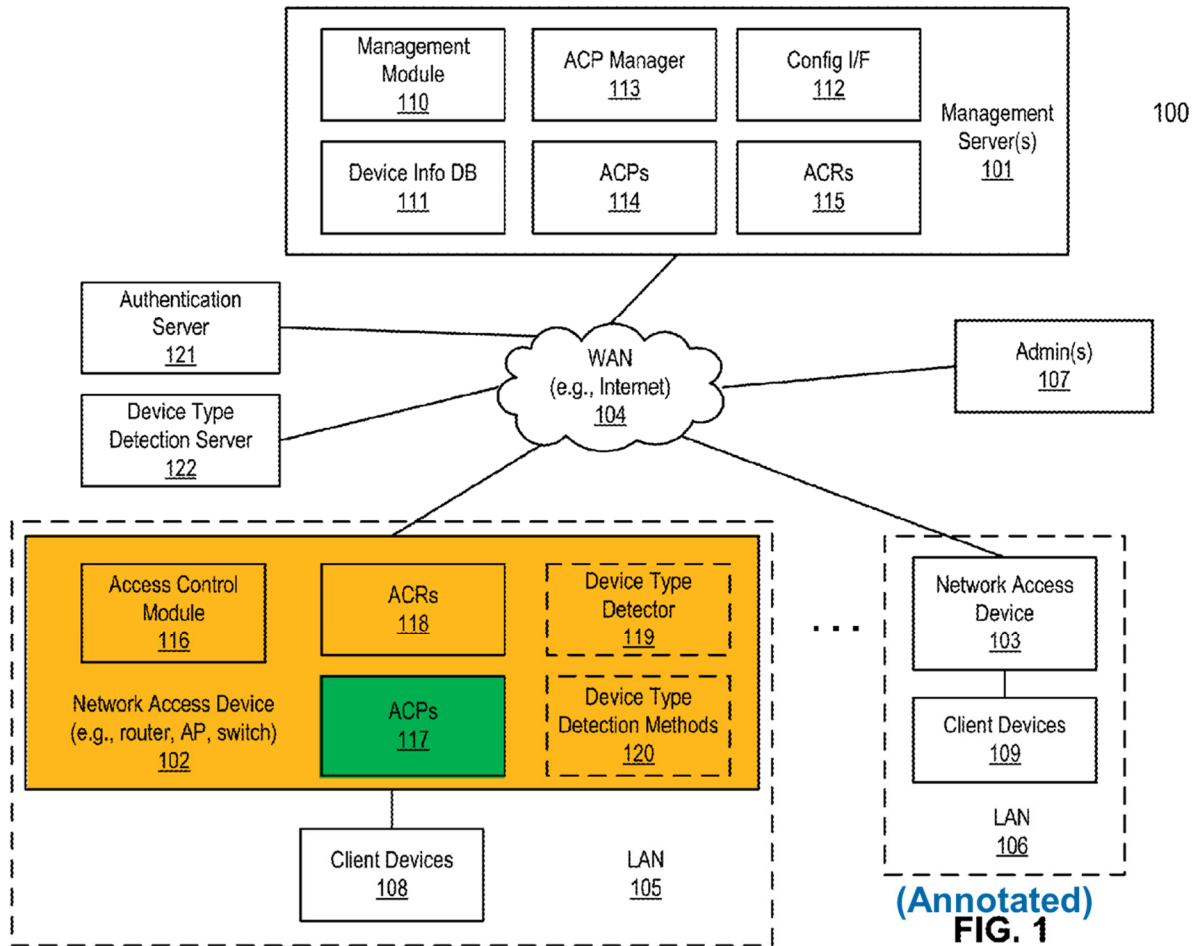
Data can Transcend Containers

- The Systems / Security Equation Changes
- Data can Straddle Containers
- Placement can optimize Access, Resilience, etc.
- New Challenges are Consistency, Versioning, Garbage Collection, etc.

FIG. 6 (Annotated)

Assigning the Policy: While Auradkar disclosed assigning a policy to users, it did not explicitly disclose a **gateway device** assigning the policy. Chambers disclosed this feature. (Lee, ¶¶195-198.)

Chambers disclosed a network access device (**NAD**) 102 (gateway device) that “operate[d] as a **gateway device**, an access point (AP), a network switch, *or a combination thereof*” and assigned an access policy to a client device. (Chambers, ¶24, FIG. 1; Lee, ¶196.)



Chambers’ NAD 102 included an access control module (ACM) 116 that “select[ed] and applie[d] appropriate...[access control policies (]**ACPs**)] 117,” stored in NAD 102, to client devices 108. (*Id.*, ¶46.) After selecting and applying “an appropriate ACP,” NAD 102 (via its ACM) sent “update information” concerning the selected ACP to management server 101, which then broadcasted the update information to other NACs. (*Id.*) A POSITA would have understood that by *selecting and applying an appropriate* access control policy to a client device without intervention from an administrator or the management server, Chambers’ NAD 102 assigned this policy to the client device. Indeed, Chambers

described its “[t]echniques for **automatically assigning**” and applying policies.
(*Id.*, ¶21.)

Chambers’ Figure 4A (below) also illustrated that “Group policy” assigned to “staff” and “iPad group,” which a POSITA would have recognized constituted assigning policies to groups of users.

400

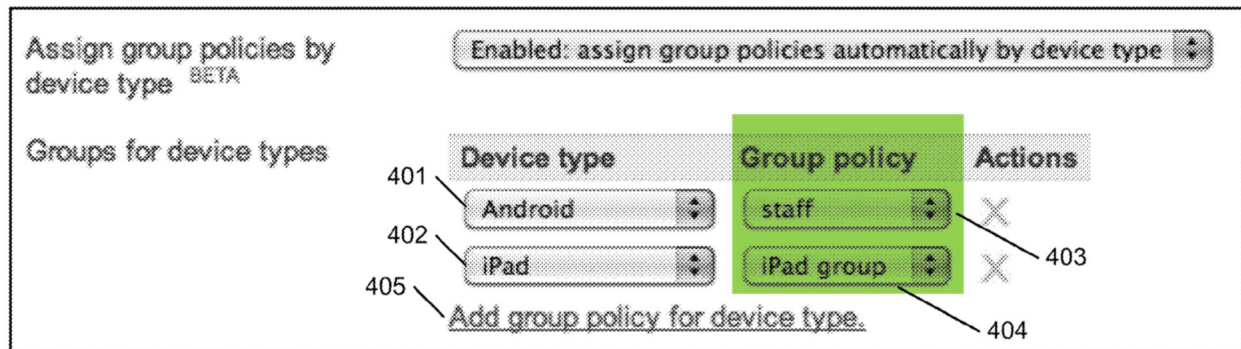


FIG. 4A (Annotated)

While Auradkar disclosed the claimed policy assigning function (as explained above), it did not explicitly disclose what device implemented it. A POSITA would have found it obvious to use Chambers’ gateway to implement Auradkar’s “assigning” function (and other claimed functions), and to automatically assign the pertinent security policies as Chambers described. Below is an illustration of this combination using Auradkar’s Figure 11.

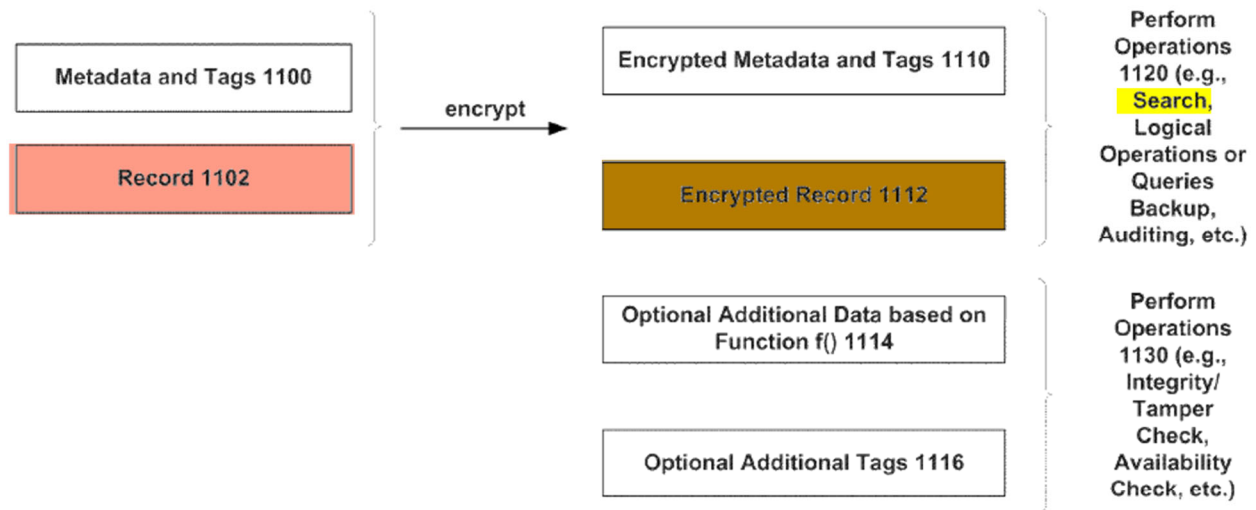


FIG. 11 (Annotated)

A POSITA would have recognized that this combination would be beneficial because an organization may have different types of users, each of whom may use multiple devices (e.g., a laptop and smartphone) to access cloud data. Incorporating Chambers’ teachings into Auradkar’s system would have allowed automatic assignment of a policy based on, e.g., a user’s role in the organization, or the user’s devices. The combination would have resulted in consistent application of the policy and would have avoided repeated policy configuration for the same user. (Lee, ¶200.) Further, automatically configuring, assigning, and implementing policies (as Chambers disclosed) would have improved system efficiency and reduced human errors. (Chambers, ¶¶5-6 (explaining that manual configuration and assignment of policies could be “cumbersome, time-consuming

and error-prone”), 21; Lee, ¶200.) The modified system would have allowed better real-time protection by automatically applying (and assigning) updated policies whenever such updates occurred. (Chambers, ¶29; Lee, ¶200.)

As noted earlier, using Chambers’ dedicated gateway as an interface between the client and protected data sources would have also enhanced network security by, e.g., shielding these sources from the client. (*See* Section VIII.D.2; Lee, ¶201.)

A POSITA would have reasonably expected success in making this combination because gateway devices were well-known for network security-related functionality, including controlling network traffic and managing access policies. (*See, e.g.*, Chambers, ¶¶66, 82; Herrmann, ¶¶18, 77; Lee, ¶202.) Thus, adding Chambers’ gateway into Auradkar’s system would have yielded predictable results. (Lee, ¶202.)

c. 1[a][ii]

Auradkar disclosed that its policy defined user access rights. For example, Auradkar disclosed that “[w]ith **policy and enforcement** within the control of users or an enterprise, data and corresponding logs [were] encrypted and **accessible only with specific capabilities granted to a user.**” (Auradkar, ¶112.) Auradkar also described that its policy could “affect how data 512 stored in a data container 510 of a cloud container 500 [could] be **accessed, manipulated,**

retrieved, searched” and “when users 508 of data 512 attempt[ed] to access such data 512, various crypto access controls 514 guided by, but separated from, enterprise access policy 530 protect the data 512 from unwarranted access by users 508.” (Auradkar, ¶117.)

d. 1[b]

Auradkar’s Figure 11 illustrated encrypting a “**record 1102** (e.g., data payload) and associated metadata and/or tags 1100” to produce encrypted metadata and tags 1110 and **encrypted record 1112**, on which “various operations 1120 can be performed,” including “search of the data.” (Auradkar, ¶124, FIG. 11.)

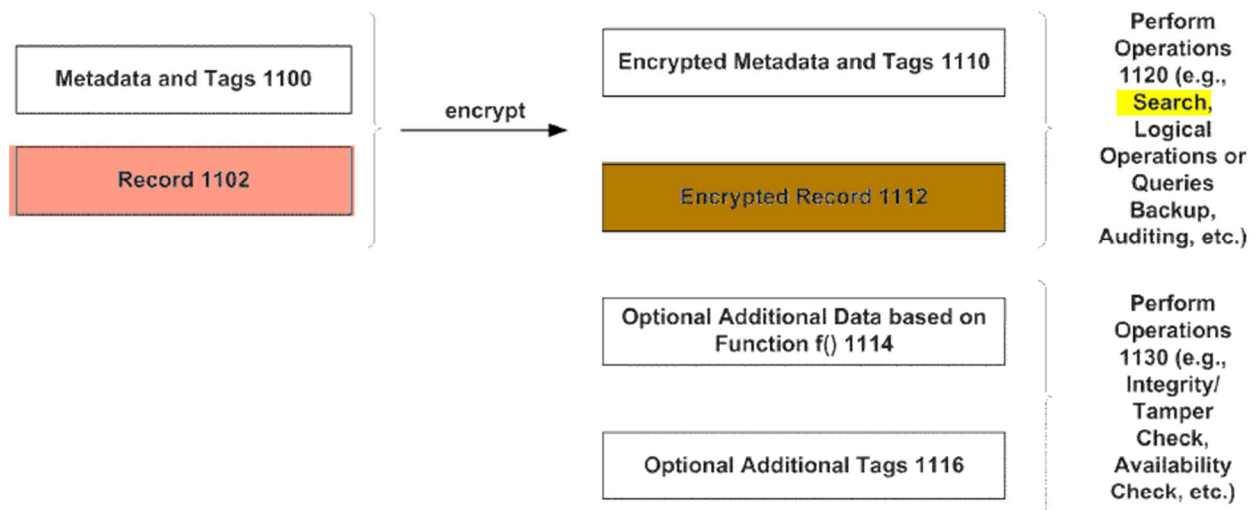


FIG. 11 (Annotated)

In Figure 12’s example, **payload 1202** and tags 1200—which included a filename (“FName: fox.txt”)—were “encrypted to form encrypted tags 1210 and

encrypted data 1212 for operations 1220,” including search operations. (*Id.*, ¶125, FIG. 12.)

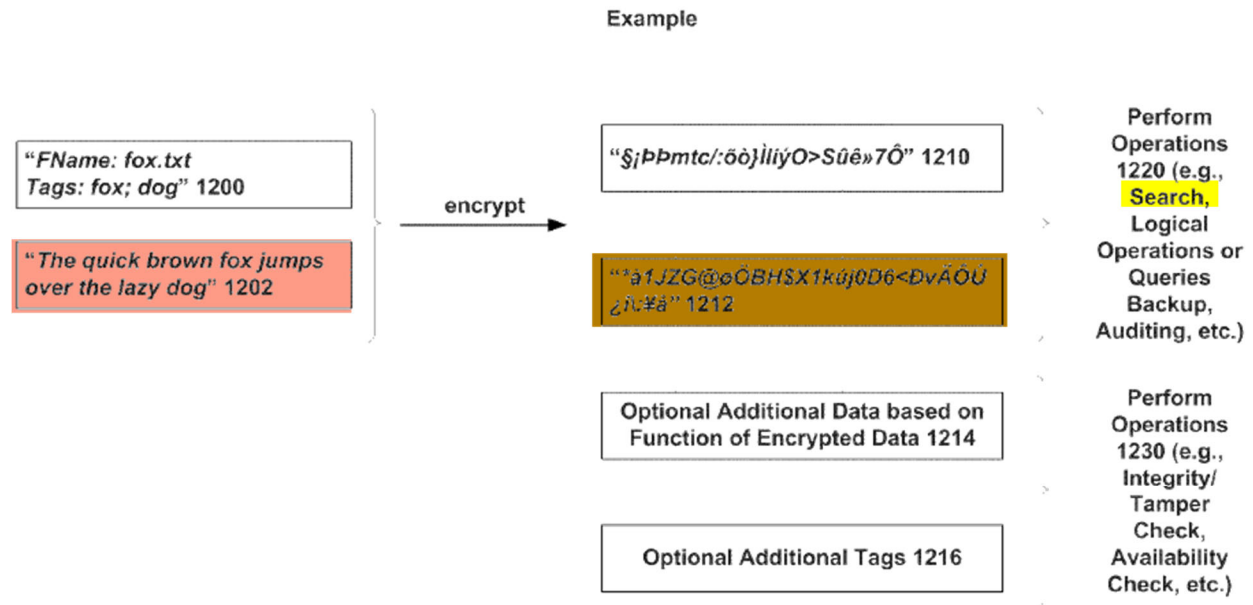


FIG. 12 (Annotated)

Auradkar also described encrypting a “document 1500 with which various keywords 1510 [were] associated” and uploading the encrypted document to “the cloud into an encrypted document store 1550,” as shown in Figure 15. (*Id.*, ¶130, FIG. 15.)

Technology for Confidentiality: Searchable Encryption

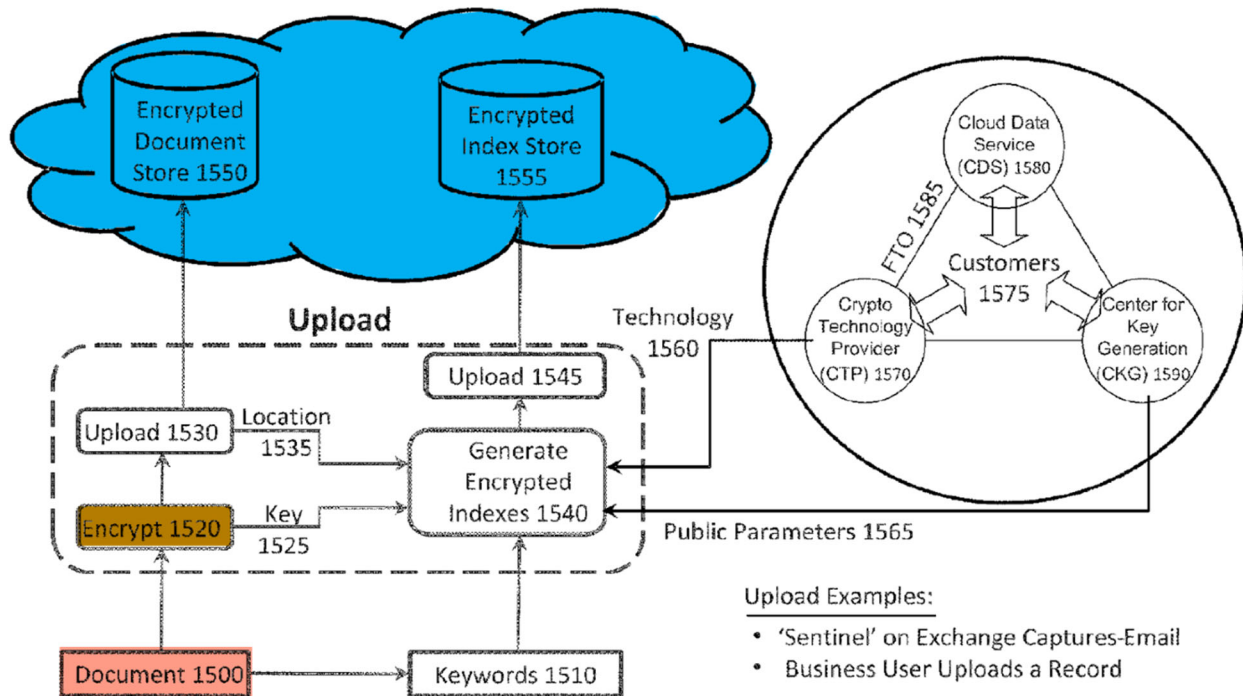
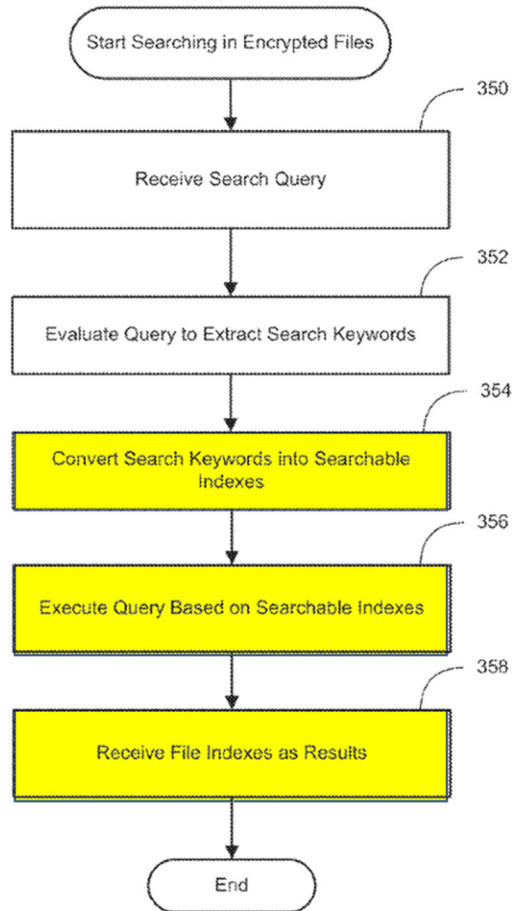


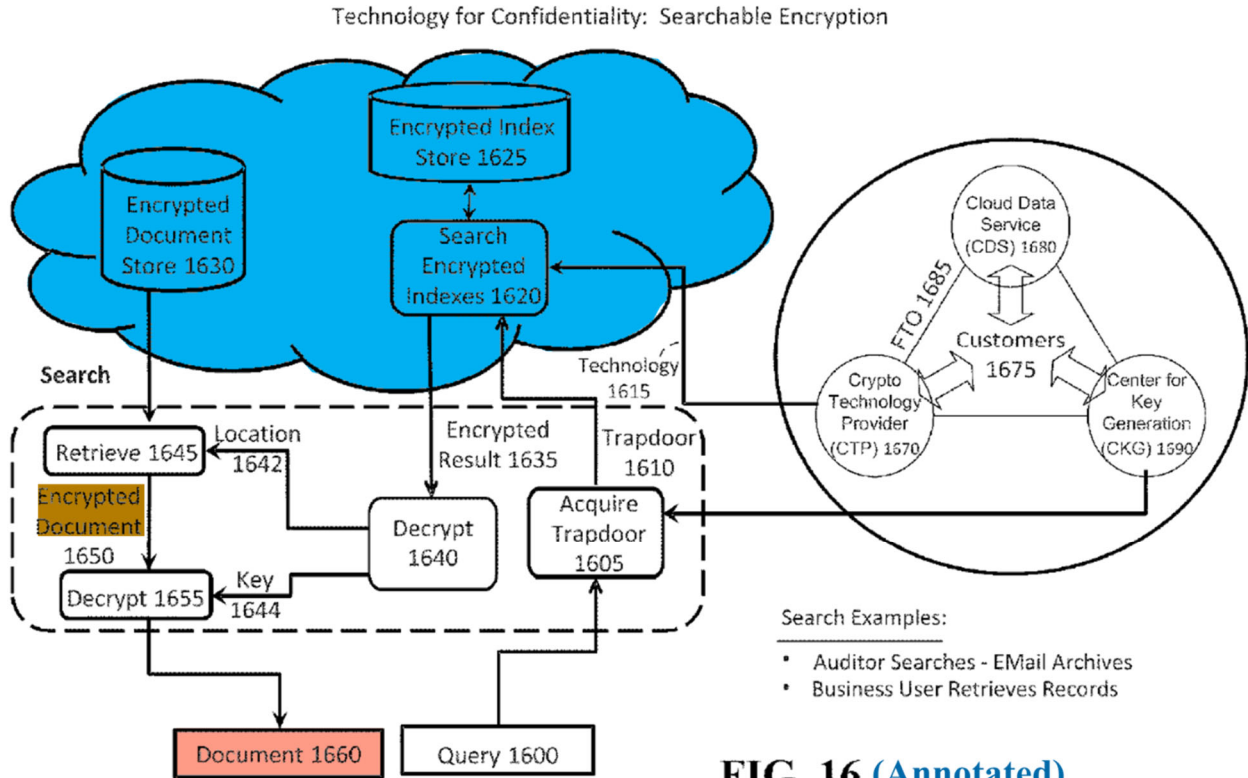
FIG. 15 (Annotated)

Similar to the '678 Patent's keywords (content of the file), which are indexed and encrypted (e.g., "search keyword 'Ronn' can be indexed as 'uevbwq' and Eric can be indexed as 'uastyb'"; '678 Patent, FIG. 3B, 12:44-47), Auradkar's keywords 1510 were also encrypted to generate encrypted indexes 1540, which were used for searching the encrypted document. (Auradkar, ¶130, FIG. 15.)



'678 Patent, Fig. 3B (Annotated)

For example, Auradkar described that to perform a search, “customer 1675 form[ed] a query 1600, and then acquire[d] a trapdoor 1610,” which was “sent along with query 1600 to the cloud.” (Auradkar, ¶133, FIG. 16.)



“[B]ased on a search of the indexes, various content or records [could] be identified matching a given index or indexes and then the user [could] either access the matching content or records **based on privileges or not.**” (*Id.*, ¶127.) Specifically, “[i]n the cloud, the encrypted indexes in encrypted index store 1625 [were] searched...[, and]results 1635 [were] then returned still encrypted and decrypted at 1640.” (*Id.*, ¶133.)

Cryptographic Key Information: While the ’678 Patent does not explicitly define “cryptographic key information,” it describes that “a selected policy can be used to define encryption keys, decryption keys, and encryption type, among other attributes for carrying out the encryption of data.” (’678 Patent, 4:4-7.) Consistent

with this description, Auradkar’s policy defined the encryption type and the length of encryption/decryption key. Specifically, Auradkar’s policy specified that certain data, such as “sensitive[] medical data,” must be encrypted using an encryption method specified in the policy (e.g., “using **AES 256 encryption**”). (Auradkar, ¶96.) A POSITA would have understood that this section of Auradkar is specifying the Advanced Encryption Standard (AES) encryption algorithm (encryption type) with a 256-bit encryption/decryption key.² (Auradkar, ¶96; Lee, ¶210.)

As explained for Element 1[a][i] (Ground 4), a POSITA would have found this obvious and would have had a reasonable expectation of success in combining Chambers’ gateway with Auradkar’s system such that the claimed operations would be carried out by the gateway. (*See supra* VIII.D.2.b; Lee, ¶211.)

e. 1[b][i]

Auradkar disclosed this limitation because it disclosed “breaking an **XML document**” (file) into XML fragments (chunks) with “potentially different protection levels,” such as “Public, Secret and Top Secret paragraphs,” and encrypting the XML fragments to generate **encrypted XML fragments**. (Auradkar, ¶¶150, 92, 95, FIG. 19; *see also* FIGS. 11-12.)

² Other common AES key lengths were 128 and 192 bits. (Lee, ¶210.)

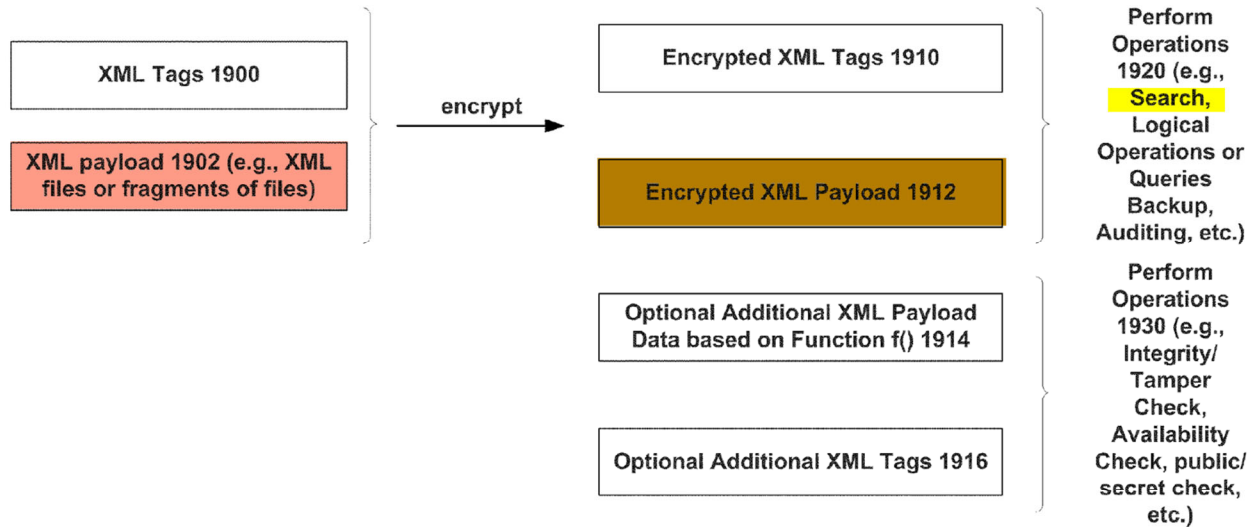


FIG. 19 (Annotated)

(*See also id.*, ¶¶98 (describing that “the confidentiality level labels of public, secret or top secret [could] be applied to each payload element of the XML document **on a fragment by fragment basis**, and included in the auxiliary encrypted metadata to achieve highly granular **policy around access to parts** of the XML document”); 95 (explaining that a “person performing a search and requesting access with a Secret clearance would only get access to Public and Secret paragraphs,” and “a policy [could] be implemented that Top Secret content [couldn’t] be accessed from a wireless or remote device”).)

f. 1[b][ii]

Auradkar and the Auradkar-Chiueh combination taught this limitation. (Lee, ¶¶212-214.) Auradkar disclosed creating tags for payloads (e.g., payload 1200,

XML fragment 1902 (chunks)), such as **tag 1200** that included the filename of the payload (FName: fox.tx) and **XML tags 1900**, and encrypting them. The encrypted tags were used in operations 1230, including searching. (Auradkar, ¶¶125, 150, FIGS. 12, 19.)

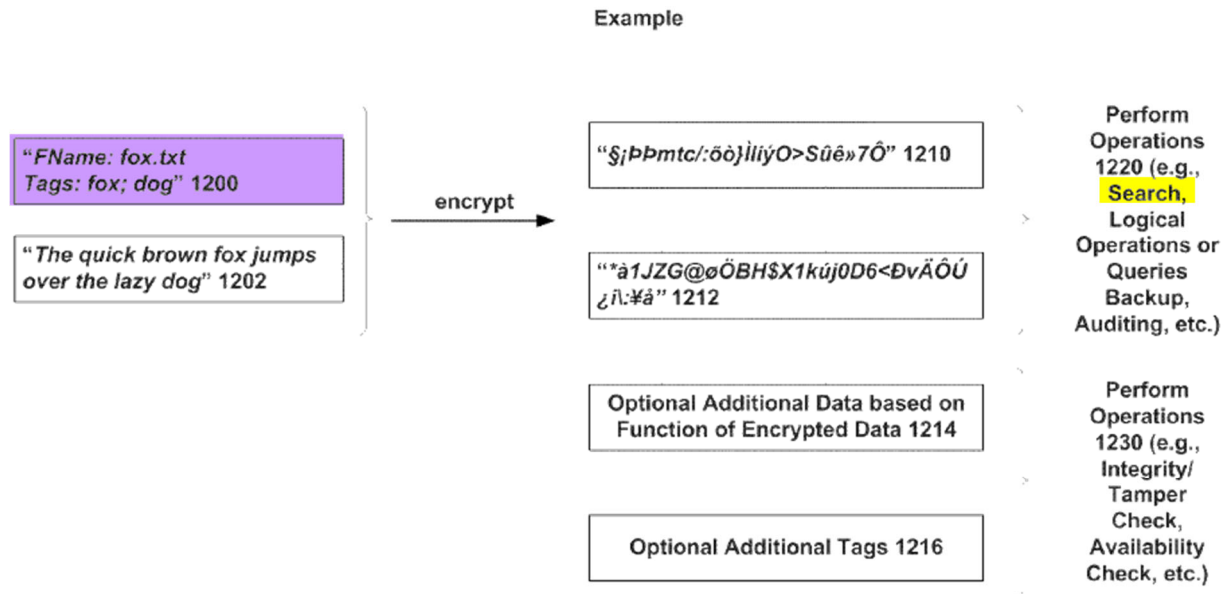


FIG. 12 (Annotated)

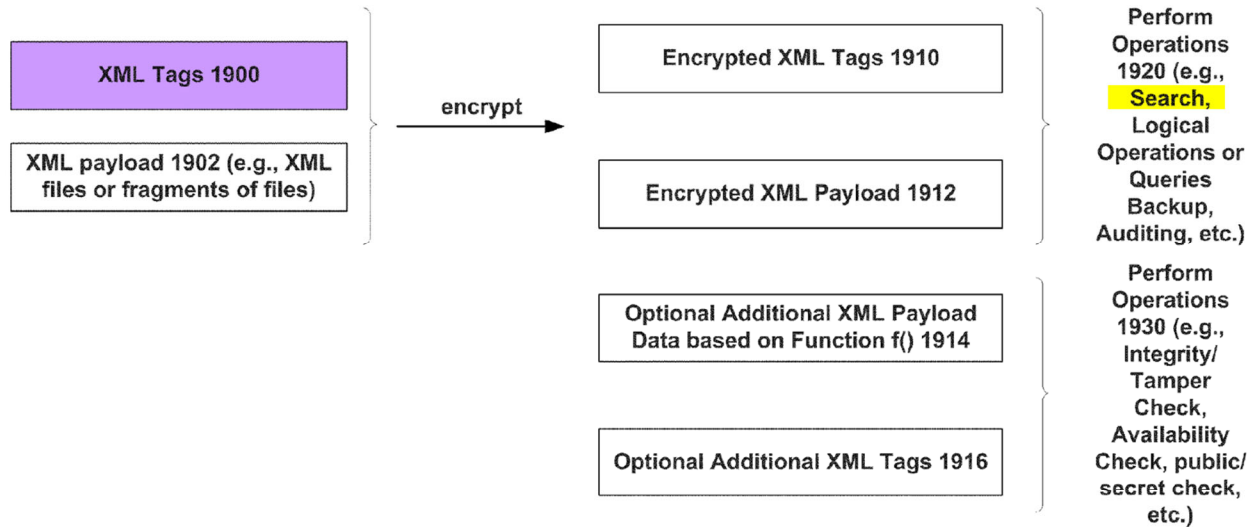


FIG. 19 (Annotated)

A POSITA would have found it obvious that these tags constituted namespaces for the payloads (e.g., file 1202), because, consistent with the plain and ordinary meaning of a namespace, they were a “set of names available for naming things” such as files or segments. (Ex. 1014; Lee, ¶217.)

If Auradkar did not disclose creating and configuring namespaces for chunks, Chiueh taught or suggested it. (Lee, ¶¶218-219.) Chiueh disclosed “break[ing] a document into a plurality of pages” and “individually encrypt[ing them]...to [produce] a plurality of encrypted pages.” (Chiueh, ¶20.) Chiueh’s system also created and assigned a document identification or ID (namespace) to each of the encrypted pages. (Chiueh, ¶20.) “Each of the encrypted words in the [global search index (]GSI[)] was] mapped to a corresponding document ID, where

the **document ID indicate[d] an encrypted page where a given word [could] be found.**” (Chiueh, ¶40.) Similar to Auradkar’s tag 1200, which included a filename with other tag information (Auradkar, FIG. 12), Chiueh’s document ID was “**combination of a file name and a page offset in the document**, and the file name and the page offset [were] combined and encrypted.” (Chiueh, ¶40.)

The Auradkar-Chiueh combination would have allowed creating document IDs that identified portions of a document (e.g., a document page, an XML fragment) and searching the encrypted files using the document IDs (as explained in Limitation 1[b][iii] below). (Lee, ¶220.) A POSITA would have made this modification because, in addition to Auradkar’s existing whole-document search, the modified system would have allowed retrieval of encrypted document portions stored in the cloud, “so that a bandwidth resource of the communication network [could] be effectively saved.” (Chiueh, ¶28; Lee, ¶220.)

A POSITA would have reasonably expected success in incorporating Chiueh’s functionalities into Auradkar’s system because Auradkar already disclosed generating tags, which could include filenames for payloads, and using these tags to search the encrypted payloads. Additionally, creating identifiers for portions of a document had been well-known, would have been compatible with Auradkar’s system, and the combination would have yielded expected results. (See Auradkar, ¶¶125, 150, FIGS. 12, 19; Chiueh, ¶40, FIG. 2; Lee, ¶221.)

g. 1[b][iii]

The Auradkar-Chiueh combination disclosed this limitation.

The '678 Patent does not describe what namespaces are or how they are configured, but it uses this term in the context of a search process that involves generating searchable keyword indexes ('678 Patent, 12:14-18), encrypting keywords used in a search query (*Id.*, 12:42-47), and searching the encrypted index to identify any file chunks that match the encrypted search keywords (*Id.*, 12:48-55).

Both Auradkar and Chiueh described similar procedures. In the Auradkar-Chiueh combination, the documents IDs/tags would be configured to allow search, identification, and retrieval of document segments (e.g., a document page, an XML fragment). Specifically, Auradkar disclosed creating and encrypting tags for a payload (e.g., an XML fragment); the tags included a filename (see e.g., “FName: fox.txt” in Figure 12), were encrypted and used to perform various “operations 1220,” such as searching the encrypted files. (Auradkar, ¶¶125, 150, FIGS. 12, 19.)

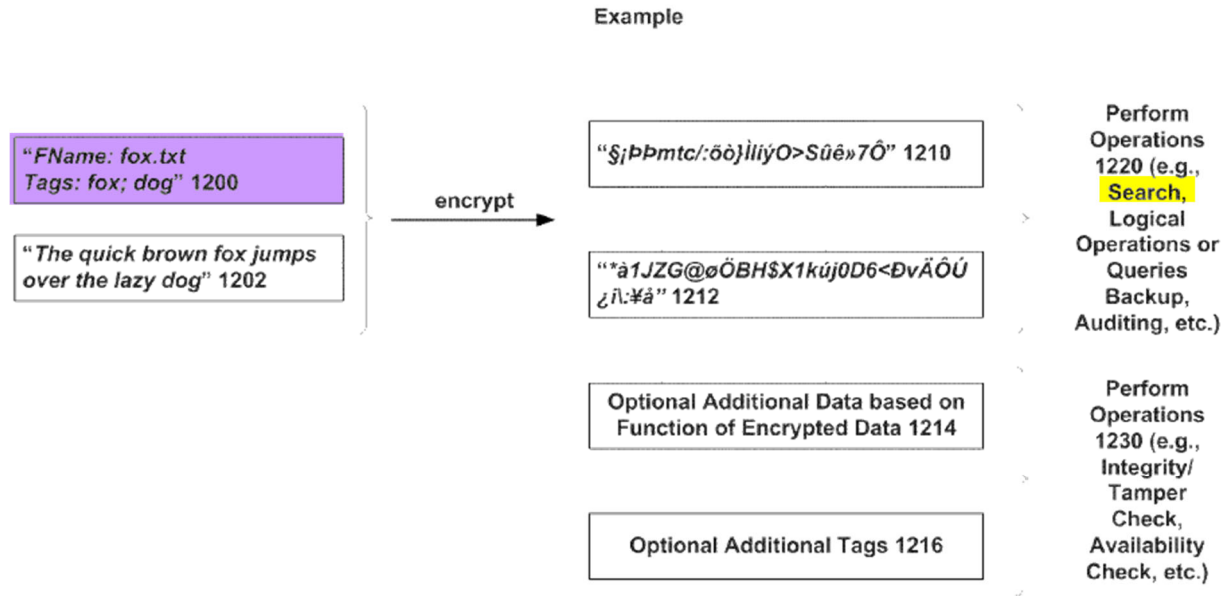


FIG. 12 (Annotated)

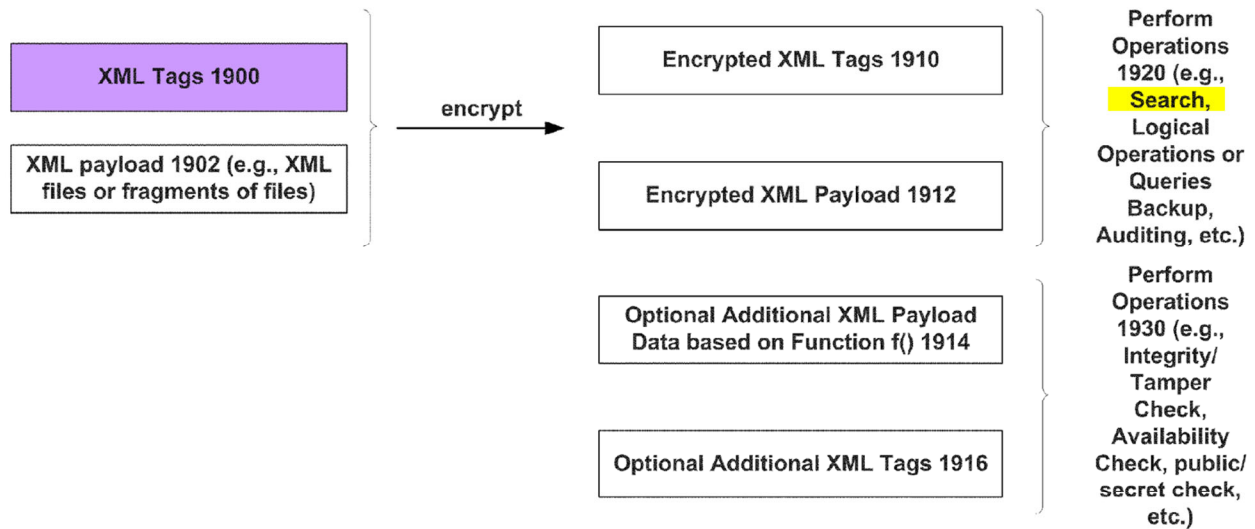


FIG. 19 (Annotated)

Auradkar also disclosed encrypting a **document 1500** and various keywords associated with the document to generate an **encrypted document** and

associated encrypted indexes (containing the encrypted keywords), which were uploaded to the cloud. (Auradkar, ¶130, FIG. 15.)

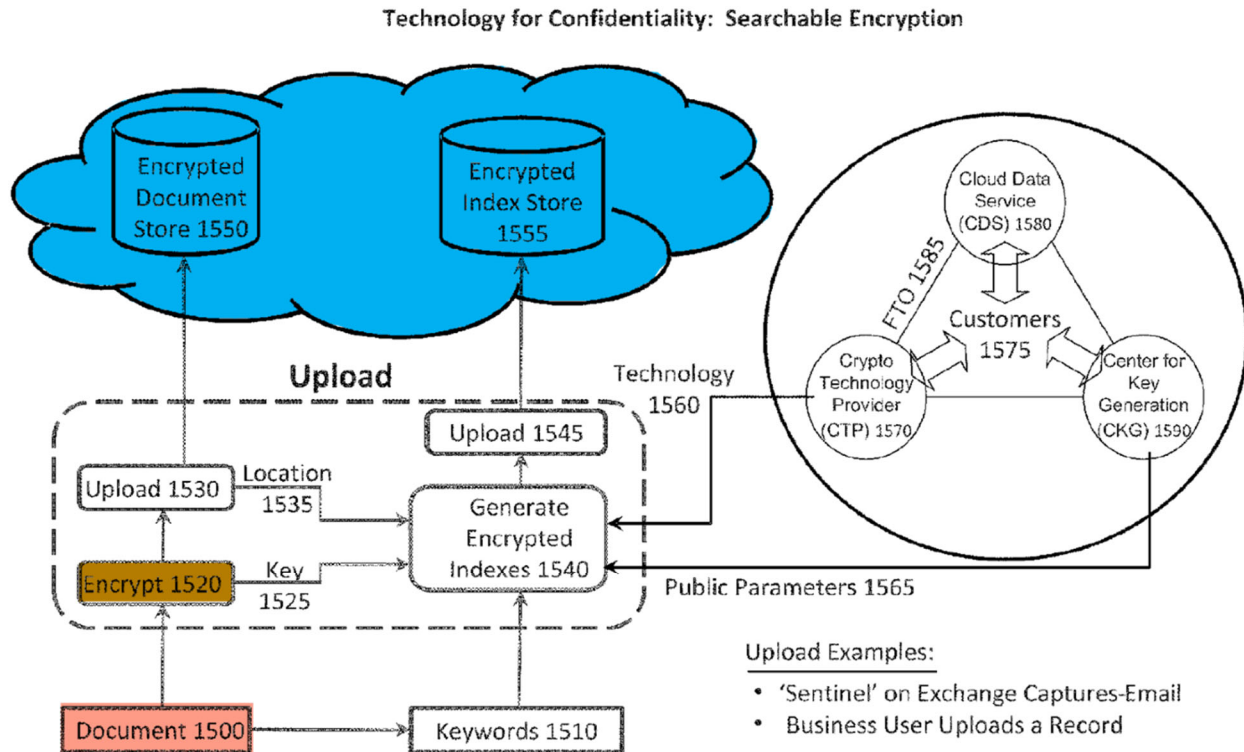


FIG. 15 (Annotated)

When searching the encrypted documents, Auradkar formed a search query that included encrypted search keywords (¶133, FIG. 16.), and searched the encrypted indexes stored in the cloud to identify “content or records [could] be identified matching a given index” (¶127).

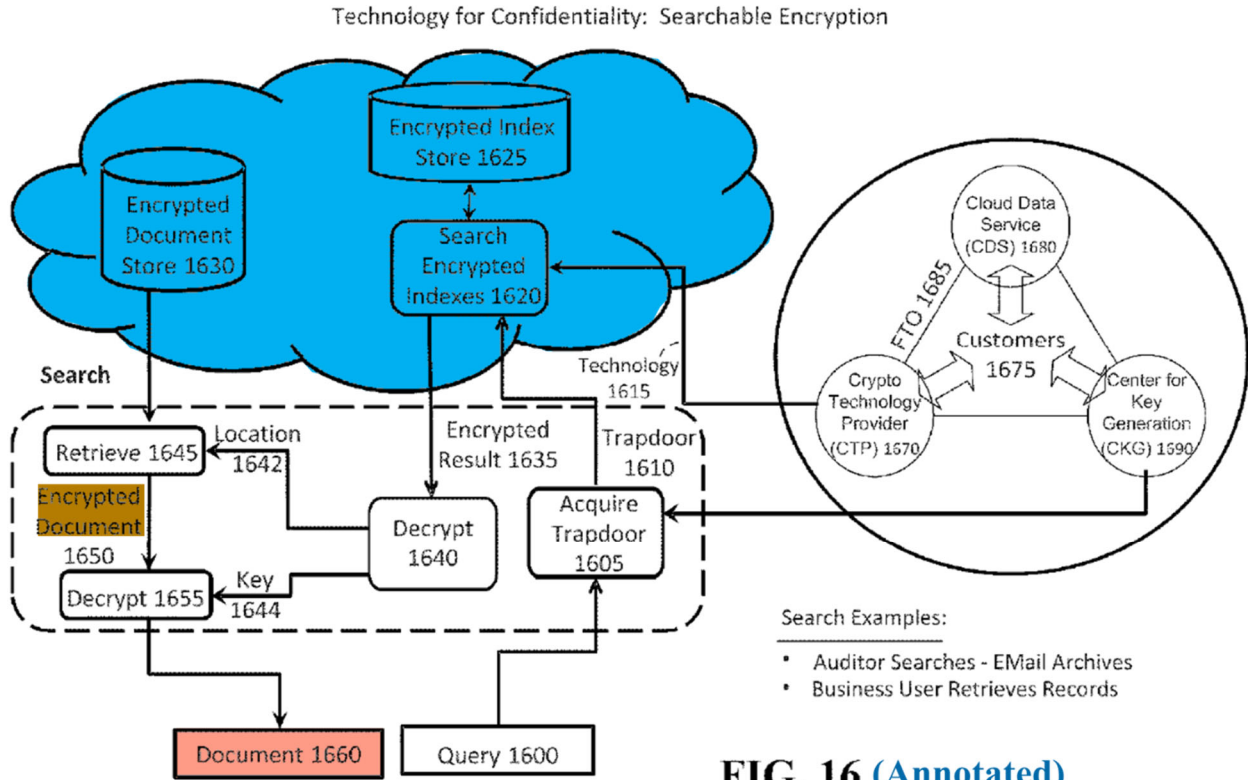


FIG. 16 (Annotated)

Chiueh disclosed encrypting individual document pages (chunks) to obtain corresponding encrypted pages and assigning an ID (namespace) to each of the encrypted pages. (Chiueh, ¶¶20, 40, FIG. 2; *supra* VIII.D.2.f.)

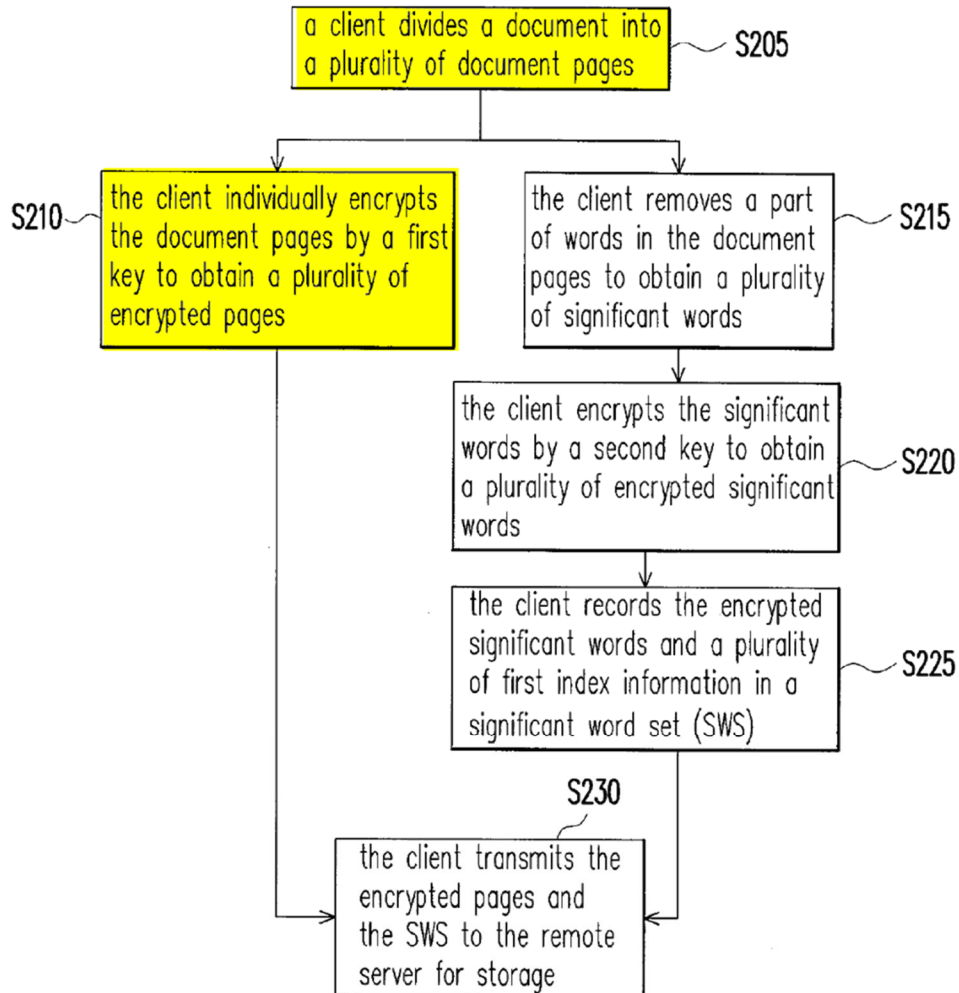


FIG. 2 (Annotated)

Chiueh also disclosed creating an index (e.g., a significant word set (SWS)) that mapped encrypted keywords to corresponding document ID(s) of the encrypted pages, so that the content of the encrypted pages could be searched using those keywords. (*Id.*, ¶¶21-23, FIG. 2.)

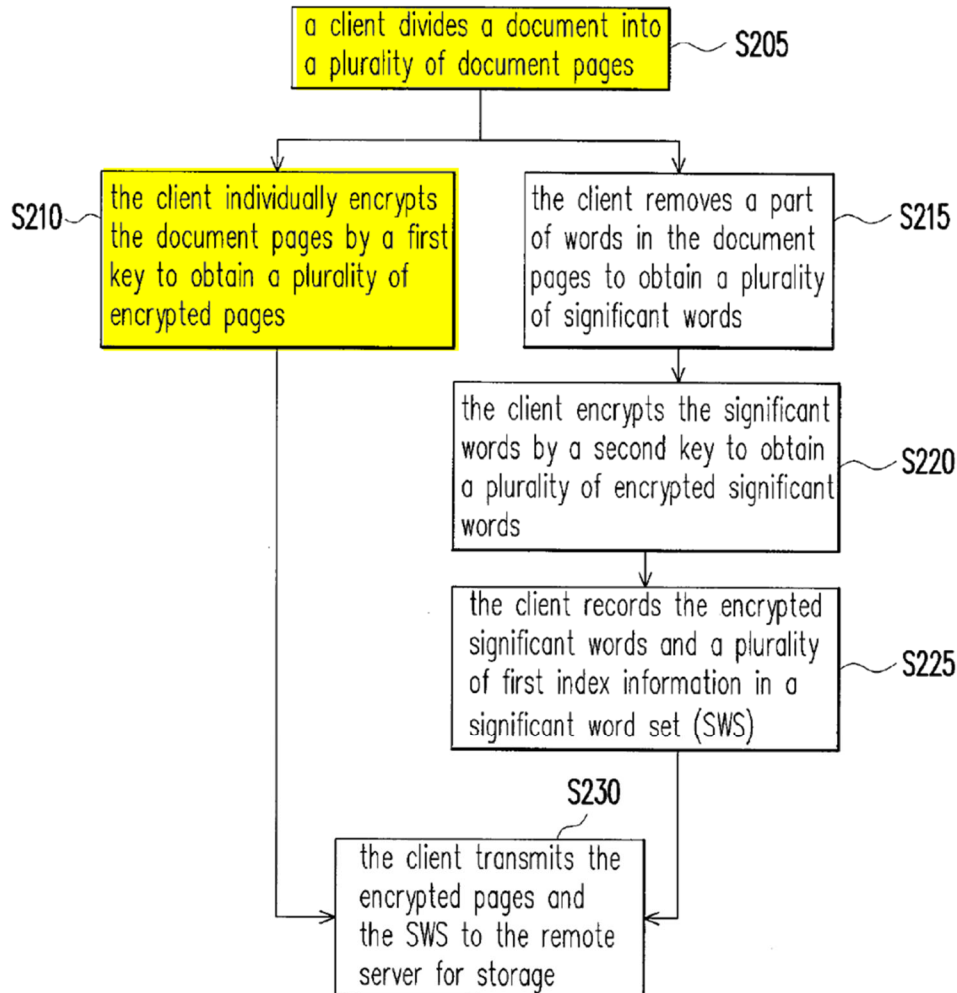


FIG. 2 (Annotated)

For example, the encrypted pages and index (SWS) were transmitted to a remote server 12, which then stored the SWS in a global search index (GSI) that included a key field that “record[ed] the encrypted significant words” and a value field that “record[ed] the first **index information.**” (*Id.*, ¶24.) This “index information” “indicate[d] a page in the encrypted pages where the encrypted significant word [came] from,” such as document IDs. (*Id.*, ¶¶24, 40.)

Similar to the '678 Patent's search process, Chiueh involved “encrypt[ing]

the keyword KW...to obtain an encrypted keyword” and “search[ing] the SWS according to the encrypted keyword, i.e., search[ing] the GSI to obtain the **index information** corresponding to the encrypted keyword.” (Chiueh, ¶25, FIG. 3.) And “[i]f the remote server 120 [found] the encrypted [keyword] from the SWS, the remote server 120 obtain[ed] at least one target encrypted page from the encrypted pages stored in the database according to the **index information** corresponding to the encrypted keyword...[and] transmit[ted] the target encrypted page to the client 110.” (*Id.*, ¶25, FIG. 3.)

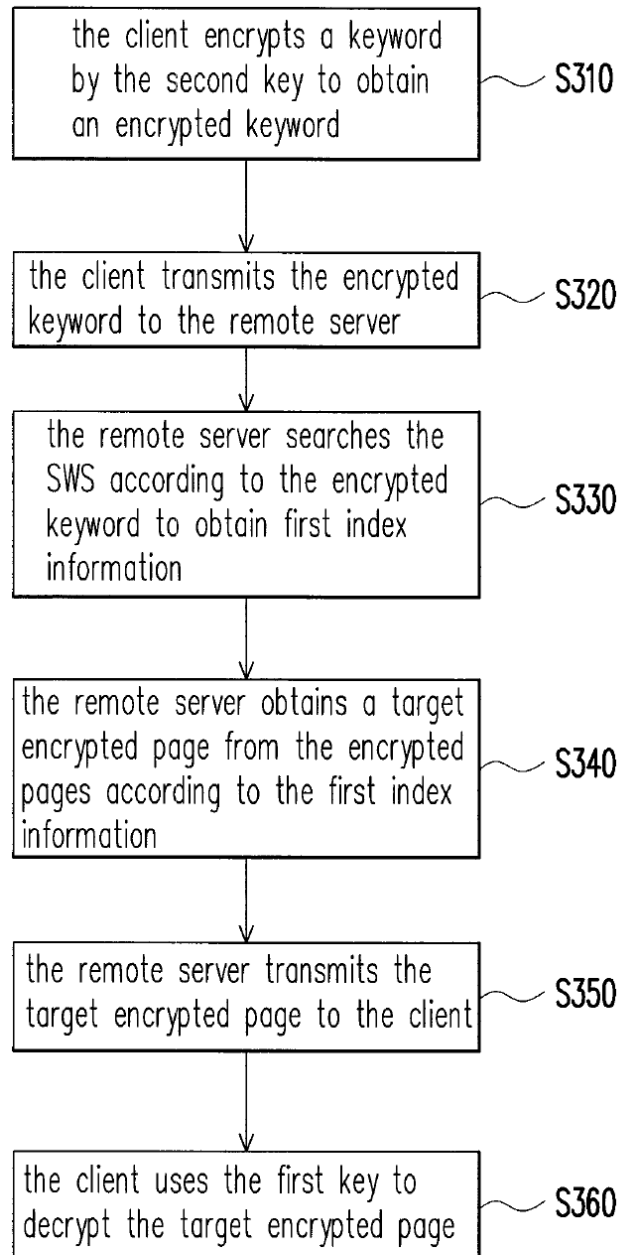


FIG. 3

A POSITA would have found it obvious that Chiueh's system configured document IDs (namespaces) of the encrypted pages (chunks) because it mapped these IDs (indicated in the index information) to keywords appearing in the

corresponding pages in a global search index (GSI), such that the page content in encrypted form was searchable. Indeed, consistent with the description of the '678 Patent, Chiueh's system enabled users to search encrypted page content using a search keyword to identify the ID that was mapped to the keyword, and to retrieve the corresponding page based on the identified ID. (Chiueh, ¶¶23-24, 40, FIG. 3; Lee, ¶232.)

A POSITA would have found it obvious to add Chiueh's functionality of configuring namespaces that allowed search and retrieval of individual pages into Auradkar's system based at least on Auradkar's suggestion for searching encrypted XML fragments. In addition, doing so would have allowed another option for retrieving individual pages, instead of the whole document, "so that a bandwidth resource of the communication network [could] be effectively saved." (Chiueh, ¶28; Lee, ¶233.) A POSITA would have reasonably expected success in incorporating Chiueh's functionalities into Auradkar's system. (*See supra* VIII.D.2.f.)

h. 1[c]

Auradkar disclosed uploading and storing the **searchable encrypted file** on one or more cloud platforms (e.g., the **cloud storage** in FIG. 15) based on the policy because Auradkar described that its policy could specify "how data could be stored, **where it could be stored**, how long it could be stored...." (Auradkar, ¶¶96,

130 (“To perform an upload, document 1500 [was] encrypted 1520 and uploaded 1530 to **the cloud** into an encrypted document store 1550.”)), FIG. 15.)

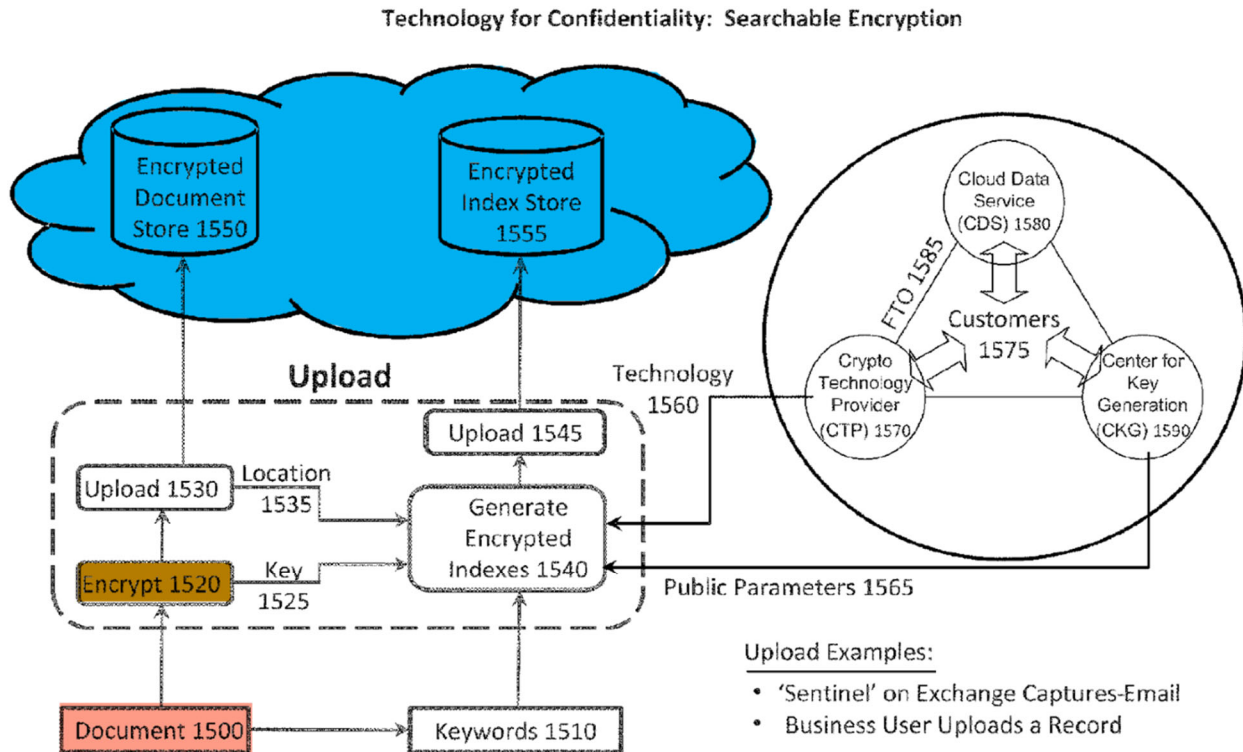


FIG. 15 (Annotated)

Auradkar also disclosed storing the encrypted data by more than one Cloud Services Providers (CSPs), such as CSPs 2872, 2874, 2876, or 2878, shown in Figure 28 (*Id.*, ¶189, FIG. 28.)

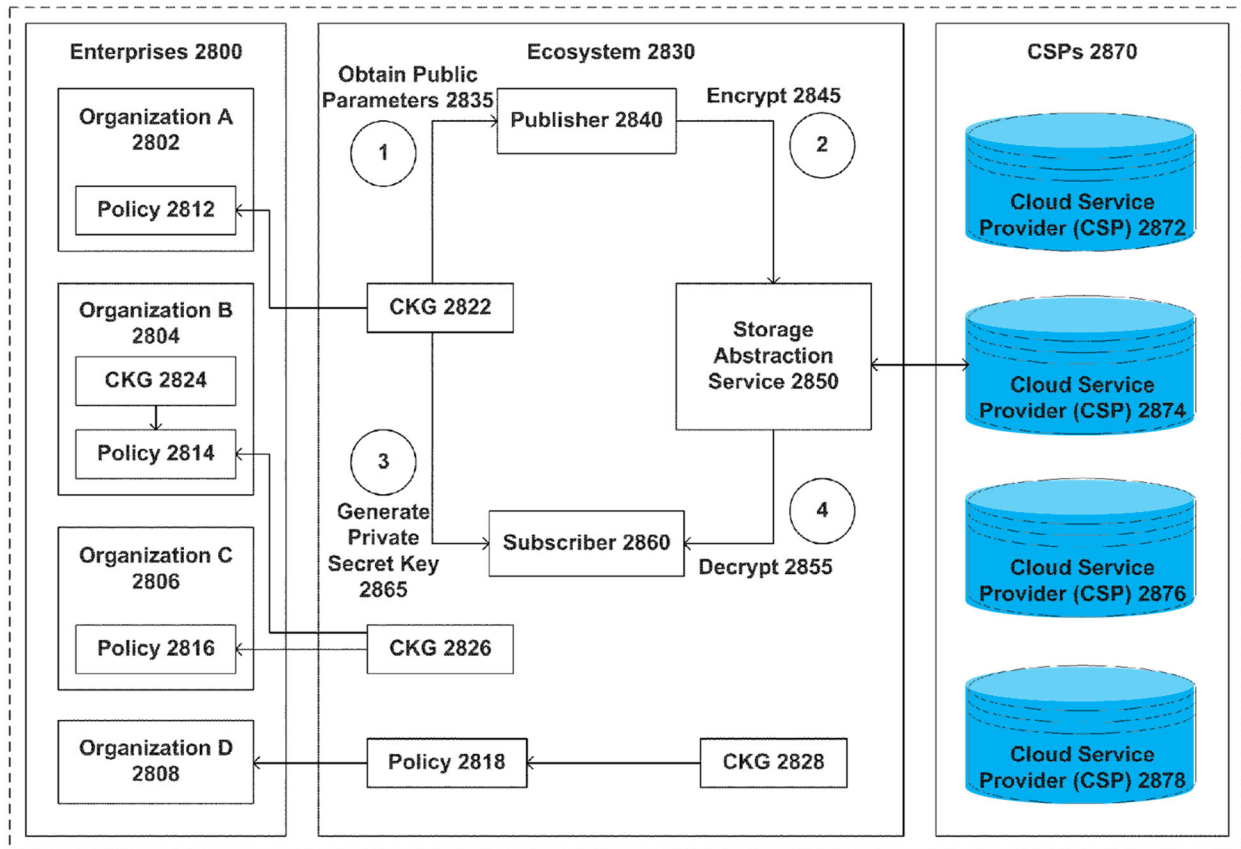


FIG. 28 (Annotated)

As explained above, a POSITA would have found it obvious to use Chambers’ gateway device to perform the functions described in Auradkar, such as storing the searchable encryption file on one or more cloud platforms based on the policy. (*Supra* VIII.D.2.b; Lee, ¶236.)

i. 1[d]

Auradkar disclosed managing access to the searchable encrypted file by the one or more users based on a policy.

Auradkar disclosed that “[w]ith policy and enforcement within the control of users or an enterprise, data and corresponding [were] encrypted and accessible

only with specific capabilities granted to a user.” (Auradkar, ¶112.) For example, “when **users 508** of data 512 attempt to access such data 512, various crypto access controls 514 **guided by...enterprise access policy 530** protect[ed] the data 512 from unwarranted access by users 508.” (Auradkar, ¶117, FIG. 5, ¶113.)

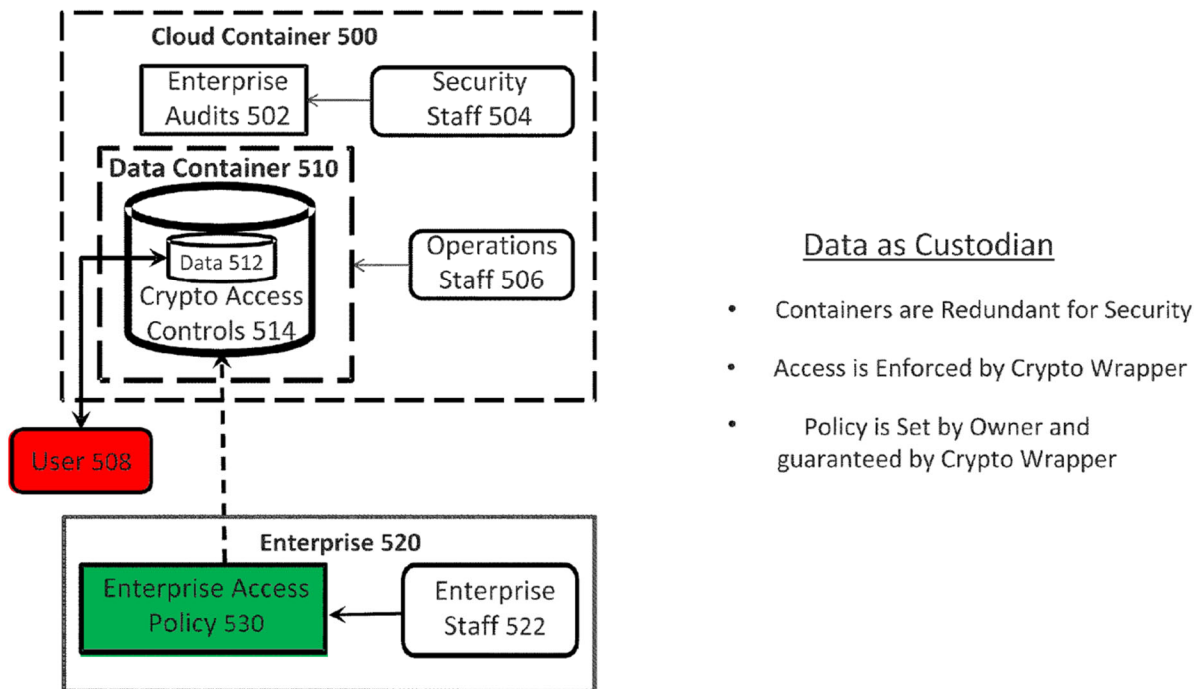


FIG. 5 (Annotated)

3. Claim 16

a. 16[pre]

Auradkar disclosed a secure cloud storage system for encrypting files and storing them in the cloud, as shown in Figures 15, 53, and 54. (See, e.g., Auradkar, Abstract, FIGS. 15, 53-54.)

Technology for Confidentiality: Searchable Encryption

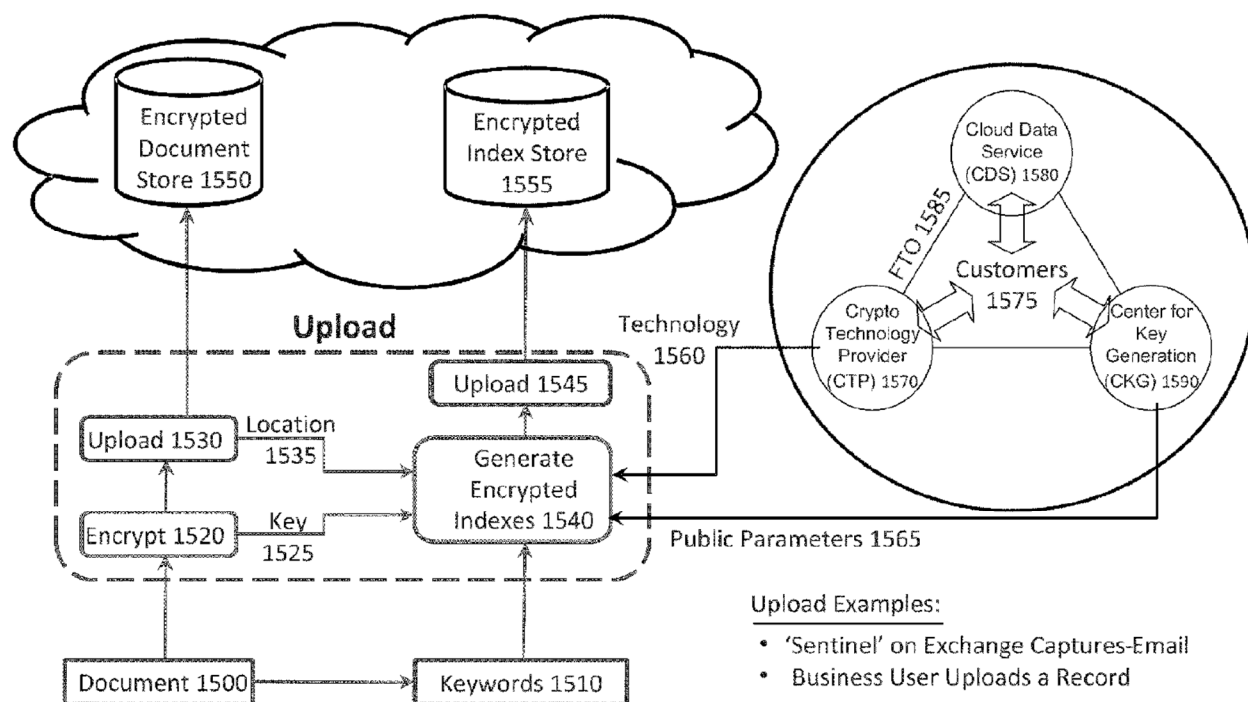


FIG. 15

Auradkar, FIG. 15

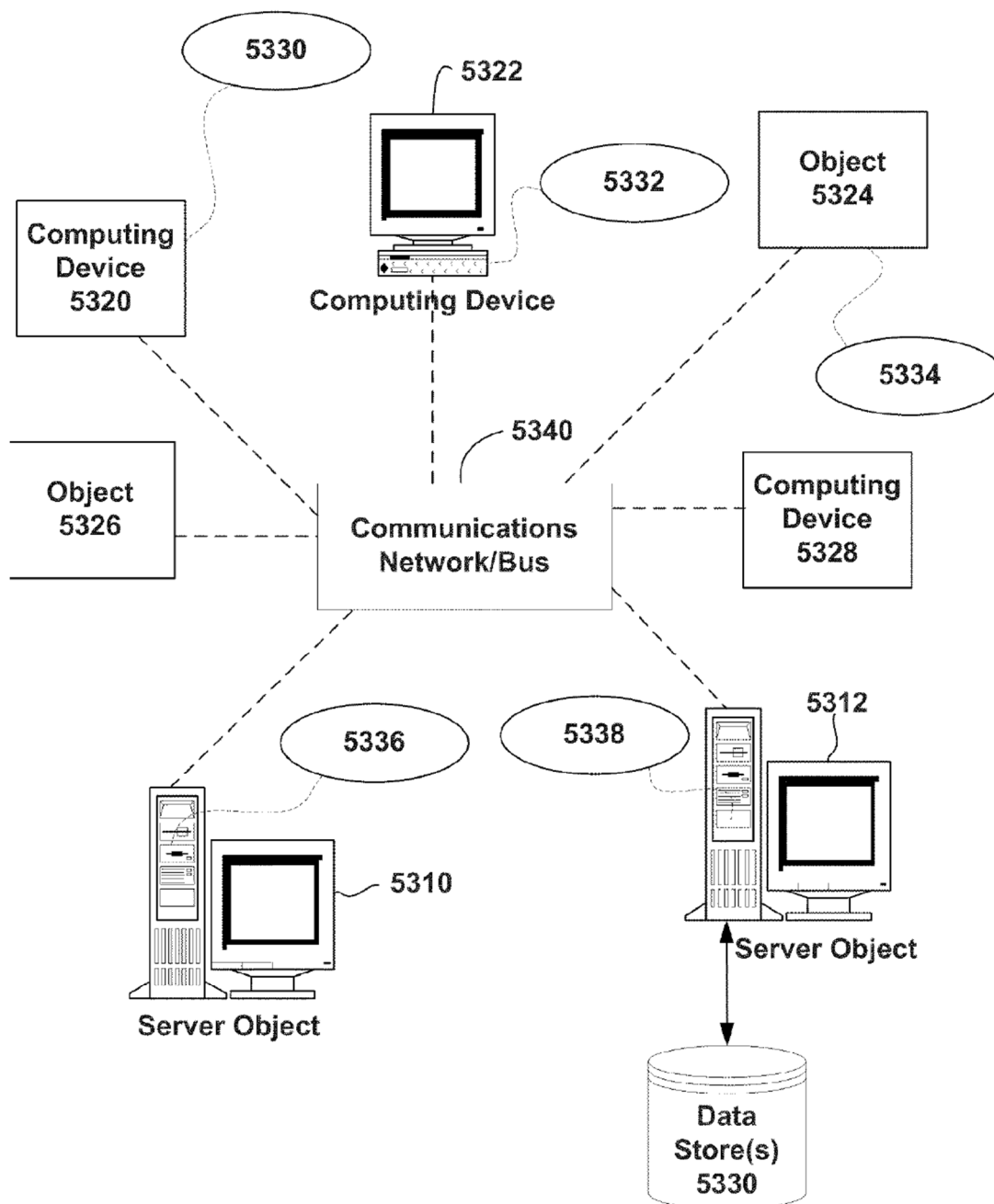


FIG. 53

Auradkar, FIG. 53

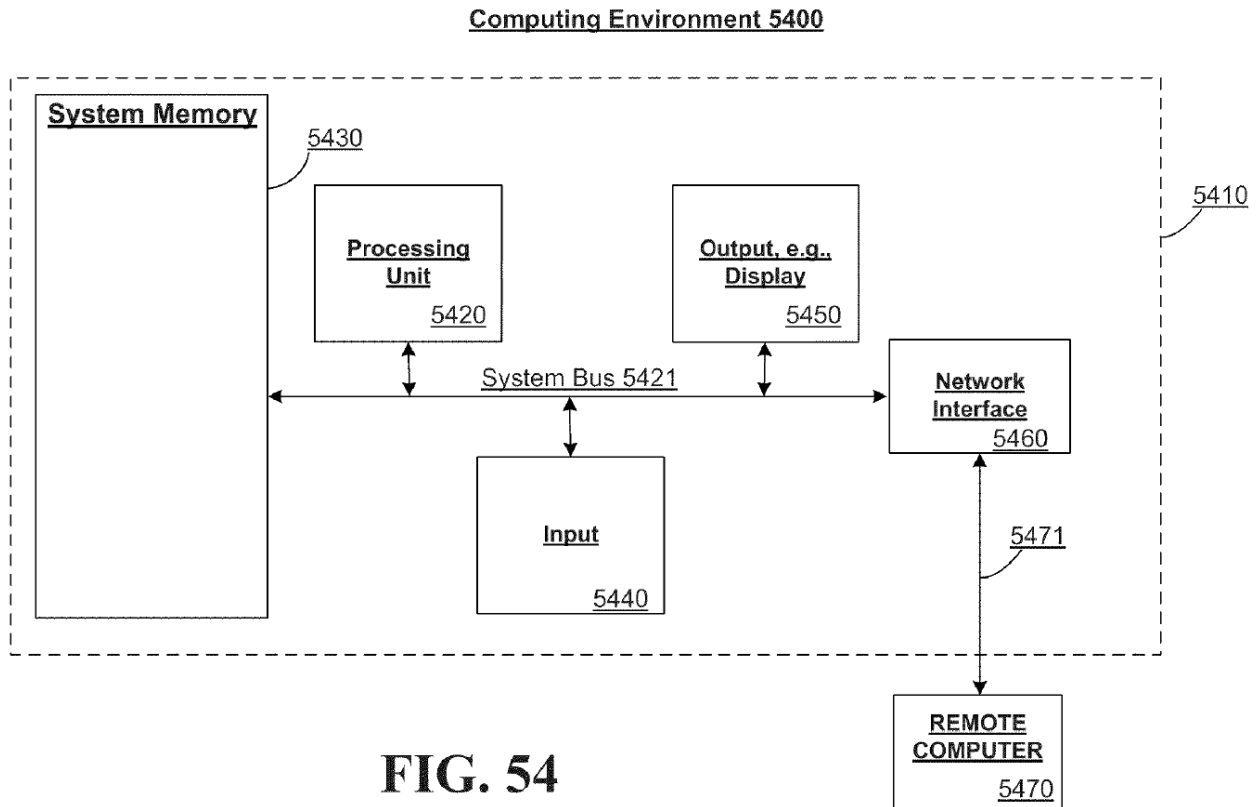


FIG. 54

Auradkar, FIG. 54

b. 16[a]

Auradkar described that its embodiments could be implemented with a computing system 5410 illustrated in FIG. 54, which included a processing unit 5420. (Auradkar, ¶¶256-257, FIG. 54.) A POSITA would have understood that Auradkar's processing unit 5420 constituted a processor. (Lee, ¶243.)

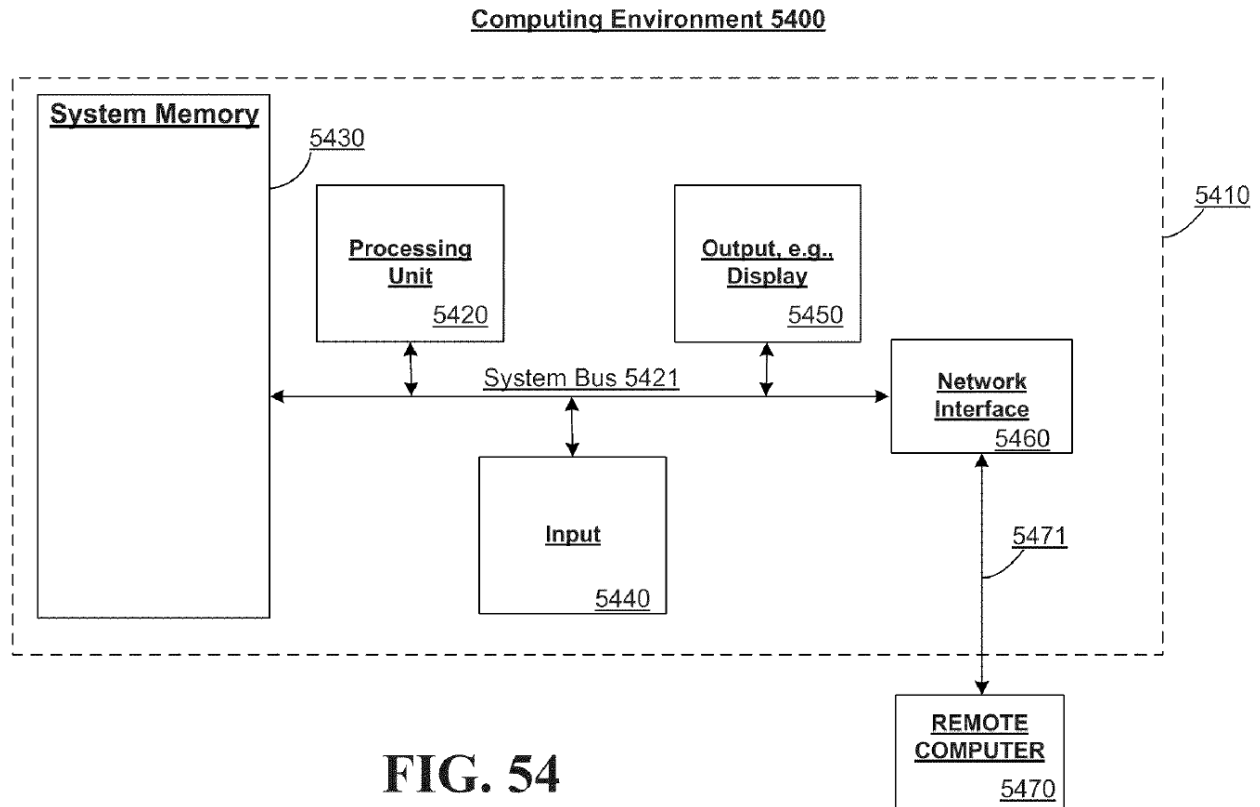


FIG. 54

Auradkar, FIG. 54

c. 16[b]

Auradkar disclosed that its computing system 5410 (system) included a network interface 5460. A POSITA would have understood this described the claimed “communication interface mechanism” because it was used to communicate with remote computer 5470 via network 5471 (e.g., local area network (LAN)). (Auradkar, ¶260, FIG. 54; Lee, ¶245.)

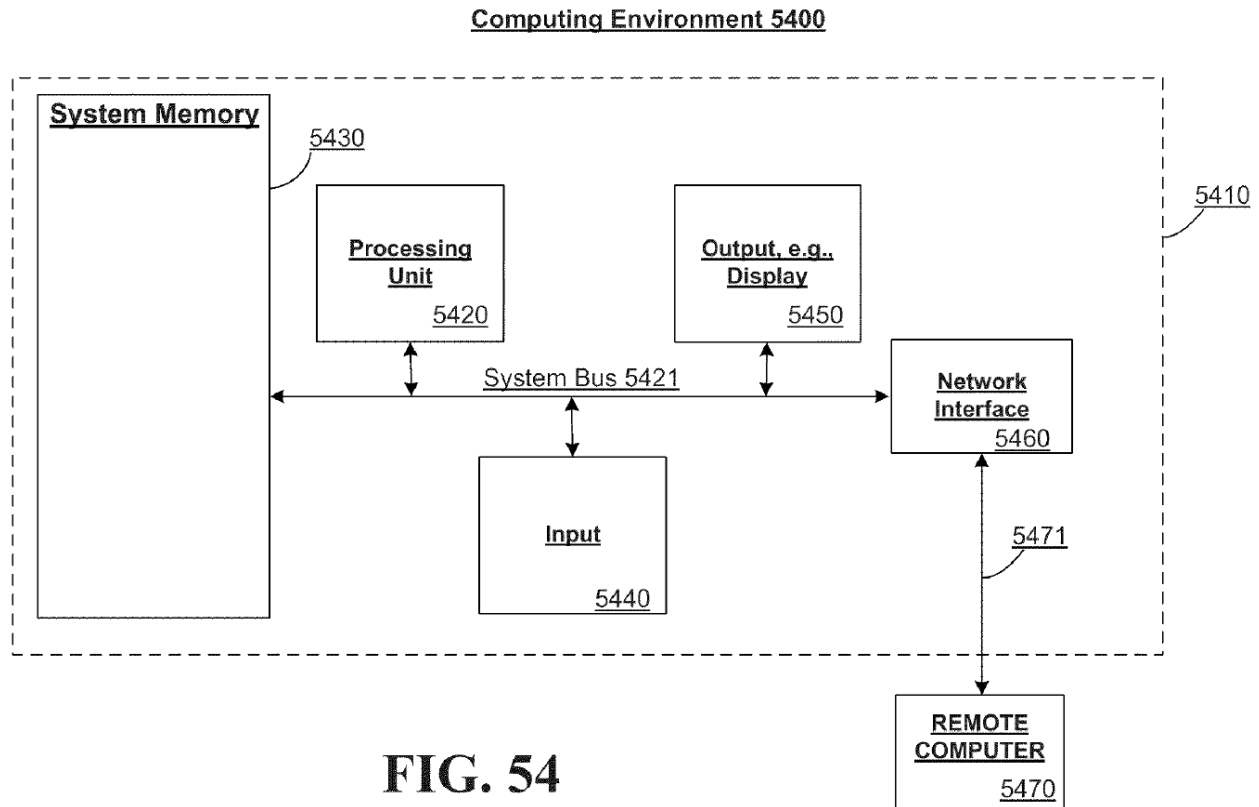


FIG. 54

Auradkar, FIG. 54

d. 16[c]

Auradkar disclosed that its computing system 5410 (system) included system memory 5430 (internal data storage device) coupled “to the processing unit 5420” (processor). (Auradkar, ¶258, FIG. 54.) Auradkar’s system memory 5430 included “computer storage media in the form of volatile and/or nonvolatile memory such as read only memory (ROM) and/or random access memory (RAM)” and stored “application programs, other program modules, and program data.” (*Id.*) A POSITA would have found it obvious that the “application programs” and “programs modules” stored in system memory 5430 could be implemented for

performing Auradkar's functions. (*Id.*, ¶¶255-256, Lee, ¶247.)

e. 16[c][i]

Auradkar and Chambers rendered this limitation obvious. (Lee, ¶¶248-251.)

a policy assignment module [and other claimed “encryption,” “storage,” “management,” and “mediation” modules], which, when executed by the one or more processors:³

Auradkar disclosed that system memory 5430 stored various “application programs” and “programs modules” that could be “executed by one or more computers,” such as computing system 5410, to implement the disclosed embodiments. (Auradkar, ¶¶255, 258, FIG. 54.) A POSITA would have found it obvious that processing unit 5420 of computing system 5410 would have executed corresponding modules to implement these prior art software functions, because executing modules by such a processing unit had been well known. (*See, e.g.*, Auradkar, ¶255; Chambers, ¶¶27, 77-78; Cidon, ¶50; Lee, ¶149.)

Auradkar and Chambers disclosed “implement[ing] a policy selected from a group of policies for a user.” (*See supra* VIII.D.2.b.)

Thus, in the Auradkar-Chambers combination, a POSITA would have added Chambers' access control module and its functionalities into Auradkar's system (as illustrated in modified Auradkar Figure 15 below) to allow Auradkar's system to

³ Claim 16's “module” limitations are addressed together here.

automatically assign and implementing policies (Chambers, ¶¶21, 46, FIG. 4A (illustrating “Group policy” for “staff” and “iPad group”)) to users. (Lee, ¶251.)

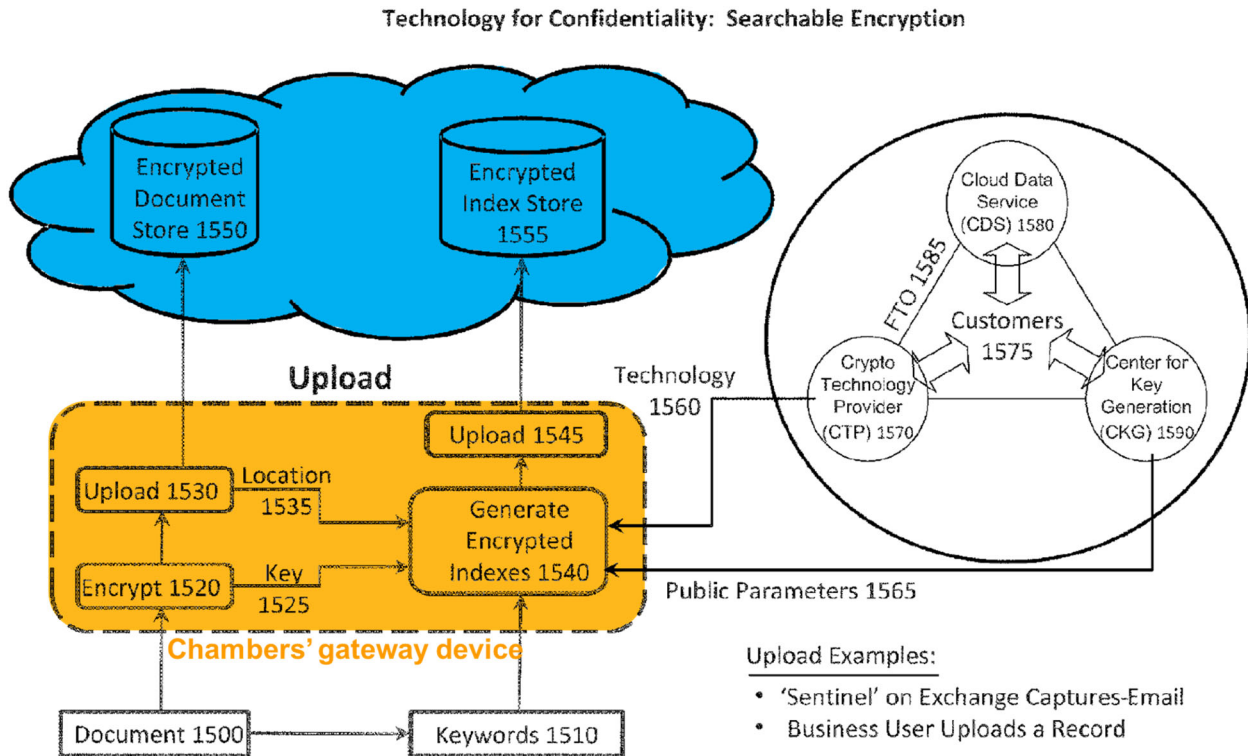


FIG. 15 (Annotated)

A POSITA would have been motivated to make this modification with a reasonable expectation of success. (See *supra* VIII.D.2.b; Lee, ¶252.)

f. 16[c][i][A]

Auradkar’s policy defined and managed access to and processing of a file to be uploaded on a cloud. (See VIII.D.2.b (Ground 4, limitation 1[a][i].))

g. 16[c][i][B]

Auradkar’s policy defined user access rights. (See VIII.D.2.c (Ground 4, limitation 1[a][ii].))

h. 16[c][ii]

A POSITA would have found it obvious that Auradkar’s system would have included various modules, including the claimed “encryption module,” that would be executed by its processing unit (processor). (*See supra* VIII.D.3.e.)

i. 16[c][ii][A]

Auradkar rendered this limitation obvious. (*See supra* VIII.A.2.d.)

j. 16[c][ii][B]

Auradkar disclosed this limitation obvious. (*See supra* VIII.D.2.e.)

k. 16[c][ii][C]

Auradkar or the Auradkar-Chiueh combination suggested this limitation. (*See supra* VIII.D.2.f.)

l. 16[c][ii][D]

Auradkar or the Auradkar-Chiueh combination rendered this limitation obvious. (*See supra* VIII.D.2.g.)

m. 16[c][iii]

A POSITA would have found it obvious that Auradkar would have included various modules, including the claimed “storage module,” when executed by its processing unit (processor) to perform, among others, the claimed “storage” function. (*See supra* VIII.D.3.e, VIII.D.2.h.)

n. 16[c][iv]

A POSITA would have found it obvious that Auradkar would have included

various modules, including the claimed “management module,” when executed by its processing unit (processor) to perform various functions, including the claimed function. (*See supra* VIII.D.3.e, VIII.D.2.i.)

4. Claims 2/17

Auradkar rendered claims 2/17 obvious. (Lee, ¶¶262-265.)

Auradkar disclosed “enterprise 220 ‘owns’ its policy 224 and enforcement 222 of the policy 224 with respect to...external users 230 (e.g., **mobile workers**)” and the policy could “affect how data 512 stored in a data container 510 of a cloud container 500 [could] be **accessed**, manipulated, **retrieved**, searched.” (Auradkar, ¶¶113, 117.)

Auradkar also described subscribers, which could be mobile workers, “**download**[ed]...the data based on the dynamically definable capabilities granted by owner/publisher,” such as enterprise 220. (*Id.*, ¶183; *see also* ¶¶248, 112; Lee, ¶264.)

A POSITA would have found it obvious that mobile workers downloading files in Auradkar’s system would store the encrypted content on their local devices, such as “PDAs, audio/video devices, mobile phones, MP3 players, laptops” associated with the mobile workers. (Auradkar, ¶¶248, 112; Lee, ¶265.)

5. Claim 3

Auradkar rendered this claim obvious because it disclosed mobile workers

could download encrypted files to various computing devices, such as “PDAs, audio/video devices, mobile phones, MP3 players, laptops.” (Auradkar, ¶248, FIG. 53.) (*See supra* VIII.D.5.)

6. Claim 8

Auradkar disclosed claim 8 because its file **encryption** occurred outside of the **cloud**, which did not “[s]ee” encrypted records stored therein. (Auradkar, FIG. 14, ¶¶127-128; Lee, ¶267.)

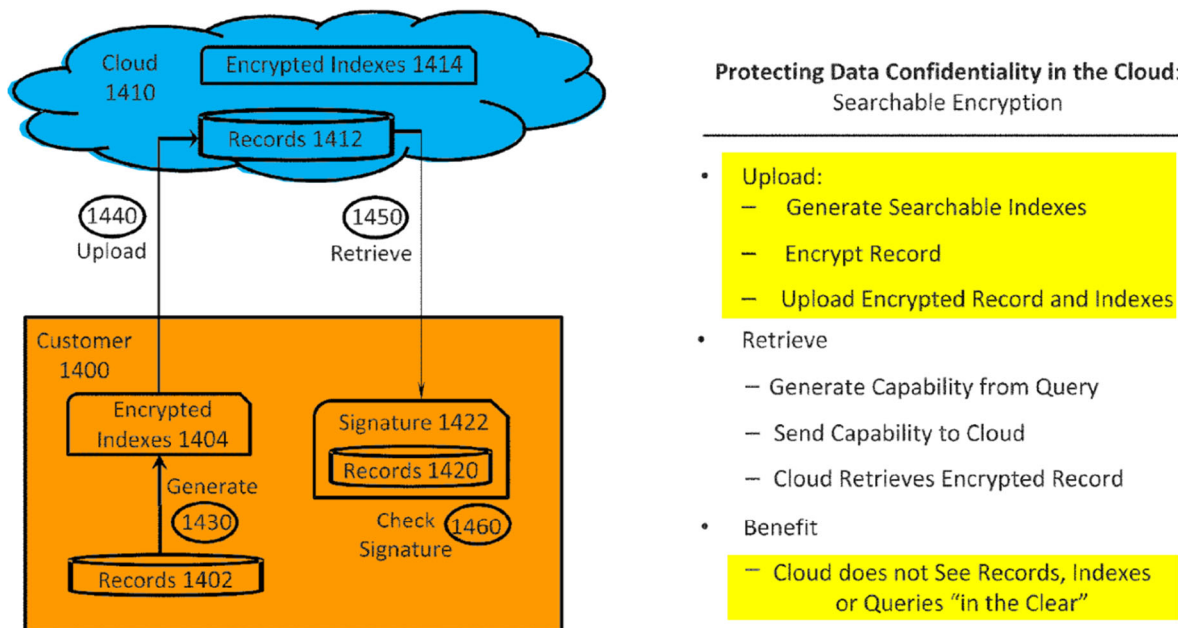


FIG. 14 (Annotated)

Indeed, Auradkar contemplated that a cloud service provider (CSP) where encrypted data were stored could be “another entity that [was] not fully trusted.” (Auradkar, ¶138; *see also id.*, ¶75 (explaining that cloud data service (CDS) could be “provided by any storage provider”).) (Lee, ¶268.)

7. Claims 9/23

Auradkar disclosed claims 9/23 because it described that “enterprise 220 ‘owns’ its policy 224 and enforcement 222 of the policy 224 with respect to **users** 226 and **their** use of system resources of the enterprise 220 as well as with respect to external **users** 230 (e.g., mobile **workers**). (Auradkar, ¶113; *see also id.*, ¶117 (explaining that different access entities had different access policies); Chambers, FIG. 4A (illustrating “Group policy” for “staff” and “iPad group”); Lee, ¶269.)

8. Claims 10/24

Auradkar disclosed claims 10/24 because its policy could specify “how data could be stored, where it could be stored, how long it could be stored” (Auradkar, ¶96), how a file was encrypted (processed) (e.g., specifying “medical data must be backed up once a day using AES 256 encryption; *id.*, ¶96), and how a file was accessed (e.g., “Top Secret content cannot be accessed from a wireless or remote device”; *id.*, ¶160. (Lee, ¶270.)

9. Claims 11/25

Auradkar disclosed claims 11/25 because it disclosed “storing the encrypted data by one **or more** **CSPs 2870**, such as CSPs 2872, 2874, 2876, or 2878.” (Auradkar, ¶189, FIG. 28; Lee, ¶¶271-276.)

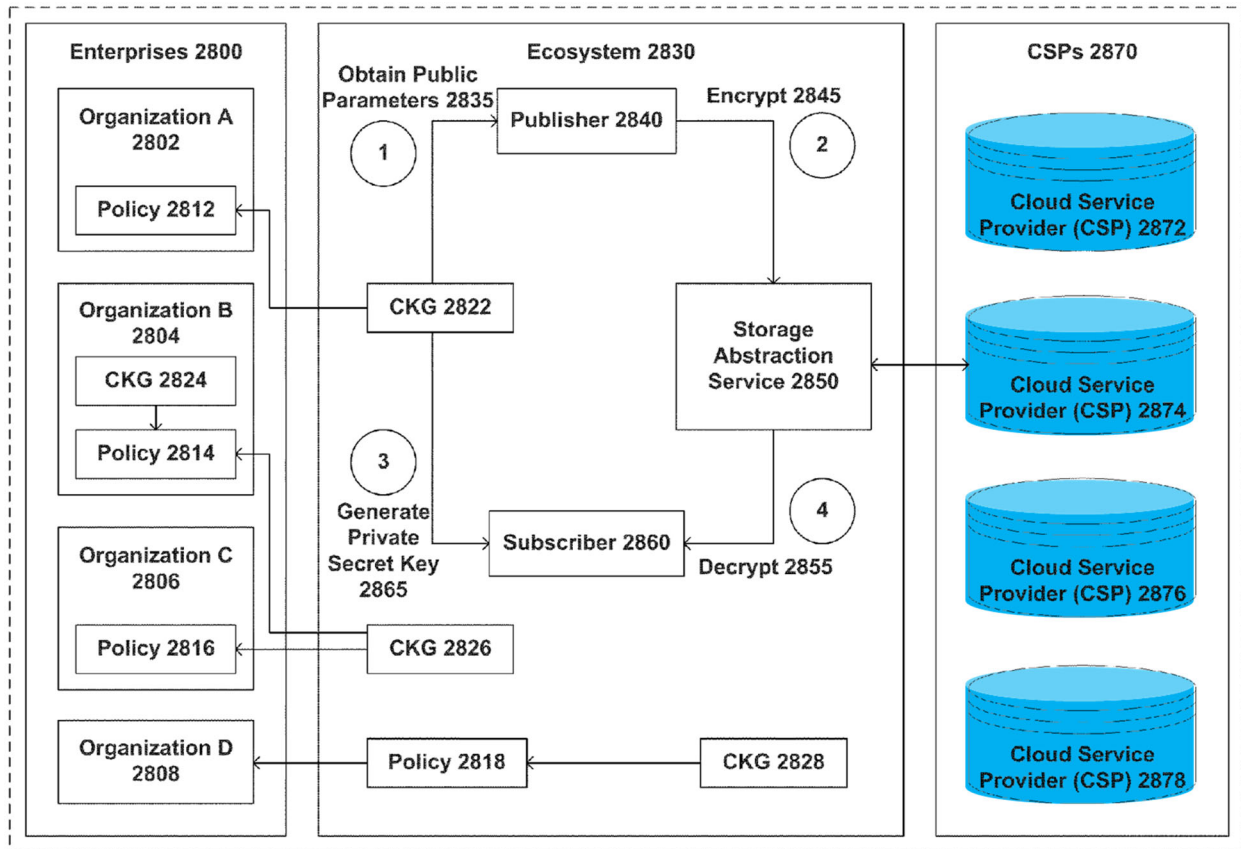
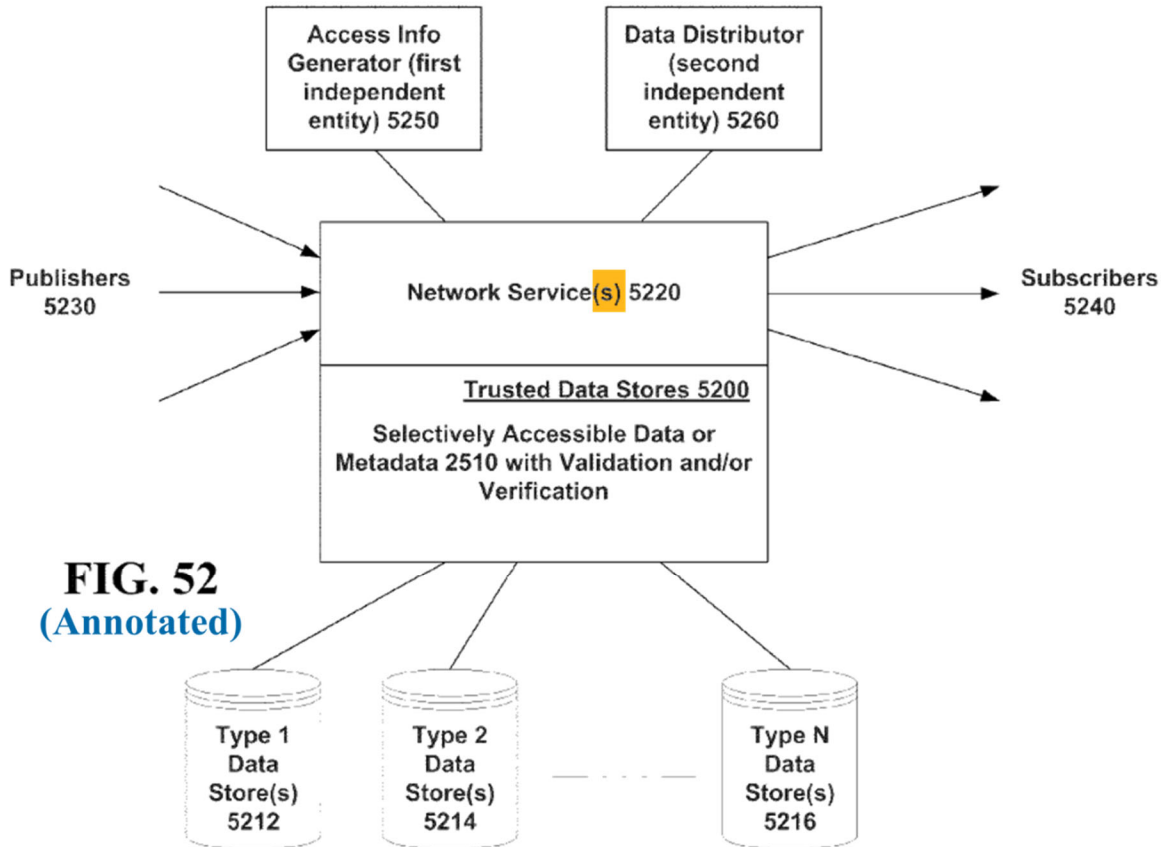


FIG. 28 (Annotated)

(See also *id.*, ¶230 (describing data (which could be an encrypted file) were “automatically...*distributed across a set of data stores*, which [could] be of the same type, or as shown in FIG. 52, containers of different types 5212, 5214,...5216”), FIG. 52, ¶9 (explaining that cloud data servers were “provided in a way that distributes trust across multiple entities to avoid a single point of data compromise”).)



10. Claim 12

Auradkar disclosed claim 12 because Auradkar’s system could upload and store searchable encrypted data to a cloud data service (CDS) that was “provided by **any** storage provider” (Auradkar, ¶75), “thereby avoiding vendor lock-in” (’678 Patent, 13:54-59). (Lee, ¶273.)

For example, Auradkar described a **storage abstraction service** (SAS), separate from **CSPs** 2870, stored encrypted data to these CSPs, “hid[ing] storage semantics in connection with storing the encrypted data by” CSPs 2870. (Auradkar, ¶189, FIG. 28.)

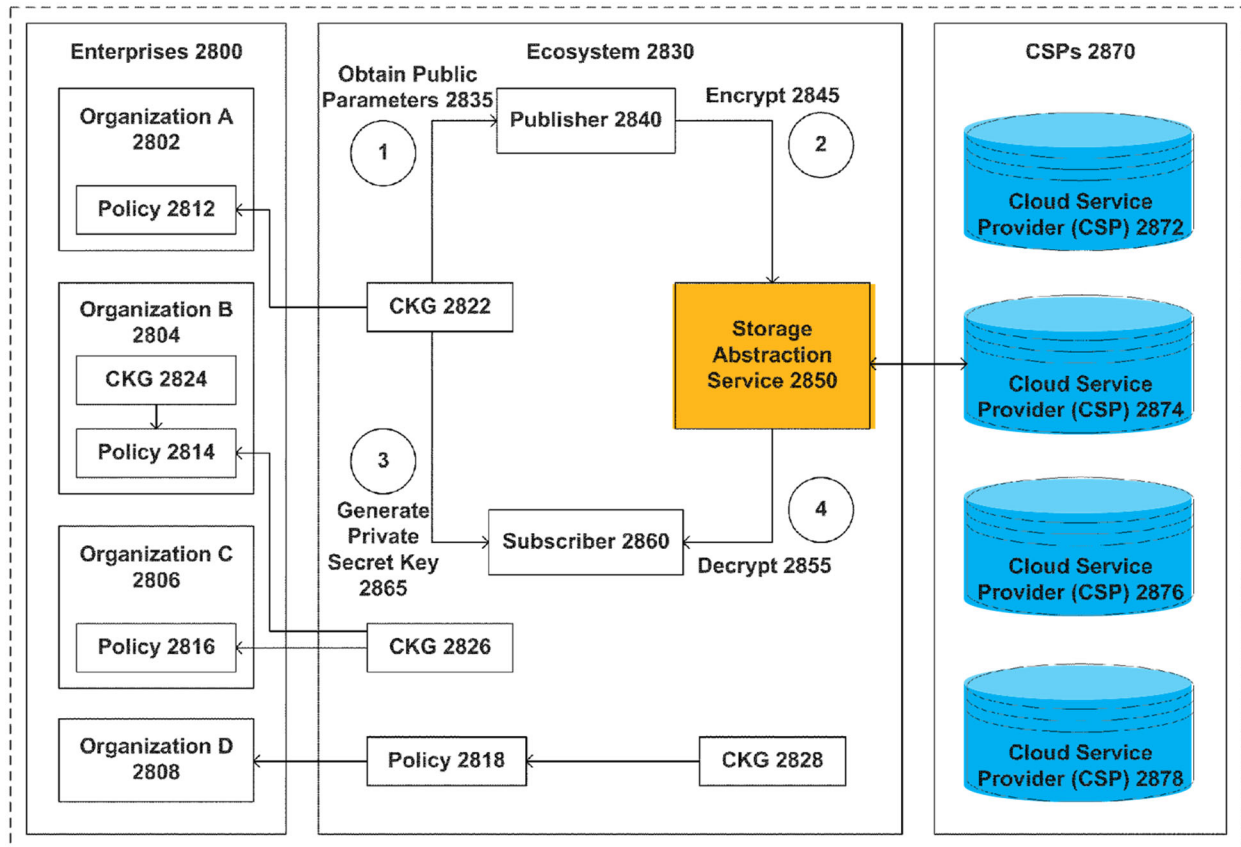


FIG. 28 (Annotated)

Auradkar explained that its SAS used storage transfer format (STF), “a standard interchange format that [could] be used for exchanging encrypted data and metadata **across repositories**,” such as private cloud store 3000, SQL data store 3002, simple storage web service store 3004. (Auradkar, ¶193, FIG. 30.)

STF is a Standard Interchange Format for Exchanging Encrypted Data and Metadata Across Repositories

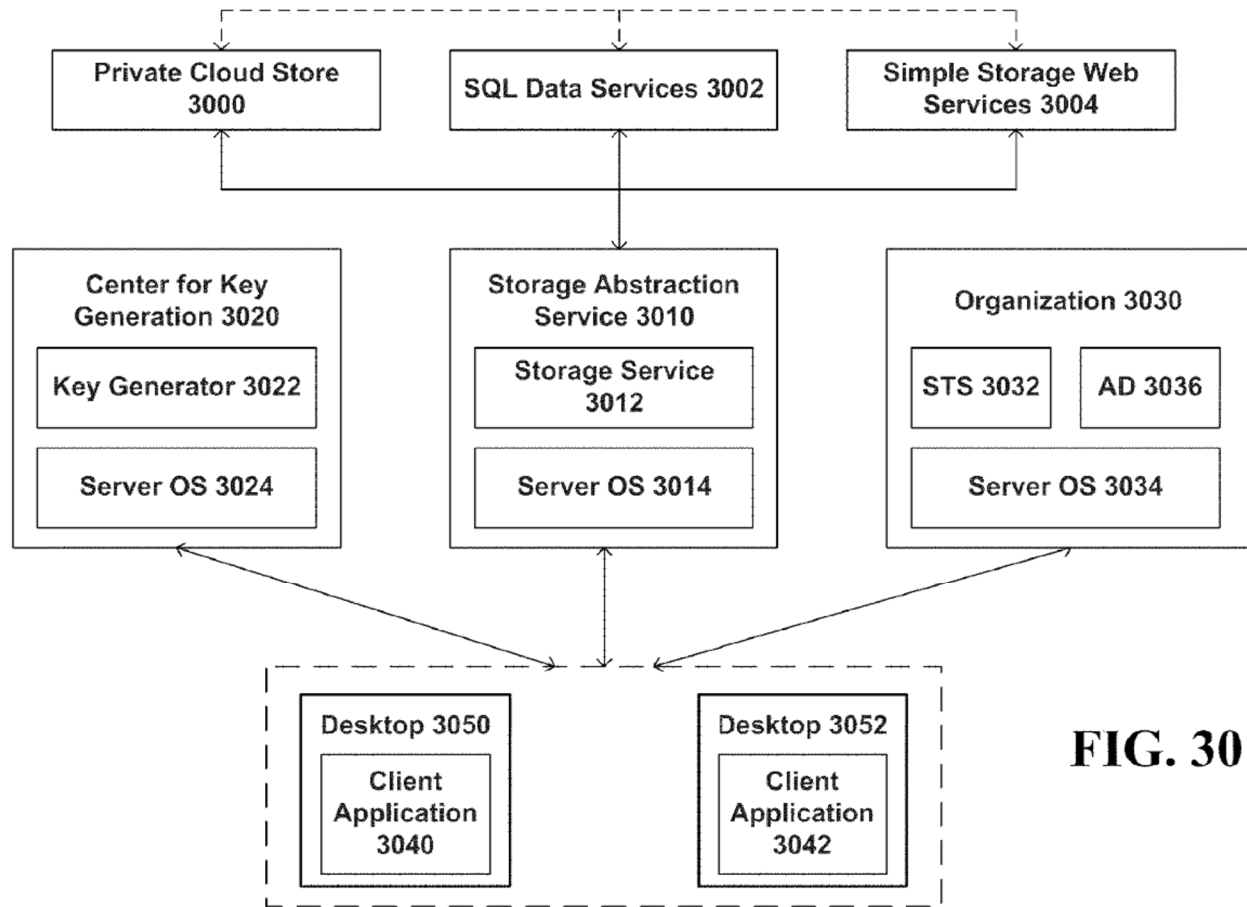


FIG. 30

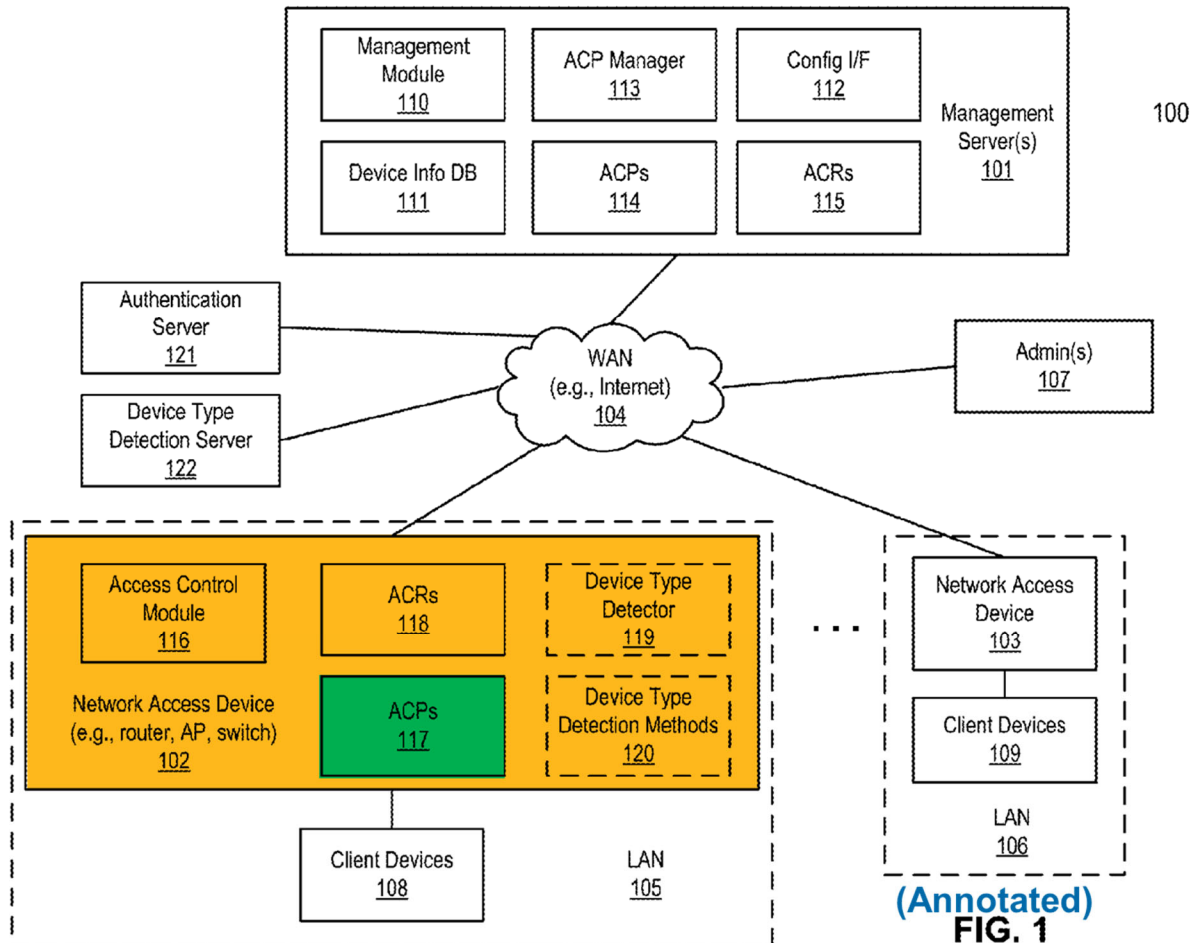
Auradkar, FIG. 30

Thus, based on Auradkar’s teachings, a POSITA would have understood that Auradkar’s system stored encrypted data independent of the cloud service provider(s), because it used a standard format for storing data to multiple CSPs, which could be a cloud storage provided by “**any** storage provider” or “**another** entity that [was] **not fully trusted.**” (Auradkar, ¶¶75, 138; Lee, ¶276.)

11. Claims 13/26

Auradkar and Chambers disclosed claims 13/26. (Lee, ¶¶277-280.)

Chambers disclosed that a NAD 102 could receive access control policies (ACPs) 114 from management server 101, which operated as a policy database storing policies for NAD 102. (Chambers, ¶27; Lee, ¶278.) Chambers also disclosed that its network access device (NAD; gateway device) included an access control module (ACM) 116 that “select[ed] and applie[d] appropriate...[access control policies ([ACPs])] 117” to a client device. (Chambers, ¶¶46, 21, FIG. 1; *supra* VIII.D.3.e.)



A POSITA would have found it obvious that Chambers’ ACPs 117 (policies) could have been stored in a database operatively coupled to the NAD

(gateway), because storing ACPs in a database would have allowed easier retrieval of a policy and maintenance of data integrity, plus Chambers already contemplated storing ACPs in a database (Chambers, claim 1). (Lee, ¶279.)

A POSITA would have been motivated to incorporate Chambers’ policy database, which was coupled with the NAD (gateway), for storing policies and functionality for selecting a policy from policies into Auradkar’s system, because the modified system would have allowed selecting and assigning a policy from existing policies, instead of generating a policy ad-hoc for a user, thereby improving efficiency for assigning and implementing policies. (Chambers, ¶¶21, 46; Lee, ¶280.) Additionally, a POSITA would have been motivated to add these features to Auradkar’s system to reap the aforementioned benefits of storing policies in a database. (Lee, ¶280.) A POSITA would have reasonably expected success in combining Chambers’ gateway and functionalities with Auradkar’s system. (*See supra* VIII.D.2.b.)

12. Claim 14/27

Auradkar rendered obvious claims 14/27. (Lee, ¶¶281-284.)

Auradkar disclosed that its policy could define and manage “where it could be stored” (one or more authorized locations to which the file may be stored), a type of encryption (e.g., AES) and the 256-bit encryption key. (Auradkar, ¶96; *see supra* VIII.D.2.))

Auradkar also disclosed generating encrypted indexes 1540 to be used in searches (search indices) using an encryption key 1525 and decrypting the encrypted document using a decryption key 1644. (Auradkar, ¶¶130, 133, FIGS. 15-16.)

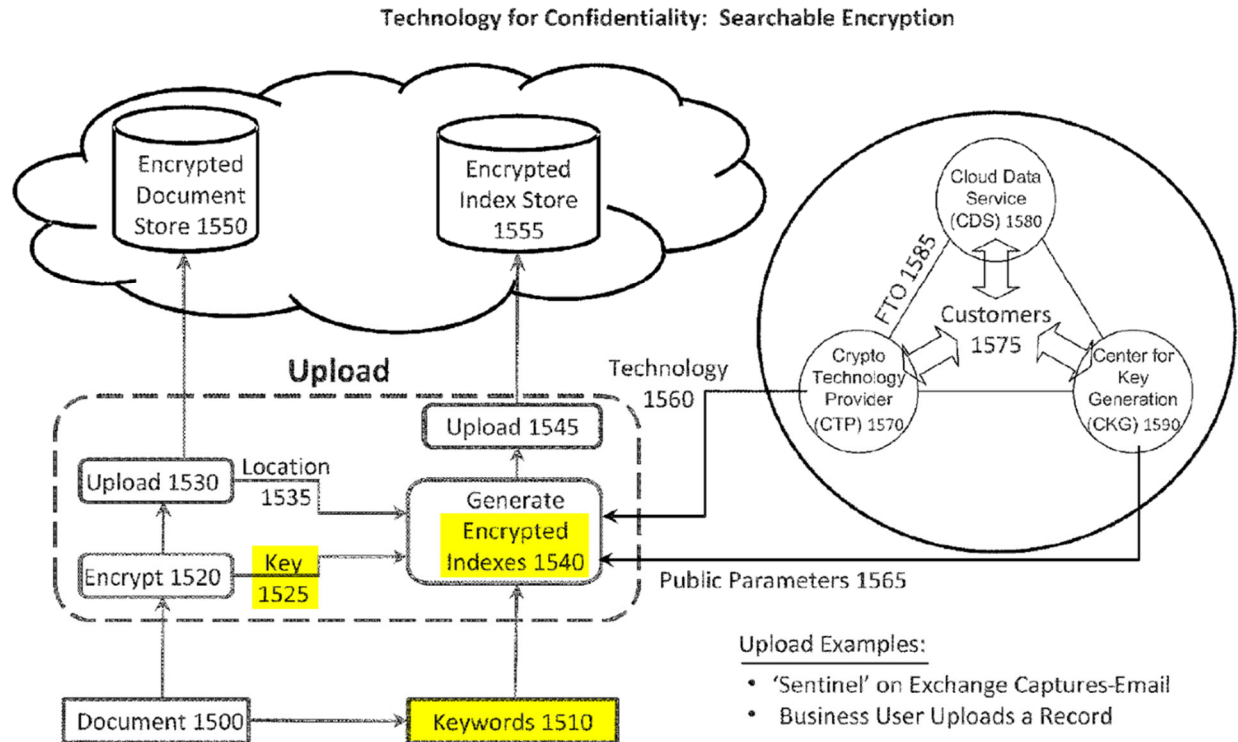


FIG. 15 (Annotated)

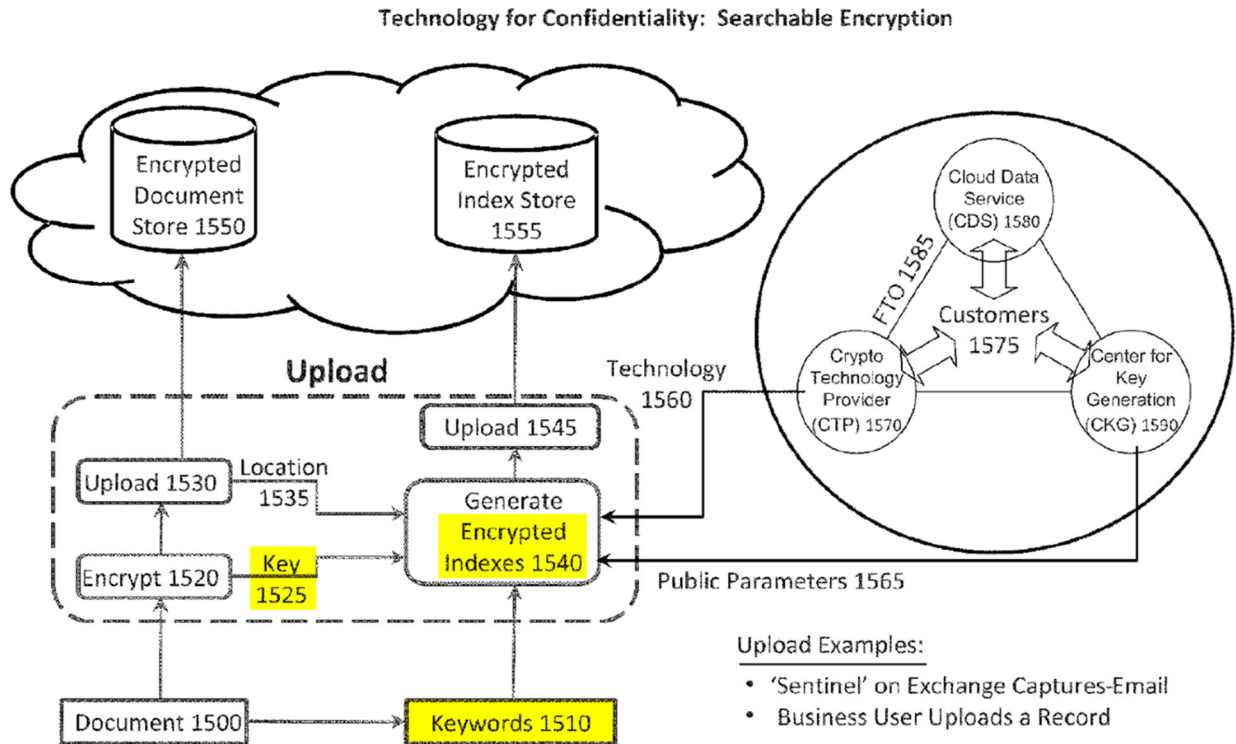


FIG. 15 (Annotated)

A POSITA would have found it obvious that Auradkar’s policy for a user would have also defined and managed encrypted indexes 1540 (e.g., the key for encrypting plaintext keywords to obtain encrypted indexes) and the encryption/decryption keys 1525/1644 to ensure encryption, decryption, and search operations consistent with the policy. (Lee, ¶284.) For example, Auradkar explained that the policy could require “sensitive[] medical data” to be encrypted using AES 256 encryption, and a POSITA would have found it obvious the policy could also have defined and managed the key for encrypting indexes, encryption key 1525, and decryption key 1644 such that they would also comply with those requirements. (Auradkar, ¶96; Lee, ¶284.)

13. Claim 15

Auradkar disclosed claim 15.

Auradkar disclosed encrypting a tag 1200 that included the filename of the payload (FName: fox.tx) and keywords (“fox,” “dog”)⁴ into an encrypted tag 1210, which included an **encrypted filename and keywords**. (Auradkar, ¶125, 150, FIG. 12.)

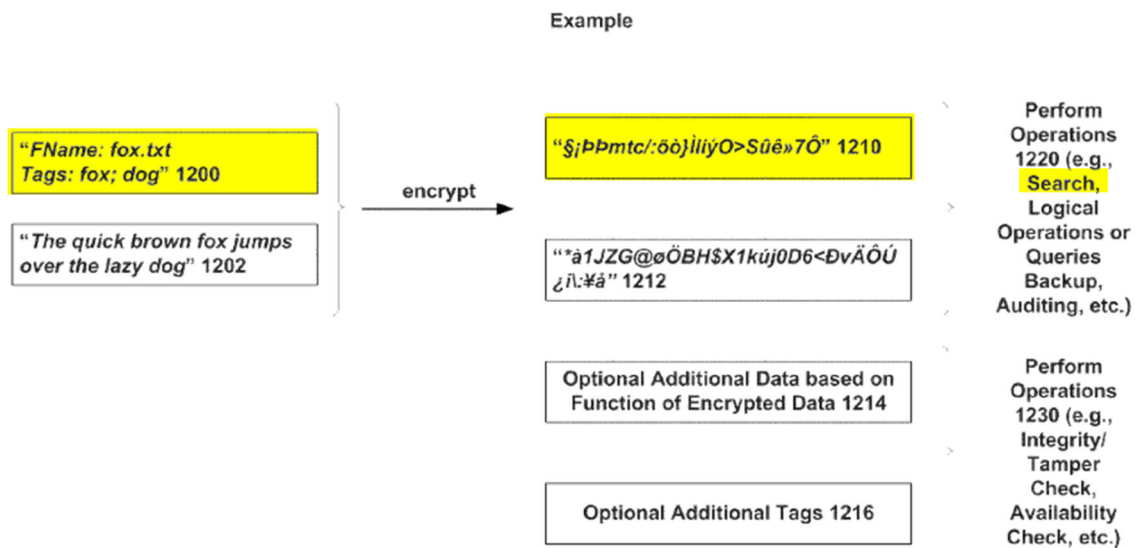


FIG. 12 (Annotated)

Auradkar also disclosed encrypting search indexes. For example, Auradkar disclosed encrypting keywords 1510 associated with document 1500 to generate encrypted indexes 1540, which were used for searching the encrypted document. (*Id.*, ¶130, FIG. 15.)

⁴ The '678 Patent also refers “indices” as “keywords.” ('678 Patent, 12:7.)

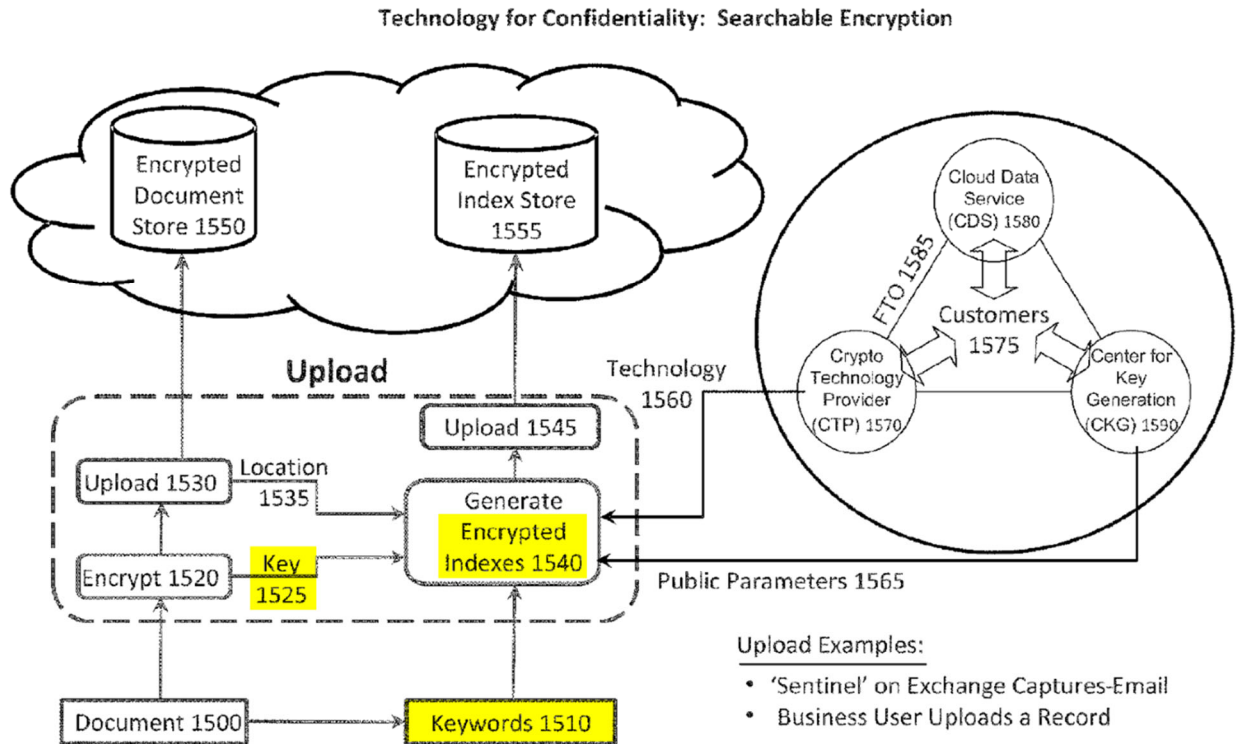


FIG. 15 (Annotated)

14. Claim 21

As explained in Ground 1, claim 16, from which claim 21 depends, does not require (or prohibit) that the recited functions or modules be performed by a single gateway device. (*See supra* VIII.A.2-VIII.A.3.) But claim 21 requires that various recited modules (or functions) are implemented on different devices. (*See supra* VIII.A.13.) Auradkar rendered claim 21 obvious. (Lee, ¶288.)

Software Implementation of Modules: As explained above, a POSITA would have found it obvious that Auradkar's system would have included various software programs or modules, including the claimed modules for performing corresponding functions. (*See supra* VIII.D.3.e, VIII.D.3.h, VIII.D.3.m-

VIII.D.3.n.) Auradkar also contemplated that “various embodiments...described herein [could] be implemented in connection with any computer or other client or server device, which [could] be deployed...in a distributed computing environment,” such as the one illustrated in Figure 53, including various computing devices and servers. (Auradkar, ¶¶247-248, FIG. 53, ¶¶246, 252.)

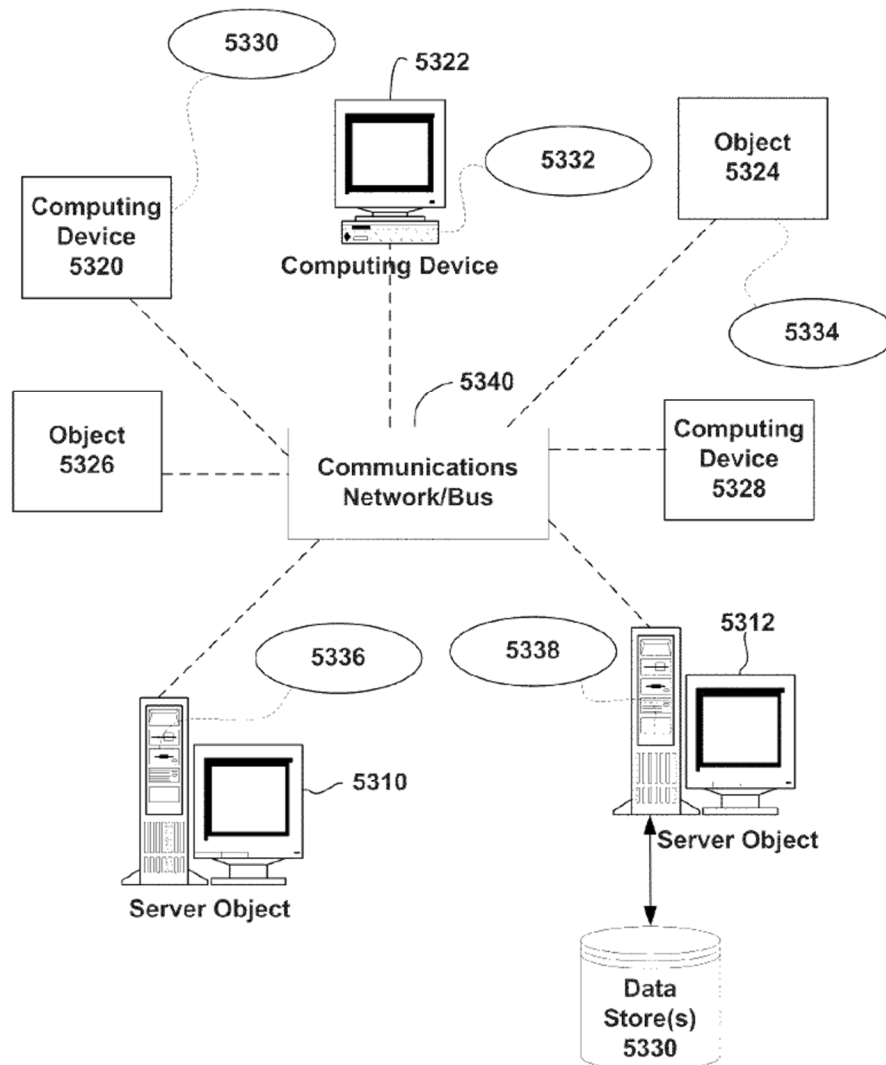


FIG. 53

Implementation using multiple devices: Chambers also contemplated

having multiple devices, such as “a gateway device...an access point (AP)...[and] a network switch,” as part of its network access device (NAD) to perform the functions described therein. (Chambers, ¶81.)

It was also well-known that implementing functions with a distributed computing environment would increase processing power and efficiency and improve task load balance. (Lee, ¶292) Therefore, a POSITA would have been motivated to implement these disclosed programs or modules on different devices, such as Auradkar’s servers for access management and Chambers’ network access device (gateway; *see supra* VIII.D.2.b) and network switch (Chambers, ¶81; network controller) to reap these benefits. (Lee, ¶292.)

A POSITA would have reasonably expected success in implementing these modules in the aforementioned devices because these references already contemplated that the functions could be implemented in a distributed computing environment, and the modification would have yielded expected results. (Lee, ¶293.)

15. Claim 22

Auradkar rendered claim 22 obvious because it described a software abstraction service (mediation module) that made the operations for generating a searchable encrypted content independent of the cloud storage service. (Lee, ¶294.)

As explained above, Auradkar disclosed that storing the searchable encrypted file on the cloud platform was performed independent of the cloud platform service provider. (*See supra* VIII.D.10.) For example, Auradkar disclosed a storage abstraction service (SAS) that used storage transfer format (STF), “a standard interchange format that [could] be used for exchanging encrypted data and metadata **across repositories**,” such as private cloud store 3000, SQL data store 3002, simple storage web service store 3004. (Auradkar, ¶193, FIG. 30; *see also id.*, ¶¶189, 191, FIGS. 28-29.)

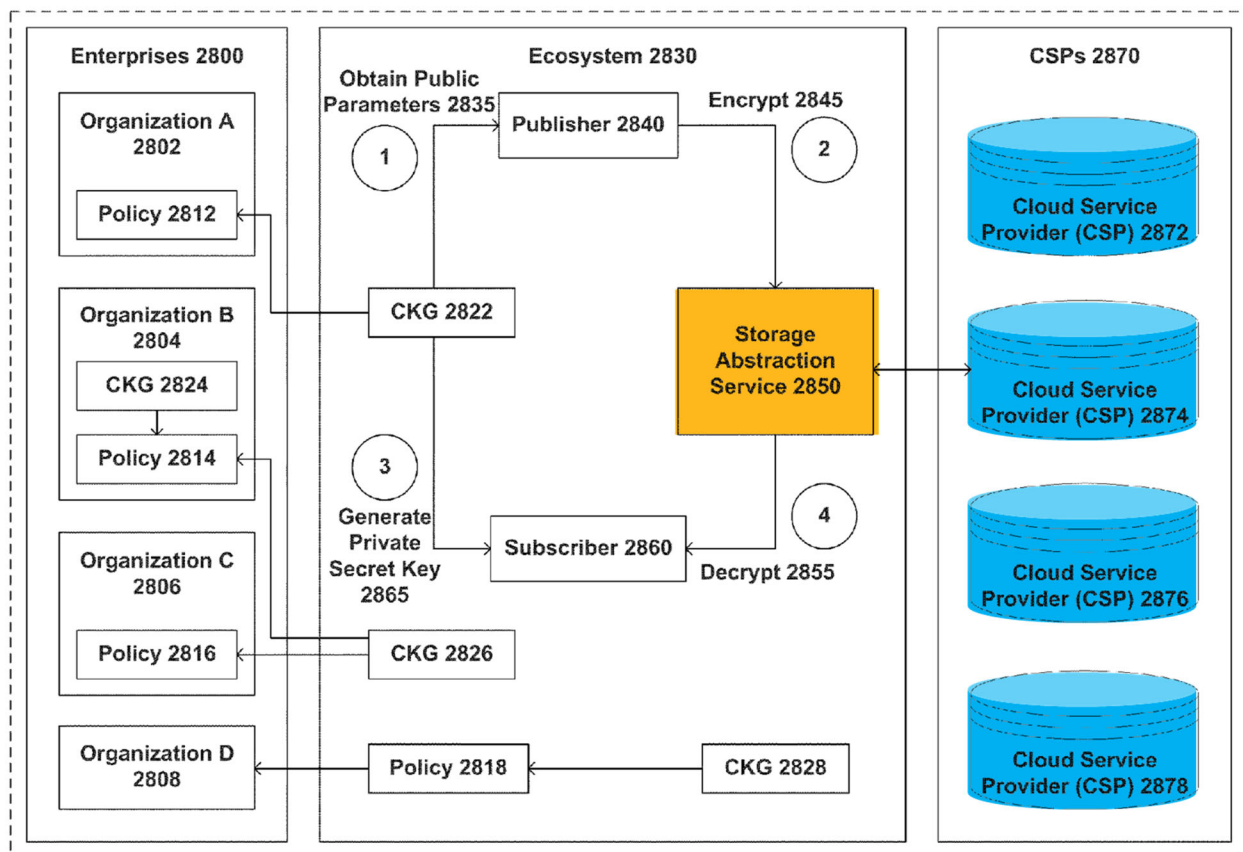


FIG. 28 (Annotated)

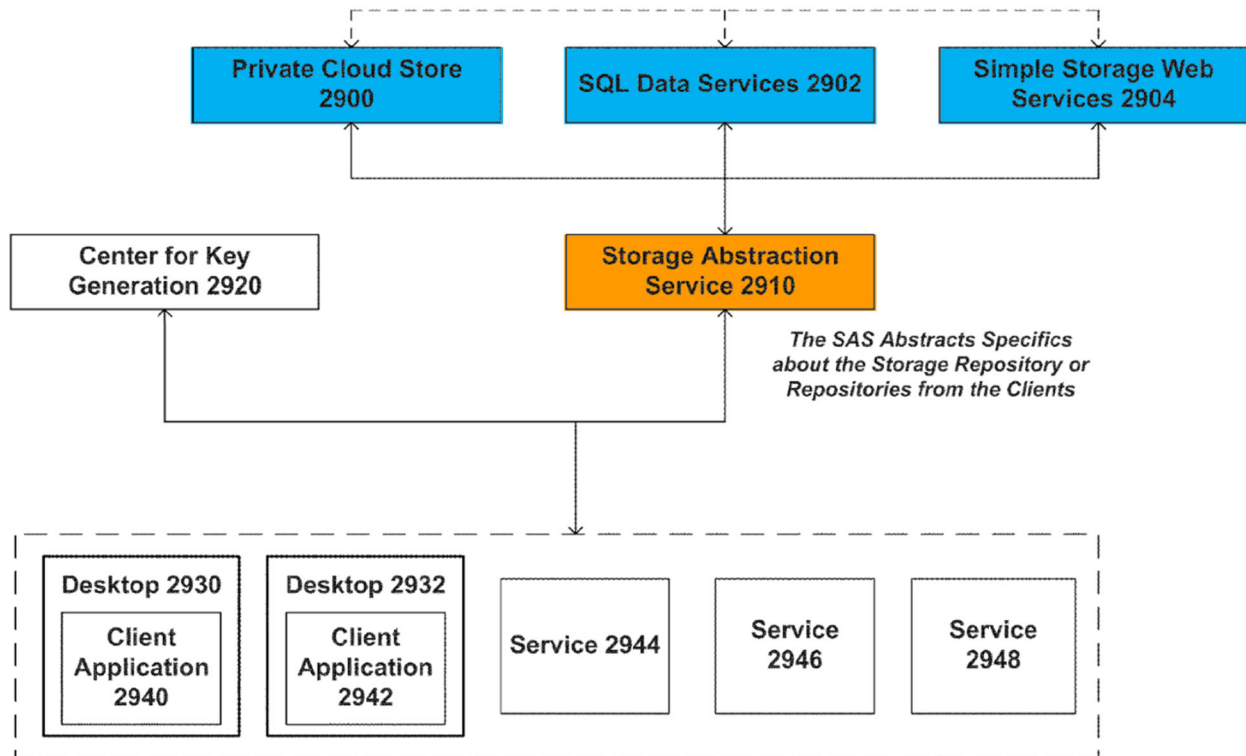


FIG. 29 (Annotated)

A POSITA would have found it obvious that Auradkar’s SAS constituted the claimed mediation module, because it “facilitat[ed] geographical and provider diversity in data placement, *making the system agnostic with respect to specific cloud vendor APIs*” (’678 Patent, 4:46-49.) by, e.g., using STF for exchanging encrypted data across different cloud storages. (Auradkar, ¶193; Lee, ¶296.)

A POSITA would have further understood Auradkar’s SAS’s role as the claimed mediation entity because Auradkar also described that its system was designed to use a cloud service provider which may “**not fully trusted**” (Auradkar, ¶¶75, 139, 127-128, FIG. 14) and its federal trust overlay (FTO) services (e.g., encryption) were “independent of the data services in the cloud...”

and] **operated by parties other than the operator of the data services.”**

(Auradkar, ¶¶237, 75.)

From these teachings, a POSITA would have found it obvious that Auradkar’s functionalities for policy assignment and management, encryption, and storage would have been carried out independent of the cloud storage service provider. Notably Auradkar contemplated that encrypted data could be stored to a cloud hosted by an untrusted entity, which should not be given access to any of these functionalities. (Auradkar, ¶¶237, ¶78 (describing that “enterprises [could] own policy and control enforcement in a granular manner, even if data and application [were] hosted in the cloud”); Lee, ¶298.)

E. Ground 5: Auradkar, Chiueh, Chambers, and Inoue render Claim 6 obvious.

1. Claim 6: “wherein the step of configuring the namespaces comprises storing content of the file as filenames.”

Auradkar and Inoue rendered claim 6 obvious. (Lee, ¶¶299-306.) The ’678 Patent describes storing keywords from file content, which “would typically be used as search terms in the form of search queries,” as filenames. (’678 Patent, 12:7-9.)

As explained, Auradkar disclosed generating and encrypting keywords (file content) for searching encrypted documents that include these keywords. (*See supra* VIII.D.2.d.) For example, Auradkar described encrypting a “document 1500

with which various **keywords 1510** [were] associated” and uploading the encrypted document to “the cloud into an encrypted document store 1550.” (Auradkar, ¶130, FIG. 15.)

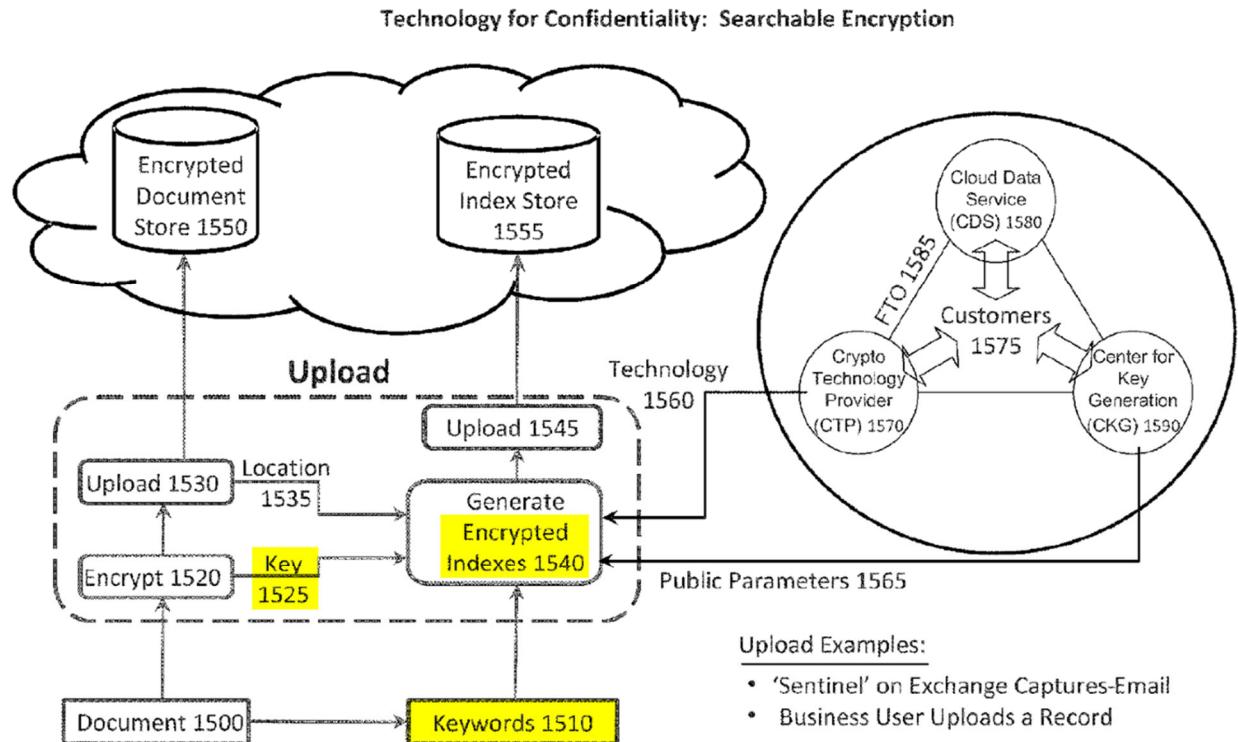
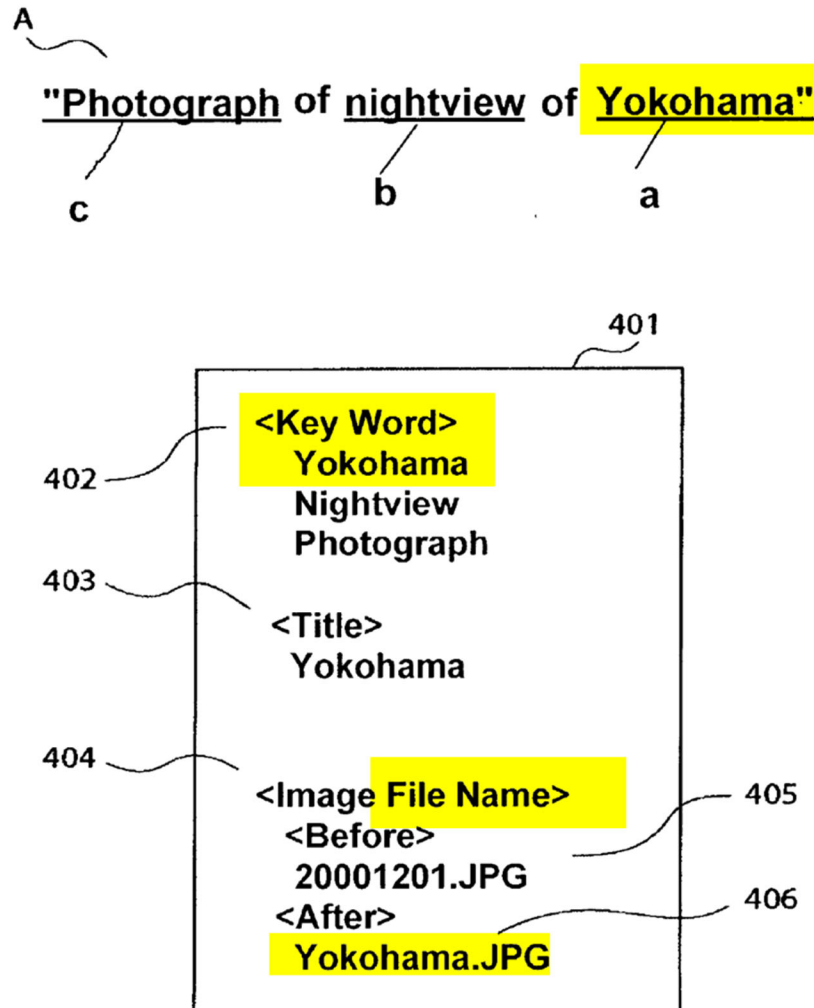


FIG. 15 (Annotated)

While Auradkar did not explicitly disclose storing keywords as filenames, Inoue did. (Lee, ¶302.) Inoue disclosed “correlate[ing] one or more extracted keywords (character string data) with the image data as keywords for image data searches” (content of the file) and “set[ting] one of the keywords as the title (a filename) of the image data.” (Inoue, ¶104, FIG. 4.)

Fig. 4
(Annotated)



A POSITA would have been motivated to incorporate Inoue's functionality of assigning a keyword extracted from file content (e.g., Auradkar's keywords 1510) into the Auradkar-Chiueh-Chambers system to produce filenames that also included a keyword. In doing so, in addition to Auradkar-Chiueh method of searching indexes (*see supra* VIII.D.2.g), the modified system would also have allowed searching files via their filenames which is faster and more efficient. (Lee,

¶304.)

A POSITA would have found further motivation to combine because Inoue—like Auradkar and Chiueh—concerned facilitating content searching based on keywords. (Auradkar, ¶130; Inoue, Abstract; Chiueh, ¶40.) And, a POSITA would have understood that those references sought to address similar problems for searching files and therefore would benefit from the complementary techniques of Inoue to facilitate and expand the search options. (Lee, ¶305.)

A POSITA would have also reasonably expected success in making the combination because assigning keywords to filenames had been well-known, was compatible with the Auradkar-Chiueh system, and the combination would have yielded expected results. (Fellenstein, ¶8; Ex. 1017, ¶8; Lee, ¶306.)

IX. CONCLUSION

The Board should institute IPR and cancel the challenged claims.

Dated: January 31, 2023

Respectfully submitted,

/Thomas Millikan/

Lead Counsel

Thomas Millikan, Reg. No. 72,316

Back-up Counsel

Babak Tehrani, Reg. No. 55,937

Andrew Klein, *PHV to be requested*

Wei Yuan, Reg. No. 71,772

Attorneys for Netskope, Inc.

PERKINS COIE LLP
11452 El Camino Real, Ste 300
San Diego, California 92130-2080
858.720.5700 (phone)
858.720.5799 (fax)

LISTING OF CHALLENGED CLAIMS

[1[pre]] A method comprising:

[1[a][i]] assigning to one or more users, by a gateway device, a policy for managing access to and processing a file to be stored on one or more cloud platforms,

[1[a][ii]] wherein the policy defines access rights of the one or more users,

[1[c]] encrypting, by the gateway device, using cryptographic key information defined by the policy, content of the file to produce a searchable encrypted file by:

[1[b][i]] dividing the file into a plurality of chunks;

1[b][ii] creating namespaces for one or more of the plurality of chunks; and

1[b][iii] configuring the namespaces of the one or more chunks such that content of the file is encrypted in a manner that makes it searchable;

1[c] storing, by the gateway device, the searchable encrypted file on the one or more cloud platforms based on the policy; and

1[d] managing access to the searchable encrypted file by the one or more users based on the policy.

[2] The method of claim 1, wherein the step of managing comprises allowing the one or more users to download the searchable encrypted file to a local

device.

[3] The method of claim 2, wherein the local device comprises a mobile computing device.

[6] The method of claim 1, wherein the step of configuring the namespaces comprises storing content of the file as filenames.

[8] The method of claim 1, wherein the step of encrypting the file is carried out independent of the service provider of the one or more cloud platforms.

[9] The method of claim 1, wherein the policy is implemented for a group of users.

[10] The method of claim 1, wherein the policy defines the manner in which the file is stored, accessed, and processed.

[11] The method of claim 1, wherein the file is stored across two or more cloud platforms.

[12] The method of claim 1, wherein said storing the searchable encrypted file on the one or more cloud platforms is carried out independent of the service provider of the one or more cloud platforms.

[13] The method of claim 1, wherein the policy is selected from a plurality of policies stored in a database, wherein the database is operatively coupled with the gateway device.

[14] The method of claim 1, wherein the policy defines one or more of a set

of encryption keys, a set of decryption keys, search indices, a type of encryption and one or more authorized locations to which the file may be stored.

[15] The method of claim 1, wherein said encrypting further comprises encrypting one or more of filename and search indices.

[16

```
pre]] A secure cloud storage system comprising:
```

[16

```
a]] one or more processors;
```

[16

```
b]] a communication interface mechanism;
```

[16

```
c]] one or more internal data storage devices operatively coupled to the one or more processors and storing:
```

[16

```
c][i]] a policy assignment module, which, when executed by the one or more processors, implements a policy selected from a group of policies for a user,
```

[16

```
c][i][A]] wherein the policy defines and manages access to and processing of a file to be uploaded on a cloud,
```

[16

```
c][i][B]] wherein the policy defines access rights of the user;
```

[16

```
c][ii]] an encryption module, which, when executed by the one or more processors:
```

[16

```
c][ii][A]] using cryptographic key information defined by the policy, produces a searchable encrypted file by encrypting content of the file to be stored across one or more cloud platforms;
```

[16

```
c][ii][B]] divides the file into a plurality of chunks;
```

[16[c][ii][C]] creates namespaces for one or more of the plurality of chunks;
and

[16[c][ii][D]] configures the namespaces of the one or more chunks such that content of the file is encrypted in a manner that makes it searchable;

[16[c][iii]] a storage module, which, when executed by the one or more processors, stores the searchable encrypted file on the one or more cloud platforms based on the policy; and

[16[c][iv]] a management module, which, when executed by the one or more processors manages and controls access to the searchable encrypted file by the user based on the policy.

[17] The system of claim 16, wherein the management module further allows the user to download the searchable encrypted file to a local device.

[21] The system of claim 16, wherein one or more of the policy assignment module, the encryption module, the storage module, and the management module are implemented on different devices, wherein the devices are selected from one or more of access management devices, proxy devices, gateway devices, and network controllers.

[22] The system of claim 16, further comprising a mediation module, which, when executed by the one or more processors makes implementation of one or more of the policy assignment module, the encryption module, the storage module,

and the policy management module independent of the service provider of the one or more cloud platforms.

[23] The system of claim 16, wherein the policy is implemented for a group of users.

[24] The system of claim 16, wherein the policy defines the manner in which the file is stored and processed.

[25] The system of claim 16, wherein the file is stored across two or more cloud storage platforms.

[26] The system of claim 16, wherein the group of policies are stored in a database operatively coupled with the system.

[27] The system of claim 16, wherein the policy defines and manages one or more of encryption keys, decryption keys, search indices, type of encryption, and location of the file.

CERTIFICATE OF WORD COUNT UNDER 37 CFR §42.24(d)

Under 37 C.F.R. §42.24(d), the undersigned certifies that the word count for this *Petition for Inter Partes Review* totals 13,902, excluding the parts exempted by 37 C.F.R. §42.24(a). Accordingly, this Petition is under the word count limit of 14,000 words.

The word count was made using the built-in word count function in the Microsoft® Word 365 software used to prepare this document.

Dated: January 31, 2023

Respectfully submitted,

/Thomas Millikan/

Lead Counsel

Thomas Millikan, Reg. No. 72,316

PERKINS COIE LLP
11452 El Camino Real, Ste 300
San Diego, California 92130-2080
858.720.5700 (phone)
858.720.5799 (fax)

Back-up Counsel
Babak Tehranchi, Reg. No. 55,937
Andrew Klein, *PHV to be requested*
Wei Yuan, Reg. No. 71,772

Attorneys for Netskope, Inc.

CERTIFICATE OF SERVICE

I hereby certify that on January 31, 2023, I caused a true and correct copy of the foregoing PETITION FOR *INTER PARTES* REVIEW of U.S. Patent No. 9,280,678 and supporting materials (Exhibits 1001-1023 and Power of Attorney) to be served via Federal Express and electronically on the Patent Owner at the following correspondence address of record as listed on PAIR:

MICHAEL A DESANCTIS
JAFFERY WATSON MENDONSA & HAMILTON LLP
7501 Village Square Drive, Ste. 206
Castle Pines, CO 80108
mike.desanctis@jwmhlaw.com

A courtesy copy was also sent via electronic mail to the Patent Owner's litigation counsel at the following email addresses:

Andrew Michael Holmes - drewholmes@quinnemanuel.com
Jordan Brock Kaericher - jordankaericher@quinnemanuel.com
Sean S. Pak - seanpak@quinnemanuel.com
Tigran Guledjian - tigranguledjian@quinnemanuel.com

Dated: January 31, 2023

Respectfully submitted,

/Thomas Millikan/

Lead Counsel

Thomas Millikan, Reg. No. 72,316

PERKINS COIE LLP
11452 El Camino Real, Ste 300
San Diego, California 92130-2080
858.720.5700 (phone)
858.720.5799 (fax)

Back-up Counsel
Babak Tehranchi, Reg. No. 55,937
Andrew Klein, *PHV to be requested*
Wei Yuan, Reg. No. 71,772

Attorneys for Netskope, Inc.