

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

NETSKOPE, INC.,

Petitioner,

v.

FORTINET, INC.,

Patent Owner.

PTAB Case No. IPR2023-00456

Patent No. 11,036,856 B2

**PETITION FOR *INTER PARTES* REVIEW
OF U.S. PATENT NO. 11,036,856 B2**

TABLE OF CONTENTS

Page

EXHIBIT LIST	III
I. INTRODUCTION	1
II. MANDATORY NOTICES	1
A. Real party in interest.....	1
B. Related matters	1
C. Counsel and service information.....	2
III. IPR REQUIREMENTS	2
A. Standing.....	2
B. Challenged claims and relief requested.....	2
1. Prior art	2
2. Challenged claims and grounds	3
IV. TECHNOLOGY DESCRIPTION	3
A. Overview	3
1. Anti-Malware	4
2. Isolating Malware	4
3. Anti-Malware on the Internet.....	5
4. Network Security Devices	7
5. Mounting Storage.....	9
B. '856 Patent Overview	9
1. Alleged Invention.....	9
2. Prosecution History.....	13
V. SECTION 325(D) ANALYSIS	13
VI. LEVEL OF ORDINARY SKILL	15
VII. CLAIM CONSTRUCTION	16
VIII. SPECIFIC GROUNDS.....	16
A. Ground 1: Combining Burdett and FortiSandbox renders	

TABLE OF CONTENTS
(continued)

	Page
claims 1-7, 9-18, 20-26, and 28 obvious.....	16
B. Ground 2: Combining Sharpe, Seese, and Kamalapuram renders claims 1, 2, 4, 5, 7-10, 12-14, 16, 18-22, 24, and 26-28 obvious.	50
C. Ground 3: Combining Sharpe, Seese, Kamalapuram, and Dods renders claims 3, 6, 11, 15, 17, 23, and 25 obvious.....	79
IX. CONCLUSION.....	81

EXHIBIT LIST

Ex. 1001	U.S. Patent No. 11,036,856 to Graun et al. (“the ’856 patent”)
Ex. 1002	Supporting Declaration of Dr. Michael Franz (“Franz”)
Ex. 1003	File History of U.S. Patent No. 11,036,856
Ex. 1004	U.S. Patent No. 11,281,775 to Burdett et al. (“Burdett”)
Ex. 1005	FortiSandbox – Administration Guide, Version 2.3.2 (“FortiSandbox”)
Ex. 1006	U.S. Patent No. 9,811,662 to Sharpe et al. (“Sharpe”)
Ex. 1007	U.S. Patent Publication No. 2015/0100357 to Seese et al. (“Seese”)
Ex. 1008	U.S. Patent Publication No. 2020/0050686 to Kamalapuram et al. (“Kamalapuram”)
Ex. 1009	U.S. Patent No. 10,382,468 to Dods (“Dods”)
Ex. 1010	Microsoft Computer Dictionary, Fourth Edition (1999)

I. INTRODUCTION

For decades, antivirus and anti-malware technologies have pursued the same objective: preventing malicious code from infecting computer systems, storage, and networks. U.S. Pat. No. 11,036,856 (“the ’856 patent,” Ex. 1001) is no different. It merely combines well-known components and techniques (e.g., anti-malware technologies, isolating malware, a “network security device,” and mounting network storage), and applies them to files located in public cloud storage. Merely rebranding these old approaches as “cloud” technology does not make them novel, however. Because the claimed invention adds nothing over the prior art, Netskope, Inc. (“Petitioner”) requests IPR and cancellation of claims 1-28.

II. MANDATORY NOTICES

A. Real party in interest

Netskope, Inc. 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 ’856 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* IPR2023-00457, IPR2023-00458, 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 Tehranchi (Reg. No. 55,937), Andrew N. Klein (*pro hac vice* forthcoming), and Kristen Schunter (Reg. No. 76,519).

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

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

Netskope-FortinetIPRTeam@perkinscoie.com.

A Power of Attorney for Petitioner is being filed concurrently.

III. IPR REQUIREMENTS

A. Standing

The '856 patent is available for IPR and Petitioner is not barred or estopped from challenging the claims on the grounds herein.

B. Challenged claims and relief requested

Petitioner requests the challenged claims be cancelled under §103.

1. Prior art

Petitioner relies on the references listed in the Table of Exhibits, including:

- U.S. Patent No. 11,281,775 (“Burdett”) (Ex. 1004)
- FortiSandbox – Administration Guide, Version 2.3.2

(“FortiSandbox”) (Ex. 1005)

- U.S. Patent No. 9,811,662 (“Sharpe”) (Ex. 1006)
- U.S. Patent Publication No. 2015/0100357 (“Seese”) (Ex. 1007)
- U.S. Patent Publication No. 2020/0050686 (“Kamalapuram”) (Ex. 1008)
- U.S. Patent No. 10,382,468 (“Dods”) (Ex. 1009)

All references are prior art under AIA 35 U.S.C. §§102(a)(1) or 102(a)(2).

Petitioner also relies on the declaration of Dr. Michael Franz (“Franz”) (Ex. 1002).

2. Challenged claims and grounds

Ground	Basis	Reference(s)	Challenged Claims
1	§103	Burdett, FortiSandbox	1-7, 9-18, 20-26, 28
2	§103	Sharpe, Seese, Kamalapuram	1-2, 4-5, 7-10, 12-14, 16, 18-22, 24, 26-28
3	§103	Sharpe, Seese, Kamalapuram, Dods	3, 6, 11, 15, 17, 23, 25

IV. TECHNOLOGY DESCRIPTION

A. Overview

The challenged claims merely combine well-known concepts into a commonly understood configuration for defending against malicious code on the

Internet, and apply that arrangement to files located in public cloud file stores.

1. Anti-Malware

Self-replicating malware has plagued computers for decades. Early malware (including the “original internet worm” of 1988, “Melissa” in 1999, “ILOVEYOU” in 2000, and “Code Red” in 2001) received broad general press coverage:



New York Times article describing “ILOVEYOU” virus¹

Public awareness of malware and its potential damage, led to the establishment of an anti-malware industry by 2000. (Franz, ¶¶35, 36.)

2. Isolating Malware

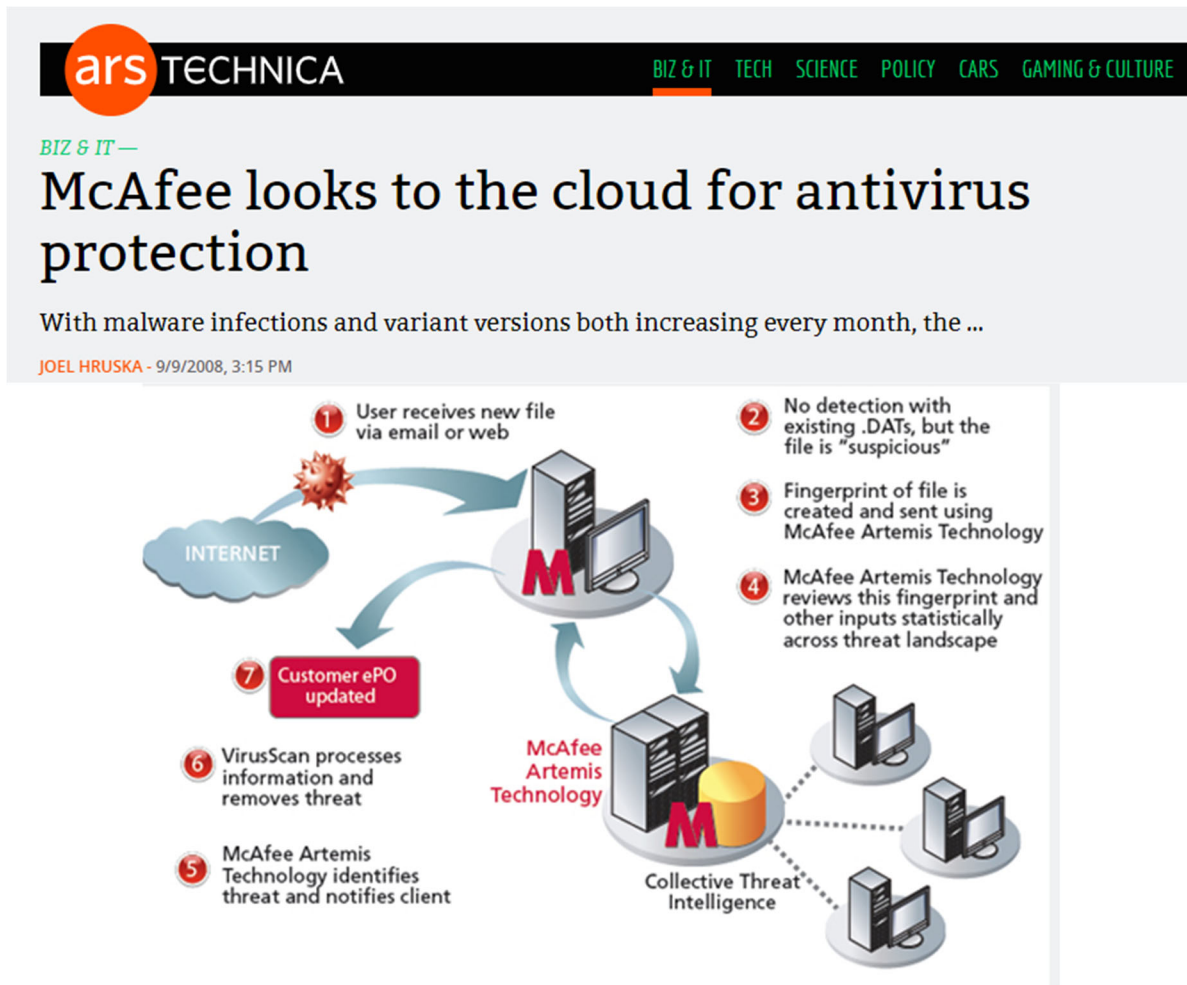
To mitigate potential damage from malware, a suspected file is separated from

¹ John Markoff, *An 'I Love You' Virus Becomes Anything But*, NEW YORK TIMES, May 7, 2000, at 584.

the main computing system. Thus, for decades, analyses of suspected malware have been conducted inside a controlled, isolated environment referred to as a “sandbox.” (See Survey on Automated Dynamic Malware-Analysis Techniques and Tools, ACM Computing Surveys, published 5 March 2008, revised 1 February 2010). (Franz, ¶37.)

3. Anti-Malware on the Internet

Performing “executable” dynamic analysis in the cloud to benefit all users of a network rather than repeating the same process on individual target computers is not a new technique. For example, by 2008, McAfee had a cloud-based virus detection system to scan new files received via email or web for viruses:



McAfee Cloud-Based Virus Detection²

More than a decade before the '856 patent's priority date, a POSITA would have been familiar with concepts such as using an in-cloud network service for analyzing potential malware and selectively quarantining files deemed a potential threat.

² Joel Hruska, *McAfee looks to the cloud for antivirus protection*, ARS TECHNICA, Sept. 9, 2008, <https://arstechnica.com/information-technology/2008/09/mcafee-moving-to-the-cloud-for-antivirus-protection/>

(Franz, ¶¶38-40.)

4. Network Security Devices

The early 2000’s network security devices became standalone, so much so, that the industry saw the widespread adoption of the term “appliance computing” (e.g., “computing appliance,” “network appliance,” “security appliance,” “network storage appliance”) to describe a device connected to a network that provided specific services.

For example, by at least June 2011, Oracle had released a storage appliance – the “Sophos ZFS Storage Appliance” – which contained integrated virus scanning:

Using Sophos Anti-Virus for Virus Scanning on the Sun ZFS Storage Appliance

By Thomas Hanvey
June 2011

The Sun ZFS Storage Appliance from Oracle features a built-in scanning service that can communicate with external virus scanning engines to ensure protection against the threat of viruses and other malware on the appliance's CIFS or NFS shared volumes.

This article describes the installation and configuration of Sophos Anti-Virus software on Microsoft Windows, Linux, and UNIX operating systems for use as a virus scan engine with the Sun ZFS Storage Appliance VSCAN service.

Contents

- [Introduction](#)
- [How VSCAN Works](#)
- [Overview of Sophos Anti-Virus](#)
- [Installing and Configuring the Sophos Anti-Virus Software](#)
 - [Installing on Microsoft Windows](#)
 - [Installing on Linux and Other UNIX Platforms](#)
- [Configuring the Sun ZFS Storage Appliance](#)
- [Conclusion](#)
- [References](#)

Oracle ZFS Storage Appliance, 2011³

(Franz, ¶¶41-46.)

³ Thomas Hanvey, *Using Sophos Anti-Virus for Virus Scanning on the Sun ZFS Storage Appliance*, June 2011, <https://www.oracle.com/technical-resources/articles/storage/sophos-anti-virus-zfs-appliance.html>

5. Mounting Storage

The practice of mounting storage to a computing device has also existed for decades. For example, the *Microsoft Computer Dictionary Fourth Edition* (1999) gave the following definition:

mount \mount\ *vb.* To make a physical disk or tape accessible to a computer's file system. The term is most commonly used to describe accessing disks in Apple Macintosh and UNIX-based computers.

Definition of “mount”⁴

The Fifth Edition (2002) added the following definition of “mount” as a noun:

mount *n.* In NFS, a folder or file retrieved from elsewhere on the network and accessed locally. See also NFS.

Thus, the practice of mounting a remote network storage repository to a local computer is not new. (Franz, ¶¶47-49.)

B. '856 Patent Overview

1. Alleged Invention

The '856 patent concerns “[s]ystems and methods for continuously scanning

⁴ Ex. 1010.

and/or sandboxing files to protect users from accessing infected files by natively mounting public cloud file stores.” (’856 patent, Abstract.) The challenged claims are directed to preventing access to files in a public cloud file store until the files have been determined to be free of malicious content.

In the ’856 patent, a public cloud file store is natively mounted on a network security device. Figures 1A and 1B (both annotated below) depict the architecture in which the invention operates. In Figure 1A, a public cloud file store (green) contains a “dirty file store” and a “clean file store.” (’856 patent, 7:50-54, 8:26-31.) Users of the cloud file store only have access to the clean file store. (*Id.*) A network security device (red) mounts the public cloud file store, ingests untrusted files from the dirty file store, and scans the untrusted file for malicious content. (*Id.*, 8:41:52.) If the file is clean, it is copied to the clean file store; if malicious, it is quarantined or deleted. (*Id.*, 8:26-31, 8:54-57, 9:12-15.) Figure 1B illustrates a similar architecture, but with the clean file store (teal) located outside the public cloud file store. (*Id.*, 9:16-26; Franz, ¶¶57-58.)

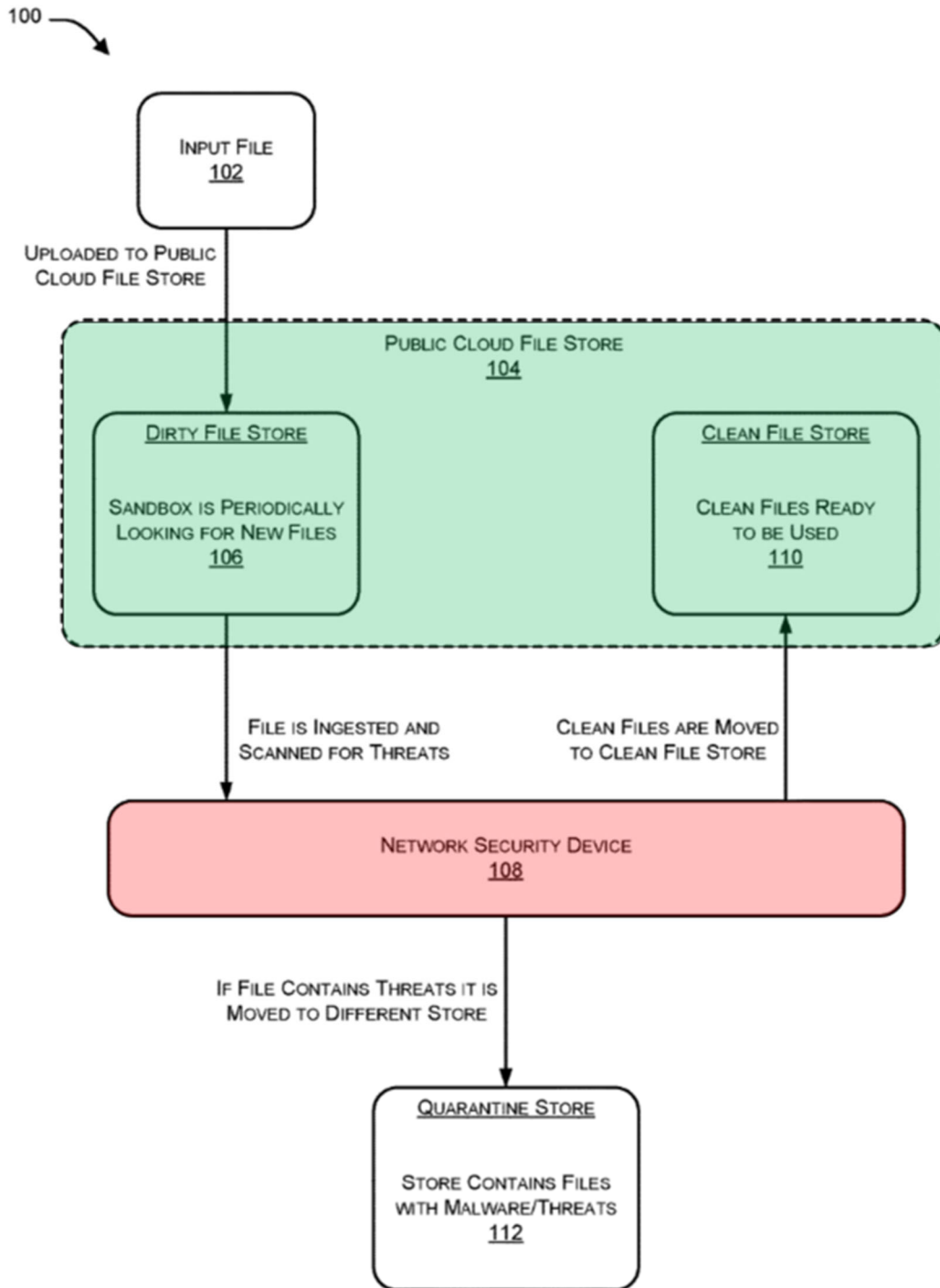


FIG. 1A (Annotated)

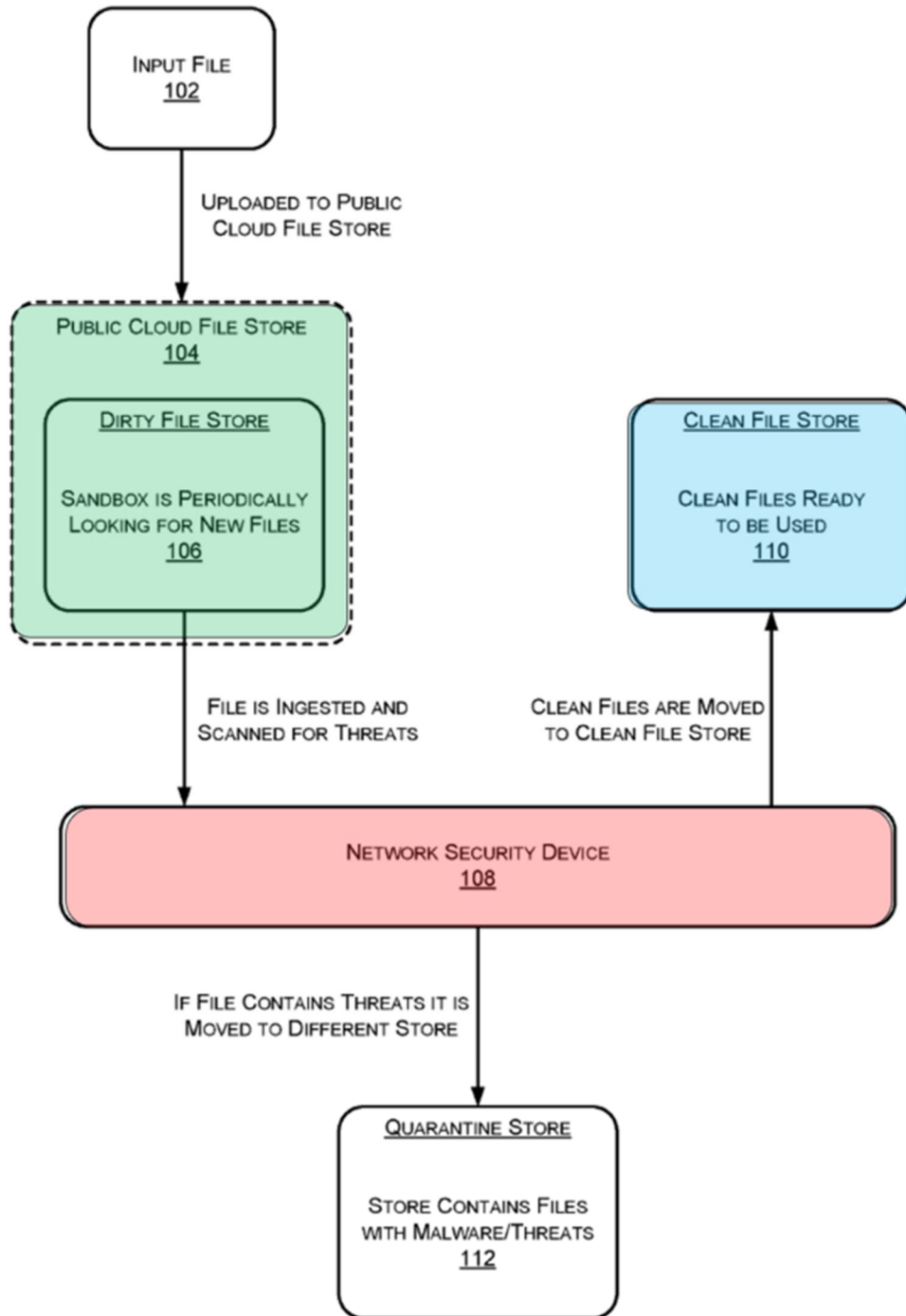


FIG. 1B (Annotated)

2. Prosecution History

The '856 claims were allowed after one Office Action rejecting then-pending claims as obvious over different combinations of U.S. Patent Publication No. 2015/0101021 A1 (“McErlean”) and U.S. Patent Publication No. 2020/0050686 A1 (“Kamalapuram”), U.S. Patent Publication No. 2018/0218155 A1 (“Grafi”) and U.S. Patent Publication No. 2017/0353496 A1 (“Pai”). (*See* Ex. 1003, 194-210.)

In response, Applicant amended pending independent claims 1 and 13 to address minor typographical issues and argued that McErlean did not disclose “a ‘public cloud file store’ being ‘natively mounted on a network security device,’” while Kamalapuram did not rectify this deficiency. (*Id.*, 249-258.) Applicant also added claims 21-28, Beauregard claims that mirrored claims 13-20. (*Id.*, 249-258.) The examiner subsequently allowed claims 1-28. (*Id.*, 265-272.)

V. SECTION 325(D) ANALYSIS

Three references cited herein—Burdett, FortiSandbox, and Seese—were not cited during original examination. Dods, used in Ground 3 against certain dependent claims, was cited in an IDS (Ex. 1003, 79) but was not used as a basis for claim rejections or discussed in any office actions.

Sharpe and Kamalapuram, used in Grounds 2 and 3 of this Petition, were cited by the Examiner during prosecution. Kamalapuram was as a secondary reference that, in combination with McErlean, rendered the as-filed claims obvious. (Ex.

1003, 196-209.) Sharpe was cited as additional art made of record but not used as a basis for rejection. (*Id.*, 208.) For at least the following reasons, the Board should institute review of the '856 patent notwithstanding its discretion under §325(d). *Becton, Dickinson & Co. v. Braun Melsungen AG*, IPR2017-01586, Paper 8 (PTAB Dec. 15, 2017) (identifying six-factor test under § 325(d); *Advanced Bionics, LLC v. Med-El Elektromedizinische Geräte GMBH*, IPR2019-01469, Paper 6 at 8 (PTAB Feb. 13, 2020) (precedential)).

Factors (a), (b), and (d): Although Sharpe, Kamalapuram, and Dods were before the Examiner, this Petition is not cumulative of original prosecution. First, neither Ground 1 reference was before the Office. Second, the Office did not consider Sharpe, Kamalapuram, and Dods in combination with Seese as presented in Grounds 2 and 3. Notably, Seese was not before the Examiner and disclosed the very limitation Applicant argued was missing in McErlean to obtain allowance. (Ex. 1003, 249-251; *Oticon Medical AB v. Cochlear Limited*, IPR 2019-00975, Paper 15 at 9-20 (P.T.A.B Oct. 16, 2019) (declining to exercise discretion under §325(d) when a new reference, not considered by the original examiner and not cumulative over the original art, was combined with references applied during prosecution).) The Office also lacked expert testimony regarding this Petition's combinations.

Factors (c), (e), and (f):

Kamalapuram: This Petition relies on different features of Kamalapuram

than those analyzed during examination. None of the portions of Kamalapuram cited herein with respect to the independent claims were cited during prosecution. (Ex. 1003, 196-200.)

Sharpe: Although the Examiner considered Sharpe, it is a lengthy, 108-column, 65-figure patent. Thus, it is unlikely that Sharpe was reviewed in full. This is corroborated by a lack of any substantive discussion of Sharpe during the original examination. (Ex. 1003, 208-210.)

Dods: There is no indication the Examiner substantively considered Dods. (Ex. 1003.) Merely listing a reference in an IDS is insufficient under §325(d). *Penumbra, Inc. v. RapidPulse, Inc.*, IPR2021-01466, Paper 7 at 51 (P.T.A.B.) March 14, 2022); *Bowtech, Inc. v. MCP IP, LLC*, IPR2019-00379, Paper 14 at 18 (P.T.A.B. July 3, 2019).

The combinations below therefore present new and distinct invalidity challenges that better reflect the state of the prior art than what was considered during examination.

VI. LEVEL OF ORDINARY SKILL

A POSITA in September 2018 would have had at least a B.S. in computer science, computer engineering, or electrical engineering (or equivalent experience) and at least two years' practical experience with computer networking and computer security.

VII. CLAIM CONSTRUCTION

The challenged claims should be construed “in accordance with the ordinary and customary meaning...as understood by one of ordinary skill in the art and the prosecution history.” 37 C.F.R. §42.200(b); *see Phillips v. AWH Corp.*, 415 F.3d 1303 (Fed. Cir. 2005). For this proceeding, Petitioner does not believe any constructions are needed.

VIII. SPECIFIC GROUNDS

A. Ground 1: Combining Burdett and FortiSandbox renders claims 1-7, 9-18, 20-26, and 28 obvious.

Burdett: Like the '856 patent, Burdett disclosed scanning files in a cloud store for malware, while preventing read access until the scan was completed. (Burdett, Abstract, 1:54-55, 25:18-33, 25:56-67, 28:42-29:63, 30:63-31:3.) In Burdett, **a scanning service** was directly coupled to a **cloud storage system** to detect potentially malicious content in files stored in the cloud storage system. (Burdett, Abstract, 1:42-53, 25:18-21, 30:57-62.)

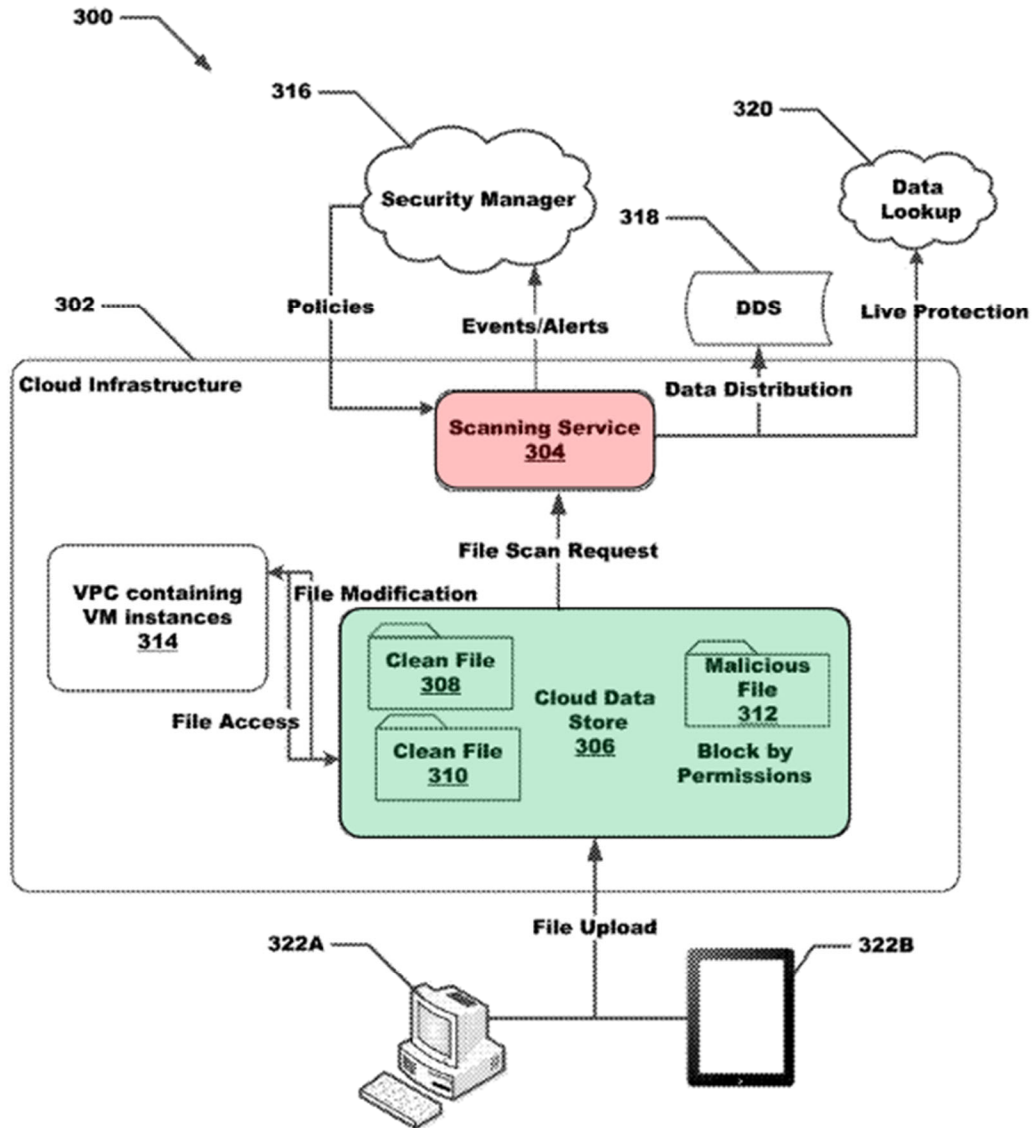


FIGURE 3 (Annotated)

The scanning service was configured with permissions that enabled it to access and make changes to a data store within the cloud storage system. (*Id.*, 28:1-32.) When the scanning service received an event notification associated with the configured data store (e.g., indicating upload or modification of a file in the data

store), the service would access the file and scan it using one or more scanners such as an anti-virus or anti-malware scanner. (*Id.*, 29:31-62.) Based on the results, the scanning service controlled access to the file. (*Id.*, 25:26-28, 25:56-67.) Among various access control mechanisms, Burdett described moving files to different locations and setting permissions to prevent file access until a scan could be performed. (*Id.*; Franz, ¶¶64-65.)

FortiSandbox⁵: FortiSandbox disclosed configuration and management of a “FortiSandbox system,” a network security device configurable to scan files on a network share, execute suspicious files within a virtual machine module, and quarantine suspicious files. (FortiSandbox, 8, 10.) The guide instructed a system administrator to schedule network share scans by mounting the shares to the system using any of the following protocols:

- Network File System (NFS) v2,
- NFSv3,
- NFSv4, or
- Common Internet File system (CIFS).

⁵ FortiSandbox was publicly available more than one year before the ’856 patent’s filing date. The document is dated November 9, 2016 (FortiSandbox, 1), with a 2016 copyright claimed by Fortinet (*id.*, 191.) (Franz, ¶¶66-67.)

(*Id.*, 10, 83, 88, 114-115; Franz, ¶68.)

1. Motivation to combine

A POSITA would have used FortiSandbox’s network security device to natively mount the public cloud storage data and implement Burdett’s malware scanning services. A POSITA would have been motivated to combine Burdett and FortiSandbox with a reasonable expectation of success.

Notably, the Burdett-FortiSandbox combination would have produced a malware detection system for an enterprise that improved security for the enterprise. As detailed in the following sections, Burdett described a scanning service that scanned an enterprise’s cloud file store for malware. (*See, e.g.*, Burdett, Abstract, 28:1-5, 29:31-41, 31:8-13.) However, Burdett did not detail how the scanning service would access the file store. FortiSandbox provided this teaching, explaining how to configure a device that could be deployed within an enterprise’s network infrastructure to scan files on a network share. (FortiSandbox, 18-20 (describing configuration of system settings of “FortiSandbox unit,” including whether its virtual machine can accessing an “outside network”); 8 (“FortiSandbox can be configured to...scan files on a network share”); *see also* 88.) Configuring a device like the FortiSandbox system to perform Burdett’s malware scans would have enabled an enterprise to preserve the privacy of its files in cloud storage while preventing enterprise users from accessing malicious files. (Franz, ¶¶69-70.)

2. Claim 1

a. 1[pre]: A secure data transfer system

Burdett disclosed a secure data transfer system including one or more computing devices within a cloud infrastructure that executed a scanning service as a virtual machine. (Burdett, Abstract, 23:33-35, 24:33-36.) Figure 3 (annotated below) illustrated a data transfer system in which files, such as those uploaded by a user (*id.*, 24:22-25), were stored in (transferred to) a cloud data store (green) where a scanning service (red) scanned them. (*Id.*, 23:33-38, 24:19-20, 25:18-19.) Burdett explained that file access in the cloud data store could be blocked until the scanning service completed its scan. (*Id.*, 25:56-64: “...an application stores a file in the cloud data store 306 *with default permissions that do not permit access* by applications running in the VPC 314. The scanning service 304 ... scans the file. If access to the file needs to be restricted based on the scan, the scanning service 304 does not change the permission of the file so that applications in the VPC 314 still cannot access the file.”⁶) If the scanning service determined that file access did not need to be restricted (e.g., because the file was malware-free), the scanning service would make the file accessible to one or more applications running in virtual machines

⁶ All **bold** or *italic* emphases that appear in quoted text herein are added by Petitioner unless otherwise specified.

(orange in annotated Figure 3 below). (*Id.*, 25:26-28, 25:64-67, 29:42-43, 31:8-14.)

By scanning files for malicious content in the cloud data store, Burdett's scanning service attempted to ensure secure data transfers in a cloud environment while providing protection against malware or other exploits. (*Id.*, 1:42-53; Franz, ¶¶71-73.)

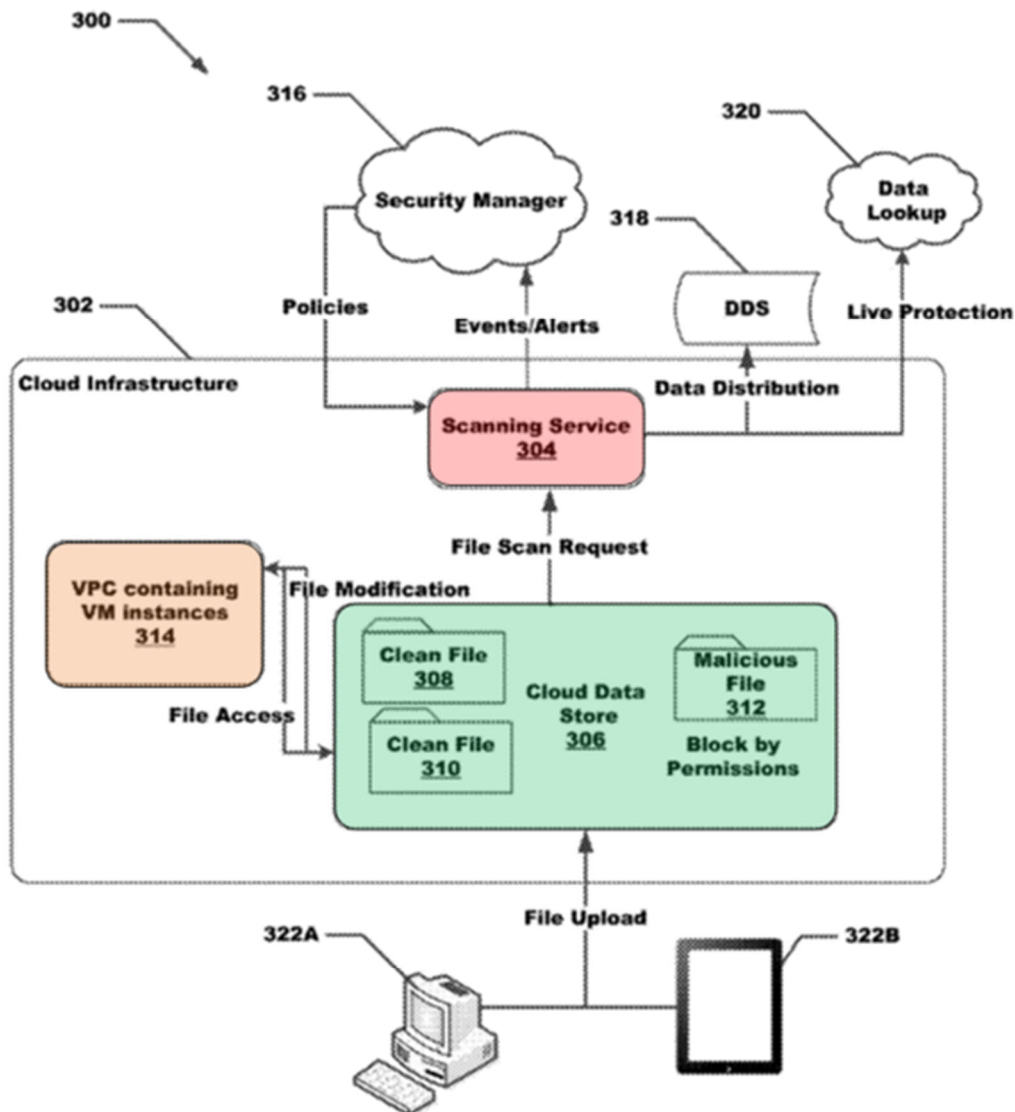


FIGURE 3 (Annotated)

b. 1[a]: a non-transitory storage device having embodied therein one or more routines operable to prevent users of an enterprise network from accessing malware infected files that are stored within public cloud file stores

Burdett taught implementing a scanning service as a virtual machine within cloud infrastructure. (Burdett, 24:33-36: “... The cloud infrastructure service 302 may include a scanning service 304 ...[which] may be implemented with one or more scanning applications running on one or more virtual machines within the cloud infrastructure 302.”) Burdett’s “scanning service simplifie[d] workflow for *deploying and managing data protection (e.g., anti-malware, data loss prevention)* for applications making use of cloud resources.” (*Id.*, 2:14-16; Franz, ¶74.)

Non-transitory storage device having embodied therein one or more routines: Burdett disclosed a non-transitory computer readable storage medium including processor-executable instructions, including those that caused the processor to scan a file accessed from cloud storage. (*Id.*, 2:22-64.) Additionally, several Burdett claims recited a non-transitory computer readable storage medium with computer readable code thereon for a scanning service. (*Id.*, 32:10-67; 34:1-50.) (Franz, ¶75.)

Enterprise network: Generally, Burdett described providing threat protection to an enterprise facility, including protecting enterprise’s network components against malicious content in files. (*Id.*, 7:14-16 (describing a security management facility that “provided protection from malware to enterprise facility

102 computer resources,” for example by “scan[ing] the client facility files for malicious code”); 14:18-21 (describing a threat management facility that “provide[d] threat protection to the enterprise facility...includ[ing] a plurality of networked components”), 5:29-31; Franz, ¶76.)

Routines operable to prevent users of an enterprise network from accessing malware infected files: Burdett disclosed one or more routines operable to control access to files in a cloud data store used by applications, as a mechanism to prevent distribution of malware infected files within a cloud repository. For example, in response to a notification of file event, the scanning service was configured to “request a copy of the file to be examined” and add the file to a scan queue. (*Id.*, 29:31-36.) Queued files were then processed by “one or more scanners,” including anti-malware or anti-virus scanners. (*Id.*, 29:58-61.) The scanning service prevented applications from accessing malware-infected files by, for example, moving the infected file (*id.*, 25:26-28), restricting its permissions (*id.*, 25:55-67), or replacing the infected file with a clean version. (*Id.*; *see also* 26:44-58.) Additionally, by denying access by applications, Burdett prevented users of the enterprise network from accessing malware-infected files. (*Id.*, 9:29, 11:57-66, and 14:19-24 (discussing users in an enterprise and restricting user access to portions of an enterprise network); *see also* 24:22-25 and 26:14-28 (illustrating examples in which a user uses an application to access the cloud data store); Franz, ¶77.)

Public cloud file store: Burdett described a cloud data store (public cloud file store) with respect to its cloud-based infrastructure as shown in Figure 3.

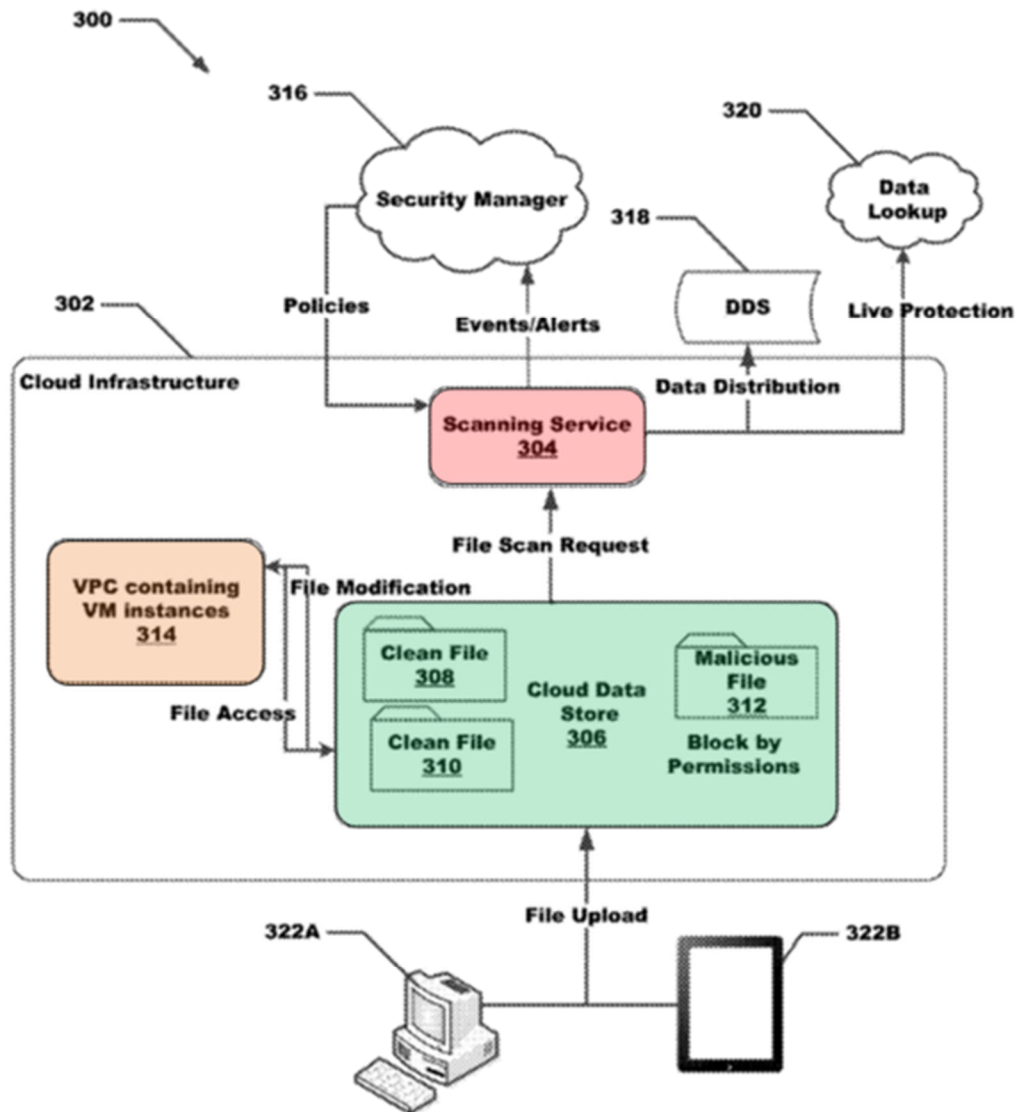


FIGURE 3 (Annotated)

A POSITA would have understood that a “public cloud file store” generally refers to a file store maintained on shared infrastructure, as opposed to an enterprise’s private infrastructure. Burdett suggested its cloud data stores were

hosted by public infrastructure. For example, Burdett described that file permissions enabled an enterprise to “take[] advantage of the benefits of the cloud,” including “avoid[ing] the need for custom infrastructure.” (*Id.*, 25:42-46.) Burdett’s cloud infrastructure, including the cloud data store, could be implemented “with any suitable cloud service and related storage system,” such as Amazon S3® or “Microsoft Azure, CenturyLink Cloud, VMware, Rackspace, Joyent, [or] Google...[or] other cloud infrastructure or infrastructure-as-a-service providers.” (*Id.*, 23:43-56; *see also* 24:28-31, 28:3.) Additionally, the scanning service was described as configured to scan multiple cloud data stores, which were accessible both by applications “instantiated on virtual machines within [] virtual private clouds 314” (*id.*, 23:36-38) and by “applications implemented elsewhere.” (*Id.*, 2:19-21; *see also* 26:6-22, 28:1-5; Franz, ¶¶78-80.)

c. 1[b]: one or more processors coupled to the non-transitory storage device and operable to execute the one or more routines

Burdett disclosed a processor configured to execute instructions on a computer-readable storage medium and thereby perform one or more routines. (Burdett, 2:22-53, 26:59-67; 32:10-15; Franz, ¶81.)

d. 1[c][i]: wherein the one or more routines include: an untrusted file processing module, which when executed by the one or more processors: accesses an untrusted file stored within a first repository of a public cloud file store

Burdett disclosed an untrusted file processing module including a processor

executing a set of instructions or routines to operate the scanning service on files in the cloud data store. (Burdett, 26:59-67, 31:4-14, 31:24-34, 32:10-15; *see also* elements 1[a] and 1[b]; Franz, ¶84.)

Burdett described the scanning service accessing a file to be scanned – an untrusted file – stored within a cloud data store. (Burdett, 25:18-19, 26:37-41.) Specifically, Burdett disclosed “...an application stores a file in the cloud data store 306 with default permissions that do not permit access by applications running in the VPC 314. The scanning service 304 ... [then] scans the file.” (*Id.*, 25:56-61.)

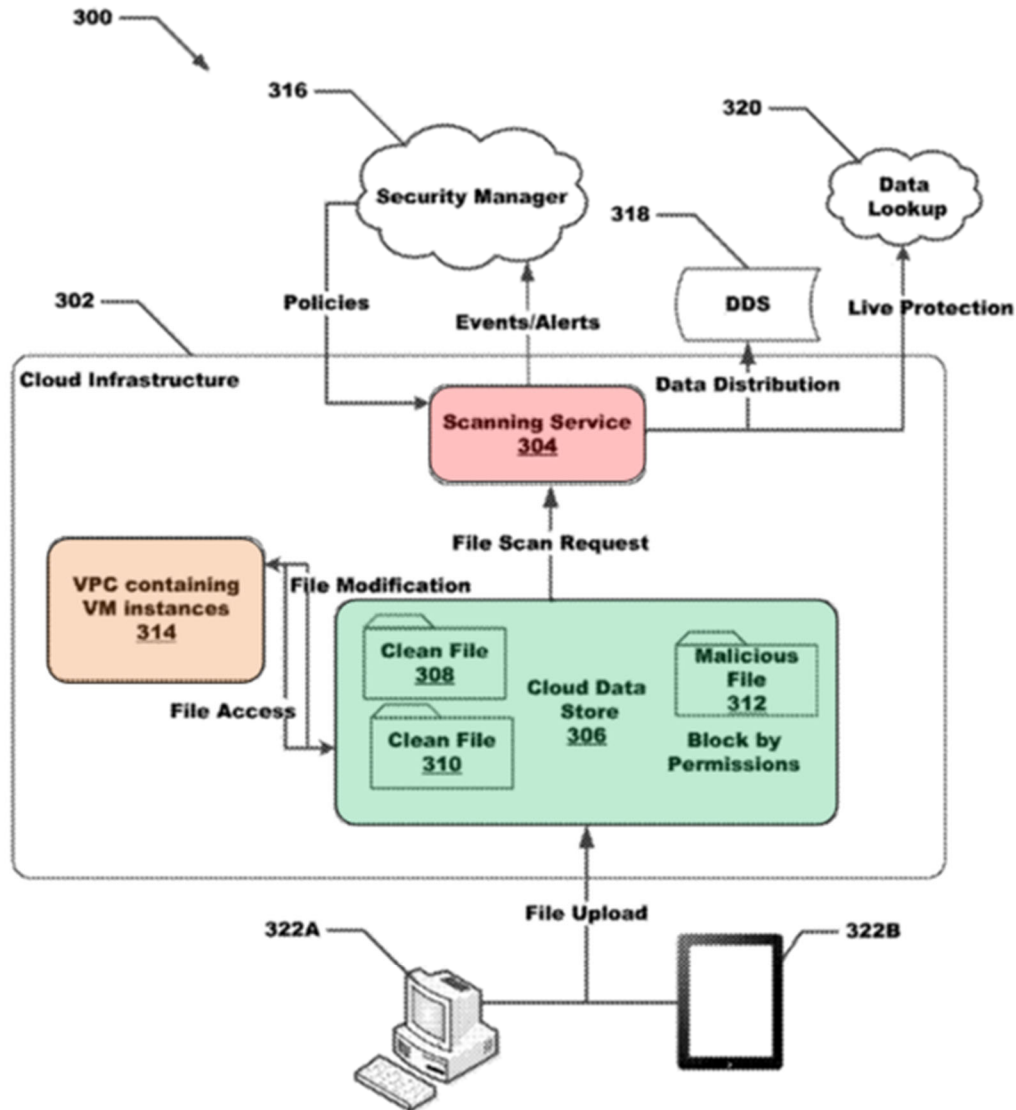


FIGURE 3 (Annotated)

Burdett’s disclosure of initially storing the files in the cloud data store 306 prior to conducting the scan so that they are inaccessible to applications constitutes storing an untrusted file within a first repository. Burdett’s description is similar to the ’856 patent’s dirty file store (first repository):

“...untrusted files, such as input file 102...are initially

shared/sent/uploaded [to a] dirty file store 106 of public cloud file store 104, before it is made accessible to its intended user/any user of the enterprise network. In a similar manner, all files being requested by different users of the network can first be stored in dirty file store 106, which can be configured to be inaccessible to the end users.” (’856 patent, 8:8-17.)

Additionally, a POSITA would have found it obvious to store Burdett’s file in cloud data store 306 prior to scanning service 304’s scan because it was well known to use repositories to logically organize files in a file store. (*Id.* at 28:1-5, 28-10-13.) Burdett explains that the scanning service could be “associated with multiple data stores, cloud infrastructure zones, applications, etc.,” and could scan “multiple data stores (e.g., multiple S3 buckets).” (Burdett, 28:2-5.) Thus, in an example, the first repository could be a first S3 bucket. (Franz, ¶¶82-83.)

e. 1[c][ii]: wherein the public cloud file store is natively mounted on a network security device that is protecting the enterprise network

Burdett did not explicitly describe natively mounting the public file store on a network security device, but FortiSandbox did. Specifically, FortiSandbox—a network security device configured to perform scans of network shares (FortiSandbox, 8, 10)—was configured to mount a network share, selected from multiple mount types including CIFS, NFSv2, NFSv3, and NFSv4. (*Id.*, 114-115, Franz, ¶85.) For example, the table below shows that a FortiSandbox user could create a new network share using those mount types:

To create a new network share:

1. Go to *Scan Input > Network Share*.
2. Click the + *Create New* button from the toolbar.
3. Configure the following options:

Enabled	Select to enable network share configuration. If network share is not enabled, its scheduled scan will not run.
Network Share Name	Enter the network share name.
Mount Type	Select the mount type from the drop-down list. The following options are available: • CIFS • NFSv2 • NFSv3 • NFSv4

⋮

(FortiSandbox, 114-115.) FortiSandbox’s use of mount types including CIFS and NFS protocols, which are native filesystem protocols, is consistent with the ’856 patent’s discussion of “natively mounting” a file store on a network security device. For example, the ’856 patent describes that a data store “may be mounted by the network security device as NFS and resulting clean files may be copied to file shares accessible to end users via CIFS.” (’856 patent, 11:7-10.) These protocols enabled the FortiSandbox system to communicate with the filesystem maintaining the network share in the filesystem’s native language – and thus the filesystem was natively mounted on the FortiSandbox system. (Franz, ¶¶85-87.)

In the Burdett-FortiSandbox combination, at least some of the operations

conducted by Burdett’s scanning service would have been performed by a device, like a FortiSandbox device (a network security device). A POSITA would naturally position the FortiSandbox device in a position to protect an enterprise network because that would improve Burdett’s disclosed system for protecting an enterprise network. (Franz, ¶88; *see also, supra*, “enterprise network” in 1[a].)

Moreover, FortiSandbox corroborated Burdett’s disclosure of scanning files in a file store. For example, once a new network share had been configured as described above, the FortiSandbox device would “scan files stored on [the] network share and optionally quarantine any malicious files.” (FortiSandbox, 113.) Scans could be “scheduled or run on-demand.” (*Id.*, 114.) Thus, performing Burdett’s scanning service operations on the FortiSandbox device would have been a combination of prior art elements according to known methods to yield predictable results. (Franz, ¶¶85-89.)

A POSITA would have found it obvious to make this combination by natively mounting the public cloud file store on the network security device. Doing so would have enabled the network security device to access files in the cloud file store and to natively communicate with the cloud file store to perform actions such as controlling file access. (Franz, ¶89.) It also would have enabled the network security device to perform scheduled or on-demand bulk scans within the cloud file store. (Burdett, 2:5-17; FortiSandbox, 114; Franz, ¶¶85-89; *see also* ’856 patent, 11:66-

12:4.)

f. 1[c][iii]: wherein the users do not have read access to the first repository of the public cloud file store

As discussed in connection with Element [c][i], Burdett described storing files in the cloud data store with default permissions that did not allow access by external applications—or users of those applications—until after the file had been scanned and deemed clean. (Burdett, 25:57-64; *see also* 24:22-25 and 26:14-28; Franz, ¶90.)

As an alternative to using file permissions, Burdett described the scanning service moving a file to control access. (Burdett, 25:26-28.) Although this disclosure suggests that a file was first stored in another repository before the scanning service moved it into a repository that was inaccessible to applications, it would have been obvious for untrusted files to initially be stored in such an inaccessible repository (e.g., a designated S3 bucket) because this would have achieved Burdett’s objective to prevent file access until a scan has been performed. (Burdett, 25:57-64; Franz, ¶91.) Burdett’s disclosure that storing files in an inaccessible location has the same effect as changing file permissions with respect to preventing user access to untrusted files would motivate a POSITA to establish such a repository with no user read access. (Burdett, 26:44-58; Franz, ¶91.)

g. 1[d]: wherein the one or more routines include: an untrusted file processing module, which when executed by the one or more processors: ... causes the network security device to make a determination regarding whether the untrusted file is a clean file that is free of malicious content

by applying one or more security checks to the untrusted file

Burdett's scanning service included functionality as part of an untrusted file processing module that applied one or more security checks (e.g., anti-malware scan or an anti-virus scan) to an untrusted file. (Burdett, 29:58-61 ("the scanning services uses one or more scanners (e.g., anti-malware, anti-virus,...) to scan the file and return a result."), *see also* 1:64-66, 8:1-12, 24:51-56, and 25:12-13.) In one embodiment, Burdett's system restricted access to the files that were transferred to the cloud data store before performing a scan (*see id.*, 25:59-61), and "[i]f access to the file need[ed] to be restricted based on the scan, the scanning service 304 d[id] not change the permission of the file so that applications in the VPC 314 still c[ould not] access the file. *If access to the file d[id] not need to be restricted based on the scan*, the scanning service 304 change[d] the permission of the file so that applications in the VPC 314 may access the file." (*Id.*, 25:61-67; *see also* 29:42-52, 31:4-14).

Based on the scan, Burdett's scanning service could also prevent the file from being distributed if the scan determined that the file contained malware, disruptive computer instructions, exploits, or content that may compromise infrastructure. (*Id.*, 31:24-34.) (Franz, ¶¶92-93.)

- h. 1[e]: wherein the one or more routines include: ... a clean file transfer module, which when executed by the one or more processors, makes the clean file accessible to the users**

**by, when a result of the determination is affirmative,
copying the clean file from the first repository to a second
repository that is accessible to the users.**

Burdett taught or suggested this limitation. After a file was scanned by Burdett's scanning service and deemed clean, Burdett explained that in "*some implementations*," the file was made accessible to the user by changing the file permission. (Burdett, 25:64-67.) Burdett, however, further described other interchangeable mechanisms to control access to a file, including changing a file name of the file, or *changing the storage location of the file*: "In some implementations, *the scanning service may move files* or change the name of files *in order to control access*." (*Id.*, 25:26-27.)

Guided by Burdett, a POSITA would have found it obvious that one alternative to control access to a contaminated file was to move the file to a different location—to a second repository - as opposed to changing the file permissions, and that this alternative would have been more efficient than managing access via file-level permissions. A POSITA would have found a further suggestion for this in Burdett's Figure 3, which depicted storing **clean** (**aqua** in annotated Figure 3 below) and **malicious** (**purple**) files in separate locations. (Franz, ¶¶94-95.)

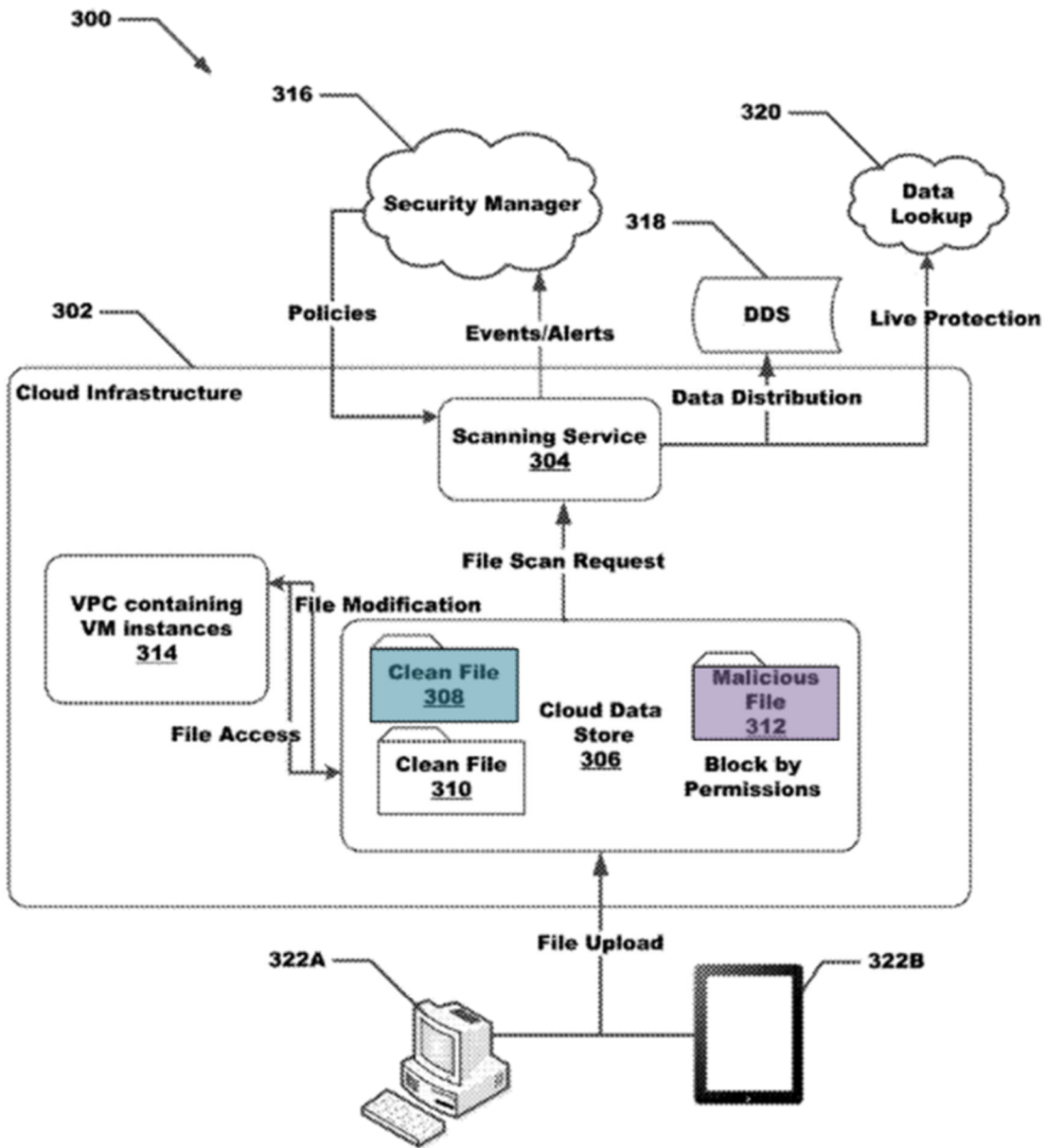


FIGURE 3 (Annotated)

In light of Burdett's disclosure, it would also have been obvious to copy a clean file to a second repository that is accessible to the applications. Additionally, by making a file accessible to one or more applications, a POSITA would have

understood that the file would be available to users of those applications. (Franz, ¶96.)

Burdett further disclosed a clean file transfer module in the form of the set of instructions or routines executed by the scanning service to make the clean file accessible to users. (Burdett, 26:59-67, 31:4-14, 31:24-34, 32:10-15; see also elements 1[a] and 1[b]; Franz, ¶97.) As described with respect to elements 1[a] and 1[b], it would have been obvious for these instructions or routines to have been executed by one or more processors.

3. Claim 13

a. 13[pre]: A method comprising:

Burdett disclosed a method for scanning files in a cloud data for malicious content. (Burdett, 26:59-61, 29:31-52, Figures 4A, 4B, 5A, and 5B; Franz, ¶98.)

b. 13[a][i]: receiving, by a network security device that is protecting an enterprise network, a notification, via an Application Programming Interface (API) call from a notification service monitoring a first repository of a public cloud file store, regarding existence of an untrusted file stored within the first repository

Burdett and FortiSandbox taught this feature of claim 13. Burdett disclosed a scanning service that performed anti-virus or anti-malware scans on files stored in a cloud data store. (Burdett, 29:31-38, 29:58-61; see element 1[c][i].) It would have been obvious that the cloud storage service (which included the first repository) also included a notification service because the scanning service received “a notification

regarding storage activity related to a file.” (Burdett, 30:43-49; *see also* 28:42-43 and 29:31-32.) Such “storage activity” could include upload of a file to a cloud data store monitored by the cloud storage service, download of a file from the cloud data store, or a change made to a file in the cloud data store. (*Id.*, 30:43-49.) The notification could be received by the scanning service “using an applications programming interface (API) of the cloud infrastructure service or the cloud data store.” (*Id.*, 30:54-56.) Since a newly-uploaded or newly-modified file would not have been scanned by the scanning service, such a file would have been an untrusted file in the cloud data store. Accordingly, the notification generated in response to the “upload” or “change” file events would have effectively notified the scanning service of the existence of an untrusted file stored within the cloud data store. (Franz, ¶99.)

Finally, as described with respect to element 1[c][ii], FortiSandbox disclosed a network security device. (FortiSandbox, 8 (“More than just a sandbox, FortiSandbox deploys Fortinet’s award-winning, dynamic antivirus and threat scanning technology...” and “FortiSandbox can be configured to... scan files on a network share”).) For at least the same reasons as element 1[c][ii], it would have been obvious to implement Burdett’s scanning service on FortiSandbox’s network security device. (Franz, ¶100.)

c. 13[a][ii]: wherein the public cloud file store is natively mounted on the network security device

See element 1[c][ii].

- d. 13[a][iii]: wherein users of the enterprise network do not have read access to the first repository of the public cloud file store**

See element 1[c][iii].

- e. 13[b]: determining, by the network security device, whether the untrusted file is a clean file that is free of malicious content by applying one or more security checks to the untrusted file**

See element 1[d].

- f. 13[c]: making the clean file accessible to the users by, when a result of said determining is affirmative, copying the clean file from the first repository to a second repository that is accessible to the users.**

See element 1[e].

4. Claim 21

- a. 21[pre]: A non-transitory computer-readable storage medium embodying a set of instructions, which when executed by one or more processors of a network security device, cause the one or more processors to:**

Burdett disclosed one or more routines operable to control access to files in a cloud data store used by applications, as a mechanism to prevent users of an enterprise network from accessing malware infected files within a public cloud data store. (Burdett, 28:33-31:53.) Burdett described that its methodology could be implemented using instructions stored on a non-transitory computer medium executed by a processor. (*Id.*, 2:22-40; 34:1-50; Franz, ¶105.)

- b. 21[a][i]: receive a notification via an Application Programming Interface (API) call from a notification service monitoring a first repository of a public cloud file store, regarding existence of an untrusted file stored within the first repository**

See element 13[a][i].

- c. 21[a][ii]: wherein the public cloud file store is natively mounted on the network security device**

See element 1[c][ii].

- d. 21[a][iii]: wherein users of an enterprise network protected by the network security device do not have read access to the first repository of the public cloud file store**

See elements 1[c][ii] and 1[c][iii].

- e. 21[b]: determine whether the untrusted file is a clean file that is free of malicious content by applying one or more security checks to the untrusted file**

See element 1[d].

- a. 21[c]: make the clean file accessible to the users by, when a result of said determining is affirmative, copying the clean file from the first repository to a second repository that is accessible to the users.**

See element 1[e].

5. Claims 2/14/22⁷: ...wherein the network security device is or comprises any or a combination of a sandbox device, a

⁷ Claims 14-20 and 22-28 restate the substantive elements of claims 2-12 in method (14-20) and non-transitory computer-readable medium (22-28) format.

network controller, a firewall, a network gateway device, an Intrusion Prevention System (IPS), and an Intrusion Detection System (IDS).

As described for claim 1, FortiSandbox disclosed a network security device, i.e., a sandbox device. (FortiSandbox, 8, 10.) Burdett also taught or suggested having a “intrusion detection system” (IDS) because its system provides “a host intrusion prevention through behavioral based protection, which may guard against unknown threats by analyzing behavior before software code executes.” (Burdett, 8:1-4; Franz, ¶111.)

6. Claims 3/15/23

a. 3[a]: ...wherein the network security device comprises a sandbox appliance

FortiSandbox disclosed a sandbox appliance. (FortiSandbox, 8, 10; Franz, ¶112.)

b. 3[b]: wherein the one or more security checks comprise behavioral-based malware detection.

Burdett disclosed a system that “may provide host intrusion prevention through behavioral based protection, which may guard against unknown threats by analyzing behavior before software code executes. Behavioral based protection may monitor code when it runs and intervene if the code is deemed to be suspicious or malicious.” (Burdett, 8:1-12.) It would have been obvious for Burdett’s scanning service to similarly employ behavioral-based malicious code detection, since this was a well-known technique. Moreover, FortiSandbox corroborated the teachings

of Burdett by also describing behavior-based malware detection. (FortiSandbox, 8; Franz, ¶113.)

7. Claim 4: ...wherein the network security device comprises a virtual machine that is controlled by a sandbox appliance to determine whether the untrusted file is a clean file that is free of malicious content.

The Burdett-FortiSandbox combination taught this claim. As described for element 1[d], Burdett disclosed a scanning service that determined whether an untrusted file was free of malicious content. (Burdett, 1:64-66, 24:51-56, 29:58-60, 31:4-14, 31:24-34.) Burdett further disclosed that its “scanning service 304 may be implemented with one or more scanning applications *running on one or more virtual machines* within the cloud infrastructure 302.” (*Id.*, 24:33-36.)

The network security device described in FortiSandbox included a sandbox appliance that controlled a virtual machine. (FortiSandbox, 8 (“FortiSandbox executes suspicious files in the VM host module”); *see also* 88 (“FortiSandbox ... executes files in a VM host environment.”).) Using the virtual machine, the sandbox appliance also determined whether an untrusted file was free from malicious content. (*Id.*, 8 (“FortiSandbox executes suspicious files in the VM host module to determine if the file is High, Medium, or Low Risk based on the behavior observed in the VM sandbox module”); *see also* 88 (“FortiSandbox utilizes Fortinet antivirus to scan files for known threats and then executes files in a VM host environment.”).)

A POSITA would find it obvious that, in the Burdett-FortiSandbox

combination, Burdett’s scanning service would utilize a sandbox appliance, such as that disclosed by FortiSandbox. Executing a file in a sandbox environment would reduce the likelihood that the file would infect other network components while it underwent evaluation. Under the sandbox appliance’s control, the virtual machine taught by both Burdett and FortiSanbox would execute the file scan. (Franz, ¶¶114-116.)

8. Claims 5/16/24: ...wherein the network security device copies the file to the second repository by sharing it by means of any or a combination of network file system (NFS), file transfer protocol (FTP), common Internet file system (CIFS), Internet Small Computer Systems Interface (iSCSI), Storage Area Network (SAN), and local storage.

FortiSandbox explained how to configure a “mount type” option to mount a network share folder to the FortiSandbox system, where the mount type was selected from CIFS, NFSv2, NFSv3, and NFSv4. (FortiSandbox, 83, 115.) It would have been obvious to copy the file to the second repository, as taught by Burdett (see element 1[e]), using CIFS and/or NFS in view of FortiSandbox’s disclosure. Since CIFS and NFS are network filesystem protocols that can be used on clouds and that enable communication with a cloud file store, a POSITA would have found it obvious to use the protocols taught by FortiSandbox because they would have enabled the network security device to copy files into a second repository. Additionally, other claimed protocols—FTP, SAN and local storage—were known alternatives to CIFS and NFS protocols and would have thus been obvious to a

POSITA. (Franz, ¶117.)

9. Claims 6/17/25: ...wherein the untrusted file processing module further removes any malware from the untrusted file that is detected by the one or more security checks when the result of said determining is negative.

As described for claim 1, Burdett’s scanning service performed routines that rendered obvious an “untrusted file processing module.” The scanning service was also configured to “replace a file with another file that is ‘clean’” when it detected malware by one or more security checks. (Burdett, 25:31-33.) A POSITA would find it obvious that, in at least some instances, a scanning service that replaces an infected file with a clean file would do so by removing a malware from the infected file—an operation that was routinely performed by typical antivirus software at the time of the alleged invention. (Ex. 1010; Franz, ¶118.)

10. Claims 7/18/26: ...wherein the untrusted file processing module quarantines or deletes the untrusted file when the result of the determination is negative.

Burdett taught or suggested these claims by describing preventing access to a file that was determined to contain malicious content. (Burdett, 25:26-33, 26:44-58, 29:42-52.) In particular, if the scanning service detected malicious content in a file, the service would take an action such as “to quarantine the file by altering permissions on the file so that at least some other applications...may not access it without administrator action.” (*Id.*, 1:64-2:3; *see also* 7:23-26.) Likewise, the scanning service could notify an administrator when malicious content was detected

in a file (*id.*, 29:38-41 and 50-52, 31:39-44), enabling the administrator to delete it.

FortiSandbox corroborated Burdett’s teaching. *See*, for example, FortiSandbox, 8 (“quarantine malicious files”), 76 (“delete original files of Malicious or Suspicious ratings after a specified time”). (Franz, ¶119-120.)

11. Claims 9/20/28: ...wherein the second repository forms part of the public cloud file store.

As described for element 1[e], a POSITA would have found a second repository obvious in view of Burdett’s disclosure of changing the storage location of a file to control access to the file. (Burdett, 25:64-67; Franz, ¶121.)

A POSITA also would have found it obvious for such a second repository to form part of the public cloud file store based on the scanning service’s permissions described by Burdett. In particular, during configuration of the scanning service, the service was assigned a role including permissions that enabled the scanning service to access data in a cloud data store and to “effect changes, such as to change permissions, move files, change file names, etc. in order to control access.” (Burdett, 27:24-29.) Burdett taught that a scanning service could not access files nor change file permissions if the scanning service was not given role permissions in the cloud data store that was to be scanned. (*Id.*, 27:38-40 (“For permissions to be used for access control, the scanning service needs permissions to enable it to access files and change the permissions of files”), *see also* 28:13-21 (“If the scanning service can

access files but not change permissions, the scanning service will operate in a Notification Only protection mode ... If the scanning service has the permissions to change permissions or perform other file changes, the scanning service may operate in a Block File protection mode”).) When the scanning service copied a clean file into a second repository to enable access to the clean file, a POSITA would have found it obvious to locate the second repository in the same public cloud file store that contained the first repository, since the scanning service would have already had permissions configured to access the public cloud file store holding the first repository. Using a second repository that forms part of the public cloud file store thus would have eliminated the need for an administrator to configure the scanning service with permissions for a second cloud file store. (Franz, ¶122.)

12. Claim 10: ...wherein the public cloud file store is provided by a third-party cloud storage service provider.

Burdett described cloud data stores that were provided by a third-party cloud storage service provider – namely, Amazon Simple Storage Service (S3)®. (Burdett, 24:28-30, 28:3, 30:33-34; Franz, ¶123.)

13. Claim 11: ...wherein the network security device determines whether the untrusted file is a clean file that is free of malicious content based any or a combination of a file signature, a file hash, a file path, file attributes, a file source, file pre-scan parameters, a file extension, file content, and a file name, and any part thereof.

Burdett taught or suggested this claim because, in its background section, it

described malware detection systems that employed signature-based detection. (Burdett, 1:21-22.) The malware detection methods disclosed by Burdett further included other techniques including those based on: file source (*see id.*, 8:13-17 (“target or identify sources of known malware”)), file extension (*see id.*, 31:29-30 (“executable files...block all zip files”)), or file content (*see id.*, 24:59-64 (“identifying files or portions of files that fit a specific pattern...[or] identifying code that has the same effect of code that is known to be malicious”), 31:26-30 (“content that may compromise customer’s infrastructure” or “source code,” *inter alia*)). (Franz, ¶124.)

14. Claim 12:

- a. 12[a]: ...wherein the network security device makes said determination regarding whether the untrusted file is a clean file that is free of malicious content by first moving the untrusted file from the first repository to a third repository**

Burdett disclosed moving an untrusted file to a third repository in two ways.

First, Burdett described once a file is uploaded to the **cloud data store** (a first repository; **green** in annotated Figure 3 below) and a corresponding notification is issued, the file is copied to a data store of the **scanning service** (a third repository; **red**) to enable the scanning service to process the file for malicious content (that is, making the determination regarding whether the file is a clean file free of malicious content):

Based on a notification of a file event, ***such as a notification of an upload***, download, or change to a file, the scanning service may request a copy of the file to be examined. ***Once the file is copied to the data store of the scanning service*** (along with meta-data to identify the customers' cloud account and cloud data store) the file may be added to the scan queue and process all required rules as part of the scan of the file. (Burdett, 29:31-38.)

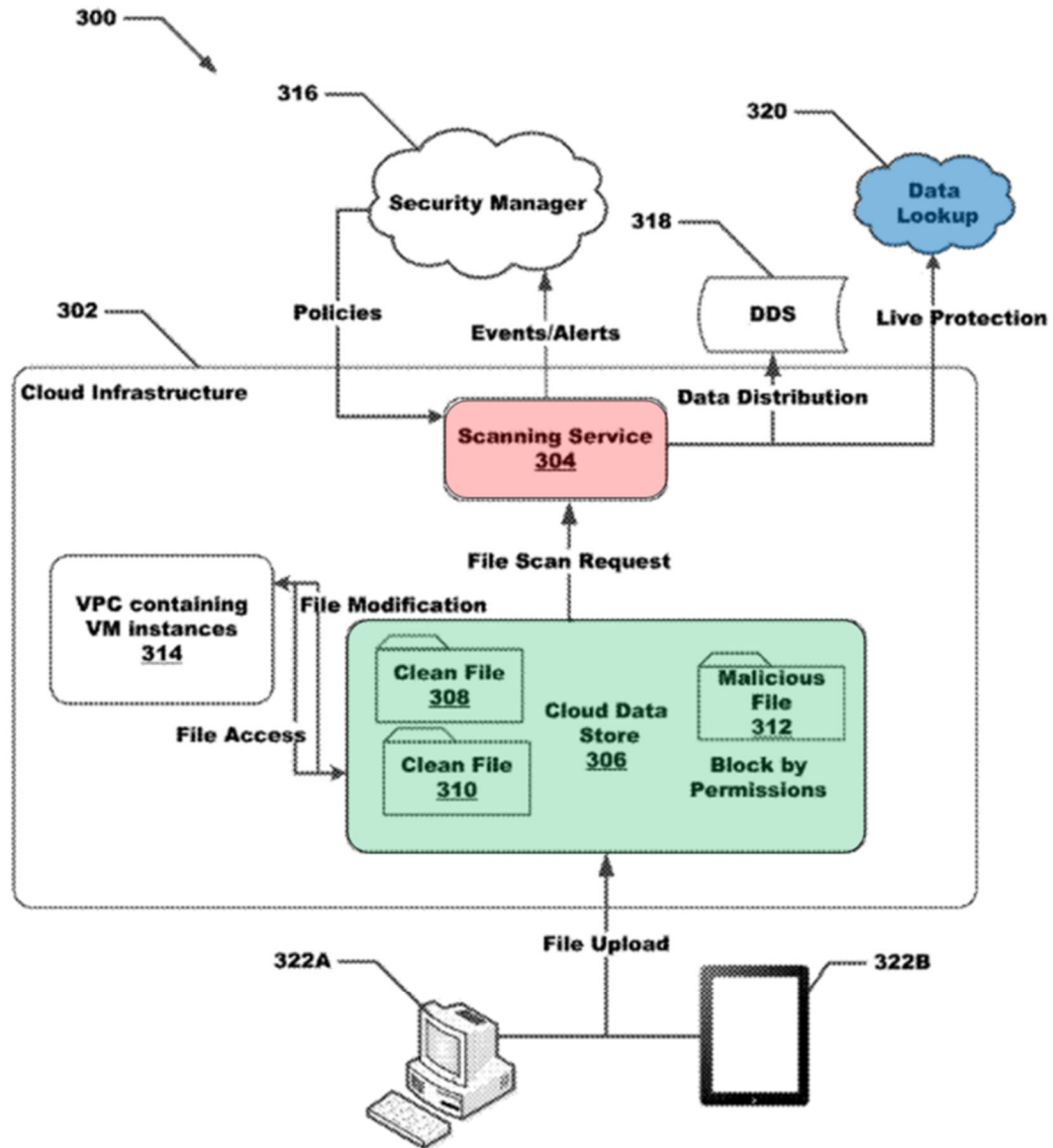


FIGURE 3 (Annotated)

Second, Burdett described the scanning service could employ a “data lookup facility” (blue in annotated Figure 3 above) to further analyze files if the scanning service was unable to determine whether a file was malicious. (*Id.*, 25:7-16.) For example, the data lookup facility would “check patterns identified by the scanning

service 304 [or] determine reputations of resources identified or provided by the scanning service 304,” thereby making a determination regarding whether a file contains malicious content. (*Id.*, 25:4-6.) Burdett’s scanning service would “provide” a file to the data lookup facility for this additional processing. A POSITA would have found it obvious that, as part of this process, the scanning service would move the file into a data repository accessible to the data lookup facility (e.g., a third repository) to enable the facility to analyze the file, because storing the file in a data repository would have enabled the facility to process files in a queue without risking data loss. (Franz, ¶¶125-127.)

Additionally, as described for element 1[c][ii], it would have been obvious for the scanning service disclosed by Burdett to be implemented on a network security device as taught by FortiSandbox. (Franz, ¶128.)

b. 12[b]: applying one or more security checks to the untrusted file in the third repository

With respect to Burdett’s disclosure of a “third repository” in the form of the data store associated with the scanning service (i.e., first mapping of third repository in Element 12[a]), a security check was applied to a file in the scanning service’s data store when the scanning service analyzed the file for malicious content. (Burdett, 1:64-66, 24:51-56, 26:37-41, 29:58-60, 30: 59-62, 31:8-13.)

With respect to the disclosure of a “data lookup facility” that also included a third repository (i.e., second mapping of third repository in Element 12[a]), Burdett

taught that the facility was configured to perform security checks such as checking “patterns identified by the scanning service” or determining “reputations of resources identified or provided by the scanning service.” (*Id.*, 25:3-6.) (Franz, ¶¶129-130.)

- a. **12[c]: moving back the checked file as a clean file into the first repository for onward copying to the second repository or moving the checked file as a clean file directly from the third repository to the second repository.**

Once Burdett’s scanning service of completed its analysis of a file in the “third repository” associated with the scanning service, a POSITA would have found it obvious for the scanning service to move the checked file directly from the third repository to a second repository. For example, Burdett described the scanning service replacing a file in the cloud data store with another, clean file. (Burdett, 25:31-33.) Additionally, Burdett described making a file available to applications after the scanning service’s scan was complete by, *inter alia*, changing permissions of the file to allow access (*id.*, 25:64-67) or moving the file. (*Id.*, 25:26-28, 31:6-7; Franz, ¶131.)

With respect to the data lookup facility, Burdett described the scanning service providing files or data to the facility “for further analysis,” such as “if the scanning service 304 is unable to determine whether a file or a portion of a file is malicious.” (Burdett, 25:7-17.) It would have been obvious for the data lookup facility to return a result of its analysis to the scanning service in order to enable the

scanning service to take an action based on the analysis. (Franz, ¶132.) In addition, once the scanning service received the analysis from the data lookup facility, it would have been obvious for the scanning service to move the checked file directly to the second repository, for the same reasons discussed above. (*Id.*)

It would have been obvious to a POSITA to move cleaned files to a repository that would be accessible to users since moving, renaming and changing file permissions were known alternatives to control access to clean or infected files. (Burdett, 25:25-27; Franz, ¶133.)

B. Ground 2: Combining Sharpe, Seese, and Kamalapuram renders claims 1, 2, 4, 5, 7-10, 12-14, 16, 18-22, 24, and 26-28 obvious.

Sharpe: Sharpe was directed to anti-virus capabilities deployed within a distributed filesystem. (Sharpe, 96:42-58.) Sharpe described a distributed filesystem in which two or more cloud controllers wrote data to a cloud storage system and served files to clients. (*Id.*, 96:59-66, 98:33-46.) At least one of the cloud controllers also initiated anti-virus scans for files written to the cloud storage system. (*Id.*)

Sharpe’s cloud storage system used file metadata to ensure that a file could not be accessed until it had undergone anti-virus scanning. (*Id.*, 2:32-40, 96:59-66, 97:56-58.) A metadata field in each file included “[a] ‘clean’ field that indicate[d] whether a file has been checked,” which was cleared upon upload (indicating the file is not clean) and remained that way until the file was scanned. (*Id.*, 96:61-64; 97:56-

58.) Cloud controllers would not serve a file to a requesting client unless the “clean” field was marked in the file’s metadata. (*Id.*, 2:32-40, 96:61-66.)

As depicted in annotated Figure 42 below, a **cloud controller 4208 (red)** would access a file from the cloud storage system 302 (**green**) that did not have the “clean” metadata field marked. (*Id.*, 97:32-50.) At **operation 5 (blue)**, the **cloud controller** transferred a file to the **anti-virus service 4214 (purple)** for analysis, which generated a scan result based on the anti-virus scan and conveyed the scan result to the cloud controller (**operation 6, olive**). (*Id.*, 97:50-53.) If the file was deemed clean, the cloud controller would update the metadata; if infected, the file could be removed, quarantined, or marked. (*Id.*, 97:56-62.) (Franz, ¶¶134-135.)

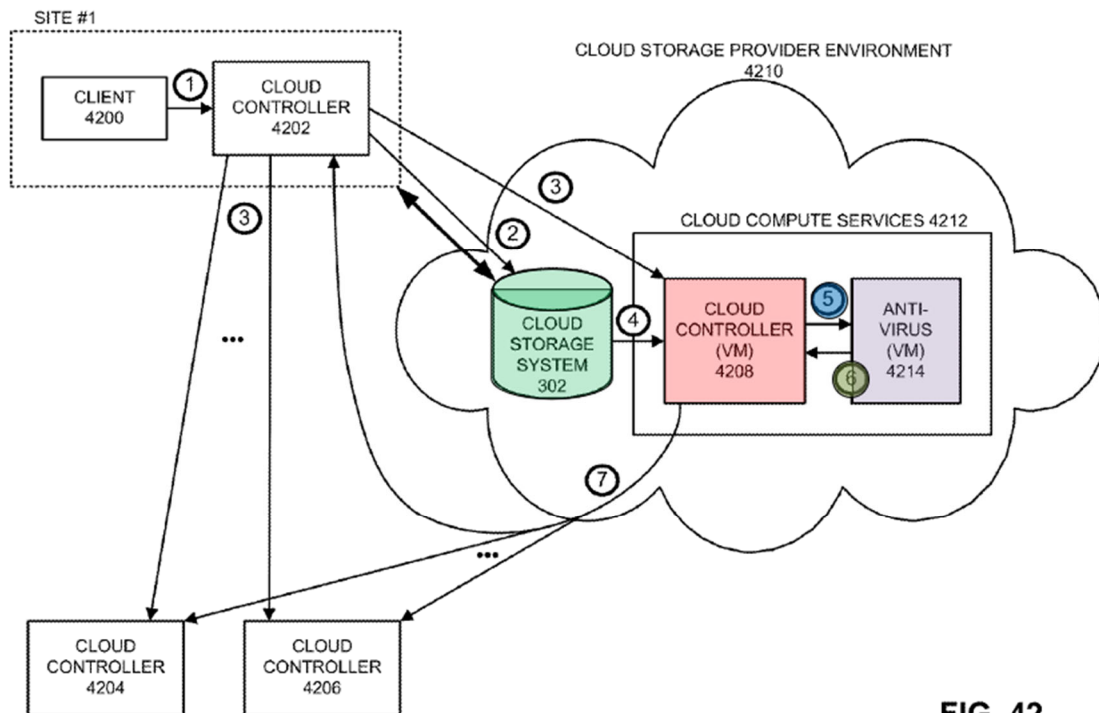


FIG. 42
(Annotated)

Seese: Seese was directed to “integrating a cloud service with information systems of an enterprise.” (Seese, [0002].) In the integrated storage system, an enterprise server (a data loss prevention (DLP) server) mounted user shares of a cloud service and inspected files within the mounted user shares against enterprise policies and rules. (*Id.*, [0063], [0067]-[0068]; Franz, ¶136.)

Kamalapuram: Kamalapuram related to security analysis of files in a network storage system, which can be implemented in a “cloud only system option” wherein a host system is a client of an enterprise network or cloud storage system. (Kamalapuram, [0121]; *see also* [0058].) When the host system uploaded a new file or changed an existing file, Kamalapuram taught that the file could be stored in an “isolated area of the network storage system” until it could be scanned by a “security analysis system.” (*Id.*, [0121], [0122].) Upon a determination that the file was not compromised, the file would be moved to a “general storage area” of the network storage system. (*Id.*, [0122]; Franz, ¶137.)

1. Motivation to combine

A POSITA would have used Seese’s disclosed network security device to perform Sharpe’s malware scan. Additionally, Sharpe’s malware scan would have been used to scan files in a first repository and copy clean files to a second repository, as Kamalapuram taught. A POSITA would have been motivated to combine Sharpe, Seese, and Kamalapuram with a reasonable expectation of success.

First, both Sharpe and Kamalapuram relate to malware detection within shared file repositories. (Sharpe, 96:42-58; Kamalapuram, [0121]-[0122].) Likewise, Seese is directed to scanning shared file repositories for compliance with a policy. (Seese, [0067]-[0068].) Accordingly, a POSITA would have been motivated to combine these complementary disclosures. (Franz, ¶¶138-139)

The Sharpe-Seese-Kamalapuram combination also would have produced a malware detection system for an enterprise that improved security. For example, the Seese's DLP server was a physical device deployed within an enterprise's network infrastructure. (Seese, [0063].) Configuring Seese's server to perform Sharpe's malware scans would have enabled an enterprise to preserve privacy of the enterprise's files stored within a cloud storage system, one of Sharpe's expressly-stated goals. (Sharpe, 98:27-32; Franz, ¶140.)

2. Claim 1

i. 1[pre]:

Sharpe disclosed a secure data transfer system in the form of one or more computing devices that executed a cloud controller. (Sharpe, 97:14-25, 97:67-98:2.) Sharpe's cloud controller can be implemented as either a virtual machine or as a dedicated physical device. (*Id.*, 97:20-24, 97:67-98:2, 98:28). Figure 42, annotated below, illustrated the **cloud controller (red)** configured to access a **cloud storage system (green)** and send files to the anti-virus service, implemented as an “**anti-virus**

virtual machine” (purple), to scan files stored in the cloud storage system. (*Id.*, 97:16-25.)

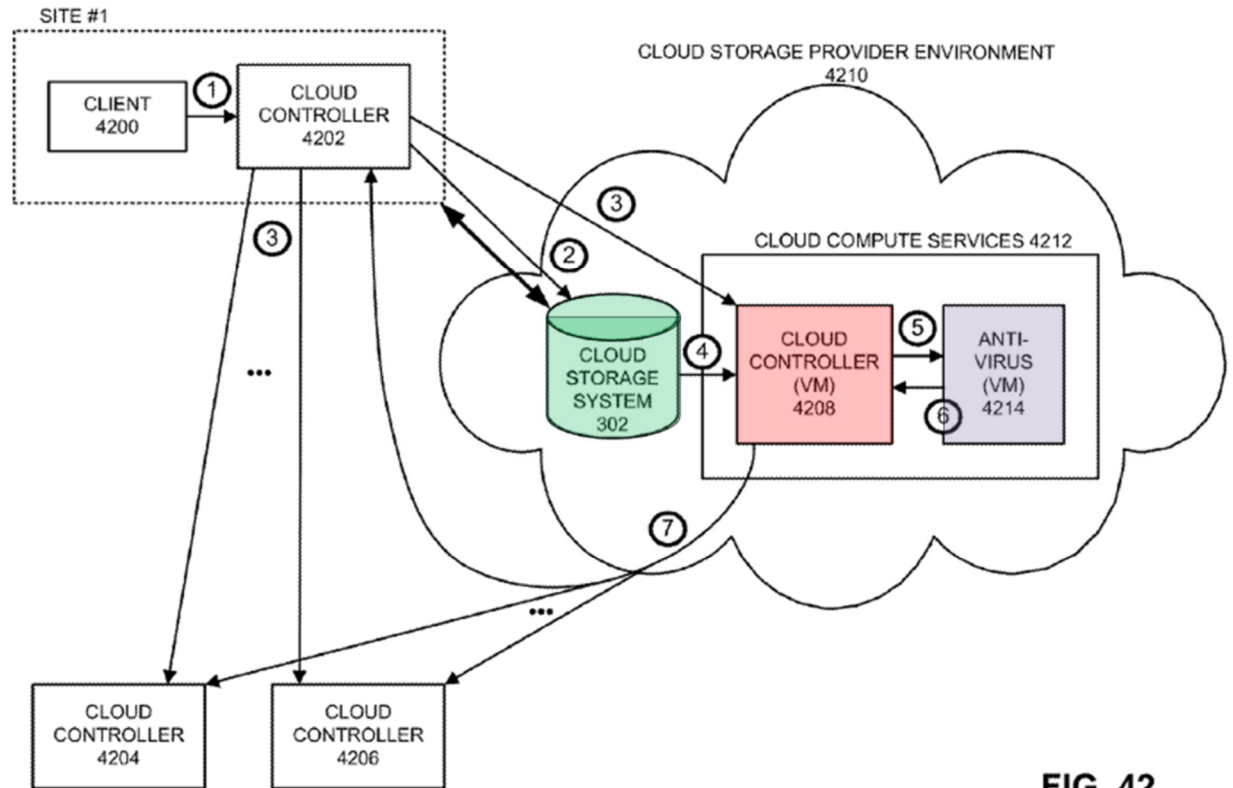


FIG. 42
(Annotated)

In Sharpe’s cloud storage environment, one or more cloud controllers ensured that “every file that is written to the distributed filesystem (and/or every file that is modified) undergoes an anti-virus scan.” (*Id.*, 96:59-61.) A metadata field in each file included “[a] ‘clean’ field that indicate[d] whether a file has been checked,” which was cleared upon upload until the file was subjected to an anti-virus scan. (*Id.*, 96:61-64; 97:56-58.) Cloud controllers would not serve a file to a requesting client unless this “clean” field was marked in the file’s metadata. (*Id.*, 2:32-40, 96:61-66.)

Sharpe’s cloud controller was communicatively coupled to an anti-virus service to perform anti-virus scans on new or modified files in the distributed filesystem. (*Id.*, 96:59-61, 97:17-25.) Upon detecting a file in the cloud storage system, the cloud controller would send the file to the anti-virus service and “[i]f the scan indicated that the file was clean, the clean field in the metadata for the new file ...[would be] updated to indicate this clean status.” (*Id.*, 97:48-62; see also 2:53-61, 97:14-25.)

According to Sharpe, if “the new file [wa]s not marked as clean[,] ...while the other cloud controllers c[ould] indicate the presence of the file to clients, they d[id] not yet allow clients to actually access the new (or modified) file.” (*Id.*, 97:32-36.) Thus, Sharpe’s anti-virus service controlled secure transfer of data in a distributed filesystem by allowing access only to files that were free of viruses. (*Id.*, 96:54-58, 98:43-46; Franz, ¶¶141-144.)

a. 1[a]:

In Sharpe, one embodiment of the cloud controller responsible for causing an anti-virus scan to be performed on files in a cloud storage system was executed as a virtual machine by one or more computing devices. (Sharpe, 98:1-2 and 5-8 (“the virtual machines executing cloud controller 4208 and anti-virus service 4214 may execute on the same server”).) Another embodiment employed a dedicated physical cloud controller. (*Id.*, 98:28-30.) In its discussion of the computing devices/cloud

controller, Sharpe disclosed element 1[a]. (Franz, ¶145.)

Non-transitory storage device having embodied therein one or more routines: Sharpe disclosed that its methods and processes—which included the anti-virus scanning operations—could be performed by execution of instructions stored in a non-transitory computer-readable storage medium. (Sharpe, 6:42-61, *see also* 105:9-10: “[a] non-transitory computer-readable storage medium storing instructions that when executed by a computer cause the computer to perform a method for performing anti-virus checks for a distributed filesystem.”) (Franz, ¶146.)

Enterprise network: Sharpe’s disclosure related to a distributed filesystem usable by an enterprise and accessed by client systems operating on the enterprise’s network. (*Id.*, 96:54-58 (“a distributed filesystem may include anti-virus capabilities”), 100:60-62 (“Clients 910-912 can include any node on a network”), 98:10-32 (distinguishing a cloud storage provider environment from an organization’s network); *see also* 9:34-48 (describing an enterprise storing data in cloud storage provided by an external vendor).)

Seese also described operations related to an enterprise network. For example, Seese disclosed a system to “implement enterprise driven scans of files stored on a cloud service.” (Seese, [0064]; *see also* [0016]; Franz, ¶¶147-148.)

Routines operable to prevent users of an enterprise network from

accessing malware infected files: Sharpe disclosed “a distributed filesystem [that]...include[d] *anti-virus capabilities* that attempt[ed] to prevent infected files from being stored to the system.” (Sharpe, 96:54-56, see also 2:19-31, 96:59-66; Franz, ¶149.) In Sharpe’s system, enterprise clients were not able to access the files until the scan indicated those files were clear of viruses, which is a type of malware (Franz, ¶36), thus preventing clients (users) from accessing infected files. (Sharpe, 2:32-40, 96:43-46, 97:26-36; *see also* 58:28 (*users are an example type of client*), Franz, ¶149.) Sharpe also disclosed that its anti-malware protection methodology can be implemented using software routines, because its “methods and processes” can be implemented on “a dedicated or shared processor that executes a particular *software module or a piece of code*.” (Sharpe, 6:62-7:3.).

Public cloud file store: Sharpe’s malware protection system performed an anti-virus check on files that were written to the cloud storage system (such as **cloud storage system 302** in Figure 42). (*Id.*, 96:59-61; 97:17-36.)

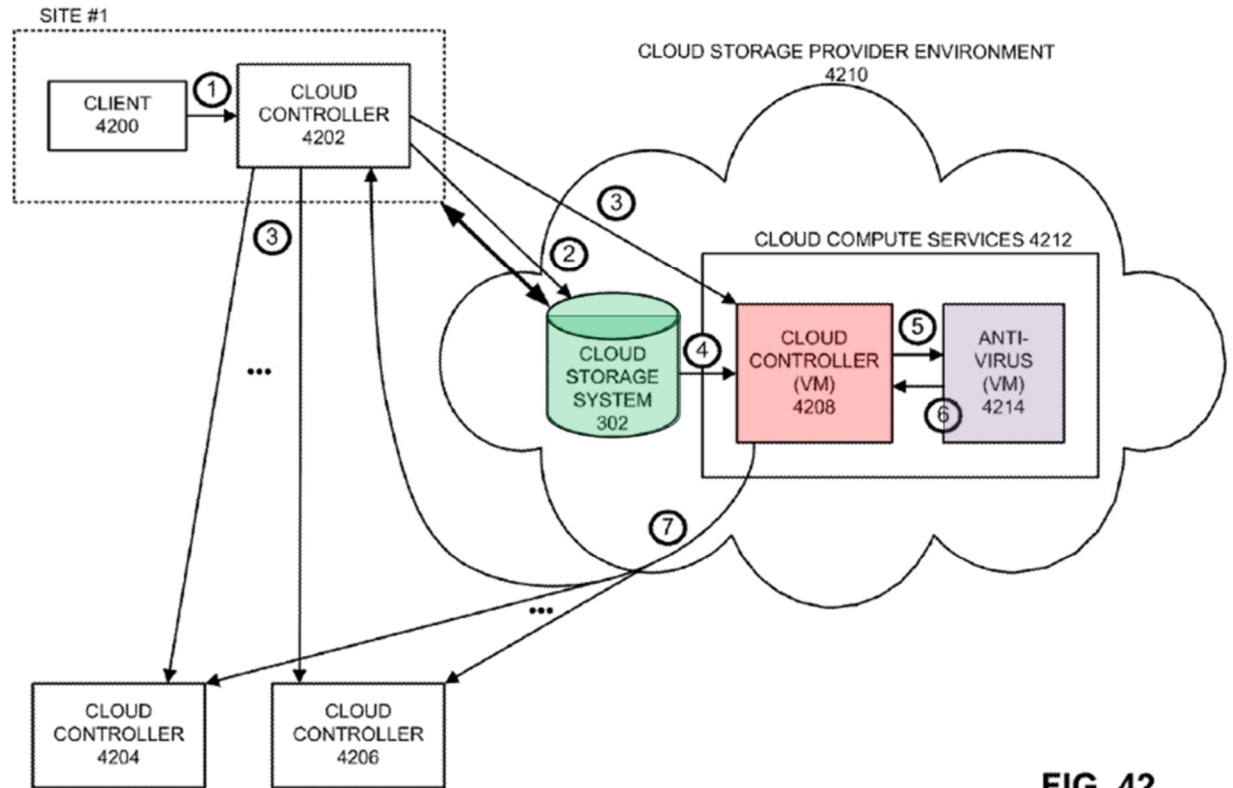


FIG. 42
(Annotated)

A “public cloud file store” generally relates to a file store maintained on shared infrastructure, as opposed to an enterprise’s private infrastructure. (Franz, ¶¶79, 151.) Sharpe suggested that its cloud data stores are part of a public cloud file store. For example, Sharpe explained “that while the cloud storage system store[d] the data for the distributed filesystem, the cloud storage capabilities may be provided by *an external vendor*.” (*Id.*, 9:34-36.) For security reasons, the external vendor was prohibited from “performing...operations that require[d] knowledge of the data and data layout” and “provid[ed] purely storage capabilities.” (*Id.*, 9:41-48.) Therefore, it would have been obvious to a POSITA that the cloud storage system provided by such an external vendor would be a public cloud storage system that

provided data storage space to the organization. (*See id.*, 98:2-32 (distinguishing between a “cloud storage provider environment” and an organization’s network); Franz, ¶¶150-152.)

b. 1[b]:

As discussed above, Sharpe’s cloud controller was implemented either as a virtual machine or as a dedicated physical system. (Sharpe, 98:1-2, 98:5-8, 98:28-30.) Additionally, Sharpe explained that its disclosed “methods and processes” can be implemented on “*a dedicated or shared processor* that executes a particular *software module or a piece of code.*” (*Id.*, 6:62-7:3; see also *id.*, 105:9-13: “[a] non-transitory computer-readable storage medium storing instructions that when executed by a computer cause the computer to perform a method for performing anti-virus checks for a distributed filesystem.”) (Franz, ¶153.)

c. 1[c][i]:

As explained for Elements 1[a] and 1[b], Sharpe disclosed that its virus-scanning operations can be implemented as software routines executed on a processor. (Sharpe, 6:62-7:3, 98:1-2, 98:5-8, 98:28-30, 105:9-13; Franz, ¶154.)

Sharpe’s cloud controller accessed an untrusted file stored on the cloud storage system. (Sharpe, 97:40-51). In doing so, the cloud controller would be retrieving the untrusted file contents from a location on the cloud storage system, which would have been a first repository. (Franz, ¶155.)

For example, in Figure 42's embodiment (annotated below), Sharpe disclosed that at operation 4 (blue) the cloud controller 4208 (red) would access a file from the cloud storage system 302 (green) that did not have the "clean" metadata field marked – *i.e.*, an untrusted file. (Sharpe, 97:32-50; Franz, ¶156.)

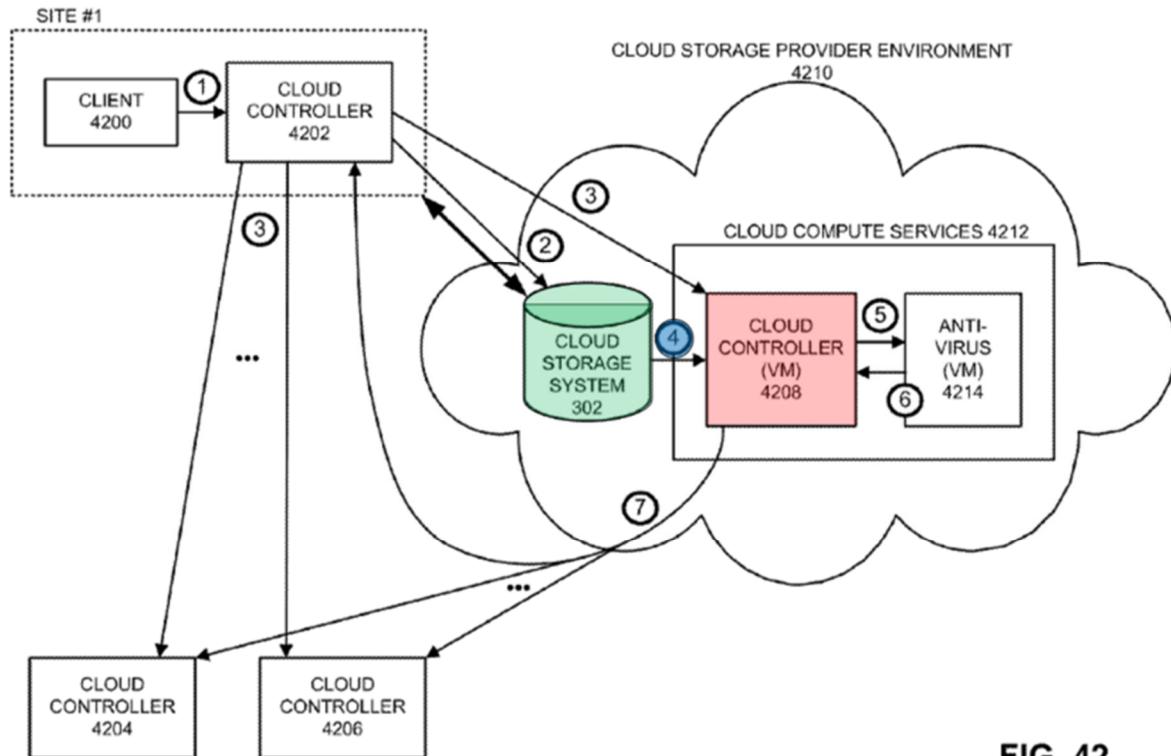


FIG. 42
(Annotated)

A POSITA would have found it obvious that Sharpe's cloud storage system would include a first repository as the repository in which the untrusted file was stored, because isolating a suspicious files or files with unknown trustworthiness was routinely done to prevent potential infection of other files or other systems. (Franz, ¶157.)

To the extent Sharpe did not explicitly disclose such a first repository to store an untrusted file, Kamalapuram did. Specifically, Kamalapuram described a security analysis system implemented in a “cloud only system option” in which a host system was a client of an enterprise network or cloud storage system. (Kamalapuram, [0121]; *see also* [0058].) When the host system uploaded a file to the network storage system or changed an existing file, Kamalapuram taught that the file could be stored in an “isolated area of the network storage system” until it could be scanned by a “security analysis system.” (*Id.*, [0121], [0122].) A POSITA would have found it obvious for Sharpe’s cloud storage system to store files in a first repository as described by Kamalapuram because isolating files prior to an anti-virus scan would have helped achieve Sharpe’s goal of ensuring that infected files could not be accessed. (Sharpe, 96:43-58; Franz, ¶158.) Using an isolated file repository would have been a combination of prior art elements according to known methods, yielding predictable results. (Franz, ¶158.)

d. 1[c][ii]:

Sharpe and Seese disclosed this limitation.

Specifically, Sharpe disclosed that its controller had the capability to mount the distributed filesystem to allow access to the contents of the filesystem:

Cloud controller instance 3102 *provides access to the distributed filesystem by exporting a filesystem mount point 3110 that exposes the structure of and the data stored in the distributed filesystem* to

server 3100 and external clients. For instance, cloud controller instance 3102 ***may include support for NFS (Network File System) protocols, and export an NFS mount point*** that presents the contents of the distributed filesystem to server 3100 and other external devices as if it was an NFS filesystem. More specifically, cloud controller instance 3102 supports an additional interface that enables full, transparent access to the data stored in the distributed filesystem.
(*Id.*, 61:11-22, FIG. 31.)

Notably, Sharpe described using, for example, the network file system (NFS) protocol to mount the filesystem—which is one of the protocols described in the '856 patent for natively mounting the data store. ('856 patent, 11:7-10.) Sharpe also noted that some embodiments of its cloud controllers were configured to communicate by another protocol used in the '856 patent to natively mount a cloud file repository – CIFS (Sharpe, 25:38-42 (describing cloud controllers that were configured to provide a storage system accessible “from multiple front-ends via an application-layer network protocol (e.g., CIFS).”); '856 patent, 11:7-10; Franz, ¶¶159-161.)

Sharpe did not describe mounting the distributed filesystem *to a network security device*, and did not discuss mounting the filesystem in the context of scanning the files (e.g., in Figure 42). However, Sharpe nevertheless provided a motivation or suggestion to a POSITA for performing this operation, making it obvious to combine Sharpe with Seese, which not only described mounting a public

cloud file system in the context of scanning the files to protect an enterprise against data loss, but also described mounting the public files on a *network security device*. Specifically, Seese disclosed a data loss prevention (DLP) server as an example type of a network security device. (Seese, [0063], [0067]-[0068].)⁸ The server described by Seese was part of an “integrated storage system” that “implement[ed] enterprise driven scans of files stored on a cloud service.” (*Id.*, [0064].) A user share of a cloud service (containing files stored on the cloud service (*id.*, [0063])) was mounted on the server to “make[] the user share accessible through the file system of the operating system executing on the DLP discover server.” (*Id.*, [0067].) It would have been obvious to a POSITA to use well-known network filesystem protocols, such as NFS or CIFS as described by Sharpe in other contexts, to mount the user share to the DLP server described by Seese. A POSITA would have understood that configuring Seese’s server to communicate by NFS, CIFS, or another filesystem protocol natively employed by the cloud storage system maintaining the user share, the user share would have been “natively mounted” to the server. Such protocols were widely used before the ’856 patent’s priority date, and would have enabled the

⁸ Seese’s disclosure of a DLP server is consistent with the ’856 patent, which describes a “network security device” providing features such as “content filtering [or] data leak prevention.” (’856 patent, 5:8-12.)

server to use native cloud APIs to promptly detect directory changes in the user share. (Franz, ¶162.)

The server described by Seese was a network security device protecting an enterprise network, for example because it scanned files in the mounted user share against data loss prevention rules to protect the enterprise against data loss. (*Id.*, [0018], [0063].)

It would have been obvious to a POSITA to natively mount a public cloud file store, as taught by Seese and suggested by Sharpe, on a network security device as taught by Seese, as a means to enable a network security device to access files in the cloud file store and to natively communicate with the cloud file store to perform actions such as controlling access to the files. (Franz, ¶¶163-164.)

e. 1[c][iii]:

As explained in connection with Element 1[c][i], the combination of Sharpe and Kamalapuram taught or suggested this limitation.

In Sharpe, users did not have read access to files in the cloud storage system until the file was scanned by the anti-virus service and the “clean” metadata field marked. (Sharpe, 96:64-66, 97:32-36.) Additionally, Kamalapuram disclosed a file being stored in an “isolated area” of a network storage system before the file had been scanned by a security analysis system, preventing user access to the file until the file had been determined to be clean. (Kamalapuram, [0122].) It would have

been obvious to a POSITA to modify Sharpe's disclosure to store files in a first repository to which users do not have read access, as taught by Kamalapuram, because this modification would have achieved the same goal as taught by Sharpe (preventing access to untrusted files) while improving usability of the cloud storage system for end users. For example, Sharpe described that cloud controllers could indicate the presence of an untrusted file to clients even when the file could not be served to the clients, which would have been frustrating to users of the clients. (Sharpe, 97:33-36.) To remedy this frustration, a POSITA would have been motivated to instead configure a repository to which users do not have read access, as disclosed by Kamalapuram, because doing so would have hidden inaccessible files from view of users until those files had been scanned and approved for access. (Franz, ¶¶165-166.)

f. 1[d]:

Sharpe and Seese disclosed this limitation.

As explained in Elements 1[a] and 1[b], Sharpe disclosed that its virus-scanning operations can be implemented as software routines executed on a processor. (Sharpe, 6:62-7:3, 98:1-2, 98:5-8, 98:28-30, 105:9-13.) Sharpe's cloud controller caused the anti-virus service to apply a security check (e.g., an anti-virus scan) to an untrusted file (e.g., a file for which the "clean" metadata field is empty). (*Id.*, 2:26-31, 2:56-61, 96:54-58, 97:48-53, 98:40-46.) For example, as

shown in Figure 42 annotated below, the **cloud controller 4208 (red)** transferred a file to the **anti-virus service 4214 (purple)** for analysis (**operation 5, blue**), which generated a scan result based on the anti-virus scan (a determination regarding whether the file is free of malicious content) and conveyed the scan result to the cloud controller (**operation 6, olive**) to indicate the file to be clean or infected.

(*Id.*)

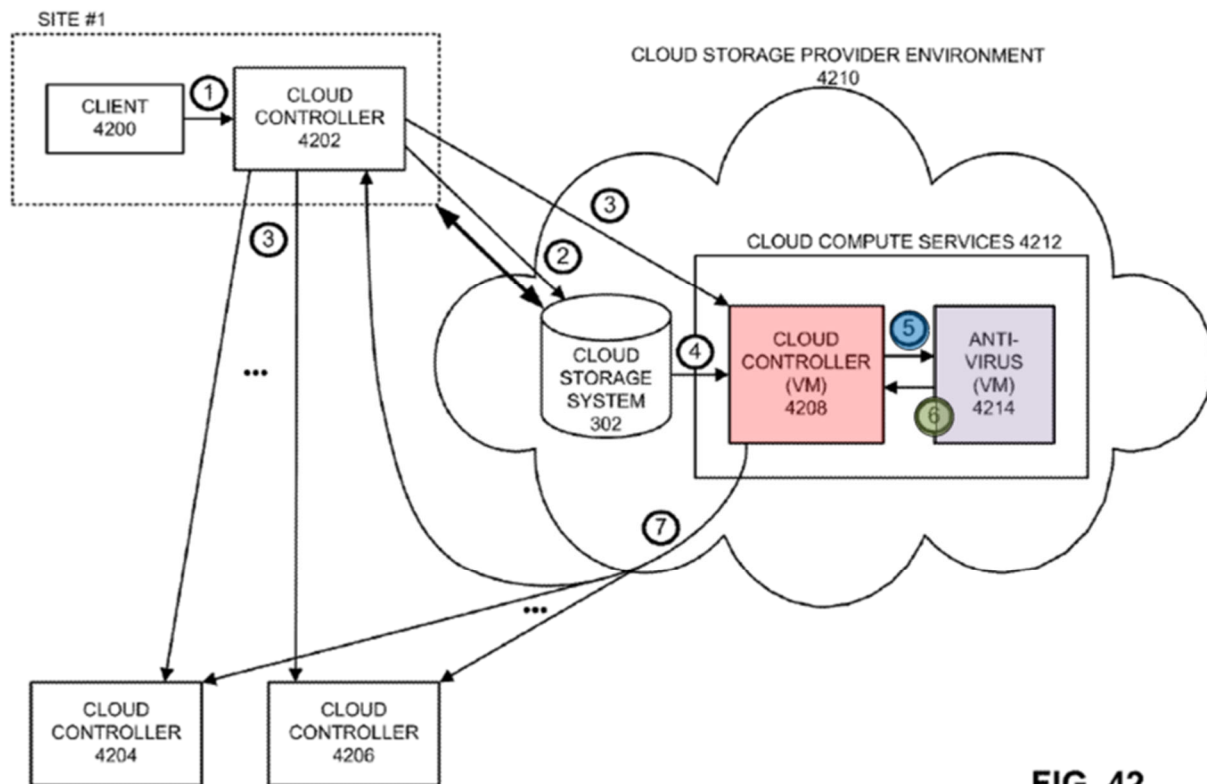


FIG. 42
(Annotated)

As described with respect to element 1[c][ii], Seese disclosed a network security device (e.g., a data loss prevention server) that scanned files for compliance with a policy. (Seese, [0067]-[0068].) A POSITA would have found it obvious to implement Sharpe's anti-virus service on Seese's network security

device for the reasons discussed for element 1[c][ii], and because this modification would have been a simple substitution of known elements with predictable results. Additionally, network security devices that performed multiple security options were well known. (Franz, ¶¶167-169.)

g. 1[e]:

Sharpe and Kamalapuram disclosed this limitation.

As explained in Elements 1[a] and 1[b], Sharpe disclosed that its virus-scanning operations can be implemented as software routines executed on a processor. (Sharpe, 6:62-7:3, 98:1-2, 98:5-8, 98:28-30, 105:9-13.) This included the cloud controller, “which execute[d] as a virtual machine in cloud compute services” (Sharpe, 97:23-24) and made a clean file available to client systems – and thus users of those client systems – after the anti-virus service’s scan of the file was complete and the file determined to be clean. (Sharpe, 2:32-40, 96:61-66, 97:56-58; *see also* 58:28 (users are an example type of client), Franz, ¶171.) Such access was enabled in Sharpe’s disclosure by modifying the file’s metadata to mark the “clean” field, which allowed cloud controllers to serve the file to requesting clients. (Sharpe, 97:56-58, 96:61-66.)

Sharpe did not explicitly describe that the “clean” file would be copied to a second repository, but Kamalapuram did. In Kamalapuram, a “file may be stored *in an isolated area of the network storage system* while waiting to be scanned, and

may be *moved to a general storage area* only after the file has been scanned by a security analysis system of the network storage system.” (Kamalapuram, [0122].) Accordingly, Kamalapuram described moving the “clean” file from a first repository (an isolated area of the network storage system) to a second repository (a general area of the network storage system). (*Id.*) Although Sharpe disclosed using file metadata to indicate that a file is clean, and is thus accessible to the users, a POSITA would have been guided by the teachings of Kamalapuram, and would have found it obvious, to copy the clean file into a separate repository to make it accessible to the users. This modification would have been one of a limited number of alternatives to make a clean file accessible to the users (the others being changing the name or metadata of the file), and would have achieved the same goal as taught by Sharpe while improving usability of the cloud storage system for end users as discussed in element 1[c][iii]. A POSITA would have been further motivated to make this combination because it would have been more efficient for the cloud storage system to manage access to files by repository-level permissions rather than file-level permissions. (Franz, ¶172.)

3. Claim 13

g. 13[pre]:

Sharpe disclosed a method for performing anti-virus scans within a distributed filesystem. (Sharpe, 2:26-52, 98:33-46; Franz, ¶173.)

h. 13[a][i]: receiving, by a network security device that is protecting an enterprise network, a notification, via an Application Programming Interface (API) call from a notification service monitoring a first repository of a public cloud file store, regarding existence of an untrusted file stored within the first repository

The Sharpe-Seese-Kamalapuram combination rendered this feature obvious. In general, Sharpe described a cloud controller that would monitor a cloud storage system (e.g., a public cloud file store) to detect existence of an untrusted file (e.g., a file for which a “clean” metadata field is empty) stored to the cloud storage system and, upon detecting an untrusted file, send the file to an anti-virus service for analysis. (Sharpe, 97:39-51.)

As described with respect to element 1[c][ii], the combination of Sharpe with Seese rendered obvious a “network security device that is protecting an enterprise network.”

The Sharpe-Kamalapuram combination taught or suggested “an application programming interface (API) call from a notification service monitoring a first repository of a public cloud file store.” Kamalapuram, like Sharpe, disclosed security scans performed on files in a network storage system. (Kamalapuram, [0121]-[0122].) In Kamalapuram’s configuration, an untrusted file (e.g., a file that had been modified) was stored in an “isolated area of the network storage system” (a first repository) until the file could be scanned by a “security analysis system.” (*Id.*) Kamalapuram also disclosed various APIs in a cloud storage environment,

including a set of APIs used by an “access manager” to receive “requests to create, modify, or destroy files” within a cloud storage repository (*id.*, [0061]) or an API used by the access manager to “register with the security scanner...to request or retrieve security scan reports.” (*Id.*, [0126].) It would have been obvious to implement the communications between Sharpe’s cloud controller and the anti-virus service as API calls, since Kamalapuram demonstrates it was well-known to use APIs to interface between systems and services associated with a cloud storage environment. (Franz, ¶176.) Thus, a POSITA would have found it obvious for Sharpe’s cloud controller to operate as a notification service monitoring a first repository of a public cloud file store. The cloud controller would have generated API calls to the network security device rendered obvious by the combination of Sharpe and Seese. (Franz, ¶¶174-176.)

i. 13[a][ii]:

See element 1[c][ii].

j. 13[a][iii]:

See element 1[c][iii].

k. 13[b]:

See element 1[d]

l. 13[c]:

See element 1[e].

4. Claim 21

f. 21[pre]:

As described with respect to element 1[a], Sharpe disclosed a non-transitory computer-readable storage medium that embodied instructions for execution by one or more processors. (Sharpe, 6:42-61, 105:9-10.)

As described with respect to element 13[a][i], the combination of Sharpe with Seese rendered obvious a “network security device” for performing the steps recited in claim 21. (Sharpe, 97:25, 98:21-32; Seese, [0018], [0063], [0067]-[0068]; Franz, ¶¶181-182.)

g. 21[a][i]:

See element 13[a][i].

h. 21[a][ii]:

See element 1[c][ii].

i. 21[a][iii]:

See elements 1[c][ii] and 1[c][iii].

j. 21[b]:

See element 1[d]

b. 21[c]:

See element 1[e].

5. Claims 2/14/22:

The combination of Sharpe, Kamalapuram and Seese rendered this claim obvious.

As explained in elements 1[c][ii] and 1[d], it would have been obvious to use Seese's DLP server (network security device) to implement Sharpe's anti-virus scanning operations. Kamalapuram further disclosed various configurations of a server including as a firewall and a gateway (i.e., a "network gateway device"):

Server 206 may be configured as any type of server, as needed, e.g., a file server, an application server, ... **a gateway**, an application gateway, a gateway server, ... **a firewall**, a web server, ... or a server executing an application acceleration program that provides firewall functionality...Other server types may also be used.

(Kamalapuram, [0045].)

Kamalapuram further described threat detection services that included **intrusion detection**. (*Id.*, [0076]; Franz, ¶¶188-190.)

It would have been obvious to modify Sharpe's disclosure to execute the anti-virus service on a network security device (*see* elements 1[c][ii] and 1[d]) that comprised any or a combination of a sandbox device, a network controller, a firewall, a network gateway device, or an IDS, since Kamalapuram described the use of such devices in the context of security analysis for shared files in a cloud file management system. (Kamalapuram, [0001], [0014] ("an illustrative remote-access system architecture that may be used in accordance with one or more illustrative aspects described herein"); Franz, ¶191.)

Furthermore, a POSITA would have found this combination obvious because

the limitations recited in claims 2/14/22 constituted routine and complementary techniques/components for protecting an enterprise from intrusions and contaminations. These techniques were well-known for many years prior to the '856 patent's 2018 priority date and were routinely integrated into a single security device. (Franz, ¶192.)

6. Claim 4:

Sharpe's anti-virus service was implemented as a virtual machine that determined whether an untrusted file (e.g., a file for which the "clean" metadata field is empty) was free of malicious content. (Sharpe, 2:26-31, 2:56-61, 96:54-58, 97:25, 98:40-46.)

Kamalapuram disclosed a security device that executed a sandbox in a virtual machine (a "sandbox appliance") to scan a file for vulnerabilities. (Kamalapuram, [0129].)

In the combination, Kamalapuram's sandbox appliance would control the virtual machine in which Sharpe's anti-virus service operated. A POSITA would have found this combination obvious because configuring the appliance executing Kamalapuram's sandbox virtual machine to additionally or alternatively execute Sharpe's anti-virus service virtual machine would have been a simple substitution of one known element for another. (Franz, ¶¶193-195.) A POSITA also would have been motivated to add a sandbox to Sharpe's anti-virus service virtual machine in

order to “mitigate system failures or software vulnerabilities from spreading,” as taught by Kamalapuram. (Kamalapuram, [0129]; Franz, ¶¶193-195.)

7. Claims 5/16/24:

Sharpe disclosed various known protocols for writing data to or reading data from a remote filesystem, including NFS (Sharpe, 9:8 and 25:6), CIFS (*id.*, 25:6-7), and SAN using iSCSI (*id.*, 8:4-5). Sharpe also described filesystems that supported local storage such as USB flash drives. (*Id.*, 25:54-59.)

Seese disclosed a network security device (see element 1[c][ii]), while Kamalapuram disclosed a second repository (see element 1[d]).

Additionally, it would have been obvious to implement the network security device of the Sharpe/Seese/Kamalapuram combination to copy files to one or more repositories using a combination of the protocols recited in claims 5/16/24. Sharpe’s distributed filesystem, for example, combined “cloud controllers with NAS [network-attached storage] capabilities and cloud-based storage” to provide a storage system accessible “from multiple front-ends via an application-layer network protocol (e.g., CIFS).” (*Id.*, 25:38-42.) Thus, since Sharpe’s storage system was accessible from multiple front-ends and Sharpe disclosed transferring files from a storage system by each of NFS, CIFS, SAN using iSCSI, and local storage, it would have been obvious for the network security device of the combination to communicate by any of these protocols when copying a file to the second repository.

(Franz, ¶¶196-198.)

8. Claims 7/18/26:

In Sharpe’s disclosure, the cloud controller was capable both of quarantining a file that was found to be infected with malicious content as well as removing the infected file (i.e., deleting the file). (Sharpe, 97:58-60: “...remove the file, quarantine the file.”) (Franz, ¶199.)

9. Claims 8/19/27:

Kamalapuram disclosed a second repository in a network storage system, as described with respect to element 1[e], to which files would be moved after undergoing a security scan. (Kamalapuram, [0122].) Kamalapuram’s second repository was located in a “general storage area” of the network storage system to which the files were uploaded, while a first repository was located in an “isolated area” of the network storage system. (*Id.*) Thus, Kamalapuram taught or suggested that the first and second repositories were in the same storage system. However, it would have been obvious to POSITA to locate the second repository taught by Kamalapuram on a network security device or an external storage device in the enterprise network. For example, as explained by Seese, cloud storage was configured in a “hybrid” model in which “some critical data reside[d] in the enterprise’s private cloud while other data [wa]s stored and accessible from a public cloud storage provider.” (Seese, [0006].) Locating the second repository on the

network security device or on an external storage device in an enterprise network would have therefore been within the capability of a skilled artisan and would have simply been an implementation decision by an enterprise network administrator based on, for example, the importance of the data. A POSITA would have been motivated to use such a hybrid cloud storage model in which the second repository is located on the network security device or an external storage device, since this configuration would have improved security of files associated with the enterprise. (Franz, ¶200.)

10. Claims 9/20/28:

Kamalapuram disclosed a second repository in a network storage system, as described with respect to element 1[e], to which files would be moved after undergoing a security scan. (Kamalapuram, [0122].) Kamalapuram taught that the second repository was located in a “general storage area” of the network storage system to which the files were uploaded, while a first repository was located in an “isolated area” of the network storage system. (*Id.*) Thus, Kamalapuram taught or suggested locating the first and second repositories in the same storage system. Since Kamalapuram also taught that the network storage system could be implemented as a “cloud only system option” in which a host system operates as a client of a cloud storage system (*see id.*, [0058], [0121]), it would have been obvious for both the first and second repositories to form part of a public cloud file store.

(*Id.*; Franz, ¶201.)

11. Claim 10:

Sharpe described that cloud storage capabilities could have been provided by a vendor external to an enterprise (third-party provider) responsible for storing data in the cloud data store. (Sharpe, 9:35-36; *see also* 98:21-32 (describing a cloud storage provider environment that is not trusted by an enterprise); Franz, ¶202.)

12. Claim 12:

c. 12[a]:

As described for element 1[d], Sharpe disclosed causing the anti-virus service to make a determination regarding whether an untrusted file is a clean file that is free of malicious content. (Sharpe, 2:26-31, 2:56-61, 96:54-58, 97:48-53, 98:40-46.) Additionally, Kamalapuram disclosed a first repository that contained the file prior to scanning, and a second repository where the clean files were placed. (*See* elements 1[c][i] and 1[c][iii].)

As part of Sharpe’s scanning procedure, the “cloud controller 4208 accesse[d] the data blocks for the new file from cloud storage system 302 (operation 4) *and the files [we]re transferred* to anti-virus service 4214 (operation 5).” (*Id.*, 97:48-51.) Annotated Figure 42 below illustrates the **cloud controller** (red) transferring a file to the **anti-virus service** (purple) for analysis (**operation 5**, blue). (Franz, ¶204.)

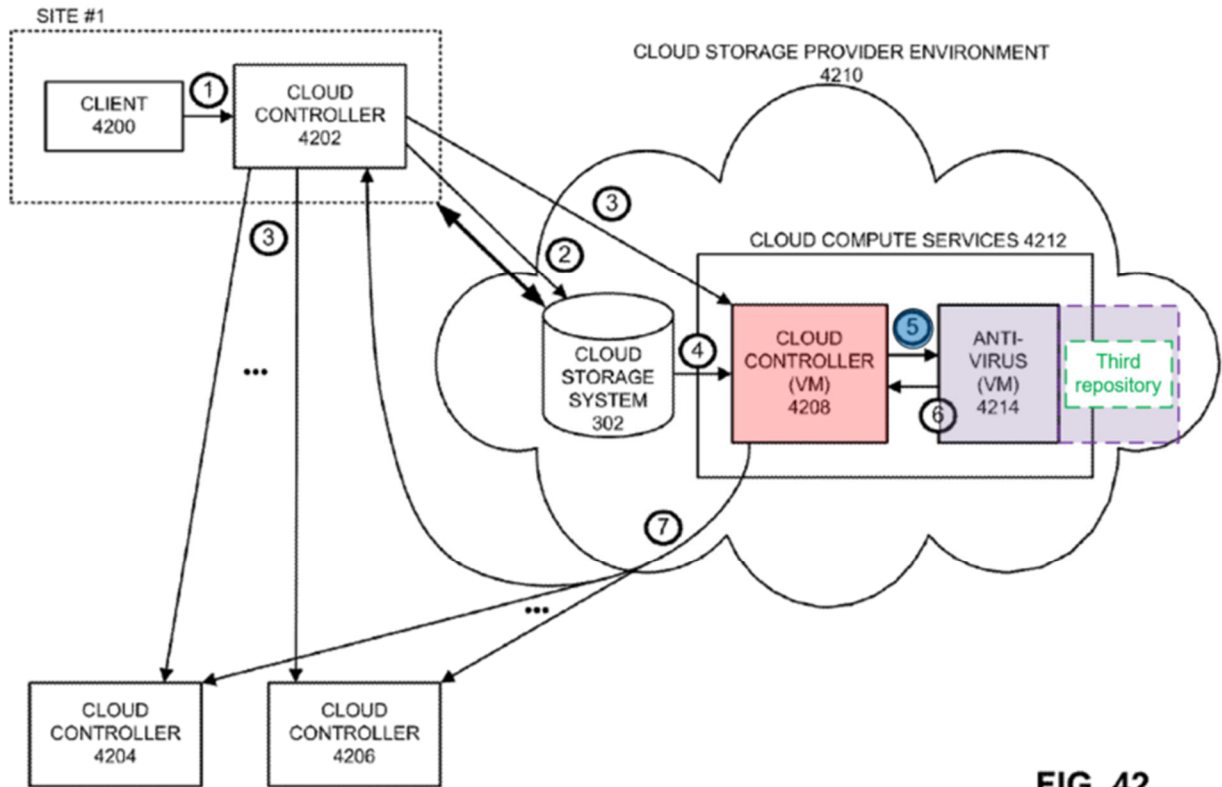


FIG. 42
(Annotated)

It would have been obvious to a POSITA that a file that is provided to the anti-virus service 4214 would be placed in a repository (**third repository – green dashed box added by Petitioner to Figure 42 above**) at or associated with the anti-virus service, since storing files to be scanned in such a repository would have enabled the anti-virus service to process files in a queue without risking data loss. (Franz, ¶205.) Thus, in view of Sharpe’s teachings, a POSITA would have found it obvious to move an untrusted file from a first repository in the cloud storage system to a third repository associated with the anti-virus service. (*Id.*)

d. 12[b]:

Sharpe described the anti-virus service applying one or more security checks

(e.g., an anti-virus scan) to the file. (Sharpe, 2:28-31, 98:40-46; Franz, ¶206.)

b. 12[c]:

The Sharpe-Kamalapuram combination rendered obvious at least “moving the checked file as a clean file directly from the third repository to the second repository.” As explained earlier, in the Sharpe-Kamalapuram-Seese combination, the “clean” file would have ultimately been placed into the second repository. (*See* element 1[e].) Since it was well-known for anti-virus programs to eliminate viruses (*see* Ex. 1010), it would have been obvious that in this combination the anti-virus service would remove any viruses it detected to produce a clean file residing at its storage device (third repository), and then to move the cleaned file to the second repository so that it is available to users. (Franz, ¶207.)

This operation would have been obvious in view of Kamalapuram’s disclosure of moving the files into the second repository after the file had been scanned. (Kamalapuram, [0122].) A POSITA would have found it obvious to modify Sharpe’s cloud controller to move a clean file into a second repository, as disclosed by Kamalapuram, for the reasons discussed in element 1[e]. (Franz, ¶208.)

C. Ground 3: Combining Sharpe, Seese, Kamalapuram, and Dods renders claims 3, 6, 11, 15, 17, 23, and 25 obvious.

1. Motivation to combine

Like Sharpe and Kamalapuram, Dods was directed to processing files for malware identification. (Dods, Title, Abstract.) Accordingly, a POSITA would

have been motivated to combine these complementary disclosures. (Franz, ¶210.)

Adding Dods to the Sharpe-Seese-Kamalapuram combination also would have improved the resulting malware detection system. For example, Dods described additional mechanisms to detect malware in a file that were not taught by the other references. A POSITA would have been motivated to use Dods to supplement the Sharpe-Seese-Kamalapuram combination in order to improve its malware detection capabilities. (Franz, ¶211.)

2. Claims 3/15/23

c. 3[a]:

The Sharpe-Seese-Kamalapuram combination did not disclose the features of claim 3/15/23, but Dods did. Dods disclosed a security platform that “identif[ied] [a] malicious payload within [a] file using a secure environment (e.g., a sandbox).” (Dods, 2:5-7; *see also* 4:40-42.) As taught by Dods, the security platform “include[d] one or more devices.” (*Id.*, 4:28.) Thus, Dods disclosed a sandbox appliance. (Franz, ¶¶212-213.)

d. 3[b]:

Dods’ security platform performed behavior-based analysis to detect malware in files. (Dods, 10:16-26 (“security platform 210 may conduct...a behavior-based analysis (analyzing results of opening the secondary file)”).) (Franz, ¶214.)

3. Claims 6/17/25: ... wherein the untrusted file processing module further removes any malware from the untrusted file that is detected by the one or more security checks when the

result of said determining is negative.

The Sharpe-Seese-Kamalapuram combination did not disclose the features of claims 6/17/25, but Dods did. Dods described the security platform as being configured to “scrub the received file to remove the malicious object” after detecting malware. (Dods, 11:42-48.) (Franz, ¶215.)

4. Claim 11:

Dods disclosed the features of claim 11. For example, Dods described the security platform determining whether a file is free of malicious content using at least “a signature-based analysis” (file signature; Dods, 10:18) and “a static analysis (analyzing the secondary file without opening the secondary file)” (file pre-scan parameters; *id.*, 10:17-18.) (Franz, ¶216.)

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*

Kristen Schunter, Reg. No. 76,519

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)

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 13,825 words, excluding the parts exempted by 37 C.F.R. §42.24(a).

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*
Kristen Schunter, Reg. No. 76,519

Attorneys for Netskope, Inc.

CERTIFICATE OF SERVICE

The undersigned hereby certifies that true copies of this *PETITION FOR INTER PARTES REVIEW* and supporting materials (Exhibits 1001-1010 and Power of Attorney) have been served via USPS Express Mail and electronically in their entirety on counsel for patent owner as follows:

Law Office of Dorian Cartwright
P.O. Box 6629
San Jose, CA 95150
UNITED STATES
info@cartwrightesq.com

In addition, the undersigned certifies that a true copy of the petition and has been served on Patent Owner via email to its legal counsel:

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*
Kristen Schunter, Reg. No. 76,519

Attorneys for Netskope, Inc.