

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

Fortinet, Inc.,

Petitioner,

v.

Netskope, Inc.,

Patent Owner.

Case No. 2026-00031

U.S. Patent 8,635,697

PETITION FOR *INTER PARTES* REVIEW

Mail Stop "PATENT BOARD"

Patent Trial and Appeal Board

U.S. Patent Trademark Office

P.O. Box 1450

Alexandria, VA 22313-1450

TABLE OF CONTENTS

	<u>Page</u>
TABLE OF ABBREVIATIONS	viii
EXHIBIT LIST	ix
I. INTRODUCTION	1
II. GROUNDS FOR STANDING (37 C.F.R. §42.104(a)).....	2
III. THE PATENT AND STATE OF THE ART	2
A. TCP/IP Stack	2
B. OS Fingerprinting.....	5
C. Malware Signatures.....	6
D. '697 Patent	6
E. File History.....	9
F. Challenged Claims	9
G. POSITA.....	10
IV. Claim Construction.....	11
V. SUMMARY OF CHALLENGE AND RELIEF REQUESTED.....	11
A. 37 C.F.R. §42.104(b)(1): The Challenged Claims	11
B. 37 C.F.R. §42.104(b)(1): Prior Art Overview and Specific Grounds of Rejection	12
1. Deridder	12
2. Tuvell	12
3. Lee.....	12
C. Grounds of Rejection	12

VI.	GROUND 1 – Claims 1-25 are Rendered Obvious by Deridder Over Tuvell and Lee	13
A.	Overview of Deridder	13
B.	Overview of Tuvell	14
C.	Overview of Lee.....	16
D.	Motivation to Combine Deridder with Tuvell and Lee.....	17
1.	Motivation to Combine Deridder with the Malware System of Tuvell and Lee	17
2.	Motivation to Combine the Malware Systems of Tuvell and Lee.....	20
E.	Claim 1	22
1.	1[pre]: "A method of network based malware detection in a service provider network, the method comprising:"	22
2.	1[a]: "receiving one or more transmission control protocol (TCP) packets originating from an access device coupled to the service provider network, the one or more TCP packets defining a TCP session between a computing device coupled to the access device, and a destination coupled to the service provider network;"	24
3.	1[b]: "determining an operating system identifier (OS ID) associated with the TCP session and the computing device;"	25
4.	1[c]: "determining if malware is present in the TCP session and an associated malware ID by comparing a malware signature to the one or more TCP packets; and"	28
5.	1[d]: "generating an alert identifying a network address associated with the access device, the malware ID and the OS ID associated with TCP session that generated the alert."	29

F.	Claim 2: "The method of claim 1 wherein determining the operating system ID further comprises:"	31
1.	2[a]: "determining a protocol associated with the TCP packets and matching an OS fingerprint from one or more protocol parameters if present in the TCP packets to determine a first operating system identifier (ID);"	31
2.	2[b]: "determining an application parameter associated with the TCP session if present, and identifying a second OS ID from the application parameter; and"	32
3.	2[c]: "wherein the OS ID in the alert comprises one of the first OS ID or second OS ID associated with one or more computing devices coupled to the access device."	33
G.	Claim 3: "The method of claim 2 further comprising sending a notification to a subscriber associated with the network address associated with the access device, the notification identifying a remediation portal to remove the determined malware in the alert."	33
H.	Claim 4: "The method of claim 3 further comprising:"	35
1.	4[a]: "identifying at the remediation portal the subscriber accessing the remediation portal;"	35
2.	4[b]: "identifying the OS associated with the computing device used by the subscriber to access the remediation portal;"	35
3.	4[c]: "comparing the OS associated with the computing device to the at least one of the first OS ID or second OS ID identified in the alert; and"	36
4.	4[d]: "providing malware remediation to the computing device when the at least one of the first OS ID or second OS ID matches the OS associated with the computing device."	36

I.	Claim 5: "The method of claim 3 wherein if the OS of the computing device is not the at least one of the first OS ID or second OS ID identified in the alert an indication is provided to the subscriber to connect to the remediation portal with another computing device connected to the access device."	37
J.	Claim 6: "The method of claim 3 wherein the alert is cleared when all computing devices associated with the access device and the computing device having the at least one of the first OS ID or second OS ID have accessed the remediation portal."	38
K.	Claim 7: "The method of claim 2 wherein determining the first OS ID is performed by comparing one or more protocol parameters of the one or more TCP packets having a SYN flag associated with the session, wherein the OS fingerprints identify the protocol parameters of the SYN flagged packet and the OS ID associated with parameters."	38
L.	Claim 8: "The method of claim 2 wherein determining the application parameter associated with the session and identifying the second OS ID from the application parameter by inspecting an HTTP user agent string to determine the second OS ID."	39
M.	Claim 9: "The method of claim 2 further comprising verifying the first OS ID against the second OS ID, wherein if the second OS ID does not match the first OS ID the session is identified by the OS ID identified as a more reliable OS ID."	40
N.	Claim 10: " The method of claim 2 further comprising verifying the first OS ID against the second OS ID, wherein if the second OS ID matches the first OS ID either OS ID is used."	41
O.	Claim 11: "The method of claim 2 further comprising verifying the first OS ID against the second OS ID, wherein if the second OS ID matches the first OS ID but the second OS ID provides additional information, the second OS ID is used."	41
P.	Claim 12: "The method of claim 3 wherein sending the notification to the subscriber associated with the access device further comprises replacing the network address of the access device with a subscriber ID."	42

Q.	Claim 13: "The method of claim 12 wherein sending the notification to the subscriber associated with the access device further comprises aggregating one or more detailed alerts having the subscriber ID into a single alert summary associated with the subscriber ID."	42
R.	Claim 14: "The method of claim 12 wherein the one or more computing devices are coupled to the access device through a local area network (LAN), the access device providing network address translation (NAT) or is coupled to NAT device to share the network address of the access device."	43
S.	Claims 15 and 24.....	44
1.	15[pre]: "A system for network based malware detection in a service provider network, the system comprising:"	44
2.	15[a]: "a plurality of network sensors coupled to the service provider network, each of the network sensors for:"	45
3.	24[pre]: "A computer readable memory containing instructions for network based malware detection in a service provider network, the instructions when executed by a processor performing:"	45
T.	Claims 16-23 and 25	47
VII.	MANDATORY Notices Under 37 C.F.R. §42.8(a)(1)	47
A.	37 C.F.R. §42.8(b)(1): Real Parties-In-Interest.....	47
B.	37 C.F.R. §42.8(b)(2): Related Matters	47
C.	37 C.F.R. §42.8(b)(3)-(4): Lead and Back-Up Counsel and Service Information.....	48
VIII.	FEES UNDER 37 C.F.R. §42.103	48
IX.	CONCLUSION	48

TABLE OF AUTHORITIES

Cases

<i>Advanced Display Systems, Inc. v. Kent State University</i> , 212 F. 3d 1272 (Fed. Cir. 2000)	26, 39
<i>Baron v. Bausch & Lomb Inc.</i> , No. CIV-79-2C, 1992 WL 46979 (W.D.N.Y. Feb. 25, 1992)	27
<i>Dow Jones & Co. v. Abblaise Ltd.</i> , 632 F. Supp. 2d 23 (D.D.C. 2009)	26
<i>Eolas Techs. Inc. v. Adobe Systems, Inc.</i> , No. 6:09-cv-446, 2012 WL 12896524 (E.D. Tex. July 19, 2012)	27
<i>Phillips v. AWH Corp.</i> , 415 F.3d 1303 (Fed. Cir. 2005)	11

Statutes

35 U.S.C. §102(a)	12
35 U.S.C. §102(b)	12
35 U.S.C. §102 (e)	12
35 U.S.C. §103(c)(1)	12
35 U.S.C. §311	1

Rules

37 C.F.R. § 42.100(b)	11
37 C.F.R. §42.103	48
37 C.F.R. §42.104(a)	2

37 C.F.R. §42.104(b)(1)	11, 12
37 C.F.R. §42.21	1
37 C.F.R. §42.8(a)(1).....	47
37 C.F.R. §42.8(b)(1)	47
37 C.F.R. §42.8(b)(2)	47
37 C.F.R. §42.8(b)(3)	48
37 C.F.R. §42.8(b)(4)	48

TABLE OF ABBREVIATIONS

Abbreviation	Full Name
'426 Patent	U.S. Patent No. 8,635,697
Challenged Claims	Claims 1-25 of the '697 Patent
Patent Owner	Patent Owner Netskope, Inc.
Petitioner	Petitioner Fortinet, Inc.
POSITA	Person of Ordinary Skill In The Art

EXHIBIT LIST

Exhibit No.	Document
1001	U.S. Patent No. 8,635,697 ("the '697 patent")
1002	File History of the '697 patent
1003	Declaration of Dr. Vijay Madesetti
1004	U.S. Patent Pub. No. 2008/0086773 to Tuvell et al. ("Tuvell")
1005	KR100765340B1 to Lee ("Lee") and Certified Translation
1006	U.S. Patent Pub. No. 2010/0161795 to Deridder et al. ("Deridder")
1007	Request for Comments: 2616, Hyper Text Transfer Protocol – HTTP/1.1, 1999 ("RFC 2616")
1008	Request for Comments: 793, Transmission Control Protocol, September 1981 ("RFC 793")
1009	A Robust Classifier for Passive TCP/IP Fingerprinting, by Beverly
1010	Index of _p0f3_releases_old_2.x, available at https://lcamtuf.coredump.cx/p0f3/releases/old/2.x/
1011	U.S. Patent No. 7,231,665 to McArdle et al. ("McArdle")
1012	Source code for p0f, v. 2.0.8, available at https://lcamtuf.coredump.cx/p0f3/releases/old/2.x/
1013	Source code for p0f, v. 1.7, available at http://lcamtuf.coredump.cx/p0f3/releases/old/1.x/
1014	U.S. Patent No. 5,319,776 to Hile et al. ("Hile")

I. INTRODUCTION

Petitioner Fortinet, Inc. ("Petitioner") respectfully petitions, under 35 U.S.C. §311 and 37 C.F.R. §42.21, for *inter partes* review ("IPR") of claims 1-25 ("the Challenged Claims") of U.S. Patent No. 8,635,697 ("the '697 patent") (EX1001) on the grounds below.¹

The '697 patent claims a method of determining the operating system ("OS") of a device from its network traffic when the system detects malware in that traffic. The patent, however, does not purport to invent how to determine the OS of a device from its network traffic. Nor could it. The HTTP standard has long included a field in which the device will include the OS of the sender, which can simply be read. Furthermore, OS fingerprinting (determining the OS of a device based upon unique signatures) dates back to the 1990s, more than a decade before the alleged priority date. Similarly, detecting malware signatures in network traffic predates the patent by well over a decade. According to the '697 patent, however, determining the OS is particularly useful in networks where multiple devices share a single IP address. Even use in that context, however, is clearly disclosed in the prior art described herein, none of which was before the examiner during prosecution, rendering the '697 patent invalid.

¹ Unless otherwise noted, all emphases and annotations have been added.

Accordingly, Petitioner respectfully requests that the Board cancel the Challenged Claims.

II. GROUNDS FOR STANDING (37 C.F.R. §42.104(a))

Petitioner certifies that the '697 patent is available for IPR and that Petitioner is not barred or estopped from requesting IPR of the Challenged Claims on the grounds in this petition. The '697 patent issued more than 9 months ago, and Petitioner was served with the complaint alleging infringement of the '697 patent less than one year ago.

III. THE PATENT AND STATE OF THE ART

The '697 patent was filed on April 8, 2011, and claims priority to a provisional application dated March 29, 2011. EX1001. For the purposes of this petition, Petitioner therefore assumes a priority date of March 29, 2011. Because it was filed before March 2013, the '697 patent is subject to pre-AIA patent laws and cites to Title 35 of the U.S.C. will be to that version unless otherwise specified.

A. TCP/IP Stack

The TCP/IP stack is a conceptual model to help explain how data is transmitted from one computer to another via a network, like the internet. For example, when a user (client) requests a webpage from a server, that request is said to travel through four layers: (1) application layer, (2) transport layer, (3) network

layer, and (4) physical layer.² EX1003, ¶19. Though there are multiple protocols that could be used for each of these layers, Petitioner focuses on only those relevant to the Challenged Claims.

Application layer. A request for a website is typically sent via the HTTP (Hypertext Transfer Protocol) protocol, which defines how to format the received data such that the application (e.g., web browser) can properly display the webpage. EX1003, ¶20. Since the 1990s, to ensure the client's web browser can properly display the data, HTTP has utilized a "User-Agent" field in which the client provides information about itself to the server, including browser type and operating system type. The User-Agent field includes "product tokens."

14.43 User-Agent

The User-Agent request-header field contains information about the user agent originating the request. This is for statistical purposes, the tracing of protocol violations, and automated recognition of user agents for the sake of tailoring responses to avoid particular user agent limitations. User agents SHOULD include this field with requests. The field can contain multiple product tokens (section 3.8) and comments identifying the agent and any subproducts which form a significant part of the user agent. By convention, the product tokens are listed in order of their significance for identifying the application.

Example: User-Agent = "User-Agent" ":" 1*(product | comment)

User-Agent: CERN-LineMode/2.15 libwww/2.17b3

EX1007, 14.43 (RFC2616).

And, as shown below the product tokens include software names and versions, which include the OS of the client.

² Although there is also a 7-layer model that is sometimes used, the four-layer model here more closely aligns to technology at issue.

3.8 Product Tokens

Product tokens are used to allow communicating applications to identify themselves by software name and version. Most fields using product tokens also allow sub-products which form a significant part of the application to be listed, separated by white space. By convention, the products are listed in order of their significance for identifying the application.

```
product           = token [ "/" product-version ]
product-version = token
```

Examples:

```
User-Agent: CERN-LineMode/2.15 libwww/2.17b3
Server: Apache/0.8.4
```

Product tokens SHOULD be short and to the point. They MUST NOT be used for advertising or other non-essential information. Although any token character MAY appear in a product-version, this token SHOULD only be used for a version identifier (i.e., successive versions of the same product SHOULD only differ in the product-version portion of the product value).

Id., 3.8; EX1003, ¶¶21-22.

In other words, HTTP itself transmits the OS of the client in the User-Agent field. EX1007, 14.43 (RFC 793); EX1003, ¶¶21-22. The server can then send different data to, for example, a Chrome browser on a Windows laptop versus a Safari browser on an iPhone. *Id.*

Transport Layer. Data from the server is typically broken up into multiple packets for ease of transport. Servers commonly use TCP (Transfer Control Protocol) to transport the packets (e.g., containing webpage data) to the client. TCP is a connection-oriented protocol in which the client and server first set up a logical connection. This connection enables the ability to check for missing or corrupted packets and to ensure all of the data is received correctly. To establish connection, the client and server engage in a three-way handshake so they can, among other things, synchronize sequence numbers for the packets so the recipient

knows how to reassemble the packets in order and knows when one is not received.

EX1008, 3.4; EX1003, ¶23.

Network Layer. This layer helps with routing and also defines the destination and source for the packets. This layer defines the network addresses for the different devices connected to the internet. The internet usually uses IP (Internet Protocol), which is where the term IP address comes from. EX1003, ¶24.

Physical Layer. This layer consists of the specific mediums for transport, e.g., Wi-Fi, Ethernet, 3G/4G/5G. EX1003, ¶25.

B. OS Fingerprinting

OS fingerprinting is the process of using unique signatures and network patterns to identify the OS of a remote computer. EX1003, ¶26. OS fingerprinting based upon network traffic dates back to the 1990s. The aforementioned User-Agent field was added before HTTP/1.1, which came out in June 1999. It would have been a matter of common sense for a POSITA to check the user-agent field for the OS type of the client. EX1003, ¶26. There was also a widely used tool called nmap that used active fingerprinting by checking for open ports on a device. EX1009, 2 (Beverly). nmap dates back to 1998. EX1003, ¶27. Another popular open-source tool was called p0f, which used the signatures in different TCP fields, including in the three-way handshake messages, to determine the OS of the client. *Id.* p0f was on version 2.0.8 by 2006. EX1010; EX1009, 2 (Beverly); EX1011,

1:27-2:14 (McArdle). There were also many other studies and tools used to determine the OS type of a client using various techniques. *E.g.*, EX1009 (Beverly); EX1011, 1:26-2:14 (McArdle); EX1003, ¶¶26-27.

C. Malware Signatures

Signature-based malware detection dates back to the 1980s. The technique derives from the fact that malware often contains certain well-known patterns (i.e., signatures) that can be searched for within network traffic. As such, devices such as firewalls (which also date back to the 1980s) will search for different signatures and can, for example, block traffic containing those signatures. EX1003, ¶63.

D. '697 Patent

The '697 patent relates to identifying computers in a network that have been infected with malware (malicious software), like a computer virus. EX1001, 1:17-21. The patent explains that residential and business networks are typically connected to the internet via a gateway or router that uses "network address translation (NAT)." *Id.*, 1:25-27. This means that, although there may be multiple devices connected through the gateway or router, "[a]ll traffic emerging from that network will share the same IP address that is assigned to the gateway or router" because "Internet service providers typically only provide one wide area network (WAN) address" per premises. *Id.*, 1:30-34. This means that, from the internet

(WAN) side, "it is difficult to determine which device behind a NAT device, is originating a particular stream of traffic." *Id.*, 1:38-40.

According to the patent, this is problematic because, when packets on the WAN side are analyzed and it is determined there is malware in the traffic originating from an infected device behind a NAT device, the "inability to determine a specific device means that any remediation procedures can not be applied in a focused manner." *Id.*, 1:43-45. To solve this alleged problem, the patent purports to identify additional information about the infected device from the TCP packets, including "the malware ID and the OS ID associated with TCP session that generated the alert." *Id.*, 1:53-67.

Figure 1 depicts the alleged invention.

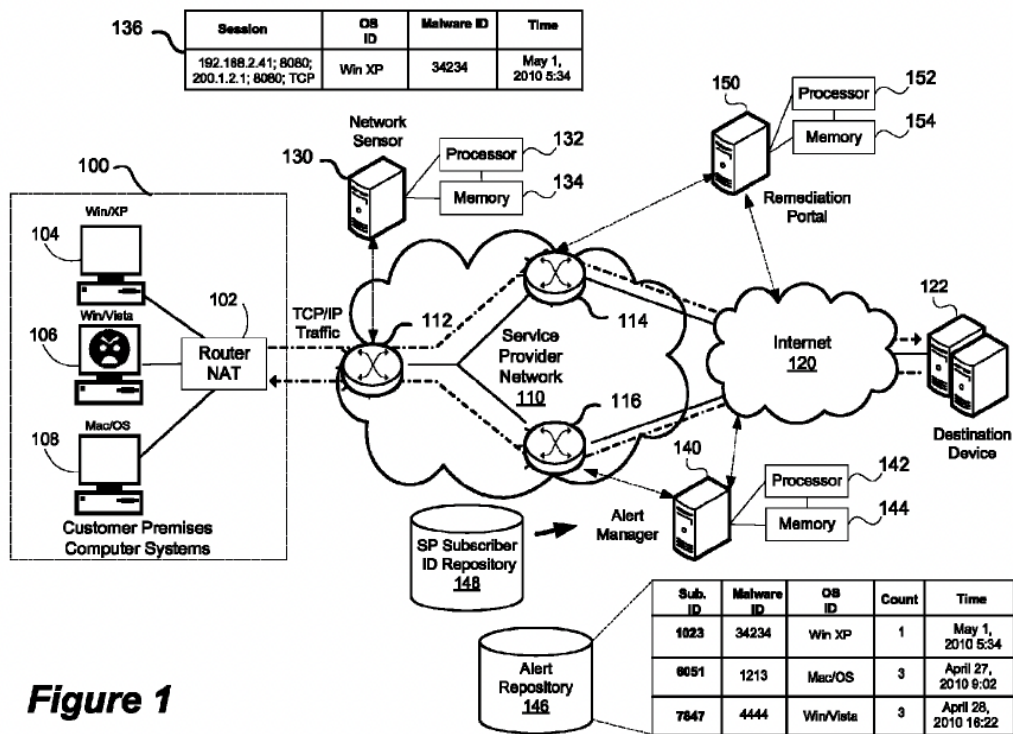


Figure 1

There are three computers (104, 106, 108), each with a different OS, that are part of a LAN 100 connected to the broader internet via a router NAT 102. All of the traffic emanating from the NAT will have the same IP address, regardless of which of the three computers sent the message. The traffic from the NAT will be routed to the service provider network 110, which consists of a plurality of routers and switches (112, 114, 116). *Id.*, 4:11-33. As traffic travels through the service provider network, the network sensor 130 will analyze packets and determine which contain evidence of malware infection on the computers in the LAN 100. *Id.*, 4:50-60. When malware is identified, the network sensor sends an alert to the alert manager 140, which stores the alert in the alert repository 146. *Id.*, 4:60-5:25.

The alert repository stores information like the malware ID and OS ID (e.g., Win XP, MacOS, WinVista). *Id.*, 5:24-27.

E. File History

On February 1, 2013, the Examiner entered a non-final rejection of all pending claims, finding that they were anticipated by Roesch (U.S. Patent No. 7,317,693). EX1002, 69-72. Applicant traversed the rejection, arguing that the claims require determining "if malware is present by comparing the TCP packets to one or more malware signatures." *Id.*, 91. Applicant continued by arguing that, when a malware signature matches a TCP packet, the system will generate an alert "identifying the network address of the access device as well as an identification of the malware and the OS ID associated with the TCP packets." *Id.* According to Applicant, Roesch scans the network to determine what operating systems and services are running on devices connected to the network to create a network topology. The network topology could then be used to manage the network and look for vulnerabilities. *Id.*

The examiner subsequently allowed the claims without amendment. *Id.*, 105.

F. Challenged Claims

The '697 patent has 25 claims, three of which are independent. Claim 1 is representative:

A method of network based malware detection in a service provider network, the method comprising:

receiving one or more transmission control protocol (TCP) packets originating from an access device coupled to the service provider network, the one or more TCP packets defining a TCP session between a computing device coupled to the access device, and a destination coupled to the service provider network;

determining an operating system identifier (OS ID) associated with the TCP session and the computing device;

determining if malware is present in the TCP session and an associated malware ID by comparing a malware signature to the one or more TCP packets; and

generating an alert identifying a network address associated with the access device, the malware ID and the OS ID associated with TCP session that generated the alert.

Claims 15 and 24 are corresponding system and computer-readable medium claims, respectively. The challenged dependent claims add limitations regarding how the OS ID is determined (claims 2, 7-11, 16, 19-23); what alerts to send or other actions to take (claims 3-6, 12-13, 17-18, 25); or that a LAN and NAT are part of the network (claim 14).

G. POSITA

The relevant art for the '697 patent is the field of computer science generally, and specifically, computer networking or computer security. A POSITA, as of March 2011, would have been an individual with either (1) at least a bachelor's degree in computer science or computer engineering or an equivalent field plus at least one year of experience working on computer networking or security, or (2) at

least 3 years of experience working on computer networking or computer security, even without a formal degree. EX1003, ¶16.

IV. CLAIM CONSTRUCTION

A claim is construed "using the same claim construction standard that would be used to construe the claim in a civil action," 37 C.F.R. § 42.100(b), which is governed by *Phillips v. AWH Corp.*, 415 F.3d 1303, 1312 (Fed. Cir. 2005). Under the standard set out in *Phillips*, this petition applies the plain and ordinary meaning to each term in the '697 Patent, which is "the meaning that the term would have to a [POSITA] in question at the time of the invention." *Id.* at 1313.

Petitioner respectfully submits that there are no specific constructions required to resolve the issues raised in this Petition.

V. SUMMARY OF CHALLENGE AND RELIEF REQUESTED

A. 37 C.F.R. §42.104(b)(1): The Challenged Claims

Petitioner requests IPR and cancellation of the Challenged Claims (i.e., claims 1-25) in view of the references discussed below.

B. 37 C.F.R. §42.104(b)(1): Prior Art Overview and Specific Grounds of Rejection

1. Deridder

Deridder is a U.S. Patent Application that was filed December 22, 2009, and published June 24, 2010, both before the assumed priority date of the '697 patent.

Deridder is therefore prior art under at least §102(a) and (e).³

2. Tuvell

Tuvell is a U.S. Patent Application that was published April 10, 2008, more than a year before the assumed priority date of the '697 patent. Tuvell is therefore prior art under at least §102(b).

3. Lee

Lee is a Korean patent application that was published October 9, 2007, more than a year before the assumed priority date of the '697 patent. Lee is therefore prior art under at least §102(b).

C. Grounds of Rejection

Petitioner asserts the following specific grounds:

Ground	Claims	Basis	Prior Art
1	1-25	§103	Deridder over Tuvell and Lee

³ Although Deridder shares at least one named inventor with the '697 patent, Deridder is prior art under at least §102(a) and therefore may be used for §103. *See* §103(c)(1).

VI. GROUND 1 – Claims 1-25 are Rendered Obvious by Deridder Over Tuvell and Lee

A. Overview of Deridder

Deridder discloses a network device used to help track user sessions in connection with a network address translation device (NAT). EX1006, Abstract. Deridder explains that OS fingerprinting is one way to distinguish users connected to the internet through a NAT (e.g., wireless router). According to Deridder, however, prior methods have not been able to do the fingerprinting analysis in real-time. *Id.*, ¶¶4-5. Deridder therefore discloses a method of determining the "OS type" for each device by using information available in TCP packets sent by the device. *Id.*, ¶47.

For example, a traffic inspection device in the system can use "IP and TCP header fields from the initial TCP synchronization (SYN) packet of a connection to provide a unique signature or 'fingerprint' of the OS that generated the packet." *Id.*, ¶31. Deridder explains that OS fingerprinting to determine the type of OS could be useful in network devices that provide services "such as network monitoring, targeted advertising, *malware detection* etc." *Id.*, ¶22.

Figure 1 depicts an example of the network for Deridder's system. The NAT 102 controls access to the service provider network 110. *Id.*, ¶21. And the traffic inspection device 150 inspects traffic coming from the NAT and stores information, like the OS fingerprint. *Id.*

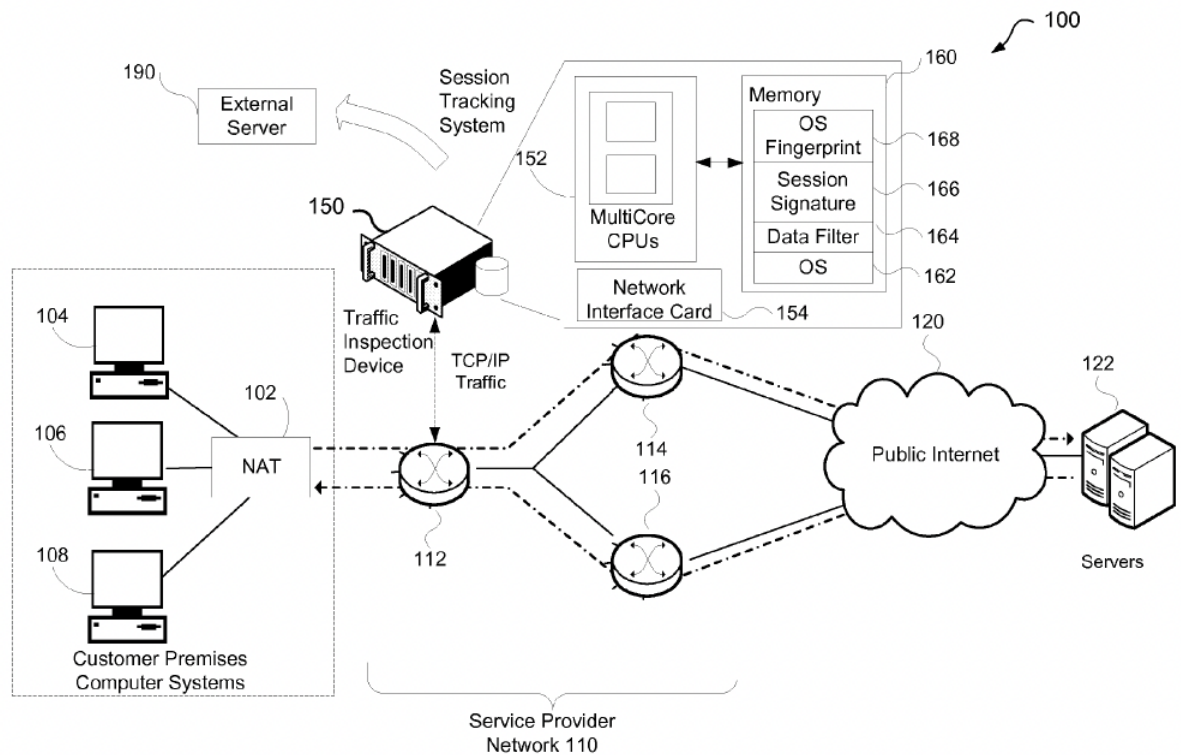


Figure 1

B. Overview of Tuvell

Tuvell discloses a malware detection system, using a "network analyzer 202" and "malware scanner 204" that are part of a network management system. EX1004, ¶¶33-34. Figure 3 depicts the network management system.

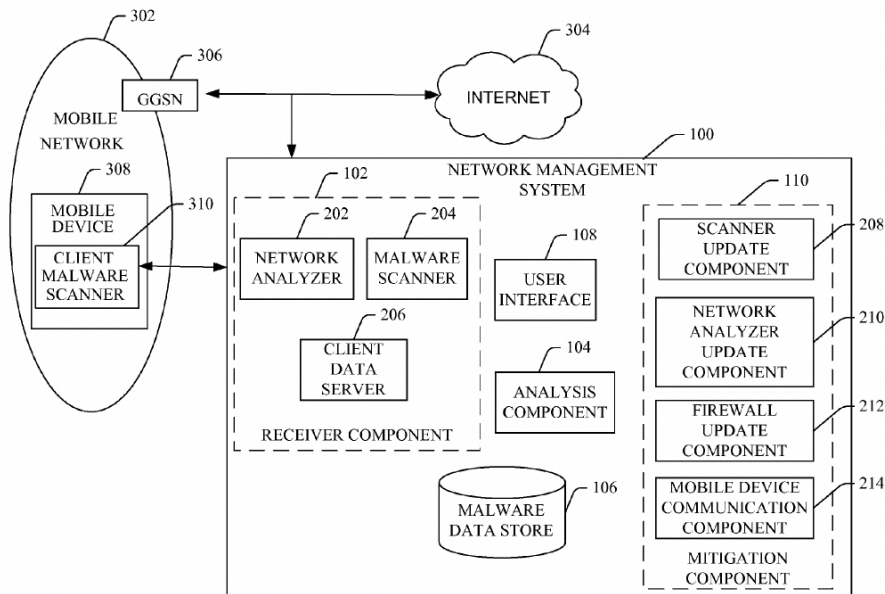


FIG. 3

As depicted in Figure 3, the network management system analyzes data coming from the mobile network **302** to the internet **304** via a GGSN (Gateway GPRS Support Node) **306**. The GGSN acts as a gateway between a subscriber network and external packet networks like the internet. According to Tuvell, the network analyzer "intercepts and logs traffic passing over a network or a portion of a network. In an embodiment, the network analyzer **202** intercepts communications between the mobile network and a data network (e.g., the Internet)." *Id.*, ¶33. The intercepted packets are then analyzed by the "malware scanner **204** to identify malware present in the data packets and the addresses of the transmitting and/or receiving mobile device." *Id.*, ¶34. This "[d]ata is analyzed

in real time." *Id.* Tuvell discloses that this analysis can include "malware signature searches" and "hash signature searches." *Id.*, ¶35.

Tuvell further discloses that the "malware data obtained by network analyzer **202** and client data server **206** includes device-specific information." *Id.*, ¶36. Tuvell explains that the "malware data received by the network analyzer **202** is correlated with data from the mobile network to identify the mobile device that transmitted the infected packet or packets." *Id.* According to Tuvell, moreover, among the received information about the mobile device is the operating system of the mobile device. *Id.*, ¶49. Tuvell discloses that the "resulting malware analysis is stored in malware data store **106** and/or presented to operators via the user interface **108**." *Id.*, ¶37. The malware data store includes information such as "virus name, infected file name, infected file size, infected packet size and infected packet number. The malware data store **106** also maintains packet source related information, such as the source IP, source port and even source identifier (e.g., phone number)." *Id.*, ¶52.

C. Overview of Lee

Lee likewise discloses a network security system, such as a firewall, that intercepts network traffic and checks for malware. EX1005, ¶13. For example, Lee explains that the payload of each packet can be checked against a malware database. If the payload matches anything in the database, it will create a log of

the incident and check whether traffic from the client is malicious. EX1005, ¶30.

As part of the logging process, Lee discloses determining the "operating system (OS)" of the client for logging. EX1005, ¶48.

D. Motivation to Combine Deridder with Tuvell and Lee

It would have been obvious to combine the NAT-based system in Deridder with the malware detection systems in Tuvell and Lee. Petitioner focuses first on why it would have been obvious to add malware detection systems like those in Tuvell and Lee to the NAT-based system in Deridder and then focuses on the combination of the malware systems from Tuvell and Lee.

1. Motivation to Combine Deridder with the Malware System of Tuvell and Lee

As an initial matter, Deridder actually discloses that its system would be useful in services "such as network monitoring, targeted advertising, *malware detection* etc." EX1006, ¶22. In doing so, Deridder recognizes that a POSITA would commonly add malware detection services to the service provider side of the NAT. When this happens, Deridder further recognizes that its technique of OS fingerprinting would be helpful to identify the computer connecting through the NAT with specificity, since all devices connecting through the NAT will have the same network address on the service provider side. EX1003, ¶85.

It would therefore have been obvious to a POSITA to add malware detection to the service provider's network and use the OS fingerprinting in Deridder based

upon Deridder's explicit recognition of this. EX1006, ¶22; EX1003, ¶86. And, since Deridder does not detail how to set up such a malware detection system, a POSITA would have turned to references in that field. Tuvell and Lee are two such references that disclose a malware detection system. Furthermore, it would have been obvious to combine Tuvell and Lee with the system in Deridder because Tuvell and Lee both call for determining the OS of the client device. EX1004, ¶49; EX1005, ¶55 ("operating system (OS) of the aforesaid extracted node").

And, although Lee at least renders obvious one way to extract the OS ID via the User-Agent field as explained more below, a POSITA would have understood that clients may sometimes not include OS information in the User-Agent field and would have been motivated to look for other methods of obtaining the OS type, especially where it is not included in the User-Agent field. EX1003, ¶91. Deridder performs OS fingerprinting using, for example, "IP and TCP header fields from the initial TCP synchronization (SYN) packet of a connection to provide a unique signature or 'fingerprint' of the OS that generated the packet." EX1006, ¶31.

Furthermore, a POSITA would have had a reasonable expectation of success implementing the malware detection system in Tuvell, which is disclosed as part of a cellular network, in the NAT-based system disclosed in Deridder for multiple reasons. For example, like the traffic inspection device in Deridder, the systems in Tuvell and Lee operate on network components that see all of the traffic passing

through a network as part of a service provider network. Tuvell discloses checking data passing through the GGSN in a cellular network, which is a service provider network. EX1003, ¶92. The traffic inspection device in Deridder similarly inspects traffic passing through the NAT in a service provider network. EX1004, ¶33; EX1005, ¶13; EX1006, ¶21, Fig. 1. And even though the GGSN performs some different functions than a NAT, both act as a gateway between a private network and the internet. And both the GGSN and NAT provide network address translation. EX1003, ¶92. A POSITA, therefore, would have known that the malware detection system in Tuvell could be combined with the NAT-based system in Deridder with only trivial modifications to identify devices in a local area network instead of in a cellular network. EX1003, ¶92.

Furthermore, a POSITA would have been motivated to use both OS fingerprinting and the User-Agent field as a cross-check to ensure that the OS ID of the mobile device is properly identified. A POSITA would know that each method could be prone to errors or failure. And a POSITA would understand the importance of correctly identifying the OS because it would have been obvious that malware may only infect certain operating systems. EX1003, ¶93.

As such, a POSITA would have been motivated to add the malware detection system disclosed by the Tuvell-Lee combination to the NAT-based system in Deridder as explicitly contemplated by Deridder. EX1003, ¶94.

2. Motivation to Combine the Malware Systems of Tuvell and Lee

It would have been obvious to a POSITA to combine the teachings of Tuvell and Lee. Both references are in the same general field: providing network security by intercepting and tracking network traffic. EX1004, ¶9 ("The system and method provides a comprehensive means for collecting, reporting, and providing visual depictions of information regarding the propagation and effect of worms, viruses and other malware on a network."); EX1005, ¶33 ("As shown, the virtual in-line network security mechanism to which the present invention is applied operates after connecting the equipment to the network in the form of a general device, unlike the conventional configuration."). Tuvell specifically discloses network components in the form of a network analyzer and malware scanner that "intercepts and logs traffic passing over a network or a portion of a network." EX1004, ¶33. Similarly, Lee discloses that its invention can be implemented in a network device, such as a firewall, that a POSITA would have understood to intercept and log network traffic going into or out of a network. EX1005, ¶13; EX1003, ¶87.

A POSITA would have been motivated to combine the system in Tuvell with that in Lee because Tuvell already teaches correlating the data traffic with the OS type. EX1004, ¶49. The OS type determination in Tuvell, however, is received from the mobile devices connected to the network. *Id.*, ¶¶49, 54. A

POSITA would also have understood that OS type is useful information to link with the virus name because viruses commonly infect only specific operating systems. EX1003, ¶88. A POSITA further would have been motivated to determine the OS type without receiving it from the mobile device in the network to reduce network traffic and make the network analyzer less dependent upon client input. *Id.* In other words, Tuvell teaches that it is helpful to correlate OS type with network traffic, and a POSITA would have been motivated to look for other ways to determine the OS type. *Id.*

A POSITA, therefore, would have been motivated to look for a system that determines the OS based upon the network traffic, and Lee discloses one such system. EX1003, ¶89. For example, Lee explains that one piece of information to extract from the packets is the "operating system (OS) of the aforesaid extracted node." EX1005, ¶55. And, because a POSITA would have had a general familiarity with the HTTP specification, a POSITA would have understood that the OS information can be extracted from the User-Agent field, in which clients disclose their OS type. EX1007, 14.43, 3.8; EX1003, ¶89.

As such, a POSITA would have been motivated to combine the system in Tuvell with that teaching in Lee and would have had a reasonable expectation of success because Tuvell and Lee both teach similar systems using similar network devices. EX1003, ¶90.

E. Claim 1

1. 1[pre]: "A method of network based malware detection in a service provider network, the method comprising:"

To the extent the preamble is limiting, Deridder discloses a NAT-based system that is useful in "malware detection." EX1006, ¶¶22. Figure 1 shows that the traffic inspection device **150** contemplated by Deridder sits in the service provider network **110** outside of the local network behind the NAT **102**. EX1003, ¶¶95-96.

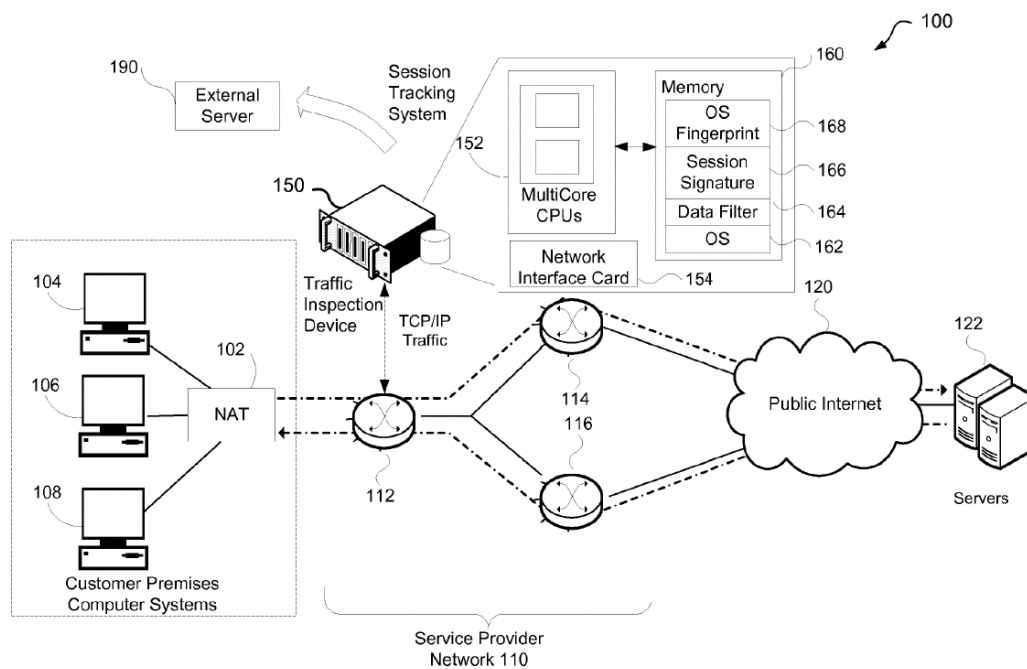


Figure 1

Notably, Figure 1 of Deridder shows a strikingly similar network architecture to Figure 1 of the '697 patent:

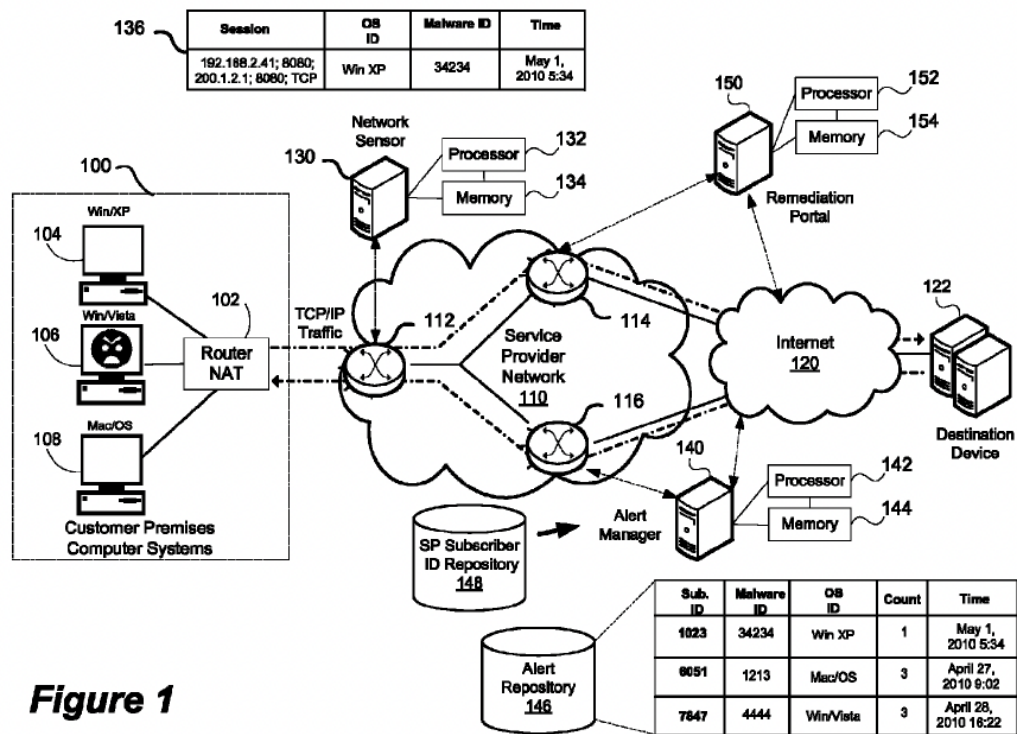


Figure 1

EX1001, Fig. 1.

Further, Tuvell discloses a network-based malware detection system in the form of the network analyzer 202 and malware scanner 204. Tuvell makes clear that these components are disposed in the network so they can intercept network traffic. EX1004, ¶33. Tuvell further discloses that the network analyzer/malware detection scanner is part of the network management system for a service provider network, such as "between the edge of a mobile network 302 and the Internet 304." *Id.*, ¶43. Tuvell also makes clear that the network may be made up of any type of mobile device, including "laptops and other mobile electronic devices." *Id.*, ¶25; EX1003, ¶¶95-96.

It would have been obvious to add the malware detection system in Tuvell to the traffic inspection device in Deridder for the reasons discussed above. EX1003, ¶¶95-96.

2. **1[a]: "receiving one or more transmission control protocol (TCP) packets originating from an access device coupled to the service provider network, the one or more TCP packets defining a TCP session between a computing device coupled to the access device, and a destination coupled to the service provider network;"**

As in 1[pre], Deridder discloses a traffic inspection device **150** that receives TCP packets so that it can use its OS fingerprinting technique based upon the TCP/IP headers. A POSITA would have understood that, because computer devices **104, 106, and 108** are connected through a NAT (access device), TCP sessions would originate from the NAT and proceed to a server (destination) that is coupled to the service provider network **110** via the internet. EX1006, Fig. 1, ¶¶21-23. Deridder makes clear that, when it refers to a NAT, it encompasses known access points "such as home wireless routers." *Id.*, ¶3; *see* EX1001, Fig. 1 ("Router/NAT" **102**). And Deridder discloses that "the inspection device may be located within the service provider network **110** with packets received from the NAT device." EX1006, ¶21. In other words, Deridder discloses that the inspection device receives TCP packets originating from the NAT (access device), which is coupled to the service provider network. EX1003, ¶97.

A POSITA would further understand that there would also be a corresponding TCP session between the NAT and the computer devices **104, 106, and 108** in the NAT network. A POSITA would likewise understand that the TCP packets from the NAT to the destination are defined by the TCP session between the computer devices and the NAT because NATs work as an intermediary network address translation device. EX1006, Fig. 1, ¶¶21-23; EX1003, ¶97.

A POSITA also would have understood that TCP is a connection-based protocol and that one or more TCP packets define a TCP session, or connection. EX1008, §3.4; EX1003, ¶97.

3. 1[b]: "determining an operating system identifier (OS ID) associated with the TCP session and the computing device;"

Deridder discloses using "IP and TCP header fields from the initial TCP synchronization (SYN) packet of a connection" to generate an OS fingerprint. EX1006, ¶31. A POSITA would have understood the IP and TCP header fields to be associated with the TCP session and the computing device as discussed in 1[a]. EX1003, ¶98. Given the limitations in later dependent claims as addressed further below, Petitioner submits that the OS ID determined using the fingerprinting techniques in Deridder is the *first* OS ID, and the OS ID determined by extracting information from the packets, as rendered obvious by Lee is the *second* OS ID. It would have been obvious to a POSITA that the labels first and second are interchangeable in this claim. EX1003, ¶98. Although this limitation requires

determining only one OS ID, both are addressed here, given the limitations in the later dependent claims.

Specifically, Deridder discloses using the open-source software tool for passive OS fingerprinting called "p0f," which Deridder incorporates by reference in its entirety. EX1006, ¶¶4, 31. In 2009 when the Deridder application was filed, the operative version of p0f was 2.0.8, which was publicly available by September 6, 2006. EX1010. The open-source code from p0f provides further detail about the different protocol parameters used to determine an OS fingerprint, (first OS ID). EX1012; EX1003, ¶¶28-62.⁴

In other words, Deridder discloses using p0f to determine a first OS ID that identifies the type of OS of the computing device. EX1003, ¶¶28-62, 98; *Advanced Display Systems, Inc. v. Kent State University*, 212 F. 3d 1272, 1282 (Fed. Cir. 2000) ("Material not explicitly contained in the single, prior art document may still be considered for purposes of anticipation if that material is incorporated by reference into the document.").⁵

⁴ The 2009 version of p0f is included as EX1012 and the 2000 version is included as EX1013 to this Petition. Petitioner provided the human readable files in those exhibits for the purposes of this submission, but can provide the native files upon request.

⁵ Publicly available source code like p0f is consistently held to be prior art. *See Dow Jones & Co. v. Abblaise Ltd.*, 632 F. Supp. 2d 23, 36-37 (D.D.C. 2009), *vacated on jurisdictional grounds*, 606 F.3d 1338, 1349 (Fed. Cir. 2010) (holding

Further, Tuvell discloses that the network analyzer/malware scanner stores information obtained during the network traffic analysis in a "malware data store **106**." EX1004, ¶52. This information includes information like IP address, virus name, and infected file name. *Id.* Tuvell also teaches that mobile devices in the service provider network send an "infection report," that includes information like "type of mobile device **308**, operating system, software and versions." *Id.*, ¶49. Tuvell then discloses that the analysis component **104** of the system "correlates data obtained from a variety of sources (e.g., network traffic, data network traffic and individual mobile devices **308**)." *Id.*, ¶54. A POSITA would therefore have understood that, after the correlation, the system will determine the operating system (OS ID) for a particular computing device associated with TCP session. EX1003, ¶98.

And, it would have been obvious to combine Tuvell with Lee for the reasons set forth above. Lee discloses a similar network security device that sits in between a client and server, analyzing network packets, such as a firewall.

that source code was prior art); *Eolas Techs. Inc. v. Adobe Systems, Inc.*, No. 6:09-cv-446, 2012 WL 12896524, at *4 (E.D. Tex. July 19, 2012) (finding that each individual codebase consisting of many files was a single prior art reference); *Baron v. Bausch & Lomb Inc.*, No. CIV-79-2C, 1992 WL 46979, at *33 (W.D.N.Y. Feb. 25, 1992) ("Although the tables were published in two parts, the two together should be considered as a single publication for purposes of determining anticipation.").

EX1005, ¶30. Lee explains that one piece of information to extract from the packets is the "operating system (OS) of the aforesaid extracted node." *Id.*, ¶55.

Although Lee does not describe from where to extract the OS information, it would have been obvious to a POSITA that the User-Agent field in the HTTP header contains that information and could be easily extracted from there. EX1007, §§14.43, 3.8; EX1003, ¶98. A POSITA reviewing Lee would therefore have found it obvious to check the User-Agent field in the HTTP header to determine the OS. EX1003, ¶98.

And Tuvell already discloses correlating the network traffic data with the OS of the mobile device. EX1004, ¶54. The combination of Deridder, Tuvell, and Lee therefore renders obvious determining an OS ID associated with the TCP session and the computer device in multiple ways, including OS fingerprinting and the User-Agent field. EX1003, ¶98.

4. 1[c]: "determining if malware is present in the TCP session and an associated malware ID by comparing a malware signature to the one or more TCP packets; and"

Tuvell discloses determining if malware is present in the TCP session by using malware scanning algorithms, including "malware signature searches [and] hash signature searches." EX1004, ¶35. A POSITA would have understood that these searches involve comparing a malware signature to one or more TCP packets to determine if malware is present. EX1003, ¶99. Tuvell further discloses storing

information about the malware when it is detected, including the "virus name."

EX1004, ¶52. A POSITA would have understood that the virus name is a malware ID associated with the detected signature. EX1003, ¶99. Further, it would have been obvious to a POSITA that the virus name must be associated with a detected malware signature, or else the virus name could not be included in the record after a signature is matched. EX1003, ¶99.

5. 1[d]: "generating an alert identifying a network address associated with the access device, the malware ID and the OS ID associated with TCP session that generated the alert."

As in 1[b], Tuvell discloses storing the malware ID, OS ID, and network address information in the "malware data store **106**." EX1004, ¶¶49-52. Tuvell then discloses determining "whether to generate output, such as a report or alert." *Id.*, ¶60. These alerts could include "notify[ing] network administrators" or requiring further "mitigating actions," such as "forc[ing] an update of mobile device **308** software, or even disable the mobile device's **308** data connection." *Id.*, ¶60, 62; EX1003, ¶100.

It would have been obvious to a POSITA that the OS types determined in 1[b] as disclosed in Deridder and Lee are stored with the malware ID in the malware store **106**, as disclosed in Tuvell. EX1003, ¶100. For example, Tuvell discloses that the analysis component **104** of the system "correlates data obtained from a variety of sources (e.g., network traffic, data network traffic and individual

mobile devices **308**)." EX1004, ¶54. It would have been obvious to a POSITA that this correlation would have included the OS type determined by the OS fingerprinting technique in Deridder and the User-Agent field from Lee, since Tuvell already contemplated using other data to determine the OS type (OS ID). EX1003, ¶100. Furthermore, a POSITA would have understood that the network address would have been associated with the access device (router/NAT), given that Deridder is detecting traffic from the NAT on the service provider side. EX1003, ¶100.

It further would have been obvious to a POSITA to include all of the stored information, including the network address of the router/NAT, malware ID, and the OS ID in the alert to the network administrator disclosed in Tuvell. A POSITA would have found it obvious that the alert in Tuvell should contain at least as much information as needed for the network administrator to be able to take the disclosed mitigating actions, such as "forc[ing] an update of mobile device **308** software, or even disabl[ing] the mobile device's **308** data connection." EX1004, ¶62. A POSITA would have understood that the network administrator would need the network address of the NAT so the network administrator could identify where in the network the alert was coming from. A POSITA further would have understood that the network administrator would need the OS ID and malware ID to know what software is needed to update the device, since software updates are OS

specific and different updates may be required for different types of malware.

EX1003, ¶100.

F. Claim 2: "The method of claim 1 wherein determining the operating system ID further comprises:"

See claim 1. It would have been obvious to combine the OS fingerprinting techniques disclosed in Deridder with the Tuvell-Lee system that already determines the OS ID from the network data packets for the reasons explained above. Given the limitations in claim 2 and in later dependent claims, Petitioner submits that the OS ID determined using the fingerprinting techniques in Deridder is the *first* OS ID, and the OS ID determined by extracting information from the packets, as rendered obvious by Lee, is the *second* OS ID. It would have been obvious to a POSITA that the labels first and second are interchangeable in this claim. EX1003, ¶¶101-102. This is further explained below.

1. **2[a]: "determining a protocol associated with the TCP packets and matching an OS fingerprint from one or more protocol parameters if present in the TCP packets to determine a first operating system identifier (ID);"**

Deridder discloses using "IP and TCP header fields from the initial TCP synchronization (SYN) packet of a connection" to generate an OS fingerprint.

EX1006, ¶31. A POSITA would have understood that, by determining that there are IP fields present in a TCP packet, the inspection device in Deridder is determining that the IP protocol is associated with the TCP packet. EX1003, ¶103.

Furthermore, a POSITA would have understood the IP and TCP header fields to be *protocol* parameters because they are specific to the TCP and IP *protocols*.

EX1003, ¶103.

Specifically, as in 1[b] Deridder discloses using the open-source software tool for passive OS fingerprinting called "p0f," which Deridder incorporates by reference in its entirety. EX1006, ¶¶4, 31. In 2009, when the Deridder application was filed, the operative version of p0f was 2.0.8, which was publicly available by September 6, 2006. EX1010. The open-source code from p0f provides further detail about the different protocol parameters used to determine an OS fingerprint (first OS ID). EX1012; EX1003, ¶¶28-62, 103.

In other words, Deridder discloses using p0f to determine a first OS ID that identifies the type of OS. EX1003, ¶¶28-62, 103.

2. 2[b]: "determining an application parameter associated with the TCP session if present, and identifying a second OS ID from the application parameter; and"

As in 1[b], it would have been obvious to a POSITA that the Tuvell-Lee system renders obvious reading the HTTP User-Agent field to determine the OS ID since the HTTP specification specifically contemplates the OS ID being transmitted in that field. EX1003, ¶104. Because HTTP is an *application*-layer protocol, a POSITA would have understood the User-Agent field to be an *application* parameter that identifies a second OS ID. EX1003, ¶104.

3. 2[c]: "wherein the OS ID in the alert comprises one of the first OS ID or second OS ID associated with one or more computing devices coupled to the access device."

As in 1[c], Tuvell discloses generating an alert, including "notify[ing] network administrators" or requiring further "mitigating actions," such as "forc[ing] an update of mobile device 308 software, or even disabl[ing] the mobile device's 308 data connection." EX1004, ¶¶60-62. It would have been obvious to a POSITA that the alert to the network administrator should include the first or second OS ID because, for example, it would have been obvious to a POSITA that the OS type would be useful information for the network administrator, so she would know what type of OS may be vulnerable to the detected malware. EX1003, ¶105. It further would have been obvious to a POSITA to include the first or second OS ID because forcing an update of software, as contemplated by Tuvell, would have required knowing what OS was on the device. *Id.*

G. Claim 3: "The method of claim 2 further comprising sending a notification to a subscriber associated with the network address associated with the access device, the notification identifying a remediation portal to remove the determined malware in the alert."

When malware is detected, Tuvell discloses notifying a specific "mobile device 308 user." EX1004, ¶62. Deridder further discloses using the TCP timestamp option to try to uniquely identify which mobile device is associated with a TCP session. EX1006, ¶37. It would have been obvious to a POSITA that the

malware detection system would only notify the mobile device user associated with the TCP session in which malware was detected. EX1003, ¶¶106-107.

Further, when malware is detected, Tuvell discloses notifying a specific "mobile device **308** user" (subscriber associated with the network address associated with the access device). EX1004, ¶62. It would have been obvious to a POSITA that the mobile device user in the context of the NAT architecture of Deridder would mean the computer device user. EX1003, ¶¶106-107. Tuvell further discloses that the system may force an update of the mobile device software. EX1004, ¶62. Tuvell explains that this is handled by the "mitigation component **110** [remediation portal] that initiates and takes actions to mitigate or alleviate the impact of malware in a mobile network." *Id.*, ¶30. It would have been obvious to a POSITA that the notification to the user device forcing the update would have identified the mitigation component so the device could download and install the update. EX1003, ¶107.

It would also have been obvious to a POSITA that forcing the update would require the remediation portal to remove the identified malware. For example, a POSITA would have understood that updating the mobile device would have included downloading and installing the updated software from the mitigation component and that, by updating the device, the malware would be removed. EX1003, ¶¶106-107. A POSITA would have understood that forcing an update

that did not remove the malware would have been pointless because, if the malware were not removed, the device would remain infected. *Id.*

H. Claim 4: "The method of claim 3 further comprising:"

See claim 3. EX1003, ¶109.

1. 4[a]: "identifying at the remediation portal the subscriber accessing the remediation portal;"

As in claim 3, the mitigation component **110** in Tuvell acts as a remediation portal, which can send a notification to "mobile device **308** user." EX1004, ¶62. It would have been obvious to a POSITA that the mobile device user in the context of the NAT architecture of Deridder would mean the computer device user. EX1003, ¶110. Because one of the actions the mitigation component takes is to notify a device *user*, it would have been obvious to a POSITA that the mitigation component identifies the user (subscriber). EX1003, ¶110.

2. 4[b]: "identifying the OS associated with the computing device used by the subscriber to access the remediation portal;"

As in claim 3, one of the mitigation actions taken by the mitigation component is to force an update of the user's computer device. EX1004, ¶62. It would have been obvious to a POSITA that, to update the user's device, the mitigation component would first need to verify the OS associated with the user's device. EX1003, ¶111.

3. **4[c]: "comparing the OS associated with the computing device to the at least one of the first OS ID or second OS ID identified in the alert; and"**

As in 4[b], to update the device, the mitigation component identified the OS. It would have been obvious to a POSITA to compare that information with the first or second OS ID as determined in 2[a] and 2[b]. EX1003, ¶112. A POSITA would have known to do this check in case, for example, the user had multiple devices that connect via the same network address. A POSITA, therefore, would be motivated to compare the determined OS by the mitigation component to the determined OS information from claim 2. EX1003, ¶112.

4. **4[d]: "providing malware remediation to the computing device when the at least one of the first OS ID or second OS ID matches the OS associated with the computing device."**

It would have been obvious to a POSITA that forcing an update of mobile device software should only be done if the OS device determined by the mitigation component matched the OS type identified in claim 2. A POSITA would have known that the malware was identified on a certain OS, as determined in claim 2, and would have known that the update only needed to be forced on that OS. This is because a POSITA would have understood that the same malware would likely not be present on a device with a different OS. It would have been obvious to a POSITA that malware commonly targeted vulnerabilities on specific applications for a specific OS and could not necessarily be used against a device with a

different OS. EX1003, ¶113. Indeed, it would have been obvious to a POSITA that the update to the device would likely be specific to a particular OS to fix the vulnerability targeted by that malware. *Id.* As such, it would have been obvious that, if the user was connecting with a device with a different OS, the update would not be needed by that device. *Id.*

I. Claim 5: "The method of claim 3 wherein if the OS of the computing device is not the at least one of the first OS ID or second OS ID identified in the alert an indication is provided to the subscriber to connect to the remediation portal with another computing device connected to the access device."

Because different devices may be connected to the NAT device in Deridder at any given time, a POSITA would have understood that the device associated with the detected malware may not be connected to the NAT when the notification needs to be sent. If the correct device is not connected, it would have been obvious to the POSITA to send a notification to a different device of the user, notifying the user to connect her other device to the mitigation component. EX1003, ¶¶114-115.

As in 4[c], a POSITA would have understood that the user may have multiple devices, each of which may have a different OS, that connect via the same network address. *Id.* A POSITA further would have understood that, if the OS associated with the computing device did not match the OS from the fingerprinted data, the infected device may be a different device owned by the user. A POSITA, therefore, would have been motivated to notify the user that one of her other

devices may be infected and that she should connect to the mitigation component with her other device. *Id.* A POSITA would have been motivated to notify the user to connect with the other device because it would have been obvious to a POSITA that the device with malware should be fixed as soon as possible to prevent damage to the computer or loss of data. *Id.*

J. Claim 6: "The method of claim 3 wherein the alert is cleared when all computing devices associated with the access device and the computing device having the at least one of the first OS ID or second OS ID have accessed the remediation portal."

As in claim 3, one of the mitigating actions could be a forced update of the user's mobile device. It would have been obvious to a POSITA that the alert to the user should be cleared after the update occurs because it would not make sense to continue to display an alert that is no longer relevant. EX1003, ¶¶116-117. And, a POSITA would have understood that if the user had only one device with a particular OS, the alert would have been cleared when all computing devices having an OS matching the first or second OS ID have been remediated. *Id.*

K. Claim 7: "The method of claim 2 wherein determining the first OS ID is performed by comparing one or more protocol parameters of the one or more TCP packets having a SYN flag associated with the session, wherein the OS fingerprints identify the protocol parameters of the SYN flagged packet and the OS ID associated with parameters."

As in 2[a], Deridder discloses using "IP and TCP header fields from the initial *TCP synchronization (SYN) packet* of a connection" to generate an OS

fingerprint. EX1006, ¶31. A POSITA would thus have understood that this fingerprinting requires comparing the protocol parameters of packets with the SYN flag to determine the OS ID type associated with those parameters in those packets. As detailed in the TCP standard, the SYN flag appears in the first two packets in the three-way handshake. EX1008, §3.4. The source code of p0f, which is incorporated into Deridder by reference, confirms that parameters from these two packets are used to determine the OS fingerprint as explained in more detail by Dr. Madesetti. EX1003, ¶¶28-62, 118-119; *Advanced Display Systems, Inc. v. Kent State University*, 212 F. 3d 1272, 1282 (Fed. Cir. 2000) ("Material not explicitly contained in the single, prior art document may still be considered for purposes of anticipation if that material is incorporated by reference into the document.").

L. Claim 8: "The method of claim 2 wherein determining the application parameter associated with the session and identifying the second OS ID from the application parameter by inspecting an HTTP user agent string to determine the second OS ID."

As in 1[b] and 2[b], it would have been obvious to a POSITA that the system described in Lee extracts the second OS ID from the User-Agent field because a POSITA would have known that the HTTP specification calls for OS information to be in that field. A POSITA, therefore, would have been motivated to inspect the User-Agent field to determine the second OS ID. EX1003, ¶¶120-121.

- M. Claim 9: "The method of claim 2 further comprising verifying the first OS ID against the second OS ID, wherein if the second OS ID does not match the first OS ID the session is identified by the OS ID identified as a more reliable OS ID."**

Because the Tuvell-Lee-Deridder combination renders obvious identifying the OS ID in two distinct ways, it would have been obvious that there could be situations where the OS IDs did not match. For example, the fingerprinting of the TCP SYN packets may not have yielded the correct OS ID. Or, alternatively, the OS ID in the User-Agent field may be out of date or intentionally misstated as part of a security measure. And there is also a possibility that packets got corrupted, causing a mismatch. EX1003, ¶¶122-123. It would have been obvious to verify the two OS IDs against each other for error checking purposes. A POSITA would have known that error checking is a very common computer science technique, including specifically in computer networking. Indeed, even TCP discloses sending a checksum that will be used to confirm reliability of the packet. EX1008, §3.1. A POSITA, therefore, would have been motivated to perform this check of the OS IDs. EX1003, ¶¶122-123. Where the two values did not match, it would have been obvious to a POSITA to use the one that is more reliable. EX1003, ¶¶122-123.

N. Claim 10: " The method of claim 2 further comprising verifying the first OS ID against the second OS ID, wherein if the second OS ID matches the first OS ID either OS ID is used."

As in claim 9, it would have been obvious to a POSITA to compare the first and second OS IDs for error checking purposes. It further would have been obvious that, if the two IDs matched, either could be used. EX1003, ¶¶124-125.

O. Claim 11: "The method of claim 2 further comprising verifying the first OS ID against the second OS ID, wherein if the second OS ID matches the first OS ID but the second OS ID provides additional information, the second OS ID is used."

As in claim 9, it would have been obvious to a POSITA to compare the first and second OS IDs for error checking purposes. If the two did not match, a POSITA would have been motivated to look at what other information was included in the User-Agent field that provided the second OS ID. A POSITA would have known, based upon the HTTP specification, that the field could also have information about, for example, the type of browser that was used. EX1007, §§14.43, 3.8; EX1003, ¶¶126-127. It would have been obvious to a POSITA that, if the User-Agent field provided browser information as well, it would have been more likely to be correct and therefore should be used. EX1007, §§14.43, 3.8; EX1003, ¶¶126-127.

P. Claim 12: "The method of claim 3 wherein sending the notification to the subscriber associated with the access device further comprises replacing the network address of the access device with a subscriber ID."

Deridder discloses detecting the traffic from the NAT on the service provider side, where all traffic emanating from the NAT has the same IP address (i.e., network address of the access device). EX1006, Fig. 1, ¶¶21-23. It would have been obvious to a POSITA that, to send a notification to a specific user, the IP address would have to be replaced with a different ID that is specific to the user. EX1003, ¶¶128-129. A POSITA would have recognized that this could be, for example, a local IP address or MAC address, which a POSITA would understand to be a subscriber ID. *Id.*

Q. Claim 13: "The method of claim 12 wherein sending the notification to the subscriber associated with the access device further comprises aggregating one or more detailed alerts having the subscriber ID into a single alert summary associated with the subscriber ID."

Tuvell discloses that alerts may be generated to network administrators "periodically or upon operator request." EX1004, ¶60. Tuvell continues that the alerts can be "stored for later use, or used in determining appropriate mitigation." *Id.*, ¶61. One of the mitigations disclosed by Tuvell includes notifications to "a mobile device 308 user." *Id.*, ¶62. As such, Tuvell contemplates that the notifications to the user can be created periodically and/or stored for later use. It would therefore have been obvious to a POSITA that all alerts for the user during a

period or that have been stored up would be aggregated into a single alert summary for the user. EX1003, ¶¶130-31. And, as in 4[a], it would have been obvious that the report is associated with a user at least because Tuvell discloses notifying the *user* of the mobile device. EX1004, ¶62; EX1003, ¶¶130-31. It would have been obvious to a POSITA that the mobile device user in the context of the NAT architecture of Deridder would mean the computer device user. EX1003, ¶¶130-31.

R. Claim 14: "The method of claim 12 wherein the one or more computing devices are coupled to the access device through a local area network (LAN), the access device providing network address translation (NAT) or is coupled to NAT device to share the network address of the access device."

As shown in Figure 1, Deridder discloses a system in which multiple computing devices **104, 106, 108** are coupled via a local area network to a NAT or router (access device) that provides network address translation services. EX1006, ¶¶3, 21-23. Further, a Deridder discloses that the computing devices that are coupled to the NAT share the network address of the NAT. *Id.*; EX1003, ¶¶132-133.

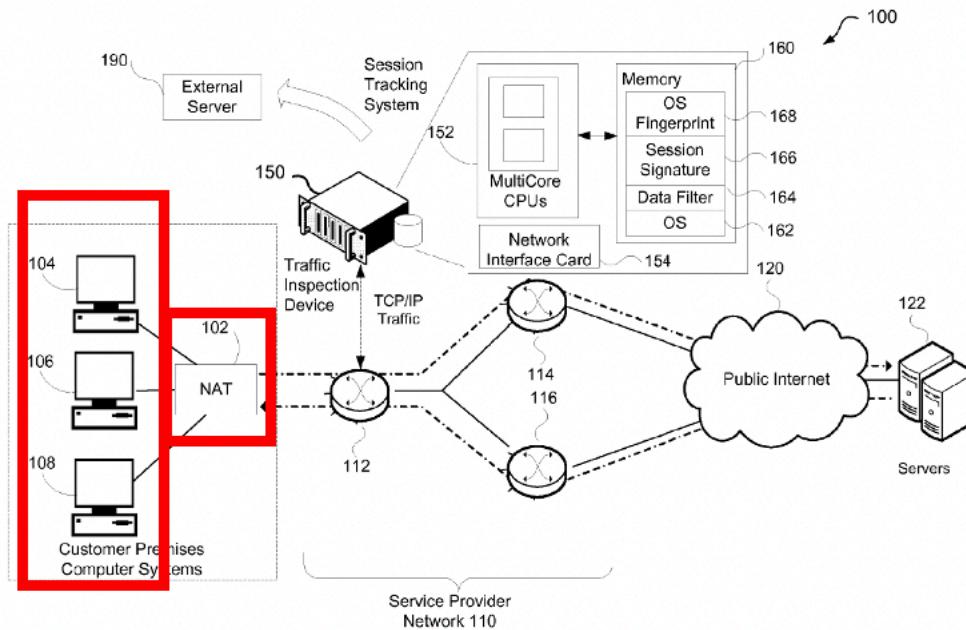


Figure 1

S. Claims 15 and 24

Claims 15 and 24 are substantially similar to claim 1, other than the preambles and preliminary limitations. Petitioner addresses the differences below. The other limitations are invalid for the reasons set forth in claim 1, as detailed in the mapping below. The analysis from claim 1 is incorporated by reference herein. EX1003, ¶¶134-139.

1. 15[pre]: "A system for network based malware detection in a service provider network, the system comprising:"

To the extent the preamble is limiting, as in 1[pre], Deridder discloses a system with a traffic inspection device **150**, which sits in the service provider network **110** outside of the NAT **102**. EX1006, Fig. 1, ¶¶21-23; EX1003, ¶136. And, Deridder explains that the system can be used for malware detection, for

which the Tuvell-Lee combination provides details. EX1006, Fig. 1, ¶¶21-23; EX1003, ¶136.

2. 15[a]: "a plurality of network sensors coupled to the service provider network, each of the network sensors for:"

It would have been obvious to a POSITA that the duties of the traffic inspection device in Deridder could have been distributed over multiple sensors throughout the service provider network, and Deridder further contemplates multiple possible implementations for the traffic inspection device, including functioning "inline with the network traffic." EX1006, ¶21. A POSITA would have understood that functioning inline would include a plurality of network sensors. EX1003, ¶137. Indeed, Tuvell discloses one such arrangement, and it would have been obvious to a POSITA that such system could be used in the service provider's system in Deridder as explained above. *See* EX1004, ¶¶73-93, EX1003, ¶137.

3. 24[pre]: "A computer readable memory containing instructions for network based malware detection in a service provider network, the instructions when executed by a processor performing:"

To the extent the preamble is limiting, as in 1[pre], Deridder discloses a traffic inspection device **150**, which sits in the service provider network **110** outside of the local network behind the NAT **102**. EX1006, Fig. 1, ¶¶21-23; EX1003, ¶138. A POSITA would have understood that a traffic inspection device

is necessarily implemented on computer-readable memory with instructions to be executed on a processor. EX1003, ¶138. And, Deridder explains that the traffic inspection device can be used for malware detection, for which the Tuvell-Lee combination provides details. EX1006, Fig. 1, ¶¶21-23; EX1003, ¶138.

Claim 15	Claim 24	Claim 1
15[b] receiving one or more transmission control protocol (TCP) packets originating from an access device coupled to the service provider network, the one or more TCP packets defining a TCP session between a computing device coupled to the access device, and a destination coupled to the service provider network;	24[a] receiving one or more transmission control protocol (TCP) packets originating from an access device coupled to the service provider network, the one or more TCP packets defining a TCP session between a computing device coupled to the access device, and a destination coupled to the service provider network;	1[a] (EX1003, ¶139)
15[c] determining an operating system identifier (OS ID) associated with the TCP session and the computing device associated;	24[b] determining an operating system identifier (OS ID) associated with the TCP session and the computing device;	1[b] (EX1003, ¶139)
15[d] determining if malware is present in the TCP session and an associated malware ID by comparing a malware signature to the one or more TCP packets; and	24[c] determining if malware is present in the TCP session and an associated malware ID by comparing a malware signature to the one or more TCP packets; and	1[c] (EX1003, ¶139)
15[e] generating an alert identifying a network address associated with the access device, the malware ID and the OS ID associated with	24[d] generating an alert identifying a network address associated with the access device, the malware ID and the OS ID associated	1[d] (EX1003, ¶139)

Claim 15	Claim 24	Claim 1
TCP session that generated the alert.	with TCP session that generated the alert.	

T. Claims 16-23 and 25

The remaining dependent system claims are substantively the same as the corresponding method claims already addressed and are invalid for the reasons set forth above based upon the following mapping. The analysis from the method claims is incorporated herein by reference. EX1003, ¶¶140-141.

System Dependent Claims	Method Dependent Claims
16	2
17	3
18	12
19	7
20	8
21	9
22	10
23	11
25	4

VII. MANDATORY NOTICES UNDER 37 C.F.R. §42.8(a)(1)

A. 37 C.F.R. §42.8(b)(1): Real Parties-In-Interest

Petitioner is the real party-in-interest.

B. 37 C.F.R. §42.8(b)(2): Related Matters

The '697 patent is asserted against Petitioner in *Netskope, Inc. v. Fortinet, Inc.*, No. 3:25-cv-2360 (N.D. Cal.).

C. 37 C.F.R. §42.8(b)(3)-(4): Lead and Back-Up Counsel and Service Information

Designated Counsel for Petitioner and service information is below:

Lead Counsel	Back-Up Counsel
Andrew D. Gish (Reg. # 67,562) GISH PLLC 41 Madison Avenue New York, New York 10010 Telephone: (212) 518-7380 andrew@gishpllc.com	Ryan Iwahashi (Reg. # 63,378) GISH PLLC 50 California Street, Suite 1500 San Francisco, CA 94111 Telephone: (415) 630-8960 ryan@gishpllc.com

Petitioner consents to service by email at the addresses above.

VIII. FEES UNDER 37 C.F.R. §42.103

Petition and Post-Institution fees totaling \$58,925 have been paid by electronic funds transfer.

IX. CONCLUSION

Petitioner requests the Board institute IPR of the Challenged Claims and cancel them.

Respectfully Submitted,

Dated: October 8, 2025

/s/Andrew Gish

Andrew Gish (Reg. #67,562)

ATTORNEY FOR PETITIONER

CERTIFICATE OF WORD COUNT UNDER 37 C.F.R. §42.24(a)

I, the undersigned, do hereby certify that the attached petition contains 9,904 words, as measured by the Word Count function of Microsoft Word. This is less than the limit of 14,000 words as specified by 37 C.F.R. §42.24(a)(i).

Dated: October 8, 2025

/s/ Andrew Gish

Andrew Gish (Reg. # 67,562)

ATTORNEY FOR PETITIONER

CERTIFICATION OF SERVICE

The undersigned certifies service pursuant to 37 C.F.R. §§42.6(e) and 42.105 on the Patent Owner of a copy of this Petition for *Inter Partes* Review and supporting materials via FedEx at the following correspondence address of record:

OSHA BERGMAN WATANABE & BURTON LLP
Enterprise Plaza
1100 Louisiana, Suite 4900
HOUSTON, TX
UNITED STATES

Dated: October 8, 2024

/s/ Andrew Gish

Andrew Gish (Reg. # 67,562)

ATTORNEY FOR PETITIONER