Trials@uspto.gov
571-272-7822

Paper 30
Entered: January 10, 2022

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

JUNIPER NETWORKS, INC.,
Petitioner,

v.

HUAWEI DIGITAL TECHNOLOGIES (CHENG DU) CO., LTD.,
Patent Owner.

IPR2020-01130
Patent 10,027,693 B2

Before PATRICK M. BOUCHER, MICHELLE N. WORMMEESTER, and NABEEL U. KHAN, *Administrative Patent Judges.*

KHAN, *Administrative Patent Judge*.

JUDGMENT
Final Written Decision
Determining Some Challenged Claims Unpatentable
35 U.S.C. § 318(a)

I. INTRODUCTION

On June 17, 2020, Juniper Networks, Inc. ("Petitioner") filed a Petition requesting *inter partes* review of claims 1–9 (the "challenged claims") of U.S. Patent No. 10,027,693 B2 (Ex. 1001, the "'693 patent"). Paper 1 ("Pet."). Huawei Digital Technologies (Chengdu) Co., Ltd. ("Patent Owner") filed a Preliminary Response. Paper 7 ("Prelim. Resp."). Petitioner and Patent Owner filed additional briefing with our authorization. *See* Papers 10, 12. On January 22, 2021, upon consideration of the Petition, Preliminary Response, the additional briefing, and the evidence cited, we determined that Petitioner established a reasonable likelihood that it would prevail with respect to at least one of the claims challenged in the Petition and instituted review to determine the patentability of the challenged claims on all grounds. Paper 13 ("Dec. Inst.").

After institution, Patent Owner filed a Patent Owner Response (Paper 17, "PO Resp."), Petitioner filed a Reply (Paper 19, "Pet. Reply"), and Patent Owner filed a Sur-reply (Paper 23, "PO Sur-reply"). An oral hearing was held on October 26, 2021, and the hearing transcript is included in the record. Paper 29 ("Tr.").

We have jurisdiction under 35 U.S.C. § 6. This Final Written Decision, issued pursuant to 35 U.S.C. § 318(a) and 37 C.F.R. § 42.73 (2019), addresses issues and evidence raised during the *inter partes* review. For the reasons that follow, Petitioner demonstrates by a preponderance of the evidence that claims 8 and 9 of the '693 patent are unpatentable but does not demonstrate by a preponderance of the evidence that claims 1–7 are unpatentable.

II. BACKGROUND

A. Related Proceedings

The parties identify the following pending district court proceeding involving the '693 patent: *Huawei Technologies Co., Ltd. et al. v. Verizon Communications, Inc. et al*., Case No. 6:20-CV-00090 (W.D. Tex.) ("the parallel litigation"). Pet. 73; Paper 5, 2.

Patent Owner also identifies IPR2020-01131 as a related proceeding. Paper 5, 2. The Board denied institution of *inter partes* review of the '693 patent in IPR2020-01131. *Juniper Networks, Inc. v. Huawei Tech. Co.*, IPR2020-01131, Paper 13 (PTAB Jan. 22, 2021).

B. Real Parties-in-Interest

Petitioner identifies "Verizon Communications, Inc., Cellco Partnership d/b/a Verizon Wireless, and Verizon Business Network Services, Inc." as the real party-in-interest. Pet. 73. Patent Owner identifies itself as the real party-in-interest. Paper 5, 2.

C. The '693 Patent

The '693 patent is titled "Method, Device and System for Alerting Against Unknown Malicious Codes Within a Network Environment." Ex. 1001, code (54). The patent explains that when malicious codes are discovered early, network attacks can be prevented in time, and the loss caused by malicious codes will be reduced. *Id.* at 1:39–41. Specifically, the patent describes approaches that

> report source path of numerous suspicious codes proactively at the earliest possible time, enable the vendor to obtain the source path of malicious code samples shortly after the malicious codes occur, ensure comprehensiveness of the alert information source, lay a foundation for shortening the time required for overcoming

virus threats, and avoid the trouble of installing software on the terminal.

Id. at 2:39–48. Figure 1 of the '693 patent, reproduced below, is a schematic diagram of an embodiment of the method. *Id.* at 2:60–62.

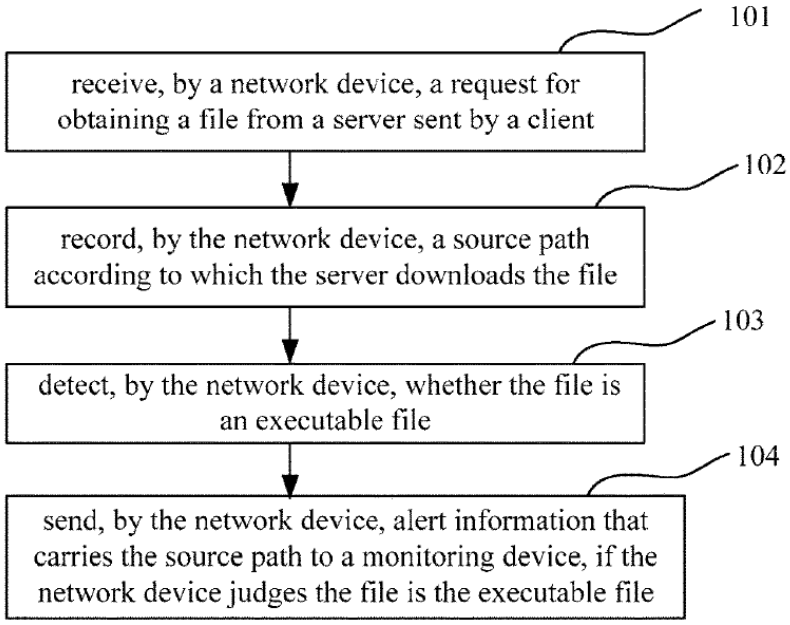


Figure 1 describes four steps in the method. *Id.* at 3:22–38.

The '693 patent describes that "a network device" performs the steps of the method, which specifies that the "source path is sent to the monitoring device so that the monitoring device could download the file according to the source path, and detect the executable file to confirm maliciousness of the executable file in time." Ex. 1001 at 3:39–43. The method determines if the requested file is an "executable file" in two ways: (1) by detecting whether a string which indicates the name of the file includes *.exe or *.dll or *.ocx (*id*. at 4:7–21) or (2) by detecting whether a file header of an executable file exists in the data returned by the server (*id*. at 4:4–6, 22–30). If the network device judges the file to be an executable file, it generates first alert information including the source path and sends the first alert information to the monitoring device. *Id*. at 4:44–49. The monitoring

device downloads the file according to the source path and confirms the maliciousness of the executable file. *Id*. at 3:39–43. In one embodiment, the monitoring device may identify maliciousness of the suspicious code through characteristics detection, sandbox test, or both. *Id*. at 5:24–26. If the monitoring device confirms that the file is malicious it may send second alarm information to the network device. *Id*. at 5:62–67. After it receives the second alarm information, the network device intercepts the suspicious file. *Id*. at 6:12–25.

D. Illustrative Claim

Of the challenged claims, claims 1, 3, 5 and 8 are independent claims. Claim 2 depends from claim 1, claim 4 depends from claim 3, claims 6 and 7 depend from claim 5, and claim 9 depends from claim 8. Claim 1 recites:[1]

> [1pre] A method for alerting against unknown malicious codes, the method comprising:
>
> [a] receiving, by a network device, a request sent by a terminal for obtaining a file from a network entity and a data stream carrying the file;
>
> [b] recording, by the network device, a source path carried in the request, wherein the network entity provides the file on the source path;
>
> [c] judging, by the network device, whether the file is an executable file according to at least one of: the request and the data stream carrying the file;
>
> [d] when the network device judges the file is an executable file, sending, by the network device, first alert information that carries the source path to a monitoring device;
>
> [e] receiving, by the network device, second alarm information sent by the monitoring device after further

[1] Reference letters in brackets mirror those provided by Petitioner. Pet. vi.

detecting the file downloaded according to the source path by the monitoring device; and

[f] intercepting, by the network device,

a) the executable file according to the second alarm information comprising a maliciousness of the executable file; or

b) the executable file and packets transmitted in a Botnet according to the second alarm information comprising the maliciousness of the executable file and Botnet topology information.

Ex. 1001, 8:13–37.

E. Evidence

The Petition relies on the following references:

1. Moshchuk, et al. "SpyProxy: Execution-based Detection of Malicious Web Content." 16th USENIX Security Symposium, presented August 8, 2007. ("Moshchuk"). Ex. 1007[2].
2. Gribble, et al., US 8,196,205 B2, issued June 5, 2012. ("Gribble"). Ex. 1010.
3. Dubrovsky et al., US 8,276.202 B1, issued Sept. 25, 2012. ("Dubrovsky"). Ex. 1011.
4. Bertman, et al., US 7,287,279 B2, issued Oct. 23, 2007. ("Bertman"). Ex. 1012.
5. Campbell, et al., US 8,316,446 B1, issued Nov. 20, 2012. ("Campbell"). Ex. 1013.

Petitioner also relies on the Declaration of Dr. Seth Nielson (Ex. 1005) and the Supplemental Declaration of Dr. Seth Nielson (Ex. 1068). Dr. Nielson was cross-examined by Patent Owner, and a copy of his deposition transcript was entered into the record (Ex. 2019). Patent Owner relies on the Declaration of Dr. Nathaniel Davis, Ph.D. (Ex. 2013), and the Second Declaration of Dr. Nathaniel Davis, Ph.D. (Ex. 2018). Dr. Davis

[2] An unannotated version is filed in Exhibit 1008.

was cross-examined by Petitioner, and a copy of his deposition transcript was entered into the record (Ex. 1050).

F. Asserted Grounds of Unpatentability

Petitioner asserts that claims 1–9 are unpatentable on the following grounds:

Ground	Claim(s) Challenged	35 U.S.C. §	Reference(s)/Basis
1	1–9	103	Moshchuk, Gribble
2A	1, 3, 5, 6	103	Dubrovsky, Bertman
2B	2, 4, 7	103	Dubrovsky, Bertman, Campbell
2C	8	103	Dubrovsky
2D	9	103	Dubrovsky, Campbell

III. ANALYSIS OF THE ASSERTED GROUNDS

For the reasons below, we determine Petitioner has shown by a preponderance of the evidence that claims 8 and 9 of the '693 patent are unpatentable but has not shown by a preponderance of the evidence that claims 1–7 are unpatentable. Before we analyze the asserted grounds we first address two matters that will underlie our analysis: the appropriate level of ordinary skill in the art and the constructions we will apply to claim terms.

A. Legal Standards

"In an [*inter partes* review], the petitioner has the burden from the onset to show with particularity why the patent it challenges is unpatentable." *Harmonic Inc. v. Avid Tech., Inc.*, 815 F.3d 1356, 1363 (Fed. Cir. 2016) (citing 35 U.S.C. § 312(a)(3) (requiring *inter partes* review

petitions to identify "with particularity . . . the evidence that supports the grounds for the challenge to each claim")); *see also* 37 C.F.R. § 42.104(b) (requiring a petition for *inter partes* review to identify how the challenged claim is to be construed and where each element of the claim is found in the prior art patents or printed publications relied upon).

A claim is unpatentable as obvious under 35 U.S.C. § 103(a) if the differences between the claimed subject matter and the prior art are such that the subject matter, as a whole, would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. *KSR Int'l Co. v. Teleflex Inc.*, 550 U.S. 398, 406 (2007). The question of obviousness is resolved on the basis of underlying factual determinations, including: (1) the scope and content of the prior art; (2) any differences between the claimed subject matter and the prior art; (3) the level of skill in the art; and (4) where in evidence, so-called "secondary considerations," including commercial success, long-felt but unsolved needs, failure of others, and unexpected results. *Graham v. John Deere Co.*, 383 U.S. 1, 17–18 (1966).

B. Level of Ordinary Skill

The level of ordinary skill in the pertinent art at the time of the invention is relevant to how we construe the patent claims. *See Phillips v. AWH Corp.*, 415 F.3d 1303, 1312–13 (Fed. Cir. 2005) (en banc). It is also one of the factual considerations relevant to obviousness. *See Graham*, 383 U.S. at 17–18.

To assess the level of ordinary skill, we construct a hypothetical "person of ordinary skill in the art," from whose vantage point we assess obviousness and claim interpretation. *See In re Rouffet*, 149 F.3d 1350,

1357 (Fed. Cir. 1998). This legal construct "presumes that all prior art references in the field of the invention are available to this hypothetical skilled artisan." *Id.* (citing *In re Carlson*, 983 F.2d 1032, 1038 (Fed. Cir. 1993)).

Petitioner contends a person of ordinary skill in the art "would have a bachelor's degree in computer science or related field and four years of industry experience in computer networking, or a master's degree in computer science and two years of industry experience." Pet. 16 (citing Ex. 1005 ¶ 88). Patent Owner does not submit a proposed level of ordinary skill in the art in its Patent Owner Response but Dr. Davis testifies that a person of ordinary skill "would have had at least a bachelor's or master's degree in computer science, computer engineering, electrical engineering, telecommunications and networking, or some similar field, along with 2–3 years of experience in networking technologies and security, or the industry equivalent thereof." Ex. 2013 ¶ 30.

In determining the level of ordinary skill in the art, various factors may be considered, including the "type of problems encountered in the art; prior art solutions to those problems; rapidity with which innovations are made; sophistication of the technology; and educational level of active workers in the field." *In re GPAC Inc.*, 57 F.3d 1573, 1579 (Fed. Cir. 1995) (citation omitted). The level of ordinary skill in the art is also reflected by the prior art of record. *See Okajima v. Bourdeau*, 261 F.3d 1350, 1355 (Fed. Cir. 2001).

Petitioner's proposed level of ordinary skill does not materially differ from that articulated by Dr. Davis. Both require either a bachelor's or master's degree in computer science or a related field (e.g. those specifically

delineated by Dr. Davis such as computer and electrical engineering) with both requiring anywhere between 2 to 4 years of experience in computer networking, or "networking technologies," as Dr. Davis puts it. We adopt Petitioner's proposed level of ordinary skill. Petitioner's proposed level is consistent with the level of expertise assumed by the '693 patent and the prior art of record. We note, however, that we would reach the same conclusions under either proposed level of ordinary skill.

C. Claim Construction

In this *inter partes* review, the Board applies the same claim construction standard as that applied in federal courts. *See* 37 C.F.R § 42.100(b). Under this standard, claim terms "are generally given their ordinary and customary meaning" as understood by a person of ordinary skill in the art in question at the time of the invention. *Phillips v. AWH Corp.*, 415 F.3d 1303, 1312–13 (Fed. Cir. 2005) (en banc) (citations omitted). The ordinary and customary meaning of a claim term "is its meaning to the ordinary artisan after reading the entire patent," and "as of the effective filing date of the patent application." *Phillips*, 415 F.3d at 1313, 1321. "In determining the meaning of the disputed claim limitation, we look principally to the intrinsic evidence of record, examining the claim language itself, the written description, and the prosecution history, if in evidence." *DePuy Spine, Inc. v. Medtronic Sofamor Danek, Inc.*, 469 F.3d 1005, 1014 (Fed. Cir. 2006) (citing *Phillips*, 415 F.3d at 1312–17). Extrinsic evidence is "less significant than the intrinsic record in determining 'the legally operative meaning of claim language.'" *Phillips*, 415 F.3d at 1317 (citation omitted).

Petitioner argues we should construe the terms "detecting the [executable] file" (Pet. 16–17), "file," and "file header" (Pet. Reply 3–4). Patent Owner responds with a proposed construction for the term "file header." PO Sur-reply 2–4.

1. "detecting the [executable] file"

Petitioner argues that "detecting the [executable] file" should be construed to mean "performing operations on the [executable] file, such as characteristic detection or sandboxing, to determine the maliciousness of the file." Pet. 16–17. Patent Owner does not address this term in its Response but did argue in its Preliminary Response that no arguments raised by Petitioner or Patent Owner rest on whether the detected file is unknown, and that therefore the term need not be construed. Prelim. Resp. 19.

Based on the issues presented, we determine that this term does not require express construction. *See Nidec Motor Corp. v. Zhongshan Broad Ocean Motor Co.*, 868 F.3d 1013, 1017 (Fed. Cir. 2017) ("[W]e need only construe terms 'that are in controversy, and only to the extent necessary to resolve the controversy.'" (quoting *Vivid Techs., Inc. v. Am. Sci. & Eng'g, Inc.*, 200 F.3d 795, 803 (Fed. Cir. 1999))).

2. "file"

In its Reply, Petitioner additionally argues that the plain and ordinary meaning of the term "file" must account for the fact that the file is at least sometimes in a state of transmission. Pet. Reply 3 (citing Ex. 1068 ¶¶ 27–30; Ex. 1049, 3). Petitioner cites the Encyclopedia of Computer Science in support of its argument, which it argues describes a file "in terms that address files in transmission, as well as stored files: 'a collection of data

representing a set of entities with certain aspects in common and which are organized for some specific purpose.'" Pet. Reply 3 (citing Ex. 1049, 3). Finally, Petitioner argues that its understanding of the term "file" is also consistent with the '693 patent and how a POSITA would understand its plain and ordinary meaning." Pet. Reply 3 (citing Ex. 1068 ¶ 31–33).

Patent Owner does not address this term in its briefing. *See* PO Sur-reply 2–4.

Based on the issues presented, we determine that this term does not require express construction.

3. "file header"

Petitioner argues that the plain and ordinary meaning of the term "file header" is "information that precedes and provides information about a file." Pet. Reply 3–4. Petitioner cites to extrinsic evidence in support of its construction, including the Microsoft Computer Dictionary, and the NIST Guide to Integrating Forensic Techniques into Incident Response. Pet. Reply 3 (citing Ex. 1051, 3–4 (defining "header" as "An information structure that precedes and identifies the information that follows, such as a block of bytes in communications, a file on a disk, a set of records in a database, or an executable program"); Ex. 1052, 43).

Patent Owner argues that the plain and ordinary meaning of "file header" is "header information included '***within a file*** that contains identifying information about the file.'" PO Sur-reply 2 (citing Ex. 2018 ¶ 119; Ex. 1050, 22:6–9). According to Patent Owner, Petitioner's proposed construction is overly broad and encompasses information far outside the ordinary understanding of "file header." PO Sur-reply 2. Patent Owner points to the same exhibit relied upon by Petitioner as providing an express

definition of the term that is consistent with Patent Owner's position. PO Sur-reply 2 (citing Ex. 1052, 105 ("File Header: Data within a file that contains identifying information about the file and possibly metadata with information but he file contents.")). Additionally, Patent Owner argues that the '693 patent also supports its proposed construction. PO Sur-reply 2 (citing Ex. 1001, 4:22–23 ("At the time of detecting whether a file header of the executable file exists in the file returned by the server . . . .")).

The parties' proposed constructions for the term "file header" are relevant to the Moshchuk-Gribble grounds (Ground 1) but not to the grounds based on Dubrovsky, Bertman, or Campbell (Grounds 2A–2D) as Patent Owner does not dispute that Campbell discloses file headers. As shown by our analysis of the Moshchuk-Gribble grounds, we need not construe the term "file header" to resolve the dispute between the parties.

D. Description of Prior Art References

Petitioner's challenges rely on Moshchuk, Gribble, Dubrovsky, Bertman, and Campbell. *See* Pet., *passim*.

1. Moshchuk (Ex. 1007)

Moshchuk, a paper from the University of Washington Department of Computer Science, describes "the use of **execution-based** Web content analysis to protect users from Internet-borne malware," which involves an approach to "observe active Web content in a disposable virtual machine **before** it reaches the user's browser, identifying and blocking pages whose behavior is suspicious," using a "proxy-based anti-malware tool called SpyProxy." Ex. 1007, 1. The tool "intercepts and evaluates Web content in transit from Web servers to the browser." *Id.*

Moshchuk observes that "modern Web pages are increasingly active, containing embedded code such as ActiveX components, JavaScript, or Flash that executes in the user's browser." Ex. 1007, 1. SpyProxy intercepts, renders, and executes downloaded web content in a temporary, isolated execution environment, separate from a requesting user's browser. *Id.* Upon evaluation, if the content is unsafe, it is blocked from a user. *Id.* When executing the web content in a "disposable virtual machine," SpyProxy "monitors the executing Web content by looking for suspicious 'trigger' events (such as registry writes or process creation) that indicate potentially malicious activity." *Id.*

An overview of the SpyProxy architecture is portrayed in Figure 1, reproduced below.

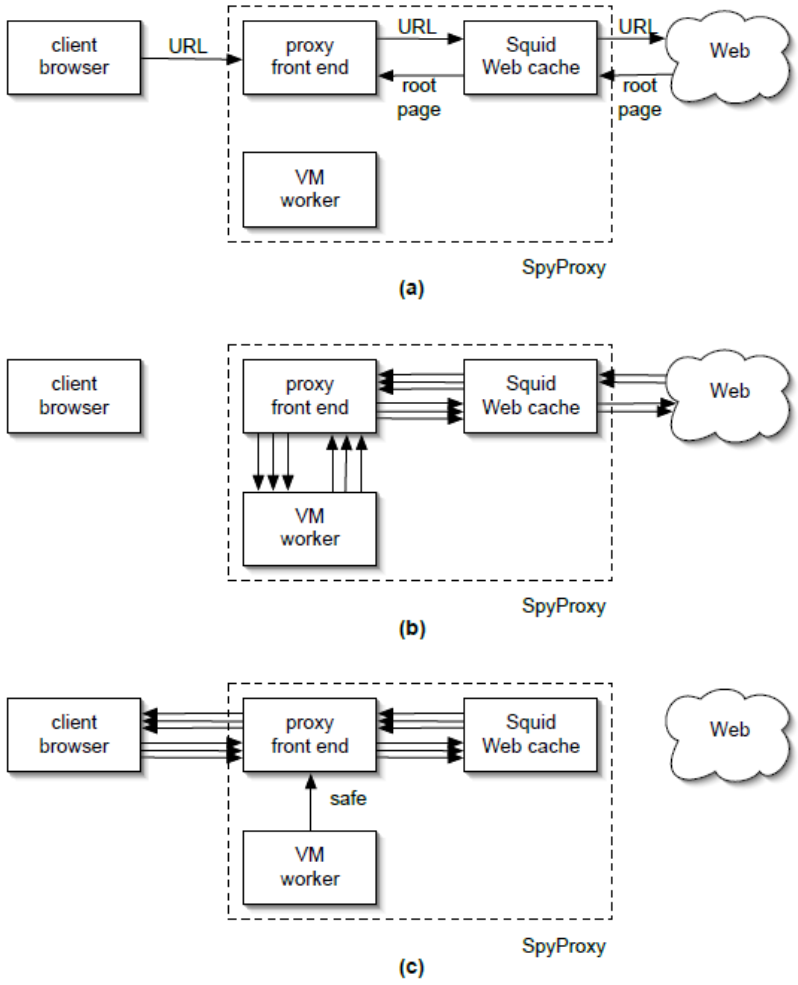


Figure 1 shows a phase (a), where web content is requested by a user, a phase (b), where content is forwarded to a virtual machine (VM) for investigation if the content cannot otherwise be declared safe "statically," and a phase (c), where content checked in the virtual machine and found to be safe is released to the requesting user. Ex. 1007, 2.

Importantly, the SpyProxy first evaluates the web page statically, before determining whether to forward the page to the virtual machine for dynamic analysis. Ex. 1007, 3. The static analysis checks content to determine if it is active, and contains "executable content, such as ActiveX, JavaScript, and other code," or is passive because it does not. *Id.* Pages that contain active content must be analyzed dynamically. *Id.* In the static

analysis, non-HTML content types are considered to be unsafe and sent for dynamic processing. *Id*. This check is primarily done for performance reasons, to filter out content that is unlikely to be harmful, but can be eliminated for better accuracy. *Id.* at 2–3.

If content cannot be deemed safe "statically," it is passed to the virtual machine, where the code is executed, and the tool determines if the code attempts any of the following actions: "the creation of a new process other than known helper applications, modifications to the file system outside of safe folders such as the browser cache, registry modifications, browser or OS crashes, and so on." Ex. 1007, 3.

2. Gribble (Ex. 1010)

Gribble describes a system which "looks for and identifies spyware-infected executables and Web pages on the Internet." Ex. 1010, 3:20–21. Four of the five listed authors of Moshchuk are listed as inventors on Gribble. Gribble states its system addresses three problems:
"(1) determining whether a Web object contains executable software; (2) downloading, installing, and executing that software within a virtual machine without direct user interaction; and, (3) analyzing whether the installation and execution of the software caused a spyware infection." *Id.* at 3:24–29.

Gribble describes the use of its system with a "web crawler," which searches the Web to find and download executable files. Ex. 1010, 5:35–38. The web crawler program uses the assumption that an executable file has one of the following characteristics: "(1) the content-type hypertext transfer protocol (HTTP) header provided by a Web server when downloading the object was associated with an executable (e.g., application/octet-stream); or,

(2) its URL contained an extension known to be associated with executables and installers (e.g., .exe, cab, or msi)." *Id.* at 5:39–46. Gribble also describes an alternative embodiment in which, "instead of performing the analysis . . . offline under the direction of a Web crawler, a similar analysis instead is performed on-the-fly, in response to a user requesting a Web page or attempting to download an executable file." *Id.* at 14:32–37.

3. Dubrovsky (Ex. 1011)

Dubrovsky relates to "cloud-based gateway anti-virus scanning." Ex. 1011, 2:8–9. Dubrovsky describes the use of firewalls to protect an entity's network from intrusion, which commonly use comparison with stored signatures, and web page ratings, to detect possible malware and prevent it from entering the network. *Id.* at 1:20–43. The limited storage of firewalls and complexity of managing multiple firewalls, however, is identified as a shortcoming of this approach. *Id.* at 1:43–51.

Figure 1, reproduced below, illustrates a general layout and operation of Dubrovsky's system.

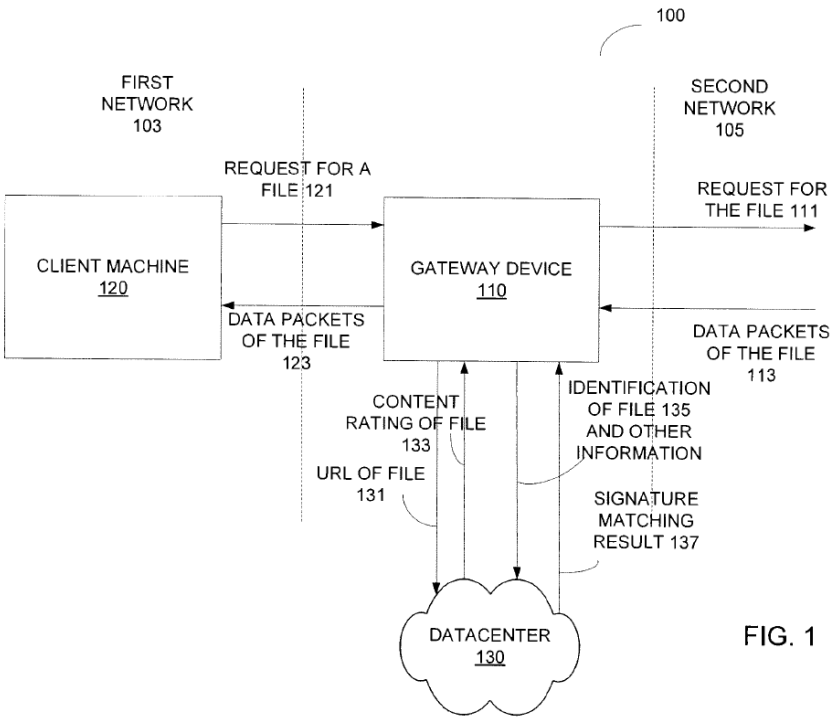


FIG. 1

Figure 1 shows a client machine 120, on a first network 103, which requests a file from a second network 105, through a gateway device 110. Ex. 1011, 3:28–34, 4:1–8. The gateway device 110 is coupled to a datacenter 130 which performs security screening related tasks, such as determining the content rating of webpages and performing signature matching. *Id*. at 3:34–44. Then, "gateway device 110 may also forward the path 131 of the file (e.g., the URL of the file) to the datacenter 130." *Id.* at 4:8–10. Using URL, "datacenter 130 looks up the content rating of the file," and if the datacenter finds the content rating, it "sends the content rating 133 to the gateway device 110," which may choose to block the request. *Id.* at 4:10–15.

If the request is forwarded to the second network, and a file is returned, as the packets of the file arrive at the gateway, the gateway "may generate an identification of the file based on the partial information of the file provided by the data packets 113 received." Ex. 1011, 4:37–41. The

identification may be based on a partial hash of the received file packets. *Id.* at 4:41–43. The identification is then forwarded to the datacenter 130, possibly along with other information such as the requested URL, and/or hostname or IP address of the server providing the file. *Id.* at 4:43–49. In order to improve efficiency, in some embodiments, the gateway may only send identifying information on "executables of software applications" to the datacenter. *Id.* at 4:49–56.

The datacenter may determine "if there is a high likelihood that the file contains malware," such as by "signature matching (e.g., pattern matching, hash comparison, etc.) on the identification 135." Ex. 1011, 4:57–61. The datacenter returns the result to the gateway, and if a match is found with records of malware file identifications, the gateway blocks the file from the client machine 120. *Id.* at 4:66–5:9.

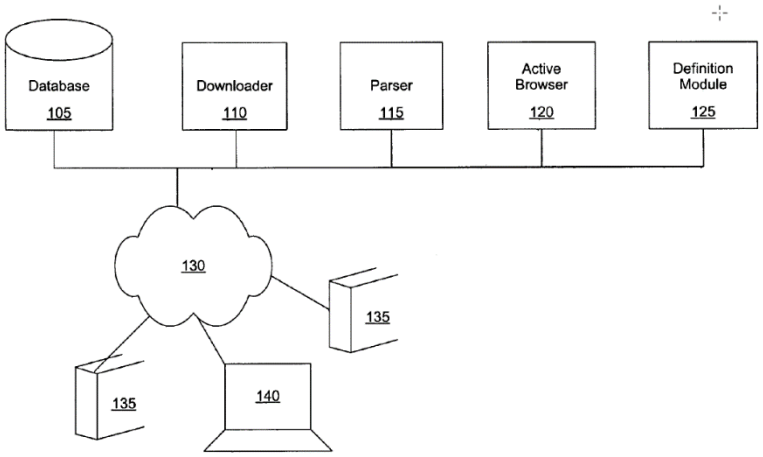
4. Bertman

Bertman concerns "searching out malware and generating corresponding malware definitions." Ex. 1012, 1:26–30. Bertman describes that its system operates to

> receive an initial URL associated with a Web site; download content from that Web site; identify any obfuscation techniques used to hide malware or pointers to malware; interpret those obfuscation techniques; identify a new URL as a result of interpreting the obfuscation techniques; and add the new URL to a URL database.

Id. at 2:8–15.

Bertman's Figure 1, reproduced below, provides a block diagram of its system components.



Bertman's Figure 1 illustrates the relationship between URL downloader 110, parsers 115, active browser 120, definition module 125, network 130, web servers 135, and protected computer 140. Ex. 1012, 2:45–50.

Bertman further describes downloading code from a URL, and sending it to three parsers: "an HTML parser, a JavaScript parser, and a form parser." Ex. 1012, 3:66–4:1. The HTML parser identifies "embedded URLs and other potential malware." *Id.* at 4:1–5. The JavaScript parser "can decode obfuscation techniques used by malware programmers to hide their malware from identification." *Id.* at 4:28–30. The form parser looks for potential malware in forms, such as "button click events that run a function rather than submitting information." *Id.* at 4:58–66.

Bertman also describes the use of an "active browser 120," which "surfs a Web site as a person would," but "any changes to the configuration of the active browser's computer system are recorded." Ex. 1012, 5:4–20. Changes may include "changes to a[n] operating system file, addition or removal of files, changing file names, changing the browser configuration, opening communication ports . . . a change to the WINDOWS registry file." *Id.* at 5:18–26.

Finally, definition engine 125 generates "malware definitions that are stored in the database and eventually pushed to the protected computers 140." Ex. 1012, 5:31–34. Bertman indicates that "applets, executable files, objects, and similar files can be added to the definitions." *Id.* at 5:62–63.

5. Campbell (Ex. 1013)

Campbell concerns "blocking unwanted software downloads." Ex. 1013, 1:7–9. Campbell discloses "a network gateway [] configured with one or more rules regarding handling of spyware, so as to afford protection for an associated network communicatively coupled to the Internet via the network-gateway by automatically implementing the rules to prevent downloads of spyware." *Id.* at 1:55–60.

Campbell notes that "spyware is often sent as executable files—such as those having .EXE (executable), .COM (command), or DLL (dynamic linked library) file extensions—or active content files—such as those having CAB (cabinet) and OCX (OLE control extension) file extensions." Ex. 1013, 4:52–58. Campbell keeps a "file extension list 316" which "associates file types with file extensions," and "includes at least the following well-known file extensions: .exe, .com, .dll, cab, .ocx, .bat, .cmd, .vbs, .vb, .pif, .hlp, .msi, .scr, .wsc, .wsh, .wsf, .hta, .class, .jar, .chm, .cpl, and .zip." *Id.* at 7:52–61.

Campbell further explains that because file extensions can be changed ("camouflaged"), "spyware scanner 204 is configured to scan incoming files for spyware signatures that cannot be hidden (e.g., through changes in file extensions)." Ex. 1013, 5:1–9. This may involve examining file "headers." *Id.* at 5:13–19.

E. Obviousness over Moshchuk and Gribble (Ground 1)

Petitioner argues that claims 1–9 of the '693 patent would have been obvious over the combination of Moshchuk and Gribble. Pet. 15. For the reasons discussed below, we determine Petitioner has not shown that the Moshchuk-Gribble combination teaches limitation 1c. Independent claims 3, 5 and 8 include limitations that are substantially similar to limitation 1c. For this reason, Petitioner relies on the same contentions for these limitations as it does for limitation 1c. *See* Pet. 38, 39–40, and 42. Similarly, Patent Owner applies the same arguments to limitation 1c as it does to the corresponding limitations of the other independent claims. *See* PO Resp. 3, 6. Thus, although we specifically refer to limitation 1c below in our discussion, our analysis applies equally to the corresponding limitations of independent claims 3, 5, and 8. The challenged dependent claims all depend from one of the aforementioned independent claims. Our conclusion as to claim 1, therefore, applies to claims 2–9 as well.

1. Petitioner's Proposed Moshchuk-Gribble combination

Petitioner's proposed Moshchuk-Gribble combination is based on the SpyProxy implementation of Moshchuk, which determines whether a webpage is active or passive by checking to see if the webpage contains executable content. Ex. 1007, 3. In Moshchuk, all non-HTML content is assumed to have executable content and is therefore forwarded to the VM worker for dynamic analysis. *Id*. In the combination proposed by Petitioner, however, instead of automatically directing all non-HTML content to be dynamically analyzed by the VM worker, the combined system would instead use Gribble's approach for identifying executables based on the file's HTTP header or URL extension. Pet. 21–22. In this proposed

combination, requested pages with executable content such as ActiveX, JavaScript, and other code, would be classified as active pages, as described by Moshchuk, but the determination of whether a page contained executable content would be based on Gribble's approach of looking at the HTTP header or to see if the URL contains an extension known to be associated with executables. Pet. 22 (citing Ex. 1007, 3 (2:38–43); Ex. 1010, 5:39–46).

2. Analysis of Independent Claims 1, 3, 5, and 8

As mentioned above, we determine Petitioner has not shown that the Moshchuk-Gribble combination teaches limitation 1c. Because this determination is dispositive of Petitioner's challenge over Moshchuk and Gribble, we begin directly by analyzing limitation 1c.

a) Parties' Arguments and Contentions

Limitation 1c recites "judging, by the network device, whether the file is an executable file according to at least one of: the request and the data stream carrying the file." Petitioner argues "Moshchuk's proxy front end comprises a 'static analyzer' which 'tries to determine whether the page is active or passive. Active pages include ***executable content***, such as ActiveX, JavaScript, and other code; passive pages contain no such interpreted or executable code.'" Pet. 29 (citing Ex. 1007, 3 (2:384–2, 18–29)). Thus, Petitioner argues "Moshchuk judges whether the requested file (Web page) is executable or not, and therefore discloses Element 1(c)." *Id*. at 29 (citing Ex. 1005 ¶¶ 242–43). Petitioner modifies Moshchuk with Gribble's "approach of dynamically analyzing files using the VM only if the file can be identified as executable based on the file's header or URL extension." *Id*. at 30 (citing Ex. 1010, 5:39–50). The combined system,

according to Petitioner, would classify active pages as those that include executable content based on Gribble's approach of looking to see if the HTTP header is associated with an executable (e.g., application/octet-stream), or the URL contains an extension known to be associated with executables and installers (e.g., .exe, .cab, or .msi). *Id*. at 30 (Ex. 1007, 3 (2:38–43); Ex. 1010, 5:39–46).

Petitioner argues that a person of ordinary skill in the art would be motivated to modify Moshchuk with Gribble's approach of identifying executable content based on the file's header or URL extension in performance-focused networks because Gribble's approach is less conservative and more performance-focused than Moshchuk's standard approach. Pet. 30–31 (citing Ex. 1005 ¶¶ 244–246).

Patent Owner argues that based on "how the Petition identifies various disclosures of the Moshchuk-Gribble combination," the Petition's Moshchuk-Gribble combination fails to judge that the requested file is an executable file. PO Resp. 7. Specifically, Patent Owner argues that in Petitioner's contentions "Petitioner identifies the 'URL' sent from the client browser to the SpyProxy server as the claimed 'request'" and that the "Petition expressly maps Moshchuk's 'root page' identified by the URL to the 'file' recited by claim." *Id.* at 7–8 (citing Pet. 26). Importantly, even if the requested file is considered to be the complete Web page, rather than just the root page, Patent Owner argues that the "'URL' identifies the root page itself" rather than any embedded objects of the root page which are not apparent to the browser until the browser begins downloading those objects. *Id*. at 8–9 (citing Ex. 2018 ¶¶ 53, 71, 77, 80, 82, 84–85, 95; Ex. 1007, 2–3, 6, 2:45–50, Fig. 2 9). Thus, according to Patent Owner, the requested file in

Moshchuk is an HTML root page that contains links to embedded objects, as opposed to the embedded object themselves. *Id*. at 9.

Patent Owner emphasizes that the Petition relies on Gribble's approach of looking to see if (1) the HTTP header is associated with an executable or (2) the URL contains an extension known to be associated with executables in order to classify pages as including executable content. PO Resp. 10 (citing Pet. 21–22). Patent Owner argues that neither of the two techniques relied upon from Gribble, when applied to Moshchuk, would have successfully distinguished active Web pages from passive Web pages. *Id*. at 10 (citing Ex. 2018 ¶ 86). Patent Owner turns to Gribble's first technique, which looks to the "Content-type hypertext transfer protocol header" to determine if a Web object was an executable (Ex. 1010, 5:40–42). Here, Patent Owner reminds us that in the Petition's Moshchuk-Gribble combination, the requested file is an HTML root page. *Id*. at 12 (citing Ex. 1007, 3 (Fig. 1, 1:24–34, 6 (2:45–50); Ex. 2018 ¶¶ 79–81, 83; Ex. 2013 ¶¶ 52, 55, 59, 79). Therefore, according to Patent Owner, "the 'Content-Type' line of the HTTP header for the HTTP response message for the requested root page in Moshchuk would have indicated that the returned file is an HTML file" and "there would have been no way of determining if the returned Web page contained active content or passive content based on the indication in the HTTP response header." *Id*. at 13 (citing Ex. 2018 ¶ 94; Ex. 2016, 11).

Patent Owner next turns to Gribble's second technique, which examines the URL to determine whether it "contained an extension known to be associated with executables and installers (e.g., .exe, cab, or msi)." PO Resp. 15–17 (quoting Ex. 1010, 5:39–46). Once again, Patent Owner

reminds us that the request in Petitioner's Moshchuk-Gribble combination is a URL for an HTML root page. *Id*. at 15–16 (citing Ex. 1007, 2 (1:1–33), 6 (2:45–48); Ex. 2018 ¶¶ 104–105). Patent Owner argues that "the URL for an HTML file would have included an extension of .html, .htm, or .xhtml" and that "there is no evidence of record that the 'URL' for an HTML file would have 'contained an extension known to be associated with executables and installers (e.g., .exe, cab, or msi).'" *Id*. at 16 (citing Ex. 2018 ¶¶ 82, 106–108; Ex. 2019, 43:7–16; Ex. 1010, 5:39–46). Thus, because the request in the Moshchuk-Gribble combination is a request for an HTML root page, Patent Owner argues that the extension in the URL would have been associated with HTML files (e.g., .html or .htm) and not an extension known to be associated with executables (e.g., .exe, cab, or msi) and therefore there would have been no way to determine if a requested Web page in the Moshchuk-Gribble combination was an active or passive page. *Id*. at 16.

In summary, Patent Owner argues that because the request in Petitioner's Moshchuk-Gribble combination is a URL for an HTML root page, neither the Content-Type field of the HTTP header nor the URL extension would indicate whether the requested Web page contained executable content.

In its Reply Brief, Petitioner contends that the Petition relies on "Gribble for its suggestion of looking at file headers *generally*, to determine file type . . . not on the specific type of file header Gribble mentions." Pet. Reply 8–9 (citing Pet. 23; Ex. 1068 ¶¶ 50–61). Petitioner argues that executable script, such as JavaScript, can be external or can be included directly in the HTML of a root page. *Id*. at 4–5 (citing Ex. 1068 ¶¶ 41–43;

Ex. 1053, 485; Ex. 1054, 463). In the case where script is embedded directly into the root page, Petitioner argues that such a root page itself, even without considering embedded objects, would be understood to be an executable file. *Id*. at 10 (citing Ex. 1068 ¶¶ 65–66; Ex. 1053, 485). Petitioner explains that in addition to having HTTP response headers, a web page has a separate file header referred to as a HEAD section or HEAD element and that this HEAD section would therefore be considered a file header. *Id*. at 11 (citing Ex. 1068 ¶¶ 68–72). Petitioner argues that in some instances, when a script is directly embedded in an HTML document, it can be embedded directly in the HEAD section as well as in the body of the HTML document. *Id*. at 11 (citing Ex. 1053, 46). When an HTML document contains a script, the HTML document declares what scripting language is used in the Content-Script-Type field of either the HEAD section or an HTTP header. *Id*. at 12 (citing Ex. 1068 ¶¶ 75–76; Ex. 1055, 239). Based on the above, Petitioner argues that a script and the script declaration can be included directly in the HEAD section and therefore the HEAD section would indicate the presence of active content. *Id*. at 13 (citing Ex. 1055, 239–240).

Moreover, Petitioner argues that the HTTP headers can also include scripting language declarations in the Content-Script-Type field which would indicate the likely presence of script. Pet. Reply 14 (citing Ex. 1068 ¶¶ 84–89; Ex. 1050, 42:14–22). Therefore, HTTP headers with the Content-Script-Type field would be judged by the Moshchuk-Gribble combination as active content. *Id*. at 14.

Finally, Petitioner argues that in addition to the root page alone, the Moshchuk-Gribble combination also analyzes the complete Web page, including the external objects, and that this analysis renders the challenged

claims obvious. Pet Reply 15. Petitioner argues that the Petition relied on not just the root page, but also the Web page as the recited "file." *Id*. at 15–16 (citing Pet. 25). According to Petitioner, the URL sent by the client is a request for a complete Web page and that the URL request would therefore necessarily include the URL requests for external content. *Id*. at 16 (citing Ex. 1068 ¶¶ 92–93). Additionally, the file header for the Web page includes the combined file headers for the external content. *Id*. at 16 (citing Ex. 1068 ¶ 94). Thus, Petitioner argues, "[t]he Moshchuk-Gribble combination would evaluate the URLs and headers of any linked content, separately from the root page itself, which would also indicate if the Web page contained active content and therefore is an executable file." *Id*. at 16 (citing Ex. 1068 ¶¶ 95–97).

In its Sur-reply, Patent Owner argues that Petitioner's arguments relying on anything other than analyzing the Content-type field of the HTTP header are new belated arguments that should be disregarded. PO Sur-reply 6–8. Thus, Patent Owner argues that Petitioner's argument stating that it relies on Gribble for a suggestion to look at file headers generally is a belated theory. *Id*. at 6. According to Patent Owner, the Petition clearly relied on Gribble's approach for identifying executables based on the file's Content-type field of the HTTP header, not on the file's HEAD section and not on the Content-Script-Type field of the HTTP header. *Id*. at 6, 10–11 (citing Pet. 21–22). Furthermore, Patent Owner argues that Petitioner's reliance on the Content-Script-Type field of the HTTP header is derived from a new exhibit (Exhibit 1055) absent from the Petition and should therefore not be considered. *Id*. at 10 (citing Pet. Reply, 12–13 (citing Ex. 1055)). Additionally, Patent Owner argues that Petitioner's arguments that

the Moshchuk-Gribble combination would evaluate not just the URL and header of the root page, but also the URLs and headers of the linked content, is also a new argument that was absent in the Petition. *Id*. at 12.

Regardless of whether Petitioner's arguments are impermissibly new, Patent Owner argues, even if Moshchuk were modified to analyze the Head section of a file, such a modification would lead to the vast majority of active pages (those pages including or linking to scripts in the body but no in the Head) to erroneously pass through without being checked for maliciousness. PO Sur-reply 9. Patent Owner argues that such a modification to Moshchuk would be contrary to Moshchuk's stated purpose. *Id*. at 10 (citing Ex. 2013 ¶¶ 115–116). Similarly, Patent Owner argues that even if analyzing the Content-Script-Type field was not a new theory, examining this field would not distinguish active and passive pages. *Id*. at 11–12. This is because Exhibit 1055 "merely teaches that ***if*** a requested file includes scripts, the browser can determine the scripting language to use by looking to see 'if any HTTP headers specify the 'Content-Script-Type.'"" *Id*. at 11 (citing Ex. 1055, 239). Finally, Patent Owner argues that Moshchuk is clear that it analyzes only the root page to perform the static safety check, not the URLs and headers of linked content in the root page. *Id*. at 12–15 (citing Ex. 1007, 3 (1:24–53), Fig. 1(a); Ex. 2018 ¶¶ 45, 60–61, 77, 92–95).

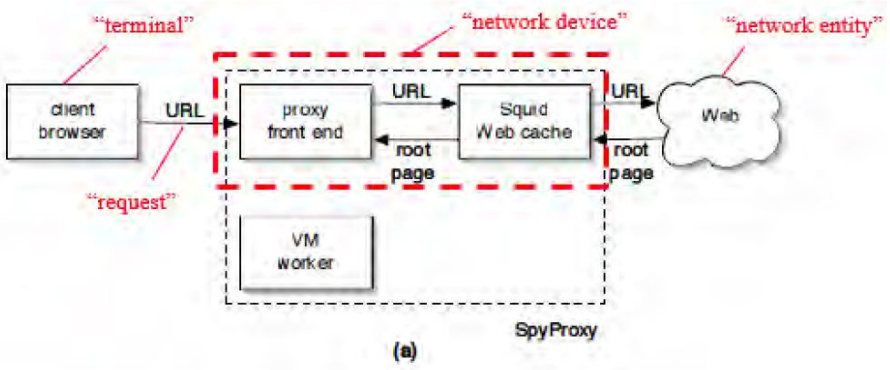
Patent Owner argues that "there is no indication in the HTTP response header as to whether the requested HTML page includes active embedded objects, or whether the HTML page includes embedded objects at all." PO Resp. 12 (Ex. 2018 ¶¶ 92, 94–101; Ex. 2016, 10–11).

b) Analysis

We begin our analysis by first determining whether Petitioner has relied on any impermissible new arguments, and if so, whether any remaining permissible arguments are persuasive in showing the Moschchuk-Gribble combination teaches limitation 1c. Petitioner has the burden to demonstrate sufficiently in the Petition that the cited prior art renders the challenged claims unpatentable, including showing that the Petition's contentions are supported by evidence. *See* 35 U.S.C. § 314(a); *see also Harmonic Inc.*, 815 F.3d at 1363 (Fed. Cir. 2016) ("In an IPR, the petitioner has the burden *from the onset* to show *with particularity* why the patent it challenges is unpatentable." (emphases added) (citing 35 U.S.C. § 312(a)(3))); *Intelligent Bio-Systems, Inc. v. Illumina Cambridge Ltd.*, 821 F.3d 1359, 1369 (Fed. Cir. 2016) (requiring "*the initial petition* identify '*with particularity*' the 'evidence that supports the grounds for the challenge to each claim.'" (emphases added)). "All arguments for the relief requested in a motion must be made in the motion. A reply may only respond to arguments raised in the . . . patent owner response, or decision on institution." 37 C.F.R. § 42.23(b).

The first issue we examine is whether, as Patent Owner argues, Petitioner has identified Moshchuk's root page as the recited "file" in the Petition (PO Resp. 7–8 (citing Pet. 26)) or whether, as Petitioner contends, it relied on not just the root page, but also the Web page as whole as the recited "file" (Pet Reply 15–16 (citing Pet. 25)). Petitioner states that the Moshchuk-Gribble combination receives "an HTTP request with a URL ('a request') for a Web page ('a file')," citing specifically to Figure 1(a) of Moshchuk. Pet. 25. Here, Petitioner identifies Moshchuk's HTTP request

with a URL as the recited "request" and a Web page as the recited "file." Later on in the Petition, referring again to Figure 1(a) of Moshchuk, Petitioner explains that the URL is used to fetch the requested file, and this time, identifies the root page as the recited "file." *See id*. at 28 ("as shown in Moshchuk's Figure 1(a) . . . the Squid Web cache receives the URL . . . uses the URL to fetch the requested file (root page), and provides the requested file (root page) back to the proxy front end."). In both instances, Petitioner relies on Figure 1(a) of Moshchuck. Figure 1(a) of Moshchuk, as annotated by Petitioner, is reproduced below:



Id. at 26. As can be seen in the above figure, Petitioner identifies the HTTP request with the URL as the recited "request," and in response to that request, the Web server (network entity) returns a root page.

In summary, we have one instance where Petitioner identifies a "Web page" as the recited "file," (Pet. 25) and another instance where Petitioner identifies the "root page" as the recited file (Pet. 26). But in both instances, Petitioner relies on Figure 1(a) of Moshchuk, which ultimately shows that it is the root page that is returned in response to the request for a file. Accordingly, we determine in light of the totality of Petitioner's contentions, that the Petition identifies the root page as the recited file. We therefore agree with Patent Owner that Petitioner's argument in its Reply Brief relying

on more than just the root page is an impermissible new argument. We will, therefore, disregard this argument and the related argument that Moshchuk's URL is a request for a complete Web page that would necessarily include the URL requests for external content. Pet. Reply 16 (citing Ex. 1068 ¶¶ 92–93).

Next, we examine what the Petition identifies from Gribble and whether, as Patent Owner contends, arguments relying on anything other than Gribble's Content-type field of the HTTP header are new impermissible arguments. The Petition states "[i]nstead of automatically directing all non-HTML content to be dynamically analyzed by the VM worker, as described in Moshchuk, the combined system would instead utilize Gribble's approach of identifying executables based on the file's HTTP header or URL extension. Pet. 21–22. Here, Petitioner clearly relies on Gribble's HTTP header to determine whether a file was an executable.

In another instance, the Petition states that "the Moshchuk-Gribble combination adopts Gribble's approach of dynamically analyzing files using the VM only if the file can be identified as executable based on the file's header or URL extension." Pet. 30. Although this statement appears to be more general in its scope, relying on "the file header" rather than specifically the file's HTTP header, the same page of the Petition immediately specifies that Gribble's approach is to "see if the HTTP header is associated with an executable." *Id*. (citing Ex. 1010, 5:39–46). Furthermore, as support for combining Moshchuk with Gribble, Petitioner relies on Dr. Nielson's testimony which again refers specifically to the content-type field of Gribble's HTTP header. Ex. 1005 ¶ 245 (stating "Gribble's proxy ("network device") could judge whether the file was executable based on the

content-type in the response or the filename's extension in the request."). Both the Petition and Dr. Nielson rely on the same passage from Gribble to show that a file could be classified as an executable by looking to its HTTP header. This passage from Gribble states

> it was assumed that a Web object was an executable if either: (1) the ***Content-type hypertext transfer protocol (HTTP) header*** provided by a Web server when downloading the object was associated with an executable (e.g., application/octet-stream); or, (2) its URL contained an extension known to be associated with executables and installers (e.g., .exe, .cab, or .msi).

Ex. 1010, 5:40–46 (emphasis added).

The contentions from the Petition, taken together with Dr. Nielson's initial testimony and the underlying disclosure from Gribble show that the Petition relies specifically on examining Gribble's content-type HTTP header to determine whether the associated file is an executable. Thus, we agree with Patent Owner that Petitioner's arguments in the Reply Brief relying on file headers generally, or those relying on other types of headers, such as the HEAD section or element, or other fields of the HTTP header, such as the Content-Script-Type field, are new arguments not presented in the Petition. Because these arguments were not raised in the Petition, it is impermissible to raise them in the Reply Brief. *See* Consolidated Trial Practice Guide[3] at 73–75.

At the oral hearing, Petitioner stated that it was appropriate for them to raise its arguments related to the Content-Script-Type field of the HTTP header in its Reply Brief. Petitioner likened its arguments to those in *Apple, Inc., v. Andrea Elec. Corp.*, 949 F.3d 697 (Fed. Cir. 2020). Petitioner argues

[3] Available at https://www.uspto.gov/TrialPracticeGuideConsolidated.

that the Content-Script-Type field of the HTTP header is just a different field of the same header as the Content-Type field mentioned in Gribble. *See* Tr. at 61:17–62:15 (arguing that "we're talking about the same header here, different line" and that it would be reasonable to "look at the various available lines per the HTTP standard.")

We disagree. In *Apple*, the reply "relie[d] on the same algorithm from the same prior art reference to support the same legal argument." *Apple*, 949 F.3d at 706. The reply in *Apple* did "not cite any new evidence or 'unidentified portions' of the [prior art] reference." *Id*. The same is not true here. Initially, we note that Petitioner's reliance on *Apple* at the oral hearing does not extend to its theories which involve headers other than the HTTP header (e.g. HEAD section or element) as those theories clearly do not rely on the "same header, different line." Petitioner's theory relying on the Content-Script-Type field of the HTTP header involves the situation where a script is directly included the HTML of the root page. Pet. Reply 4–5 (citing Ex. 1068 ¶¶ 41–43; Ex. 1053, 485; Ex. 1054, 463; Ex. 1056, 6). But neither Moshchuk nor Gribble discusses scripts directly embedded in an HTML page. Instead, in order to make this argument, Petitioner relies on newly submitted Exhibits 1053[4], 1054[5], 1055[6], and 1056[7] which explain that a script can be embedded in an HTML page (Ex. 1054, 463) and that the default scripting language can be specified in the Content-Script-Type

[4] Duckett, *Beginning HTML, XHTML, CSS, and JavaScript*, Wiley Publishing (2010).
[5] Robbins, *Learning Web Design*, 4th ed., O'Reilly.
[6] HTML 4.0 Specification, W3C, revised April 24, 1998.
[7] Flanagan, *JavaScript, The Definitive Guide*, 5th ed., O'Reilly (2006)

header (Ex. 1055, 239; Ex. 1056, 4). Gribble makes no mention of scripts embedded in an HTML page nor of the Content-Script-Type header.

Even if we accept that Petitioner relied on Gribble for the suggestion of looking at file headers generally to determine file type, the specific header and field provided as an example of this general proposition in the Petition (Gribble's the Content-Type HTTP header) does not actually show the general proposition when combined with Moshchuk for the reasons stated above. Petitioner implicitly acknowledges this, stating that "[t]he Petition relies on the general principle in Gribble that file headers can also be used to identify file types, *not on the specific type of file header Gribble mentions*." Pet. Reply 9 (emphasis added). But if the specific type of file header in Gribble is not being relied on, this leaves Petitioner in the awkward position that if we accept its theory relying on the Content-Script-Type header (rather than the Content-Script header) then Gribble adds very little evidentiary support for its contentions when compared to its newly introduced exhibits. For these reasons, we disagree with Petitioner that its situation is similar to that in *Apple*.

Having determined which of Petitioner's arguments are new, we now consider those arguments that were properly presented in the Petition. Petitioner argues that in the Moshchuk-Gribble combination "the static analyzer of the proxy front end 1 would determine whether the requested page is active or passive, per Moshchuk" but would do so based on Gribble's approach of looking to see if the HTTP header is associated with an executable (e.g., application/octet-stream), or the URL contains an extension known to be associated with executables and installers (e.g., .exe, .cab, or .msi). Pet. 30 (Ex. 1007, 3 (2:38–43); Ex. 1010, 5:39–46).

As we summarized above, Patent Owner argues that because the request in Petitioner's Moshchuk-Gribble combination is a URL for an HTML root page, neither the Content-Type field of the HTTP header nor the URL extension would indicate whether the requested Web page contained executable content. PO Resp. 10–16. Dr. Davis provides credible testimony that because, in Moshchuk, the file that is provided in response to the request is an HTML root page, the HTTP header for the file would have merely indicated that the returned file is an HTML file, which is not a file type associated with being an executable, and would not have indicated whether the file contains active or passive content. Ex. 2018 ¶¶ 94–100. Similarly, Dr. Davis provides credible testimony that "[a] POSITA would have known that the URL for an HTML file would have generally included an extension of .html or .htm" and that such extensions would not have been associated with executables. *Id*. at ¶ 106. Thus, Dr. Davis concludes "[t]he resulting Moshchuk-Gribble combination therefore would not have been able to determine if a requested Web page was an active page or a passive page based on the analysis of the URL." *Id*. at ¶ 107.

Even if we were to consider Petitioner's impermissible, new theory relying on the Content-Script-Type field of the HTTP header, the theory Petitioner identified at the oral hearing as appropriate subject matter for a reply under *Apple* (Tr. 61:17–62:15), we determine such theory does not teach "judging . . . whether the file is an executable file." Petitioner argues that a scripting language declaration in the Content-Script-Type field of an HTTP header would indicate the *likely* presence of a script and thus, executable content. Pet. Reply 14 (citing Ex. 1068 ¶¶ 84–89). Patent Owner argues that a scripting language declaration in the Content-Script-Type field

does not necessarily indicate the presence of a script. Instead, according to Patent Owner, it only teaches that *if* a requested file includes scripts, the browser can look to the scripting language declaration to determine the language of the script. PO Sur-reply 11. Based on the arguments of both parties and the evidence presented, we determine that the Content-Script-Type field does not judge whether a file is executable. The Content-Script-Type field indicates to a browser the default language for scripts so that browser can interpret the script if it encounters one in an HTML page. *See* Ex. 1055, 239. But the Content-Script-Type field does not indicate that an HTML page will necessarily include a script. Thus, the Content-Script-Type field alone does not indicate that the file is an executable file. Accordingly, even if we were to consider Petitioner's theory relying on the Content-Script-Type field of an HTTP header as appropriate, which, for the reasons stated above we do not, we are still unpersuaded by Petitioner's arguments.

Accordingly, in light of all the arguments and evidence presented by both parties, we are unpersuaded by Petitioner that the combination of Moshchuk and Gribble teaches limitation 1c.

c) Conclusion as to Obviousness over Moshchuk-Gribble (Ground 1)

For the reasons stated above, we determine that Petitioner has not shown that the Moshchuk-Gribble combination teaches limitation 1c and the corresponding limitations of independent claims 3, 5, and 8. For the same reasons, we determine Petitioner has not shown the Moshchuk-Gribble combination teaches the limitations of the challenged dependent claims. Accordingly, we determine Petitioner has not shown by a preponderance of the evidence that claims 1–9 would have been obvious over the Moshchuk-Gribble combination.

F. Obviousness over Dubrovsky and Bertman (Ground 2A)

Petitioner argues that claims 1, 3, 5, and 6 of the '693 patent would have been obvious over the combination of Dubrovsky and Bertman. Pet. 15.

1. Petitioner's Proposed Dubrovsky-Bertman Combination

Petitioner's proposed Dubrovsky-Bertman combination "is based on the cloud-based virus scanning of Dubrovsky, but in which Dubrovsky's data center is enhanced with Bertman's teachings of downloading material associated with the URL in order detect and generate signatures/definitions for previously unknown malware." *Id*. at 49. In the combination, when Dubrovsky receives a file it has not encountered before, there would be no matching signature in Dubrovsky's datacenter, and in such a situation the system would pass the requested URL to Bertman's network where the URL would be used to download the material and subsequently evaluate the material for malware, and if present, would generate a signature and pass that signature back to Dubrovsky's signature database. *Id*. at 49–50 (citing Ex. 1005 ¶ 323). Dubrovsky's signature database would then contain a matching signature to the file in question and would therefore return a signature matching result to the gateway device after which the gateway device would proceed to block the file. *Id*. at 50 (citing Ex. 1005 ¶ 324).

Alternatively, Petitioner proposes that Dubrovsky's datacenter could be modified to incorporate Bertman's downloader, parser and active browser and such a system would operate similarly to the above described system. Pet. 50–51 (citing Ex. 2005 ¶ 325).

2. Analysis of Claims 1, 3, 5, and 6

Petitioner argues the Dubrovsky-Bertman combination teaches the limitations of claim 1. Pet. 53–65. Patent Owner argues that the combination fails to teach at least limitations 1c and 1f. PO Resp. 36–60. Below we analyze Petitioner's contentions with respect to each of the limitations of claim 1. Although we ultimately conclude that Petitioner fails to make a persuasive showing with respect to limitation 1f, such that Petitioner fails to show that claim 1 is unpatentable, we discuss all limitations because our analysis bears on our analysis of independent claim 8 below. Independent claims 3 and 5 include limitations that are substantially similar to the limitations of claim 1 and claim 6 depends from claim 5. *See* Ex. 1001, 8:50–9:12, 9:21–10:14. For this reason, both parties present substantially the same arguments for these claims. *See* Pet. 65–69; PO Resp. 36–61. Our analysis for claim 1 therefore applies to claims 3, 5 and 6 as well.

a) [1pre] A method for alerting against unknown malicious codes, the method comprising:

Petitioner argues that, to the extent the preamble is limiting, the Dubrovsky-Bertman combination discloses performing cloud-based gateway anti-virus scanning in which Dubrovsky's datacenter determines whether a file contains malware by using Bertman's teaching of downloading material associated with the URL in order to generate a signature for previously unknown malware. Pet. 53 (citing Ex. 1011, 2:8–9, 4:57–59; Ex. 1005 ¶¶ 320–321, 334–336). Patent Owner does not separately dispute Petitioner's arguments with respect to the preamble.

We agree with Petitioner's aforementioned arguments. Based on the evidence of record and the parties' arguments we are persuaded by Petitioner that the Dubrovsky-Bertman combination teaches the preamble. As such, we need not determine whether the preamble is limiting.

b) [1a] receiving, by a network device, a request sent by a terminal for obtaining a file from a network entity and a data stream carrying the file;

Petitioner argues "the Dubrovsky-Bertman combination includes a 'network device' in the form of Dubrovsky's gateway device 110 and 'network entity' in the form of the host/server of the second network 105." Pet. 54 (citing Ex. 1011, 4:24-26; Ex. 1005 ¶¶ 337–338). Petitioner argues Dubrovsky's gateway device receives a request from a client machine (the claimed "terminal") for a file when a user of the client machine clicks on a link to request content such as another webpage, a document, a song, a video, a picture, an executable of a software application. *Id*. at 55 (citing Ex. 1011, 4:1–6). Petitioner further argues that Dubrovsky's gateway receives data packets of the requested file from the host/server of the second network and therefore receives a "data stream carrying the file." *Id*. (citing Ex. 1005 ¶¶ 337–340). Patent Owner does not separately dispute Petitioner's arguments with respect to this limitation.

We agree with Petitioner's aforementioned arguments. Based on the evidence of record and the parties' arguments we are persuaded by Petitioner that the Dubrovsky-Bertman combination teaches limitation 1a.

c) [1b] recording, by the network device, a source path carried in the request, wherein the network entity provides the file on the source path;

Petitioner argues the client's request in Dubrovsky can be a hyperlink in a webpage and therefore carries a "source path" on which the network entity would provide the requested file. PO Resp. 56 (citing Ex. 1005 ¶¶ 341–42). Petitioner argues Dubrovsky's gateway and datacenter would record the source path of the request in order to perform content rating screening. Pet. 56 (citing Ex. 1005 ¶¶ 345–346). Patent Owner does not separately dispute Petitioner's arguments with respect to this limitation.

We agree with Petitioner's aforementioned arguments. Based on the evidence of record and the parties' arguments we are persuaded by Petitioner that the Dubrovsky-Bertman combination teaches limitation 1b.

d) [1c] judging, by the network device, whether the file is an executable file according to at least one of: the request and the data stream carrying the file;

Petitioner argues that for efficiency, Dubrovsky sends identification of only executables of software applications to the datacenter because malware is most likely found in executables. Pet. 57 (citing Ex. 1011, 4:45–56). According to Petitioner, Dubrovsky's gateway generates an identification of the file based on information provided by the data packets it receives. *Id*. at 58 (citing Ex. 1011, 4:1–8, 4:37–41, 5:42–44, 6:24–27). Thus, Petitioner argues that a person of ordinary skill would understand that Dubrovsky judges a file to be an executable file based at least on the data stream carrying the file. *Id*. (citing Ex. 1005 ¶¶ 347–349).

Patent Owner argues that Petitioner "cites no portion of Dubrovsky (or Bertman) that discloses determining that a file is an executable file 'according to at least one of: the request and the data stream carrying the

file.'" PO Resp. 37 (citing Ex. 2018 ¶¶ 148–156). Patent Owner acknowledges that Dubrovsky's gateway generates an identification of the file based on information provided by the data packets it receives, but argues there is no discussion in Dubrovsky of using this generated identification to determine if the file is an executable file. *Id*. at 38 (citing Ex. 2018 ¶¶ 152–153); *see also id.* at 39 ("There is no description of the 'identification of the file' being processed or analyzed to determine that the file is an executable file. . . . There is no discussion of using the generated identification to actually judge that the file ***is*** an executable file."). According to Patent Owner, it is not inherent or necessary for the identification of the file to be used to determine whether the file is executable and that there may be ways to identify that a file is executable other than according to the data stream carrying the file, such as having the network entity (host/server) send a communication to the network device (gateway). *Id*. at 40 (Ex. 2018 ¶ 156).

In its Reply Brief, Petitioner argues that Dubrovsky teaches limitation 1c both explicitly and implicitly. Pet. Reply 18 (citing Ex. 106 ¶¶ 116–124). Petitioner argues that Dubrovsky discloses that the identification of the file is generated from the data packets received by the gateway. *Id*. at 19 (citing Ex. 1011, 4:41–43). Because the identification is itself a hash of the file, and because Dubrovsky makes a judgment about whether the identification corresponds to an executable, Petitioner argues Dubrovsky explicitly discloses judging if the file is executable using the data stream carrying the file. *Id*. at 20 (citing Ex. 1068 ¶ 118). Furthermore, Petitioner argues that because the only external information relating to the requested file that Dubrovksy's gateway receives is the request 121 and the data packets 113, Dubrovksy implicitly discloses using the request or the data packets to judge

whether the file is executable. *Id*. at 22 (citing Ex. 1011, Fig. 1; Ex. 1068 ¶¶ 116–124).

Patent Owner responds that a person of ordinary skill would expect many communications to be received at the gateway other than just the request and data packets and that there is no justification for the assumption that the request or the data packets must be used to determine if the file is executable. PO Sur-reply 16–17 (citing Ex. 2018 ¶¶ 148–156).

Based on the evidence of record, we are persuaded by Petitioner's argument that the Dubrovsky-Bertman combination teaches limitation 1c. Dubrovsky discloses that its gateway "may generate an identification of the file based on the partial information of the file provided by the data packets received" and that it "then sends the identification 135 of the file to the datacenter 130." Ex. 1011, 4:39–45. Dubrovsky clearly teaches that it generates an identification of the file based on the datastream carrying the file. Dubrovsky further discloses that, to improve efficiency, it may send identifications of some predetermined types of files to the datacenter, including sending only identifications of executables because malware is most likely found in executables. *Id*. at 4:49–56. Thus, Dubrovsky teaches that it identifies the file using the data stream and also identifies whether the file is an executable. Petitioner also points out the only explicitly disclosed inputs received by the gateway relating to the requested file are the data packets and the URL request. Pet. Reply 22 (citing Ex. 1011, Fig. 1; Ex. 1068 ¶¶ 116–124). Dr. Nielson provides credible testimony that because these are the only external inputs to the gateway, one of ordinary skill would have understood that Dubrovsky uses the URL request or the data packets to judge whether the file is executable. Ex. 1068 ¶¶ 116–124.

Patent Owner argues that Dubrovsky's gateway would receive many other communications besides the URL request and the data packets of the file. PO Sur-reply 16–17 (citing Ex. 2018 ¶¶ 148–156). We agree with Dr. Nielson, however, that communications from other client devices would have nothing to do with the gateway's processing of the file being requested by the client and the information used to generate an identification of that file. Ex. 1068 ¶ 122.

Based on Dr. Nielson's testimony and Dubrovsky's disclosure that the gateway generates an identification of the file using the data packets received and that it also identifies whether the file is an executable file, one of ordinary skill in the art would understand that this latter identification of the file as an executable would be based on the data packets as well. Accordingly, we are persuaded by Petitioner that the Dubrovsky-Bertman combination teaches limitation 1c.

e) *[1d] when the network device judges the file is an executable file, sending, by the network device, first alert information that carries the source path to a monitoring device;*

Petitioner argues Dubrovsky discloses a gateway device (the claimed network device) sending file identifications (the claimed first alert information) to the datacenter (the claimed monitoring device). Pet. 58–59 (citing Ex. 1011, 4:37–45, 4:45–49, 4:49–56; Ex. 1005 ¶¶ 350–51). Patent Owner does not separately dispute Petitioner's arguments with respect to this limitation.

We agree with Petitioner's aforementioned arguments. Based on the evidence of record and the parties' arguments, we are persuaded by Petitioner that the Dubrovsky-Bertman combination teaches limitation 1d.

f) [1e] receiving, by the network device, second alarm information sent by the monitoring device after further detecting the file downloaded according to the source path by the monitoring device; and

Petitioner argues that the Dubrovsky-Bertman combination teaches this limitation by disclosing that Dubrovsky's datacenter (the claimed monitoring device) sends the results of pattern matching between the file and malware signatures (the claimed second alarm information) to the gateway (the claimed network device). Pet. 60–61 (citing Ex. 1011, 4:57–59, 4:60–66, 5:6–9, 6:18–38, Fig. 3B). Petitioner argues that the Dubrovsky-Bertman combination would also download the file using the source path as disclosed by Bertman which downloads and evaluates material associated with the URL. This would occur in cases where the file has not been encountered before. Once downloaded the system would evaluate the material and send second alarm information based on whether the system detected behavior consistent with malware. *Id*. at 62–63 (citing Ex. 1005 ¶ 357). Patent Owner does not separately dispute Petitioner's arguments with respect to this limitation.

We agree with Petitioner's aforementioned arguments. Based on the evidence of record and the parties' arguments, we are persuaded by Petitioner that the Dubrovsky-Bertman combination teaches limitation 1e.

g) [1f] intercepting, by the network device, a) the executable file according to the second alarm information comprising a maliciousness of the executable file; or b) the executable file and packets transmitted in a Botnet according to the second alarm information comprising the maliciousness of the executable file and Botnet topology information.

Petitioner argues that in the Dubrovsky-Bertman combination, if the datacenter indicates that there is a match, then the gateway device determines that the file is likely to contain malware and thus, blocks the file

from the client machine. Pet. 64 (citing Ex. 1011, 2:19–27, 5:6–9, 5:57–6:2). As explained above, however, for files that have never been encountered before there would be no match. *Id*. at 49–50 (citing Ex. 1005 ¶ 323). In such a situation, according to Petitioner, the Dubrovsky-Bertman combination would use Bertman's analysis to determine if the file is malicious. Here, Bertman would be provided with the URL to search. *Id*. at 62–64. Bertman would then download and evaluate material associated with the URL. *Id*. If Bertman detects behavior consistent with malware, it will generate a malware signature that will be pushed back to update Dubrovsky's signature database so that the database would then contain a matching signature for the file. *Id*. at 64–65 (citing Ex. 1005 ¶ 323). Dubrovsky would then return a signature matching result to the gateway device as second alarm information after which the gateway device would proceed to block or allow the file. *Id*. at 65 (citing Ex. 1005 ¶¶ 323–324).

Patent Owner argues that, as proposed, Petitioner's Dubrovsky-Bertman combination would never have the opportunity to intercept a malicious file because the results of Bertman's analysis would come too late, after the file was already transferred from the gateway to the client. PO Resp. 42, 54–60. This is because, as Patent Owner emphasizes, Dubrovsky's gateway device forwards the data packets it receives to the client machine as they are received and that its "signature check is performed '***in parallel with forwarding the data packets to the client machine***.'" *Id*. at 42 (citing Ex. 1011, 4:24–38, 2:12–16; Ex. 2018 ¶ 159). Patent Owner argues that Dubrovsky's gateway device continues to forward the data packets unless and until it receives a communication indicating that the file is malicious. *Id*. at 47 (citing Ex. 1011, 4:26–30, 5:51–57; Ex. 2018

¶ 163). If no such communication is received, Patent Owner argues, Dubrovsky will continue forwarding data packets of the file. *Id*. at 48 (citing Ex. 1011, 4:32–36). According to Patent Owner, "Dubrovsky's system therefore requires a quick, simple signature check so that the result can be returned prior to all of the data packets being forwarded to the client machine." *Id*. at 42 (citing Ex. 1011, 2:8–32; Ex. 2018 ¶¶ 160–161; Ex. 2013 ¶¶ 128, 133). Bertman's process for identifying malicious files, according to Patent Owner, includes downloading the entire file to Bertman's parser/active browser and fully rendering the file and observing its behavior and such an extensive process would never complete before the file had been fully downloaded by Dubrovsky's client. *Id*. at 57–58 (citing Ex. 2018 ¶ 168; Ex. 2013 ¶ 132). Thus, Patent Owner contends, there would never be an opportunity to block the file if the results of Bertman's analysis determined that the file was malicious. *Id*. at 42 (citing Ex. 2013 ¶ 134; Ex. 2018 ¶¶ 166–170).

Patent Owner argues that the Petition never articulated modifying the Dubrovsky-Bertman combination to address the above issue by, for example, pausing or holding back the download while Bertman performed its analysis. Such a modification of Dubrovsky, according to Patent Owner "is (1) not described by either of the cited references; (2) never articulated in the Petition; (3) not supported by any rationale as to ***why*** a person of ordinary skill would have modified Dubrovsky in such manner." PO Resp. 43–44 (citing Ex. 2018 ¶¶ 171–177).

In its Reply Brief, Petitioner disagrees that Dubrovsky would let the file be downloaded to the client before the signature matching process was complete. Pet. Reply 24 (citing Ex. 1011, 2:19–24, 6:35–38; Ex. 1068

¶¶ 128–130). According to Petitioner, Dubrovsky first analyzes the file and returns a result to the gateway, and then the gateway determines whether to block the file. *Id*. (citing Ex. 1011, 2:19–27). Thus, Petitioner argues, it is a precondition to the blocking decision that the gateway be provided with a result of the file's analysis. *Id*. at 25 (citing Ex. 1068 ¶¶ 132–133). Letting the file be completely downloaded at the client would be nonsensical, according to Petitioner, because then there would be no way to block the file if it was determined that the file was malicious. *Id*. (citing Ex. 2019 at 83:22–84:4). Petitioner also argues that Dubrovsky's technique of forwarding packets to the client in parallel with the process of analyzing the file was known as "latency hiding" but that this technique would not be implemented in a way that undermines Dubrovsky's objective of preventing malware infection. *Id*. at 26–27 (citing Ex. 1068 ¶¶ 134–141; Ex. 1063, 3:11–36, 3:58–67). Instead, latency hiding involves withholding a portion of the requested file to ensure that a virus infected file will not reach the client. *Id.* at 26 (citing Ex. 1063, 3:11–67). Finally, Petitioner argues that even if Dubrovsky were to blindly forward data packets without waiting for the results of the signature check, Patent Owner has not explained why the steps performed by Bertman in addition to downloading the file would have a material impact on the time required for the Dubrovsky-Bertman combination to complete its analysis before downloading that file to the client. *Id*. at 28–30 (citing Ex. 1068 ¶¶ 146–155; Ex. 1007 at 9, 2:51–54).

Patent Owner responds that Dubrovksy includes no discussion of "latency hiding" or holding back some packets while the signature check is performed. PO Sur-reply 19. Further, Patent Owner argues that the process of downloading and forwarding of received data packets, as is done by

Dubrovsky, would have completed prior to downloading those same packets combined with the additional steps of rendering and executing the file and analyzing its behavior. *Id*. at 21.

We begin our analysis first by addressing the issue of whether Dubrovsky, without being combined with Bertman, waits for the signature check to complete before finishing the process of downloading the file to the client. Here, we determine that Petitioner has not provided sufficient evidence supporting its argument that Dubrovsky holds back data packets while the signature matching process occurs. Petitioner relies on the following disclosure from Dubrovsky as support for its argument:

> The datacenter performs signature matching on the identification and returns a result of the signature matching to the gateway device. ***Then*** the gate way device determines whether to block the file from the client machine ***based on the result*** of the signature matching from the datacenter. In some embodiments, a match indicates that the incoming file is likely to contain malware, whereas no match indicates that the incoming file is not likely to contain malware. If the gateway device determines to block the file, the gateway device may simply stop forwarding the data packets not yet forwarded to the client device and discard these data packets.

Ex. 1011, 2:19–30 (emphasis added). Petitioner argues that the above disclosure, and especially the highlighted portions, indicates that receiving the result of the signature matching process is a precondition to determining whether to block the file because that determination is "based on the result." Pet. Reply 24–25.

Patent Owner, on the other hand, argues that Dubrovsky forwards data packets to the client device in parallel to performing the signature check and that it will continue forwarding the data packets until the file has been completely downloaded by the client unless it receives a result showing that

a match has been found indicating that the file is malicious. PO Resp. 47.

Patent Owner relies on the following disclosures from Dubrovsky to support its argument:

> The gateway device 110 may forward the data packets 123 to the client machine 120 as the data packets are received at the gateway device, ***provided the gateway device 110 has not received the content rating of the file from the datacenter 130 yet*** or the gateway device 110 has determined that the content rating of the file is not in a prohibited category. ***Typically***, in some embodiments, if the data center 130 can successfully find the content rating of the file, ***the datacenter 130 can send the content rating 133 to the gateway device 110 before all data packets of the file are received at the gateway device***.

Ex. 1011, 4:26–36 (emphases added).

> As processing logic generates the identification, processing logic also forwards the data packets received to the client machine (processing block 230). Then processing logic determines if it has received any result from the datacenter (processing block 232). ***If not yet, then processing logic returns to processing block 230 to continue forwarding data packets received to the client machine***.

Ex. 1011, 5:51–57 (emphasis added).

> The identification generated may be forwarded via the third network interface 430 to the datacenter for signature matching or may be forwarded to the security Screening module 440 for signature matching locally. ***If there is a match*** between the identification and a predetermined malware signature, then the Security Screening module 440 can signal the first network interface 410 to stop forwarding data packets of the file to the private network. ***Otherwise, the first network interface 410 may continue forwarding data packets of the file to the private network***.

Ex. 1011, 7:19–28 (emphases added).

Our review of Dubrovsky shows that it makes no mention of waiting for the signature matching process to complete, or of holding back data packets while the signature check occurs, or of latency hiding. Instead, the above disclosures show that Dubrovsky continues forwarding received packets to the client "provided the gateway device 110 has not received the content rating of the file from the datacenter 130 yet." Ex. 1011, 4:38–30. Dubrovsky stops forwarding data packets of the file to the private network "if there is a match" but "[o]therwise . . . continue[s] forwarding data packets of the file to the private network." *Id*. at 7:22–28.

Figure 2B of Dubrovsky, reproduced below, illustrates one embodiment of Dubrovsky's method.

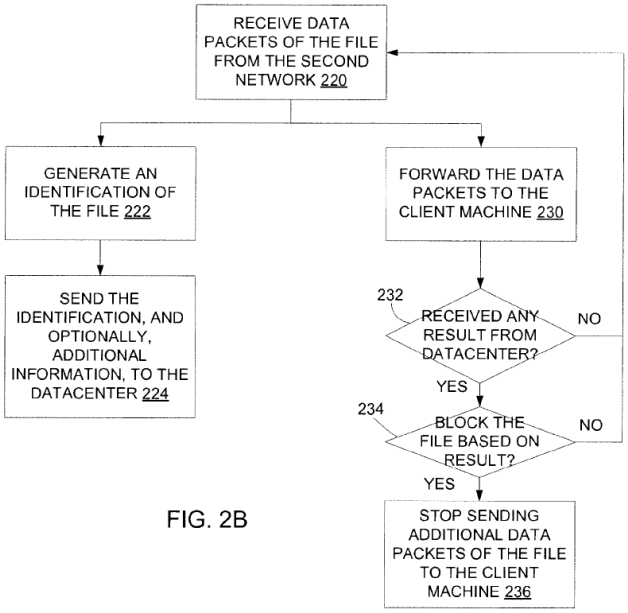


FIG. 2B

Figure 2B above, shows that when Dubrovsky receives packets of the requested file, it generates an identification of the file and sends that identification to the datacenter so that the datacenter can perform the signature matching. Ex. 1011, Fig. 2B, 5:39–50. Simultaneously, Dubrovsky forwards the data packets to the client and determines whether it

has received a result from the datacenter regarding a signature match. *Id*. at Fig. 2B, 5:51–55. If no result has yet been received, Dubrovsky continues forwarding data packets to the client. *Id*. at 5:55–57.

Crucially, Figure 2B and its associated description does not indicate any step taken to hold back packets if the download process is nearing completion and a result has not yet been received. Instead, step 232 of Figure 2B shows that Dubrovsky assumes that if a result has not yet been received, the file has not yet completed downloading and more packets are available to be received. Step 234 of Figure 2B also shows that when a result is received and "the result indicates there is no match [and thus] the file is not likely to contain malware, . . . [the] logic returns to . . . continue forwarding data packets received to the client machine." Ex. 1011, 5:60–63. In other words, even after a negative signature match result comes in, Dubrovsky assumes there are more packets to be received and forwarded to the client.

We determine the above description is consistent with Patent Owner's position that Dubrovsky relies on a quick signature check so that the result can be returned prior to all of the data packets being forwarded to the client. PO Resp. 42. Patent Owner's position is bolstered by Dubrovsky's description that "[t]ypically, in some embodiments, if the datacenter 130 can successfully find the content rating of the file, the datacenter 130 can send the content rating 133 to the gateway device 110 before all data packets of the file are received at the gateway device," which implies that if the content rating is not found in time, the datacenter may not be able send the rating before the packets have all been received. Ex. 1011, 4:32–36.

The passage from Dubrovsky that Petitioner relies on states "[t]he datacenter performs signature matching on the identification and returns a result of the signature matching to the gateway device. Then the gateway device determines whether to block the file from the client machine based on the result of the signature matching from the datacenter." Ex. 1011, 2:19–24. Although this passage states that gateway device determines whether to block the file after a result is received, it does not address the situation where downloading of the file is nearing completion and no result has yet been received. Instead, we find that it merely addresses the situation in which a result is determined quickly enough before the file has completed downloading. The same paragraph goes on to state that "[i]f the gateway device determines to block the file, the gateway device may simply stop forwarding the data packets not yet forwarded to the client." *Id*. at 2:27–29. Thus, although receiving a result may be a precondition to blocking a file, the above passage does not indicate that a result is a precondition to completing the download process and *not* blocking the file.

In summary, we determine Petitioner has not set forth sufficient evidence showing that an unmodified Dubrovsky holds back data packets of a file while it completes its signature check to identify whether the file is malicious.

We turn next to the issue of whether the Dubrovsky-Bertman combination, as proposed by Petitioner, would intercept a malicious executable file given that Dubrovsky forwards the data packets of the file in parallel to analyzing it. Importantly, Petitioner does not address Dubrovsky's packet forwarding operation in its Petition. Because Petitioner does not propose any modification to the packet forwarding operation, one

of ordinary skill would assume Dubrovsky packet forwarding operation would be included in Petitioner's Dubrovsky-Bertman combination. Patent Owner argues that as proposed, the Dubrovsky-Bertman combination would never have an opportunity to block the file if Bertman determined that the file was malicious because Bertman's analysis would never complete prior to the file's data packets being fully downloaded to the client. PO Resp. 42 (citing Ex. 2013 ¶ 134; Ex. 2018 ¶¶ 166–170). We agree with Patent Owner.

Petitioner's proposed combination "is based on the cloud-based virus scanning of Dubrovsky, but in which Dubrovsky's data center is enhanced with Bertman's teaching of downloading material associated with the URL in order detect and generate signatures/definitions for previously unknown malware." Pet. 49. In the combination, when Dubrovsky receives a file it has not encountered before, there would be no matching signature in Dubrovsky's datacenter, and in such a situation the system would pass the requested URL to Bertman's network where the URL would be used to download the material and subsequently evaluate the material for malware, and if present, would generate a signature and pass that signature back to Dubrovsky's signature database. *Id*. at 49–50 (citing Ex. 1005 ¶ 323); *id*. at 62 (citing Ex. 1005 ¶¶ 322, 355–357). Dubrovsky's signature database would then contain a match to the file in question and would therefore return a signature matching result to the gateway device after which the gateway device would proceed to block the file. *Id*. at 50 (citing Ex. 1005 ¶ 324).

Dubrovsky's parallel packet forwarding operation consists of Dubrovsky downloading the file and forwarding it to the client while sending a URL of the file to Bertman's network for its analysis. Bertman's analysis requires doing the same thing, i.e. downloading the file, in addition

to doing several other tasks, such as evaluating the downloaded file, generating the signature, and doing a signature check. PO Resp. 57–58. Because Bertman's analysis necessarily requires doing more than Dubrovsky's packet forwarding operation, the latter would always complete before the former. Dr. Davis provides credible testimony supporting this conclusion. Dr. Davis testifies that the proposed combination involves downloading of the file plus five additional tasks and that one of ordinary skill would have recognized that these functions would not have completed prior to the file being downloaded to the client. Ex. 2018 ¶ 168.

An analogy would be of two runners who run at the same speed, but the second runner has a longer distance to run than the first runner. In this situation, the first runner would always finish before the second runner. Similarly here, Dubrovsky downloads and forwards the requested file to the client. Bertman, when it receives the URL, also downloads the same file, but in addition, has several other tasks to perform. Bertman would, therefore, never complete the download plus its additional tasks before Dubrovsky completes the same download and forwarded the packets to the client, especially since Bertman does not even begin to download the file until Dubrovsky already has.

Petitioner argues in its Reply Brief that the additional tasks would have no material impact on the time required to provide a complete analyzed file to the client because the transfer times of the download would dominate over the time taken by the additional analysis steps. Pet. Reply 28–29 (citing Ex. 1068 ¶¶ 146–155, Ex. 1007 at 9, 2:51–54). Even if the additional steps took a small amount of time in comparison to the downloading step, as Petitioner argues, the total time for downloading the file and performing

these additional steps would still be larger than the time to download and forward the file without performing these additional steps.

For these reasons, we agree with Patent Owner that Bertman's proposed analysis would not complete before Dubrovsky's parallel packet forwarding operation has completed downloading the file to the client. Accordingly, we determine Petitioner has not shown the Dubrovsky-Bertman combination teaches "intercepting the executable file according to the second alarm information comprising a maliciousness of the executable file" as recited by limitation 1f.

h) Conclusion as to Obviousness over Dubrovsky-Bertman (Ground 2A)

For the reasons stated above, we determine that Petitioner has not shown that the combination of Dubrovsky and Bertman teaches limitation 1f and the corresponding limitations of independent claims 3 and 5. Claim 6 depends from claim 5. Accordingly, we determine Petitioner has not shown by a preponderance of the evidence that claims 1, 3, 5, and 6 would have been obvious over Dubrovsky and Bertman.

G. Obviousness over Dubrovsky, Bertman, and Campbell (Ground 2B)

Petitioner argues that claims 2, 4, and 7 of the '693 patent would have been obvious over the combination of Dubrovsky, Bertman, and Campbell. Pet. 15, 69–71. Claim 2 depends from claim 1, claim 4 depends from claim 3, and claim 7 depends from claim 5. Petitioner's contentions for these dependent claims do not cure the deficiencies outlined above for their corresponding independent claims. As we explained above, we determine Petitioner has not shown that the Dubrovsky-Bertman combination teaches all the limitations of independent claims 1, 3 and 5 and for the same reasons

has not shown that the Dubrovsky-Bertman-Campbell combination teaches the limitations of dependent claims 2, 4, and 7.

H. Obviousness over Dubrovsky (Ground 2C)

Petitioner argues that claim 8 differs from claim 1 in two regards: (1) claim 8 is directed to computer readable medium with software instructions stored thereon and (2) claim 8 is broader than claim 1 because the monitoring device is not recited as downloading the file according to the source path, or as detecting the downloaded file. Thus, unlike its contentions for claims 1, 3, and 5, Petitioner relies only on Dubrovsky for its contentions for claim 8. Pet. 71–72.

With respect to claim 8's recitation of "a non-transitory computer readable medium storing instructions for execution by a processor" Petitioner argues Dubrovsky's gateway includes computer hardware that includes memory for storing instructions that cause the gateway to perform various computer operations. Pet. 66 (citing Ex. 1011, 5:33–38, 6:39–51, Fig. 6), 71; Ex. 1005 ¶¶ 363–364.

With regard to the remaining limitations of claim 8, Petitioner relies on its contentions from claim 1 regarding Dubrovksy alone. Pet. 72 (arguing that "since the only limitations in Claim 1 for which Bertman was relied on under Ground 2A are not included in Claim 8, and since Claim 8 does not differ from Claim 1 in any other material respect, Dubrovsky alone renders Claim 8 obvious for the reasons set forth above under Claim 1.")

Based on the evidence of record and the arguments presented by both parties, we are persuaded by Petitioner that Dubrovsky teaches the limitations of claim 8. In particular, we agree with Petitioner that the limitations of claim 8 are similar to those of claim 1 except (1) that claim 1

is directed to a method while claim 8 is directed to a "non-transitory computer readable medium" and (2) that claim 1 recites that the monitoring device detects the file that is downloaded according to the source path while claim 8 does not include such a requirement.

We agree that Dubrovsky's gateway includes memory for storing instructions that cause the gateway to perform the various limitations of the claim. For example, Dubrovsky discloses that the gateway includes a storage device 470 and software such as instructions that run on a processing core. Ex. 1011, 5:34–38, 6:39–44. Thus, Dubrovsky teaches the preamble of claim 8.

The remaining limitations are similar to those of claim 1 and we agree Dubrovsky teaches those limitations for the same reasons explained in our analysis of claim 1. *See supra* § III.F.2. We determined, however, that the Dubrovsky-Bertman combination does not teach limitation 1f (the "intercepting" limitation) because adding Bertman's more extensive analysis, which involves downloading the file along with several other tasks, would mean that the analysis would take too long to allow Dubrovsky to intercept a malicious file before it was forwarded to the client. *See supra* § III.F.2.g. Here, because claim 8 does not require downloading the file, Petitioner does not propose combining Dubrovsky with Bertman. As Petitioner explained in its contentions regarding claim 1, Dubrovsky explicitly teaches that it would block a malicious file based on a positive signature match, which, because it does not download the file to perform its signature match[8], typically occurs quickly enough before the file is

[8] Dubrovksy discloses that it "generate[s] an identification of the file based on the partial information of the file provided by the data packets" and

completely forwarded to the client even without having to hold back data packets to wait for the results of the analysis. Pet. 64–65 (citing Ex. 1011, 5:6–9, 2:19–27; Ex. 1005 ¶ 324); Ex. 1011, 4:32–36. Thus, without the issues created by combining Dubrovsky with Bertman, we agree with Petitioner that Dubrovsky alone teaches the "intercepting" limitation of claim 8.

I. Obviousness over Dubrovsky and Campbell (Ground 2D)

Claim 9 depends from claim 8 and further recites:

> The network device according to claim 8, wherein determining the file is an executable file according to the request or the data stream carrying the file comprises:
>
> detecting whether a string which indicates the name of the executable file exists in the request; and/or
>
> detecting whether a file header of the executable file exists in the data, which is transmitted by the data stream returned by the network entity.

According to Petitioner, Campbell teaches both claimed prongs of identifying an executable file, i.e. (1) looking to the name of the file and (2) looking at the file header.[9] As to the first prong, Petitioner argues Campbell teaches identifying executable files by examining the name of the file for

---

"sends the identification to 135 of the file to the datacenter" which then "performs signature matching . . . on the identification." Ex. 1011, 4:39–61. Because it sends the identification of the file to the datacenter, the file is not disclosed as being downloaded by the datacenter.

[9] Petitioner relies on its contentions for claim 2 in arguing that the Dubrovsky-Campbell combination teaches the limitations of claim 9. Pet. 72. Because we did not analyze Petitioner's contentions for claim 2 above, we do so here.

known extensions, such as the .EXE extension. Pet. 70 (citing Ex. 1013, 7:39–42, 7:54–55; Ex. 1005 ¶ 412). As to the second prong, Campbell teaches that because file extensions can sometimes be disguised, the file's header can also be used to determine the file type. *Id*. at 70–71 (citing Ex. 1013, 5:6–24).

Petitioner argues that while Dubrovsky judges when a requested file is an executable file, it does not explain how a file is identified as being executable. Pet. 70. Thus, according to Petitioner, one of ordinary skill would have been motivated to look to known ways of identifying file types, such as Campbell's method of examining the file extension and the file header. *Id*. (citing Ex. 1005 ¶¶ 404–406). Campbell is directed to a system analogous to Dubrovsky for blocking unwanted software and Campbell, like Dubrovsky, recognizes that malware is often sent as executable files. Ex. 1013, 4:51–57. Therefore, one of ordinary skill would have been "motivated to look to Campbell given that it has the same goal and uses the same approach for identifying suspicious files." Pet. 70 (citing Ex. 1005 ¶ 406).

Patent Owner disagrees that Campbell teaches claim 9's limitations. Specifically, as to the first prong, Patent Owner argues that the claim requires detecting whether a string which indicates a name of the executable file exists ***in the request*** but that Campbell analyzes the file name of the downloaded file, not the file name in the request. PO Resp. 61–62 (citing Ex. 2018 ¶¶ 184–187).

As to the second prong, Patent Owner argues that Campbell's discussion of examining a file's header is directed to identifying the file as spyware, not as an executable. PO Resp. 63–64 (citing Ex. 1013. 5:9–19)

(arguing "Campbell is specifically directed toward identifying patterns within a ***spyware*** header to identify that a file is ***spyware***."). According to Patent Owner, "[b]ecause Campbell's spyware policy database only includes signatures for known characteristics of spyware headers it could not be used to identify executable files." *Id*. at 64 (citing Ex. 2018 ¶¶ 188–189).

Based on the evidence of record and the parties' arguments, we are persuaded by Petitioner that the Dubrovsky-Campbell combination teaches the limitations of claim 9. Initially, we note that we agree with Petitioner that one of ordinary skill would have been motivated to look to known ways of identifying file types, such as Campbell's method of examining the file extension and the file header. We give credit to Dr. Nielson's testimony in this regard that both Dubrovsky and Campbell recognize that executables were a frequent source of malware but that Dubrovsky does not provide specific details regarding how a file would be determined to be executable. Ex. 1005 ¶¶ 404–405. Because Campbell does provide these details, Dr. Nielson credibly testifies that a person of ordinary skill would have been motivated to combined Campbell's techniques of identifying executables with Dubrovsky. Ex. 1005 ¶¶ 404–406.

We agree with Petitioner that Campbell teaches that a file's header can be used to determine that the file is an executable. Campbell teaches that spyware contains certain patterns that can be used to identify the nature of the file. Ex. 1013, 5:9–12. Campbell provides an example of a .CAB file that includes a header having a certain format. *Id*. at 5:12–13. Campbell explains that spyware scanners have access to spyware signature lists that can be compared with characteristics of spyware headers to identify spyware so that the spyware can be blocked. *Id*. at 13–24.

Patent Owner's argument that Campbell's teachings regarding examining headers is directed to identifying spyware rather than executables, fails to account for the fact that Campbell explicitly states that "spyware is often sent as executable files." Ex. 1013, 4:52–53. Furthermore, Dr. Nielson provides credible testimony that one of ordinary skill would recognize that a .CAB file is an executable file. Ex. 1068 ¶¶ 173–178. Thus, even if Campbell's teachings were narrowly directed at identifying spyware using the file's header, this would still teach one instance of identifying an executable file using the file's header.

Campbell's teaching satisfies the second prong of claim 9. We therefore need not reach the merits of Petitioner's contentions regarding the first prong.[10] Accordingly, we determine Petitioner has shown by a preponderance of the evidence that the Dubrovsky-Campbell combination teaches claim 9.

IV. CONCLUSION[11]

For the foregoing reasons, we determine Petitioner has not shown by a preponderance of the evidence that claims 1–9 would have been obvious

[10] Claim 9 requires determining the file is an executable file (1) by the name of the file "and/or" (2) by the file header. Thus, a teaching of either prong would satisfy the requirements of the claim as a whole.

[11] Should Patent Owner wish to pursue amendment of the challenged claims in a reissue or reexamination proceeding subsequent to the issuance of this decision, we draw Patent Owner's attention to the April 2019 *Notice Regarding Options for Amendments by Patent Owner Through Reissue or Reexamination During a Pending AIA Trial Proceeding*. *See* 84 Fed. Reg. 16,654 (Apr. 22, 2019). If Patent Owner chooses to file a reissue application or a request for reexamination of the challenged patent, we remind Patent

over the Moshchuk-Gribble combination. We determine Petitioner has not shown by a preponderance of the evidence that claims 1, 3, 5, and 6 would have been obvious over Dubrovsky and Bertman or that claims 2, 4, and 7 would have been obvious over Dubrovsky, Bertman, and Campbell. We determine Petitioner has shown by a preponderance of the evidence that claim 8 would have been obvious over Dubrovsky and that claim 9 would have been obvious over Dubrovsky and Campbell.

In summary:

Claims	35 U.S.C. §	Reference(s)/ Basis	Claims Shown Unpatentable	Claims Not Shown Unpatentable
1–9	103	Moshchuk, Gribble		1–9
1, 3, 5, 6	103	Dubrovsky, Bertman		1, 3, 5, 6
2, 4, 7	103	Dubrovsky, Bertman, Campbell		2, 4, 7
8	103	Dubrovsky	8	
9	103	Dubrovsky, Campbell	9	
Overall Outcome			8, 9	1–7

V. ORDER

In consideration of the foregoing, it is hereby:

ORDERED claims 8 and 9 of the '693 patent are held to be unpatentable; and

Owner of its continuing obligation to notify the Board of any such related matters in updated mandatory notices. *See* 37 C.F.R. §§ 42.8(a)(3), (b)(2).

FURTHER ORDERED that, because this is a Final Written Decision, the parties to the proceeding seeking judicial review of the decision must comply with the notice and service requirements of 37 C.F.R. § 90.2.

FOR PETITIONER:

Jonathan Lindsay
IRELL & MANELLA LLP
jlindsay@irell.com

FOR PATENT OWNER:

Michael Hawkins
Patrick Bisenius
Nicholas Stephens
Jennifer Huang
Kenneth Darby
Sangki Park
Thomas H. Reger II
Brian Strand
FISH & RICHARDSON P.C.
hawkins@fr.com
bisenius@fr.com
nstephens@fr.com
jjh@fr.com
kdarby@fr.com
spark@fr.com
reger@fr.com
strand@fr.com