

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

McAFEE, INC.,
Petitioner,

v.

CAP CO., LTD.,
Patent Owner.

Case IPR2015-01876
Patent 8,544,078 B2

Before DONNA M. PRAISS, PATRICK M. BOUCHER, and J. JOHN LEE,
Administrative Patent Judges.

LEE, *Administrative Patent Judge.*

FINAL WRITTEN DECISION
35 U.S.C. § 318(a) and 37 C.F.R. § 42.73

INTRODUCTION

On September 4, 2015, McAfee, Inc. (“Petitioner”) filed a Petition (Paper 1, “Pet.”) requesting *inter partes* review of claims 1–5, 16, and 18–20 (“the challenged claims”) of U.S. Patent No. 8,544,078 B2 (Ex. 1001, “the ’078 patent”). CAP Co., Ltd. (“Patent Owner”) filed a Preliminary Response (Paper 10). An *inter partes* review of all challenged claims was instituted on March 8, 2016. Paper 12 (“Inst. Dec.”). Patent Owner then filed a Patent Owner Response (Paper 17, “PO Resp.”), and Petitioner filed a Petitioner Reply (Paper 20, “Pet. Reply”). An oral hearing was held on November 1, 2016. Paper 28 (“Tr.”).¹

We have jurisdiction under 35 U.S.C. § 6(c). This Final Written Decision is issued pursuant to 35 U.S.C. § 318(a) and 37 C.F.R. § 42.73. As discussed below, Petitioners have shown by a preponderance of the evidence that each of the challenged claims is unpatentable.

A. *Related Proceedings*

The parties identify the following cases as related to this proceeding:

- (1) *CAP Co. v. McAfee, Inc.*, Case No. 3:14-cv-05068-JD (N.D. Cal.);
- (2) *CAP Co. v. Symantec Corp.*, Case No. 3:14-cv-05071-JD (N.D. Cal.);
- (3) *CAP Co. v. Microsoft Corp.*, Case No. 2:14-cv-01899 (W.D. Wash.); and
- (4) *McAfee, Inc. v. CAP Co.*, Case IPR2015-01856 (PTAB). Pet. 1; Paper 18, 1–2.

B. *The ’078 Patent*

The ’078 patent, titled “Flexible Network Security System and

¹ A combined oral hearing was held for this case as well as related *inter partes* review IPR2015-01856.

Method for Permitting Trusted Process,” is directed to a network security system and method that permits a trusted process to communicate while using a firewall. The firewall protects a network connection by setting restrictions on information communicated over a network. Ex. 1001, 2:40–60. The ’078 patent notes that prior art firewalls (such as Microsoft’s “Internet Connection Firewall (ICF)”) presented difficulties for “inexpert users” due to the need to manually add port numbers to the firewall. *Id.* at 1:20–23, 2:1–23. To solve this problem, the invention of the ’078 patent involves automatically adding or removing a port to or from the firewall. *Id.* at 1:7–14, 2:29–36.

As described in the specification, an “internal permitted program storage” stores a list of programs permitted by the firewall. *Id.* at 6:23–40. An example of data stored in an internal permitted program storage is shown in Table 1, reproduced below:

TABLE 1

	Entire path of program	Program MD5 hash value
1	D:\Program Files\MSN Messenger\msnmsgr.exe	0x83276482764827368682376482637872
2	D:\Program Files\PcAnywhere.exe	0x93847293874298379427973928479374
3		
...		

As shown in Table 1, the internal permitted program storage may store information about the permitted programs, such as each program’s name, path, and hash value. *Id.* at 6:33–40, Tbl. 1.

When a program opens a server port to operate as a server, the system determines whether the program is registered in the internal permitted program storage. *Id.* at 6:56–60. If so, the opened port is registered in an “internal permitted port storage.” *Id.* at 6:61–63. The specification

describes a “port monitoring unit” that extracts information about the port, which is then registered in the internal permitted port storage. *Id.* at 2:45–47, 2:50–54. An example of data stored in an internal permitted port storage is shown in Table 2, reproduced below:

TABLE 2

	Entire path of program	Protocol	Port
1	D:\Program Files\MSN Messenger\msnmsgr.exe	TCP	1863
2	D:\Program Files\MSN Messenger\msnmsgr.exe	TCP	6891
3	D:\Program Files\PcAnywhere\PcAnywhere.exe	TCP	5631
4	D:\Program Files\PcAnywhere\PcAnywhere.exe	UDP	5632
...			

As shown in Table 2, the internal permitted port storage may store information about the path of the program as well as the communication protocol and port number. *Id.* at 7:26–30, Tbl. 2.

Packets of inbound traffic are examined to determine whether the destination port has been registered in the internal permitted port storage. *Id.* at 6:64–7:2. If the port is not registered, the packets are blocked; if they are registered, the packets are allowed to bypass the firewall. *Id.* at 7:3–9.

C. *Challenged Claims*

Petitioner challenges the patentability of claims 1–5, 16, and 18–20 of the '078 patent, of which claims 1 and 16 are independent. Claim 1 is illustrative of the challenged claims, and recites:

1. A network security system controlling inbound traffic by using a firewall, the firewall protecting a corresponding network connection of a computer to a network by setting restrictions on information communicated between networks, comprising:

a port monitoring unit extracting information about a server port, wherein the server port is designated as a port of a network communication program;

an internal permitted program storage storing a list of programs permitted by the firewall, wherein the internal permitted program storage adds a program to the list by extracting information about the program for which communication is to be permitted by the firewall; and

a firewall flexible device determining whether the network communication program is registered in the list of programs stored in the internal permitted program storage;

wherein the firewall flexible device automatically stores the extracted information about the server port in an internal permitted port storage if the network communication program is registered in the list of programs stored in the internal permitted program storage; and

wherein the firewall flexible device further determines whether a port of a packet of inbound traffic matches with the server port and blocks the packet of inbound traffic if the port does not match with the server port.

D. Instituted Grounds of Unpatentability and Asserted Prior Art

This *inter partes* review was instituted on the following alleged grounds of unpatentability under 35 U.S.C. § 103(a):

Challenged Claims	Asserted Prior Art
1–5, 16, 18–20	Yadav ²
1–5, 16, 18–20	Yadav and Freund ³

Inst. Dec. 21. In support of these asserted grounds, Petitioner also relies on the testimony of its proffered expert, Dr. Atul Prakash. Ex. 1004 (“Prakash Declaration”); Ex. 1023 (“Prakash Rebuttal Declaration”). In response,

² U.S. Patent No. 7,174,566 B2, filed Feb. 1, 2002, issued Feb. 6, 2007 (Ex. 1002, “Yadav”).

³ U.S. Patent No. 5,987,611, issued Nov. 16, 1999 (Ex. 1003, “Freund”).

Patent Owner relies on the testimony of its proffered expert, David Bernstein. Ex. 2007 (“Bernstein Declaration”). Both experts were deposed, and the transcripts of their depositions are in the record. Ex. 2004 (Prakash deposition); Ex. 1022 (Bernstein deposition).

ANALYSIS

A. *Claim Construction*

In an *inter partes* review, claim terms in an unexpired patent are construed according to their broadest reasonable interpretation in light of the specification. 37 C.F.R. § 42.100(b); see *In re Cuozzo Speed Techs., LLC*, 793 F.3d 1268, 1278–1279 (Fed. Cir. 2015). During trial, the parties disputed the claim construction of the phrase “adds a program to the list [of programs] by extracting information about the program for which communication is to be permitted by the firewall,” which is recited in independent claim 1. Independent claim 16 recites a similar limitation.

Patent Owner argues this phrase should be construed to mean “adds a new program to the list of programs permitted by the firewall by extracting information about the program for which communication is to be permitted, i.e., a trusted program.” PO Resp. 18. Petitioner argues it should simply be given its plain meaning. Pet. Reply 12. The parties’ dispute several issues relating to this phrase.

First, the parties disagree that the phrase should be limited to “new” programs. The Patent Owner Response does not explain sufficiently what is meant by “a new program,” though it refers to “a previously unidentified application” (PO Resp. 42) and “an unseen server application” (*id.* at 46) without explaining whether or how these relate to claim construction, much less identifying evidence supporting a claim construction based on them.

Petitioner notes that the specification of the '078 patent refers to a scenario in which *existing* software is upgraded, which alters its associated server port. Pet. Reply 10 (citing Ex. 1001, 2:18–20). Further, at the oral hearing, Patent Owner clarified that the claim should not be interpreted to preclude pre-existing applications. See Tr. 38:12–39:4.⁴ Thus, we decline to import a requirement that the added program be “new.”

Next, the parties dispute whether this phrase encompasses “copying” pre-existing information. Patent Owner argues that, because the claim recites adding a program “by” extracting information about the program, this claim phrase “cannot mean simply copying information (that was previously extracted).” PO Resp. 23; *see also id.* at 25 (“This carefully described process can in no way be equated to some mechanical internal copying of bits of memory from one data structure to another or one storage location to another.”). None of the evidence cited by Patent Owner, however, expresses any disclaimer or disavowal of adding a program to the list of programs by copying information. *See Gillette Co. v. Energizer Holdings, Inc.*, 405 F.3d 1367, 1374 (Fed. Cir. 2005) (holding that “words of manifest exclusion or explicit disclaimers in the specification are necessary to disavow claim scope” (internal quotations omitted)). In fact, most of the evidence cited by Patent Owner generally relates to *different* limitations concerning *port* information rather than adding a program to a list of programs. *See* Ex. 1001, 1:7–14 (describing the invention as relating to automatically adding ports to a firewall), 2:2–17 (describing the problem of inexperienced users

⁴ Patent Owner asserted that a pre-existing application would be “new to the firewall” if it had not previously sought to act as a server, or if it were using a different port than previously registered, but indicated this issue related to the server port limitations. *See* Tr. 38:12–39:12.

having difficulty in finding the port used by software acting as a server), 3:19–26 (discussing extraction of port information). Thus, we decline to construe the “adds a program” phrase to exclude copying pre-existing information.

The parties further dispute whether the extracted program information must itself be stored when the program is added to the list of programs. See PO Resp. 23 (“The process of adding a trusted program begins with extracting program and port information for the very purpose of registering the information in a permitted program and port storage.”); Pet. Reply 13 (arguing the extraction need not be for such a purpose). We note that the claims do not explicitly recite storing the extracted program information. Moreover, we note that the claims *do* explicitly recite extracting server port information and “automatically stor[ing] the extracted information about the server port” (claim 1) or similar limitations. Thus, we decline to construe the “adds a program” phrase to implicitly require storing the extracted program information.

Finally, the parties dispute the inclusion of “a trusted program” in the construction of the “adds a program” limitation. As Petitioner argues, however, general references to “trusted” processes and programs in the specification do not indicate that the “adds a program” limitation, in particular, relates only to “trusted” programs. See Pet. Reply 13–14. Thus, we are not persuaded that the proper construction of the “adding” phrase requires that the program being added is “trusted” (at the time it is added), as Patent Owner suggests.⁵

⁵ We note that, in any event, none of Patent Owner’s arguments for patentability relate to the “trusted program” aspect of its proposed construction.

No further construction of the “adds a program” phrase, or any other claim term, is necessary to resolve the issues presented in this *inter partes* review. See *Vivid Techs., Inc. v. Am. Sci. & Eng’g, Inc.*, 200 F.3d 795, 803 (Fed. Cir. 1999) (stating that only those terms in controversy need to be construed, and only to the extent necessary to resolve the controversy).

B. Asserted Grounds of Unpatentability

Petitioner contends that claims 1–5, 16, and 18–20 are unpatentable under § 103(a) in view of (1) Yadav, and (2) the combination of Yadav and Freund. Pet. 30–60. As discussed below, Petitioner has demonstrated by a preponderance of the evidence that all challenged claims are unpatentable on both of these grounds.

1. Legal Principles

A claim is unpatentable under § 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) objective evidence of non-obviousness, i.e., secondary considerations.⁶ *Graham v. John Deere Co.*, 383 U.S. 1, 17–18 (1966).

Additionally, the obviousness inquiry typically requires an analysis of

⁶ The record after trial does not include any evidence pertaining to secondary considerations, nor does Patent Owner assert that any such objective indicia support a conclusion of non-obviousness for any challenged claim.

“whether there was an apparent reason to combine the known elements in the fashion claimed by the patent at issue.” *KSR*, 550 U.S. at 418 (citing *In re Kahn*, 441 F.3d 977, 988 (Fed. Cir. 2006) (requiring “articulated reasoning with some rational underpinning to support the legal conclusion of obviousness”)); see *In re Warsaw Orthopedic, Inc.*, 832 F.3d 1327, 1333 (Fed. Cir. 2016) (citing *DyStar Textilfarben GmbH & Co. Deutschland KG v. C. H. Patrick Co.*, 464 F.3d 1356, 1360 (Fed. Cir. 2006)).

2. *Level of Skill in the Art*

Based on its expert’s testimony, Petitioner contends a person of ordinary skill in the art at the time of the claimed invention would have had a master’s degree in computer science, computer engineering, electrical engineering, or the equivalent; or a bachelor’s degree in one of those fields and two years of industry experience as a network computer system administrator, including working with network firewalls, systems programming, and operating systems; or four years of industry experience. Pet. 8 (citing Ex. 1004 ¶ 22). Patent Owner does not dispute Petitioner’s contention, and its expert’s opinion essentially matches that of Petitioner’s expert. See Ex. 2007 ¶ 35. Based on the expert testimony in the record, we agree with and adopt the level of skill set forth in the Petition.

3. *Scope and Content of the Prior Art*

a. *Yadav*

Yadav describes an intrusion detection system that analyzes packets to identify “particular sources of network communications” to be “singled out for greater scrutiny.” Ex. 1002, at [57]. According to Yadav, the “integrated intrusion detection system uses an end-node firewall that is dynamically controlled using invoked-application information and a network

policy.” *Id.* The system “may monitor network traffic to block traffic that violates the network policy.” *Id.*

Figure 2B, reproduced below, is a diagram of the integrated network intrusion detection system disclosed in Yadav:

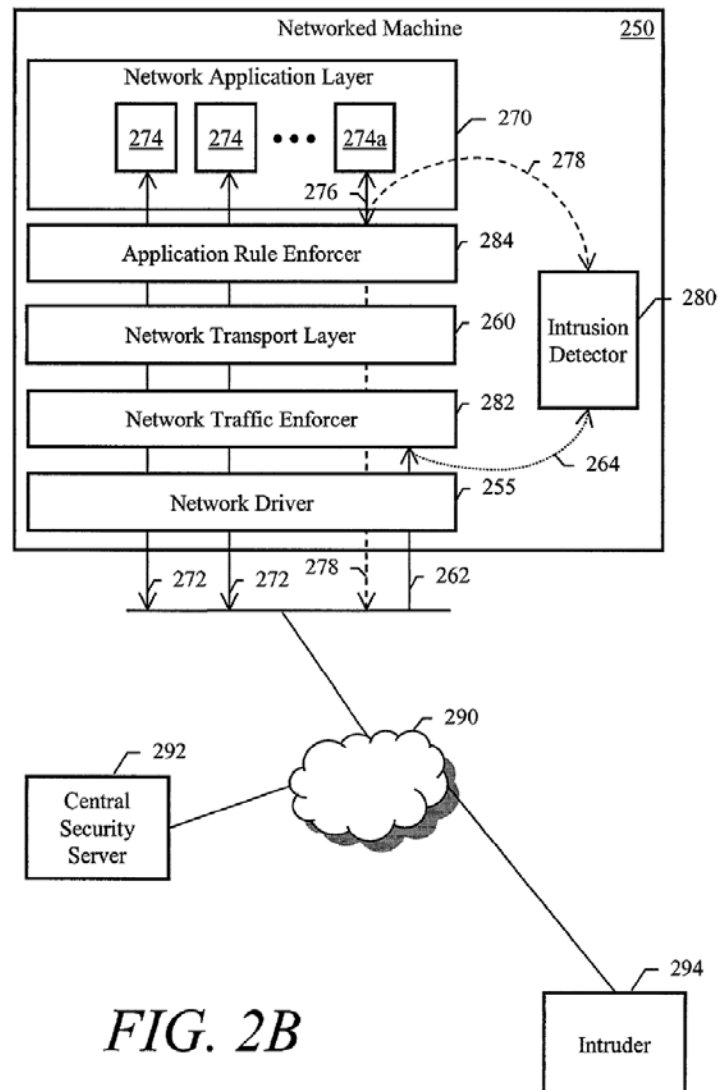


Figure 2B illustrates three main components of the system: application rule enforcer (ARE) 284, network traffic enforcer (NTE) 282, and intrusion detector 280. *Id.* at 5:59–62. The ARE “authorizes network service requests from applications invoked on the networked machine and identifies abnormal behavior by an invoked application.” *Id.* at 3:23–26. The NTE

“monitors inbound network communications and blocks those communications that fail to correspond to an authorized network service request.” *Id.* at 3:26–30. The intrusion detector “monitors the blocked communications and identifies abnormal application behavior to determine when additional traffic monitoring is needed to detect an intrusion.” *Id.* at 3:30–33. “Thus, the total number of communications (e.g., packets) that are examined may be reduced while intrusions may be detected more effectively.” *Id.* at 3:34–36.

Figure 3, reproduced below, is a flowchart illustrating the operation of these components of the system:

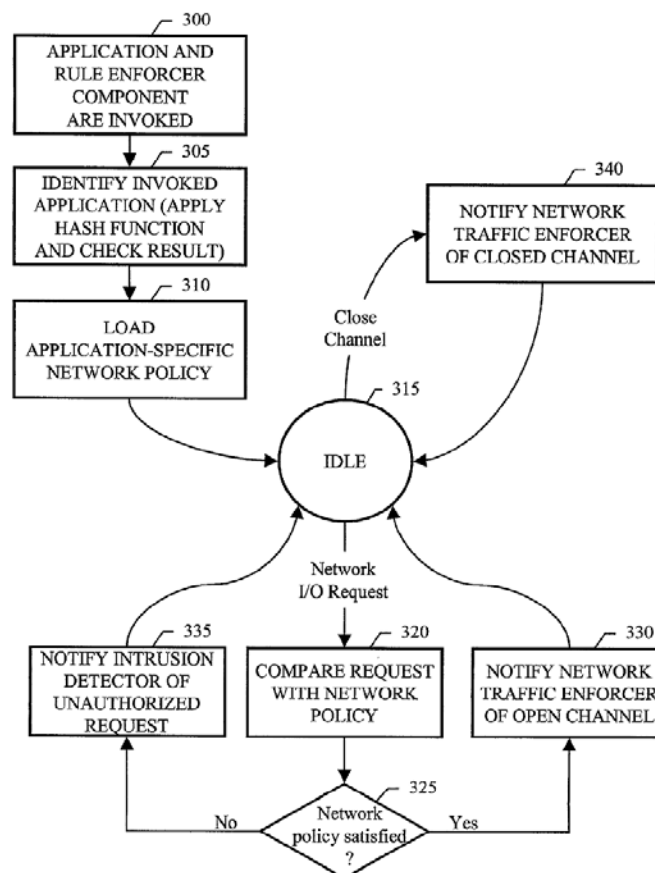


FIG. 3

As shown in Figure 3, the ARE is invoked along with an application (block 300). *Id.* at 7:22–25. The ARE then identifies the invoked application (block 305), such as by determining the full path of the application, examining machine instructions embodying the application, and/or crosschecking with file properties information (e.g., application filename, size, version number). *Id.* at 7:26–32. Examining machine instructions can involve obtaining a hash value of the executable for the application and comparing it with “predefined hash values for known applications to identify the invoked application.” *Id.* at 7:32–37. Once the application is identified, an application-specific network policy is loaded by the ARE from a local or remote repository (block 310). *Id.* at 7:45–51.

When the application makes a network I/O request, the request is compared with the application-specific network policy (block 320). *Id.* at 7:53–55. If the policy is satisfied, the NTE is notified to open a channel, such as by sending a message “specifying a source IP address, a source port, a destination IP address, a destination port and a protocol for the opened channel.” *Id.* at 7:55–60. This information may be gleaned from the network I/O request. *Id.* at 3:52–61. If the policy is not satisfied, the intrusion detector is notified of the rejected request. *Id.* at 7:61–62. When the NTE receives the message to open a channel, “the opened channel is added to an authorization list.” *Id.* at 8:21–24. The NTE monitors network communications to block unauthorized communications and notify the intrusion detector. *Id.* at 8:19–21, 8:25–28. When the NTE is notified that a channel is closed, the closed channel is removed from the authorization list. *Id.* at 8:29–31. “Thus, network communications that have not been pre-approved by the ARE component are blocked and copied to the intrusion detector.” *Id.* at 8:21–23.

b. Freund

Freund describes a system for monitoring access to an open network (such as the Internet) using client-based filtering and access rules. Ex. 1003, at [57], 5:53–64. Access rules may include, for example, “a list of applications or application versions that a user can and cannot use in order to access the Internet.” *Id.* at [57]. The system includes a “supervisor” application running on a server or client monitor that coordinates the system, including maintaining access rules and providing them to filter applications on individual clients, which may retain local copies of the rules. *Id.* at [57], 5:9–15, 14:32–33, 34:40–44. The supervisor application coordinates with a firewall to control which client applications are given access to the Internet. *Id.* at 5:21–27.

The system of Freund “intercept[s] process loading and unloading” and “keep[s] a list of currently-active processes” such that each client process can be identified, for example, by checking executable name or executable file checksums. *Id.* at 4:40–45. “With this information, the system can determine if a particular process in question should have access to the Internet and what kind of access . . . is permissible for the given specific user.” *Id.* at 4:45–49. More specifically, the system can employ a client-side monitor that “hooks” into the network communications driver (e.g., Windows Winsock driver). *Id.* at 15:22–26, 31:3–31. The monitor can intercept network requests from an application to determine whether the requests are permitted under the access rules, such as by verifying that the identified application is permitted to access the Internet by checking the rules and the application database. *Id.* at 15:26–16:7; 29:25–29. Also, the monitor can examine port address information provided with a network

command to determine the underlying protocol and whether the protocol is permitted for the application. *Id.* at 16:14–25, 29:43–53.

4. *Comparison of Claimed Subject Matter and Prior Art*

a. *Claim 1*

Independent claim 1 recites, “a port monitoring unit extracting information about a server port, wherein the server port is designated as a port of a network communication program.” We agree with Petitioner that the ARE disclosed in Yadav teaches this limitation. *See* Pet. 31–33. The operation of the ARE is depicted in Figure 3 (reproduced above). As described in Yadav, an application makes a network I/O request, and the ARE compares the request to the relevant application-specific network policy. Ex. 1002, 7:53–55, Fig. 3 (block 320). If the policy is satisfied, the ARE notifies the NTW to open a channel for that application’s request. Ex. 1002, 7:55–57. That notification may include, for example, source and destination port information. *Id.* at 7:57–60. Based on Dr. Prakash’s credible testimony, Petitioner argues persuasively that this teaches the extraction of that information. Pet. 32–33; *see* Ex. 1004 ¶¶ 240–249. Although Patent Owner disputes that Yadav teaches this limitation, its argument is conclusory and unpersuasive. PO Resp. 41.

Petitioner further contends Freund also teaches the “port monitoring unit” limitation. Pet. 33–34. We agree with Petitioner’s analysis and find both credible and persuasive the testimony of Dr. Prakash cited in support of that analysis. Dr. Prakash explains in detail how the discussion in Freund of the Client Monitor intercepting Winsock command calls teaches extracting information about a server port designated as a port of an application seeking to receive Internet communications. *See* Ex. 1004 ¶¶ 255–263.

Patent Owner argues Freund does not teach extracting server port information, asserting that Freund required manual input of such information by a user. PO Resp. 38–39, 42. Although Patent Owner notes that a user input interface depicted in Figure 7B of Freund refers to “protocols,” Patent Owner fails to persuasively rebut Dr. Prakash’s detailed testimony regarding other teachings in Freund of extracting port information. *See id.* The testimony from Mr. Bernstein identified by Patent Owner (*id.* at 42) only repeats Patent Owner’s argument without further explanation. *See Ex. 2007 ¶ 119.* Thus, we find that Freund also teaches the recited “port monitoring unit” limitation of claim 1.

Claim 1 next recites, “an internal permitted program storage storing a list of programs permitted by the firewall.” Petitioner argues Yadav teaches this limitation in its disclosure of a repository of application-specific network policies loaded by the ARE. Pet. 35–38. We agree. As discussed above, Yadav describes the ARE identifying an invoked application and loading an application-specific network policy. *See Ex. 1002, 7:45–51.* The application-specific network policy may be loaded from a remote repository of such policies to a local repository. *Id.* at 7:46–51. According to Dr. Prakash, a person of ordinary skill also would have understood Yadav to teach that the application-specific network policy is loaded to a local memory, an opinion we find credible. Ex. 1004 ¶¶ 272–273. In an example of such a policy, Yadav indicates the policy itself could include information about the application, such as the application name. *See Ex. 1002, 3:61–67* (policy for the application “Internet Explorer”). Thus, the local repository and local memory containing application-specific network policies teach or suggest an internal permitted program storage storing a list of programs permitted by the firewall, as recited in claim 1.

We also agree with Petitioner's assertion that Freund teaches this limitation. *See* Pet. 38–40. Freund discloses a “database of application[s] allowed to access the Internet,” including “what kind of activity the application is allowed to do.” Ex. 1003, 5:56–60. Further, Freund describes signaling a firewall to indicate “which client applications have been ‘certified’ so that the firewall only grants Internet access to those clients.” *Id.* at 5:21–27. A copy of the rules governing a particular client are stored locally by a filter application. *Id.* at 5:12–15. Thus, Freund also teaches or suggests an internal permitted program storage storing a list of programs permitted by the firewall, as recited in claim 1. Patent Owner does not dispute that Yadav and Freund teach this limitation.

In conjunction with the recited “internal permitted program storage,” claim 1 further recites, “wherein the internal permitted program storage adds a program to the list [of programs] by extracting information about the program for which communication is to be permitted by the firewall.” According to Petitioner, Yadav teaches this limitation in its description of loading an application-specific network policy into the local repository or local memory once an invoked application is identified based on information about the application. Pet. 41–42. We agree. Yadav describes a procedure for identifying an invoked application in which the ARE obtains information about the application, such as the full path of the application executable or a hash value. Ex. 1002, 7:22–44. When the application is identified, the appropriate application-specific network policy (which includes the application name) is loaded and added to the already-loaded policies (i.e., the list of programs) in the local repository or local memory (i.e., the internal permitted program storage). *See id.* at 7:45–46. Although the ARE is the structure that extracts the information about the application and not the local

repository/memory, we find that a person of ordinary skill in the art would have found obvious combining this function of the ARE (extracting) with the local repository/memory (storing) based on these teachings of Yadav. *See* Ex. 1004 ¶ 298; *KSR*, 550 U.S. 421 (indicating that a person of ordinary skill is “a person of ordinary creativity, not an automaton”).

Patent Owner argues Yadav’s disclosure does not teach this limitation because it “cannot mean simply copying information (that was previously extracted),” and “can in no way be equated to some mechanical internal copying of bits of memory from one data structure to another or one storage location to another,” as in Yadav. PO Resp. 22–25. Along similar lines, Patent Owner maintains that Yadav falls short because it does not teach “how one would create . . . a rule on the fly” (as opposed to copying a pre-existing rule). *Id.* at 41–42; *see id.* at 34–35. As discussed above with respect to claim construction, however, the claim does not exclude adding a program by copying pre-existing information. Yadav teaches adding an application to a list of permitted applications—i.e., adding an application-specific network policy (including application name) to a repository of such policies. Yadav further teaches that the adding of the policy is done by extracting information about the application—i.e., by extracting information to identify the application (e.g., a hash value) and, thus, load the appropriate policy. Whether the policy is new and not pre-existing is inapposite because the claim does not recite such a requirement.

At bottom, Patent Owner’s argument appears to be based on a misunderstanding of Petitioner’s position. In Patent Owner’s view, the claim is directed to the creation of “rules” (though “rules” are not recited in the claim), which associate a program with a server port. *See* Tr. 44:3–10, 45:13–46:9. According to Patent Owner, Yadav does not teach creating

“rules” because the application-specific network policies are not created, but rather are copied from a pre-existing repository. *See* PO Resp. 22–25, 34–35. Even putting aside the question of whether the claim is properly understood as requiring the creation of “rules” (which conflates multiple different limitations), the application-specific network policies of Yadav are not advanced by Petitioner as “rules” (as Patent Owner defines them), but rather as the “list of programs” recited in claim 1. *See* Pet. 31–32, 37. Petitioner relies on other teachings of Yadav to address the server port aspects of the claim. As explained above, we agree with Petitioner that the application-specific policies of Yadav teach the recited “list of programs,” and that Yadav teaches adding to that list by adding policies to the local repository or local memory storing such policies.

Petitioner also asserts that Freund teaches the “internal permitted program storage adds a program” limitation. Pet. 43–45. We disagree. Freund describes checking information about a process (i.e., a program) to determine if the process should be given access to the Internet. Ex. 1003, 4:40–49. Petitioner does not, however, explain how obtaining (i.e., extracting) this information is involved in adding to a list of programs. In fact, the identified portions of Freund relate to obtaining information about an active process to check if it is *already listed* as a program permitted to access the Internet, not to *add* it to the list of such programs. *See id.* Petitioner also cites portions of Freund describing a user manually selecting applications to be subject to a new rule being created. Pet. 44. Although Petitioner asserts that this selection “triggered addition of the rule and its details” (*id.*), the evidence cited does not describe any extraction of information about the applications. *See* Ex. 1003, 25:20–26, 27:45–47.

Nonetheless, Petitioner relies on the combination of Freund with Yadav, not Freund alone.

Claim 1 next recites, “a firewall flexible device determining whether the network communication program is registered in the list of programs stored in the internal permitted program storage.” We agree with Petitioner that the ARE of Yadav teaches this limitation.⁷ Yadav states, “When a request is received for network service (i.e., a network input/output (I/O) request) from an invoked application, a check is made as to whether the request violates a network policy.” Ex. 1002, 3:52–55; *see also id.* at Fig. 3, 7:53–60 (describing the ARE checking an application’s network request against application-specific network policies). Dr. Prakash testified that Yadav’s description of the ARE, which can “dynamically add or remove channels to the NTE based on the program that was making the request,” indicates it made firewall functions “flexible,” testimony we find credible and persuasive. Ex. 1004 ¶ 319. Patent Owner does not dispute that Yadav teaches this limitation.

With respect to the firewall flexible device, claim 1 further recites, “wherein the firewall flexible device automatically stores the extracted information about the server port in an internal permitted port storage if the network communication program is registered in the list of programs stored in the internal permitted program storage.” According to Petitioner, Yadav

⁷ Patent Owner does not contest Petitioner’s allegations on the basis that the ARE is cited as teaching both the recited “port monitoring unit” and “firewall flexible device.” Regardless, we note that the proper obviousness inquiry is not whether each specific structure of the claimed invention is “expressly suggested in any one or all of the references,” but rather what the references’ teachings “would have suggested to those of ordinary skill in the art.” *In re Keller*, 642 F.2d 413, 425 (CCPA 1981).

teaches this limitation. Pet. 47–50. We agree with Petitioner’s analysis. As discussed above, Yadav teaches that the ARE extracts server port information from an application’s network I/O request. The extracted port information is then sent to the NTE to open a channel for communications pursuant to that request. See Ex. 1002, 7:53–60. Yadav indicates that “[w]hen a notification of an opened channel is received, the opened channel is added to an authorization list.” *Id.* at 8:21–23; see *id.* at Figs. 3, 4. According to Dr. Prakash, a person of ordinary skill in the art would have understood Yadav as teaching that adding the channel to the authorization list would be automatic, i.e., performed without user intervention. Ex. 1004 ¶¶ 340–341. Based on these disclosures of Yadav and Dr. Prakash’s credible testimony, we are persuaded Yadav teaches the firewall flexible device automatically storing the extracted port information (i.e., information for the opened channel) in an internal permitted port storage (i.e., the NTE’s authorization list), as recited in claim 1.

Further, Yadav indicates that the channel is opened and the NTE provided the port information only if the application-specific network policy is satisfied. Ex. 1002, 7:55–60. As discussed above, the application-specific network policy is loaded when an application is identified (i.e., the application is added to the list of programs). Thus, Yadav teaches that the ARE sends the port information (which is automatically stored) only if the application is registered in the list of programs in the internal permitted program storage, i.e., the application-specific network policy is included in the repository of such policies.

Patent Owner disputes that Yadav teaches this limitation, but its arguments are unpersuasive. According to Patent Owner, Yadav does not teach this limitation because it does not automatically store extracted port

information as part of rule creation, instead using pre-existing port information provided by an administrator for rule enforcement. *See* PO Resp. 32, 34, 46–47; Tr. 50:23–53:6. The purported distinction between rule creation and rule enforcement, however, is not linked to any specific claim limitations—indeed, as previously noted, the claims make no mention of “rules” at all, much less “creating” or “enforcing” them. Patent Owner again improperly conflates different claim limitations and misapprehends Petitioner’s contentions, focusing on the fact that the application-specific network policies are copied from a pre-existing repository and, according to Patent Owner, may have been created manually by an administrator. *See id.* Petitioner, however, does not allege that those policies are the internal permitted port storage where port information is automatically stored. Rather, as discussed above, the port information extracted from a network I/O request is stored by adding the opened channel to the NTE’s authorization list (i.e., the internal permitted port storage).

Finally, claim 1 recites, “wherein the firewall flexible device further determines whether a port of a packet of inbound traffic matches with the server port and blocks the packet of inbound traffic if the port does not match with the server port.” The parties do not dispute that Yadav teaches this limitation, as Petitioner contends. We agree with Petitioner’s analysis and are persuaded the cited evidence supports Petitioner’s contentions. *See* Pet. 50–51. As Petitioner asserts, Yadav indicates that the NTE monitors communications and blocks them if they do not use a channel on its authorization list, which teaches blocking inbound traffic if the port does not match the port information stored by the NTE. *See* Ex. 1002, 6:25–36, 8:25–33, Fig. 4. Although the NTE is the structure that monitors and blocks communications, we find that a person of ordinary skill in the art would

have found obvious combining this function of the NTE with the ARE and its teachings of the other functions of the recited firewall flexible device, based on the teachings of Yadav. *See* Ex. 1004 ¶¶ 343–348; *KSR*, 550 U.S. 421 (indicating that a person of ordinary skill is “a person of ordinary creativity, not an automaton”).

For the above reasons, we conclude a preponderance of the evidence supports Petitioner’s contention that Yadav alone, as well as the combination of Yadav and Freund, teach or suggest each limitation of claim 1. With respect to the combination of Yadav and Freund, Petitioner also articulates sufficient reasoning with rational underpinning to combine their teachings in the manner asserted, which Patent Owner does not dispute. *See* Pet. 21–25. For example, Petitioner notes that both Yadav and Freund concerned management of network communications using many common features, such as intercepting network communication requests, end-point filtering in conjunction with centralized control, use of Windows-based architecture (e.g., Winsock hooking), and integration with firewalls. *Id.* at 22–24. Petitioner also highlights that both Yadav and Freund sought to address the common problem of alleviating bandwidth demands associated with monitoring network communications. *Id.* at 23. Dr. Prakash provided detailed testimony supporting these allegations, which we find credible and persuasive. *See* Ex. 1004 ¶¶ 221–231.

b. Remaining Challenged Claims

Petitioner contends that claims 2–5, 16, and 18–20 are unpatentable as obvious in view of Yadav alone, or Yadav combined with Freund. Pet. 52–60. Patent Owner does not argue these claims separately, relying on the

same arguments as it advances for claim 1. PO Resp. 47. Upon review of the full record after trial, we agree with Petitioner's analysis of these claims.

More specifically, claim 2 recites, "the information about the program includes information about at least one of a program name, an entire path of the program, and a program hash value." Yadav teaches identifying invoked applications using, for example, the full path of the application and/or its hash value. Ex. 1002, 7:26–37.

Claim 3 recites, "the information about the server port includes information about at least one of an entire path of the program, a protocol, and a port." Yadav teaches extracting, for example, source and destination ports and the protocol from the network I/O request. Ex. 1002, 7:53–60.

Claim 4 recites, "the firewall flexible device allows the packet of inbound traffic to bypass the firewall if the port matches with the server port." As discussed above, Yadav teaches that the NTE monitors communications to block them if they do not occur on an authorized channel (e.g., the port does not match the port of an opened channel on the NTE's authorization list). See Ex. 1002, 6:25–36. Authorized communications, conversely, are allowed to pass, including bypassing the firewall. *Id.* at 6:9–14; Ex. 1004 ¶ 371.

Claim 5 recites, "the firewall flexible device stores the extracted information about the server port in the internal permitted port storage if the server port is determined to be opened." As discussed above, Yadav teaches that the ARE notifies the NTE to open a channel, and the opened channel is added to the authorization list. Ex. 1002, 7:53–57, 8:21–23.

The remaining challenged claims recite limitations substantively similar to those discussed above, and Petitioner relies on essentially the same evidence and arguments as set forth above. Petitioner further relies on

the same reasoning to combine Yadav and Freund for all challenged claims, which is persuasive and supported sufficiently by the record for the same reasons as discussed above with respect to claim 1.

For the reasons explained above, we conclude a preponderance of the evidence supports Petitioner's contention that Yadav alone, as well as the combination of Yadav and Freund, teach or suggest each limitation of claims 2–5, 16, and 18–20.

5. *Conclusions as to Obviousness of the Challenged Claims*

As explained above, Petitioners have demonstrated sufficiently that Yadav alone, as well as the combination of Yadav and Freund, teach or suggest each limitation of the challenged claims. Petitioners also have articulated sufficient reasoning with rational underpinnings to combine Yadav and Freund in the manner asserted by Petitioner. Patent Owner has not presented any evidence of objective indicia of non-obviousness, or argued any such considerations compel a conclusion of patentability. Therefore, based on the full record after trial, we conclude Petitioners have established by a preponderance of the evidence that claims 1–5, 16, and 18–20 are unpatentable under § 103(a) in view of Yadav alone, as well as in view of the combined teachings of Yadav and Freund.

CONCLUSION

For the foregoing reasons, Petitioners have shown by a preponderance of the evidence that claims 1–5, 16, and 18–20 of the '078 patent are unpatentable under 35 U.S.C. § 103(a).

ORDER

It is

ORDERED that claims 1–5, 16, and 18–20 of the '078 patent are held unpatentable under 35 U.S.C. § 103(a);

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

PETITIONER:

James F. Valentine
Intel-CAP-IPR@perkinscoie.com
Ryan McBrayer
Intel-CAP-IPR@perkinscoie.com

PATENT OWNER:

Keith Kline
kkline@carrferrell.com
Bruce J. Wecker
bwecker@hausfeldllp.com