

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

CISCO SYSTEMS, INC.
Petitioner,

v.

CENTRIPETAL NETWORKS, INC.
Patent Owner.

Case IPR2018-01436
Patent 9,124,552

**PATENT OWNER'S PRELIMINARY RESPONSE
UNDER 37 C.F.R. § 42.107**

Table of Contents

	<u>Page</u>
I. Introduction.....	1
II. The ’552 Patent.....	3
A. Overview	3
B. Challenged Claims	5
III. Claim Construction.....	5
IV. Overview of the Prior Art.....	6
A. Summary of Sourcefire	6
V. Specific Reasons Why The Cited References Do Not Invalidate The Claims, And Why <i>Inter Partes</i> Review Should Not Be Instituted	9
A. Petitioner Improperly Incorporates Arguments by Reference and Fails to Make Its Arguments With the Required Specificity.....	9
B. Sourcefire in view of knowledge, skill and creativity (“Knowledge”) of a POSA Does Not Render Obvious Claims 1–21.....	13
1. Petitioner Has Not Demonstrated that Sourcefire in view of the Knowledge of a POSA Discloses applying an operator specifying “one or more application-header-field-value criteria identifying one or more transport layer security (TLS)-version values” (claims 1-21)	14
(a) Sourcefire Does Not Disclose the Recited Operator.....	15
(b) Sourcefire Does Not Disclose Blocking Packets Based on the Claimed Operator	23
2. Petitioner Has Not Demonstrated that Sourcefire in view of the Knowledge of a POSA Discloses applying an operator specifying “a determination by the at least one processor that one or more packets, of the at least a portion of the plurality of HTTPS packets, have one or more application-header-field	

values corresponding to one or more TLS-version values” (claims 1-21)	26
3. Petitioner Has Not Demonstrated that Sourcefire in view of the Knowledge of a POSA Discloses applying an operator specifying “at least one packet-transformation function specified by the operator” (claims 1-21).....	27
4. Petitioner Has Not Demonstrated That a POSA Would Have Modified Sourcefire to Reach the Invention Claimed in the ’552 Patent.....	28
(a) Petitioner Describes No Motivation to Modify Sourcefire to Prevent Exfiltrations.....	28
(b) Petitioner Describes No Motivation to Modify Sourcefire to Practice the “operator” Element of the Challenged Claims	30
(c) Petitioner Describes No Motivation to Modify Sourcefire to Practice the Blocking Element of the Challenged Claims	32
VI. CONCLUSION.....	35

TABLE OF AUTHORITIES

	Page(s)
Cases	
<i>ams AG v. 511 Innovations, Inc.</i> , IPR2016-01792, Paper 15 (PTAB Mar. 16, 2017)	6
<i>Belden Inc. v. Berk-Tek LLC</i> , 805 F.3d 1064 (Fed. Cir. 2015)	31
<i>Broadcom Corp. v. Emulex Corp.</i> , 732 F. 3d 1325 (Fed. Cir. 2013)	29
<i>Cisco Sys., Inc., v. C-Cation Techs., LLC</i> , IPR2014-00454, Paper 12 (PTAB Apr. 4, 2016)	10, 12
<i>In re Cuozzo Speed Techs., LLC</i> , 793 F.3d 1268 (Fed. Cir. 2015)	6
<i>DeSilva v. Dileonardi</i> , 181 F.3d 865, 866-67 (7th Cir. 1999)	10
<i>DirectTV, LLC, v. Qurio Holdings, Inc.</i> , IPR2015-02006, Paper 6 (PTAB Apr. 4, 2016)	12
<i>In re Gordon</i> , 733 F.2d 900, 902 (Fed. Cir. 1984)	31
<i>Insite Vision Inc. v. Sandoz, Inc.</i> , 783 F. 3d 853 (Fed. Cir. 2015)	32
<i>Kinetic Technologies, Inc. v. Skyworks Solutions, Inc.</i> , IPR2014-00529, Paper 8 (PTAB Sept. 23, 2014).....	34
<i>In re McLaughlin</i> , 443 F.2d 1392 (CCPA 1971)	32
<i>Monarch Knitting Mach. Corp. v. Sulzer Morat GmbH</i> , 139 F.3d 877, 881 (Fed.Cir.1998)	32
<i>Orexo AB v. Actavis Elizabeth LLC</i> , 903 F.3d 1265 (Fed. Cir. 2018)	31

Travelocity.com L.P. v. Conos Techs., LLC,
CBM2014-00082, Paper 12 (PTAB Oct. 16, 2014) 2

Statutes

35 U.S.C. § 103.....13
35 U.S.C. § 312(a)(3).....13

Other Authorities

37 C.F.R. § 42.6(a)(3) 9
37 C.F.R. §§ 42.22(a)(2), 42.104(b)(4)–(5).....13
37 C.F.R. § 42.100(b) 6
37 C.F.R. § 42.104(b)(3).....6
37 C.F.R. § 42.104(b)(4).....12, 15, 27
37 C.F.R. § 42.108(c).....1

I. INTRODUCTION

On July 20, 2018, Cisco Systems, Inc., (“Cisco” or “Petitioner”) submitted a Petition to institute *inter partes* review (“IPR”) of U.S. Patent No. 9,124,552 (Ex. 1001, “the ’552 Patent”), challenging claims 1-21. Centripetal Networks, Inc. (“Patent Owner”) requests that the Board not institute *inter partes* review because Petitioner has not demonstrated a reasonable likelihood that it would prevail in showing unpatentability of any of the challenged claims on the grounds asserted in its Petition, as required under 37 C.F.R. § 42.108(c).

The Petition should be denied because the Sourcefire reference—the only reference cited against the claims—admittedly does not disclose all elements of the independent claims. For example, Sourcefire does not disclose applying an operator that specifies an “application-header-field-value criteria identifying” a “transport layer security (TLS)” version value to received HTTPS packets. Rather than inspecting received HTTPS packets to determine whether those packets include application field values that match an identified TLS version value, Sourcefire discloses extracting SSL version information from non-HTTPS packets during the SSL handshake procedure and associating that version number with the entire SSL session. Thus, Sourcefire does not disclose applying an operator specifying one or more application-header-field-value criteria to the received

HTTPS packets or determining that one or more of those HTTPS packets has an application-header-field values corresponding to a the specified TLS-version value.

Furthermore, Petitioner's alleged motivation to modify the Sourcefire reference to teach the claims of the '552 Patent lacks any reasonable foundation. For example, although Petitioner notes that the '552 Patent is designed to prevent exfiltration of data from a protected network (e.g., data leaving the network), it provides no explanation of why a person of skill in the art (POSA) would have modified Sourcefire in order to prevent the type of exfiltrations described in the '552 Patent. As described in the '552 Patent, exfiltration "has proven to be especially difficult for conventional cyber defense systems to prevent." '552 Patent at 1:16–18. Accordingly, Petitioner's failure to address why a POSA would modify a "conventional cyber defense system[]"—Sourcefire—to stop the types of exfiltrations addressed by the '552 Patent lacks any reasonable foundation. Nor has Petitioner met its burden to demonstrate that a POSA would modify Sourcefire to practice at least the "operator" or "blocking" elements of the challenged claims.

Although there are a variety of reasons why the '552 Patent is valid over Petitioner's asserted prior art references, this Preliminary Response focuses on only limited reasons why *inter partes* review should not be instituted. See *Travelocity.com L.P. v. Conos Techs., LLC*, CBM2014-00082, Paper 12 at 10 (PTAB Oct. 16, 2014) ("[N]othing may be gleaned from the Patent Owner's

challenge or failure to challenge the grounds of unpatentability for any particular reason.”). The deficiencies of the Petition noted herein, however, are sufficient for the Board to find that Petitioner has not met its burden to demonstrate a reasonable likelihood that it would prevail in showing unpatentability of any of the challenged claims.

II. THE '552 PATENT

A. Overview

The '522 Patent generally discloses systems and methods for “filtering network data transfers.” '552 Patent at Title. In particular, the '552 Patent discloses exemplary techniques for preventing “[a] category of cyber attack known as exfiltrations (e.g., stealing sensitive data or credentials via the Internet). . . .” *Id.* at 1:15–16. The exemplary techniques involve a two stage process whereby *first*, “[a] determination may be made that a portion of the packets have packet header field values corresponding to a packet filtering rule,” and *second* “[a] further determination may be made that one or more of the portion of the packets have one or more application header field values corresponding to one or more application header field criteria specified by the operator.” *Id.* at 1:49–51; 1:54–58.

For example, as described with respect to FIG. 4, “a determination may be made as to whether a portion of the packets received from network 104 have packet header field values (e.g., one or more of one or more data section protocols,

one or more source IP addresses, one or more source ports, one or more destination IP addresses, or one or more destination ports) corresponding to rule 5 310.” *Id.* at 9:2–8. “Responsive to determining that one or more of the portion of received packets have packet header field values corresponding to the packet filtering rule, an operator specified by the packet filtering rule may be applied to the portion of the received packets.” *Id.* at 9:8–12.

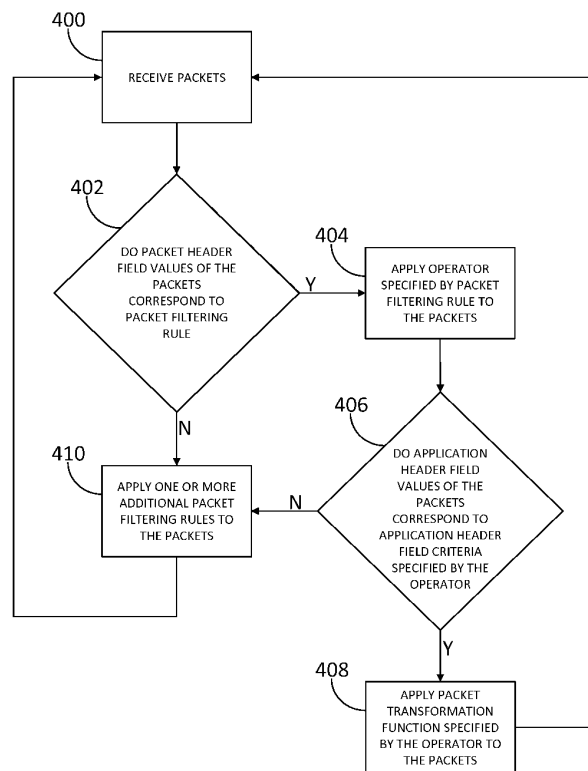


FIG. 4

'552 Patent at FIG. 4.

Next, “a determination may be made as to whether one or more application header field values of one or more of the portion of the received packets

correspond to one or more application header field criteria specified by the operator,” such as “whether one or more of the portion of the received packets have application header field values corresponding to one or more application header field criteria of the REQUIRE TLS-1.1-1.2 operator specified by rule 5 310 (e.g., application header field values corresponding to TLS version 1.1 or 1.2).” *Id.* at 9:17–26.

“[R]esponsive to determining that one or more of the portion of received packets have application header field values corresponding to one or more application header field criteria specified by the operator, a packet transformation function specified by the operator may be applied to the one or more of the portion of the received packets.” *Id.* at 9:26–32.

B. Challenged Claims

Petitioner challenges claims 1-21 of the '552 Patent, of which claims 1, 8 and 15 are independent. The challenged claims are reproduced in Appendix A:

III. CLAIM CONSTRUCTION

In an *inter partes* review proceeding, claims are given their broadest reasonable interpretation (“BRI”) in light of the specification of the patent in which they appear. 37 C.F.R. § 42.100(b); Office Patent Trial Practice Guide, 77 Fed. Reg. 48756 at 48766 (Aug. 14, 2012); *see also In re Cuzco Speed Techs., LLC*,

793 F.3d 1268, 1278 (Fed. Cir. 2015) (“We conclude that Congress implicitly approved the broadest reasonable interpretation standard in enacting the AIA.”).

The Petition should be rejected because Petitioner fails to identify “[h]ow the challenged claim is to be construed” under 37 C.F.R § 42.104(b)(3). By not providing any explicit constructions for any claim terms, “Petitioner improperly attempts to shift the burden of construing the claim... to the Board.” *ams AG v. 511 Innovations, Inc.*, IPR2016-01792, Paper 15 at 8 (PTAB Mar. 16, 2017).

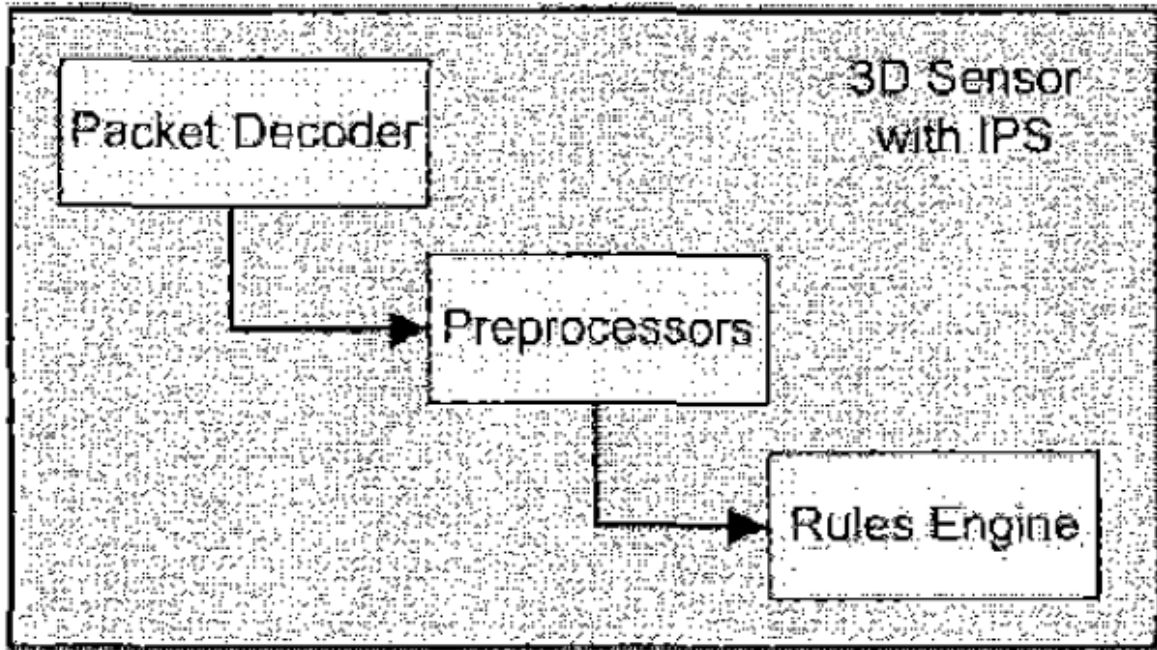
IV. OVERVIEW OF THE PRIOR ART

A. Summary of Sourcefire

Sourcefire discloses a three-stage process for its Intrusion Prevention System (“IPS”):

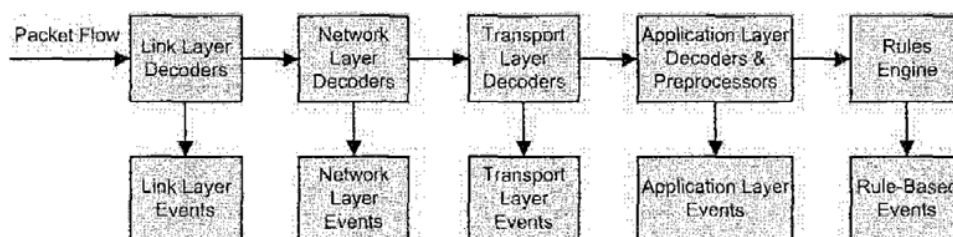
First, packet decoders and preprocessors detect anomalous traffic that might signal an intrusion attempt and, when you have enabled accompanying decoder and preprocessor rules, report on detected anomalies. Next, the detection engine uses intrusion rules to examine the decoded packets for attacks based on patterns.

Sourcefire at 254.



Sourcefire at 259.

One of the Application Layer Preprocessors disclosed in Sourcefire is the “SSL Preprocessor,” which “maintains state information as it inspects the SSL handshake, tracking both the state and SSL version for that session.” Sourcefire at 597. As may be appreciated with reference to the figure shown below, the SSL Preprocessor extracts the SSL version information for a session during the SSL handshake using the SSL Preprocessor (in the Application Layer Decoders & Preprocessors Block), well before the Rules Engineer applies any rule incorporating the `ssl_version` keyword.



Sourcefire at 256. Further, when the rules engine encounters the `ssl_version` keyword, Sourcefire only discloses identifying, but not blocking as required by the independent claims of the '552 Patent, traffic using a particular “version of SSL.” Sourcefire at 827 (stating that “you could use the `ssl_version` keyword with the `sslv2` argument to identify traffic using that version of SSL.”).

Because Sourcefire fails to inspect the application header of HTTPS packets and does not disclose blocking packets that include application header field values, it should be apparent then, that Sourcefire fails to disclose or suggest at least the following elements of the '552 Patent's independent claims:

- applying, by the at least one processor, an operator specified by the packet-filtering rule to the at least a portion of the plurality of HTTPS packets,
- wherein the operator specifies one or more application-header-field-value criteria identifying one or more transport layer security (TLS)-version values
- for which packets should be blocked from continuing toward their respective destinations;
- responsive to a determination by the at least one processor that one or more packets, of the at least a portion of the plurality of HTTPS

packets, have one or more application-header-field values corresponding to one or more TLS-version values of the one or more TLS-version values for which packets should be blocked from continuing toward their respective destinations,

- applying, by the at least one processor, at least one packet-transformation function specified by the operator to the one or more packets to block each packet of the one or more packets from continuing toward its respective destination..

'552 Patent, claim 1.

V. SPECIFIC REASONS WHY THE CITED REFERENCES DO NOT INVALIDATE THE CLAIMS, AND WHY *INTER PARTES* REVIEW SHOULD NOT BE INSTITUTED

Petitioner proposes that claims 1-21 of the '552 Patent are obvious over the Sourcefire reference in view of the knowledge of a POSA. As detailed below, however, the Petition should be denied at least for the several reasons set forth below.

A. Petitioner Improperly Incorporates Arguments by Reference and Fails to Make Its Arguments With the Required Specificity

Petitioner certifies that the Petition includes 13,995 words. See Petition at "Certificate of Compliance." However, the Petition incorporates vast amounts of information from other documents by reference and fails to explain with the required specificity how that information renders the challenged claims obvious.

Under the Board's rules, "[a]rguments must not be incorporated by reference from one document into another document." 37 C.F.R. § 42.6(a)(3). And in one

of the Board's informative opinions on this topic, the Board found that the citing "large portions of another document, without sufficient explanation of those portions, amounts to incorporation by reference." *See Cisco Sys., Inc., v. C-Cation Techs., LLC*, IPR2014-00454, Paper 12 at 8 (PTAB Apr. 4, 2016) (informative opinion). Nevertheless, Petitioner consistently makes conclusory assertions and then cites to large page ranges of its exhibits, including the Staniford Declaration, without making the underlying evidence available in the Petition. This tactic is improper because it fails to make the actual evidence "accessible to the judges" and "amounts to a self-help increase in the length of the brief." *See Cisco* at 10 (informative opinion) (citing *DeSilva v. Dileonardi*, 181 F.3d 865, 866-67 (7th Cir. 1999)) ("Incorporation by reference amounts to a self-help increase in the length of the brief, and is a pointless imposition on the court's time. A brief must make all arguments accessible to the judges, rather than ask them to play archeologist with the record.") (modifications omitted)).

The overwhelming majority of paragraphs in the Petition follow the same format: Sourcefire discloses X, followed by a string-cite to vast page ranges of the Sourcefire reference as well as one or more paragraphs of the Staniford declaration. For example, the following paragraph contains two conclusory statements that baldly cite to 17 pages of the Sourcefire reference and a paragraphs

of the Staniford declaration without actually presenting the evidence disclosed therein:

Thus, a user could protect a network against a security vulnerability associated with a specific encryption protocol version (*e.g.*, TLS 1.0) by writing a rule using the `ssl_version` keyword to invoke the SSL preprocessor to identify traffic using that version of SSL or TLS, and either *pass* (allow), *alert* (and pass), or *drop* (block) packets identifying that protocol version. Sourcefire explicitly discloses that such rules could be implemented by the rules engine invoking the SSL preprocessor to inspect the SSL/TLS traffic. *CS-1004* pp.827-828, 491, 597-601, 435-439, 697-700; *CS-1003* ¶121.

Petition at 30. Because Petitioner fails to make the underlying evidence available in the Petition itself, it is impossible to gauge whether Petitioner's assertions accurately characterize the underlying information without scouring each page of the many pages and paragraphs cited for the many conclusory assertions made in the Petition.

This tactic improperly forces Patent Owner and the Board to “play archaeologist with the record” to determine whether or not the conclusory assertion made in the paragraph is actually supported by the record evidence. And, as demonstrated below, Petitioner abuses this tactic to mischaracterize the Sourcefire reference in many respects. *See* § VI.B.1-2, *infra* (detailing numerous

mischaracterizations of Sourcefire with respect to the “applying... an operator” element of the challenged claims).

While the Board in the informative *Cisco* case, cited above, determined that the proper course of action would be to “not consider arguments that are not made in the Petition, but are instead incorporated by reference,” the problem is so pervasive in this Petition that it is unclear whether any arguments would remain standing, and outright denial of *inter partes* review is warranted here. *Cisco* at 10. Indeed, to the extent that Petitioner’s “arguments” are actually made in the Petition, those arguments so completely torture the underlying evidence as to become unreliable sources for the Board to find a reasonable likelihood that any claim of the ’552 Patent is obvious over Sourcefire.

Furthermore, Petitioner’s tactic of making a conclusory assertion and allegedly supporting that assertion with bare citations to large portions of the record evidence also results in the Petition failing to comply with “the particularity and specificity required of supporting evidence under [the Board’s] governing statute and rules,” including 37 C.F.R. § 42.104(b)(4). *DirectTV, LLC, v. Qurio Holdings, Inc.*, IPR2015-02006, Paper 6, at 10 (PTAB Apr. 4, 2016). When Petitioner makes conclusory assertions and cites to large portions of a document that allegedly supports that assertion, without making the evidence available in the Petition or explaining how the evidence actually supports the assertion, the Petition

cannot satisfy the Board's "governing statute and rules," including 35 U.S.C. § 312(a)(3) and 37 C.F.R. §§ 42.22(a)(2), 42.104(b)(4)–(5). *Id.*

Thus, due to the pervasive and improper incorporation by reference of arguments and evidence not presented in the Petition and Petitioner's failure to meet their burden to state their challenge with "particularity and specificity," Patent Owner requests that the Board decline to institute *inter partes* review of the '552 Patent.

**B. Sourcefire in view of knowledge, skill and creativity
("Knowledge") of a POSA Does Not Render Obvious Claims 1–21**

The Board should decline to institute a *inter partes* review at least because the Sourcefire reference does not render challenged claims 1-21 obvious under 35 U.S.C. § 103.

The Petition should be denied because Sourcefire fails to disclose or suggest:

- "applying... an operator specified by the packet-filtering rule to the at least a portion of the plurality of HTTPS packets, wherein the operator specifies one or more application-header-field-value criteria identifying one or more transport layer security (TLS)-version values for which packets should be blocked from continuing toward their respective destinations,"
- "a determination by the at least one processor that one or more packets, of the at least a portion of the plurality of HTTPS packets, have one or more application-header-field values corresponding to one or more TLS-version values" or

- an operator specifying “at least one packet-transformation function specified by the operator”

as recited in each independent claim of the '552 Patent. Petitioner also has not met its burden to demonstrate that it would have been obvious to modify Sourcefire in order to use any “application-header-field-value criteria” at all, let alone “criteria identifying one or more transport layer security (TLS)-version values.”

1. Petitioner Has Not Demonstrated that Sourcefire in view of the Knowledge of a POSA Discloses applying an operator specifying “one or more application-header-field-value criteria identifying one or more transport layer security (TLS)-version values” (claims 1-21)

The '552 Patent teaches and claims applying an operator that specifies one or more application-header-field-value criteria identifying one or more transport layer security (TLS)-version values. See '522 Patent at 8:10-17 (“If the application packet contained in the IP packet is a TLS Record Protocol packet, the value of the version field in the TLS Record Protocol packet header may be compared to the values that encode version numbers 1.0, 1.1, and 1.2. If a match is found, then the REQUIRE-TLS-1.1-1.2 operator may return either ALLOW or BLOCK, depending on the version number value.”). Petitioner has not met its burden to demonstrate that Sourcefire discloses applying the claimed operator or blocking packets in response to applying the operator.

(a) Sourcefire Does Not Disclose the Recited Operator

Petitioner has not met its burden to demonstrate that Sourcefire discloses applying an operator specifying “one or more application-header-field-value criteria identifying one or more transport layer security (TLS)-version values,” as recited in independent claims 1, 8, and 15 of the '552 Patent.

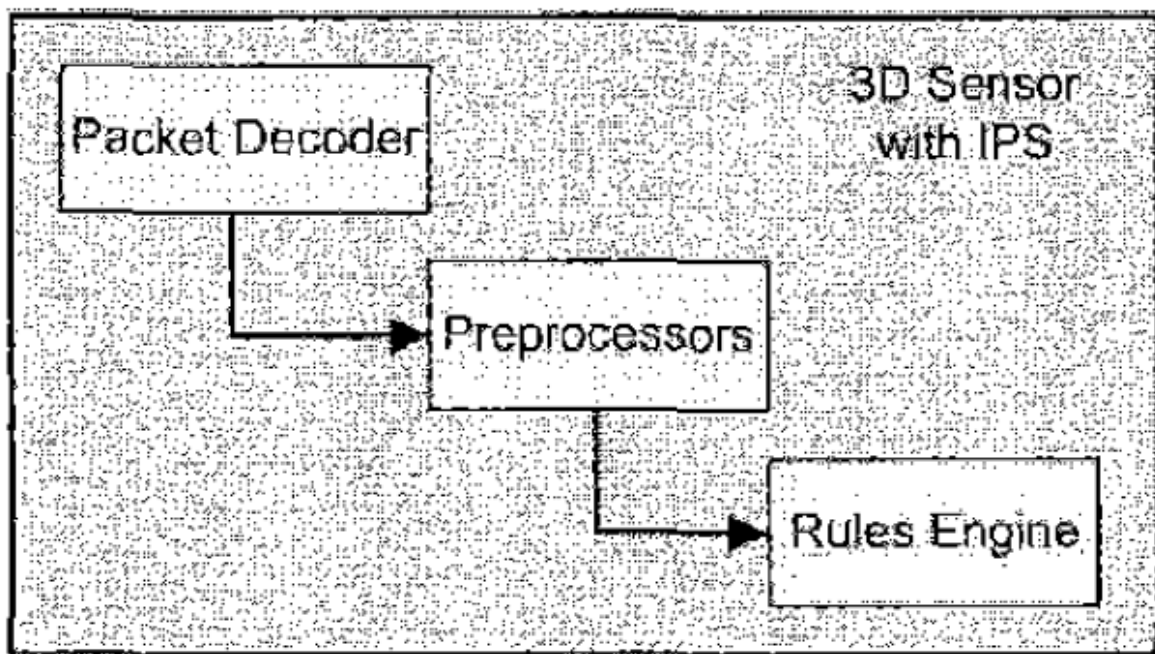
As a preliminary matter, Petitioner failed to meet its burden to “specify where each element of the claim is found in the prior art patents or printed publications relied upon.” 37 C.F.R. § 42.104(b)(4). For example, Petitioner fails to specify which disclosures in the Sourcefire reference, if any, correspond to the claimed “operator specified by the packet-filtering rule.” *See, e.g.*, Petition at 36–44 (failing to identify any teaching in Sourcefire as corresponding to the claimed “operator”). Similarly, Petitioner fails to identify any teaching in Sourcefire as allegedly corresponding to the claimed “application-header-field-value criteria.” *Id.* Instead of identifying where these claim elements are disclosed in the Sourcefire reference, Petitioner sets up and knocks down strawman arguments untethered to the language of the claims. *See, e.g.*, Petition at 39 (arguing that “a POSA would be motivated to identify and block packets including handshake Records that proposed using SSLv2 for a SSL session”). The Petition should be denied on this basis alone. Nevertheless, the following discussion identifies Petitioner’s numerous mischaracterizations of the Sourcefire reference, which

further demonstrate that Petitioner has not met its burden to demonstrate that the challenged claims are obvious.

Petitioner points to the `ssl_version` keyword disclosed in Sourcefire to suggest that “Sourcefire discloses the use of the `ssl-version` [sic] keyword for writing intrusion rules (which will invoke the SSL preprocessor) to identify the SSL/TLS version of received packets.” However Sourcefire’s use of the `ssl_version` keyword does not “specif[y] one or more application-header-field-value criteria identifying one or more transport layer security (TLS)-version values,” as recited in each of the independent claims of the ‘522 Patent, because the `ssl_version` keyword does not cause the Sourcefire appliance to look at any “application-header-field-value criteria,” let alone such criteria corresponding to a TLS-version value.

To the contrary, Sourcefire discloses using `ssl_version` keywords to detect the SSL or TLS version being used for a particular session. Sourcefire at 597 (“The SSL preprocessor maintains state information as it inspects the SSL handshake, tracking both the state and SSL version for that session.”). In particular, as Sourcefire’s SSL preprocessor encounters handshake messages, it “extracts state and version information from specific handshake fields. Two fields within the handshake indicate the version of SSL or TLS used to encrypt the session and the stage of the handshake.” Sourcefire at 825.

Sourcefire's SSL preprocessor extracts this SSL version data **prior to** Sourcefire's "Rules Engine" applying any rule to "check traffic for SSL version information." Sourcefire at 827 ("The ssl_version keyword can be used to match against version information for an encrypted session. When a rule uses the ssl_version keyword, the rules engine invokes the SSL preprocessor to check traffic for SSL version information."). The figure below further illustrates how the Preprocessor stage operates prior to the Rules Engine stage.



Sourcefire at 259.

That is, the ssl_version keyword is not part of an "operator [that] specifies one or more application-header-field-value criteria," but rather a rule that merely "invoke[s] the preprocessor to obtain SSL state and version information" that was

previously extracted and associated with the SSL session. Sourcefire at 599.

Accordingly, the “version information” disclosed in Sourcefire is not “application-header-field-value criteria,”—it is information extracted and maintained by the SSL preprocessor based on handshake packets prior to any rule being applied.

Sourcefire at 597 (“The SSL preprocessor maintains state information as it inspects the SSL handshake, tracking both the state and SSL version for that session.”). In other words, Sourcefire teaches determining the SSL version during the **SSL handshake** such that the Rules Engine can request the SSL version if a rule is written that incorporates the `ssl_version` keyword. Petitioner mischaracterizes this disclosure from Sourcefire in several ways.

First, Petitioner incorrectly argues that “a POSA understood that Sourcefire describes the use of SSL/TLS rule keywords to invoke the application-layer SSL preprocessor and extract information about SSL or TLS version and session state from Record headers in packets for an encrypted session.” Petition at 37.¹ This assertion is false. As discussed directly above, Sourcefire’s SSL Preprocessor extracts the SSL version information during the SSL handshake, well before any rule incorporating the `ssl_version` keyword invokes the SSL Preprocessor.

¹ Petitioner improperly cites to, but does not otherwise explain, 15 pages of the Sourcefire reference and 3 paragraphs of the Staniford Declaration to support this conclusory and factually incorrect argument.

The SSL preprocessor stops inspection of encrypted data, which can help to eliminate false positives. The SSL preprocessor maintains state information as it inspects the SSL handshake, tracking both the state and SSL version for that session. Once the preprocessor detects that a session state is encrypted, IPS marks the traffic in that session as encrypted. You can configure IPS to stop processing on all packets in an encrypted session once encryption is established.

Sourcefire at 597 (highlighting added). Thus, contrary to Petitioner's assertion, the SSL Preprocessor does not "extract" the version information upon being invoked by a SSL/TLS rule," let alone extract such information "from Record headers in packets for an encrypted session." Petition at 37. Rather, the SSL Preprocessor "maintains state information as it inspects the SSL handshake" and can provide that maintained information if invoked by the rules engine. Sourcefire at 597.

Second, Petitioner falsely claims that "Sourcefire discloses that the SSL preprocessor implemented the SSL preprocessor rules and intrusion rules including SSL keywords (e.g., ssl_version) for both SSL and TLS." Petition at 37. Contrary to Petitioner's assertion, the SSL preprocessor does **not** implement intrusion rules. Sourcefire at 769 ("Keywords and their associated values (called arguments) dictate how the system evaluates packets and packet-related values that the rules engine tests."). It is Sourcefire's "rules engine" that uses the "ssl_version keyword," which—rather than specifying any application-level packet-header information—merely requests the preprocessor to return the SSL version it already extracted from other packets associated with that session.

The `ssl_version` keyword can be used to match against version information for an encrypted session. When a rule uses the `ssl_version` keyword, the rules engine invokes the SSL preprocessor to check traffic for SSL version information.

Sourcefire at 827.

The SSL preprocessor in Sourcefire does have and implements “preprocessor rules.” But these rules do not relate to the SSL version:

The SSL Preprocessor Rules table describes the SSL preprocessor rules you can enable.

SSL Preprocessor Rules

Preprocessor Rule ID:SID	Description
137:1	Detects a client hello after a server hello, which is invalid and considered to be anomalous behavior.
137:2	Detects a server hello without a client hello when Server side data is trusted is disabled, which is invalid and considered to be anomalous behavior. See Configuring the SSL Preprocessor on page 599 for more information.

Sourcefire at 598. The SSL preprocessor tracks and maintains the SSL version for every session whether or not the state or session information is subsequently requested by the rules engine. Sourcefire at 597 (“The SSL preprocessor maintains state information as it inspects the SSL handshake, tracking both the state and SSL version for that session.”)

Third, Petitioner cites Sourcefire's "adaptive" rules to suggest that "a POSA understood that the SSL preprocessor and/or rules engine checks the unencrypted application-layer header (e.g., Record header) of packets for SSL/TLS service identifiers to determine if the packets were SSL/TLS traffic." Petition at 38.² This assertion is false as well as irrelevant. Sourcefire discloses that "if you are managing IPS using a Defense Center with RNA, you can enable the adaptive profiles feature to use service information from your RNA network map to identify SSL traffic traveling over non-standard ports." Sourcefire at 598. This argument is irrelevant because it concerns only whether or not the SSL Preprocessor will process a particular SSL handshake packet that was received over a non-standard port, not whether Sourcefire includes an operator that specifies "application-header-field-value criteria," for later-received HTTPS packets. *Id.* This argument is also incorrect because nowhere does Sourcefire disclose that the SSL preprocessor (and certainly not the rules engine) "checks the unencrypted application-layer header... to determine if the packets were SSL/TLS traffic." Petition at 38. Rather, Sourcefire discloses that such information is taken from the

² Petitioner improperly cites to, but does not otherwise explain, 6 pages of the Sourcefire reference and a paragraph of the Stanford Declaration to support this conclusory and factually incorrect argument.

“RNA network map,” which Petitioner fails to explain or even mention.

Sourcefire at 598.

Fourth, Petitioner falsely states that “Sourcefire describes that the ssl_version keyword could be used by the SSL preprocessor to match against such protocol version information in the Record header of received packets including unencrypted and encrypted Records exchanged in negotiating an encrypted SSL/TLS session.” Petition at 39.³ Contrary to Petitioner’s assertion, Sourcefire does not disclose using the “SSL preprocessor to match against such protocol version information in the Record header of received packets.” *Id.* Rather, “[t]he ssl_version keyword can be used to match against version information **for an encrypted session.**” Sourcefire at 827 (emphasis added). Thus, rather than using the “SSL preprocessor to match against such protocol version information **in the Record header of received packets,**” as asserted by Petitioner, Sourcefire merely teaches matching against the version information that the SSL preprocessor had previously extracted for that session. Petition at 39.

Petitioner also questions whether “the claim requires that intrusion rules using the ssl_version keyword must inspect the SSL/TLS version values in Record

³ Petitioner improperly cites to, but does not otherwise explain, 13 pages of the Sourcefire reference and a paragraph of the Stanford Declaration to support this conclusory and factually incorrect argument.

headers for encrypted *application-data* message packets, as opposed to the identical values in the identical Record headers for encrypted and unencrypted handshake packets.” Petition at 41. This question betrays Petitioner’s misunderstanding of the technology or a bad-faith mischaracterization of the claims. As recited in each independent claim, the operation is applied “to the at least a portion of the plurality of **HTTPS packets**.” ’552 Patent at claims 1, 8, and 15. As Petitioner’s exhibits demonstrate, HTTPS refers to HTTP traffic encrypted with TLS, and the “TLS handshake occurs long before any HTTP traffic.” CS-1039 at 2, 12. Thus, Sourcefire does not disclose applying an operator that specifies the recited “application-header-field-value criteria” to a portion of HTTPS packets because Sourcefire only extracts TLS version value from non-HTTPS packets.

(b) Sourcefire Does Not Disclose Blocking Packets Based on the Claimed Operator

As demonstrated directly above, Sourcefire does not disclose “applying, by the at least one processor, an operator specified by the packet-filtering rule to the at least a portion of the plurality of HTTPS packets, wherein the operator specifies one or more application-header-field-value criteria identifying one or more transport layer security (TLS)-version values.” See § VI.A. Petitioner also fails to demonstrate that Sourcefire discloses applying such an operator “identifying one or more transport layer security (TLS)-version values for which packets should be

blocked from continuing toward their respective destinations,” as recited in each independent claim of the '552 Patent.

As with the several, non-exhaustive mischaracterizations of the Sourcefire reference noted in the previous section, Petitioner also mischaracterizes the Sourcefire reference to assert that “Sourcefire discloses that the keyword `ssl_version` could be included in such intrusion rules and used to block harmful, or allow benign, Record Protocol packets.”⁴ Petition at 39. On the contrary, Sourcefire **never** discloses blocking packets based on applying an “operator specified by the packet-filtering rule to the at least a portion of the plurality of HTTPS packets, wherein the operator specifies one or more application-header-field-value criteria identifying one or more transport layer security (TLS)-version values,” as required by each independent claim of the '552 Patent. At best, Sourcefire vaguely references identifying, but does not disclose blocking, traffic using a “version of SSL”:

⁴ Petitioner improperly cites to, but does not otherwise explain, 13 pages of the Sourcefire reference and a paragraph of the Stanford Declaration to support this conclusory and factually incorrect argument.

The `ssl_version` keyword can be used to match against version information for an encrypted session. When a rule uses the `ssl_version` keyword, the rules engine invokes the SSL preprocessor to check traffic for SSL version information. For example, if you know there is a buffer overflow vulnerability in SSL version 2, you could use the `ssl_version` keyword with the `ssl_v2` argument to identify traffic using that version of SSL.

Sourcefire at 827 (highlighting added). And as described below, Petitioner provides insufficient motivation that a POSA would have modified Sourcefire to practice the claims of the '552 Patent. See § VI.B.4, *infra*.

For at least the foregoing reasons, Petitioner has not demonstrated that Sourcefire discloses “applying, by the at least one processor, an operator specified by the packet-filtering rule to the at least a portion of the plurality of HTTPS packets, wherein the operator specifies one or more application-header-field-value criteria identifying one or more transport layer security (TLS)-version values,” as recited in each independent claim of the '552 Patent. Accordingly, Patent Owner respectfully requests that the Board decline to institute *inter partes* review of the '552 Patent.

2. Petitioner Has Not Demonstrated that Sourcefire in view of the Knowledge of a POSA Discloses applying an operator specifying “a determination by the at least one processor that one or more packets, of the at least a portion of the plurality of HTTPS packets, have one or more application-header-field values corresponding to one or more TLS-version values” (claims 1-21)

As discussed directly above, Sourcefire does not disclose applying the “operator” disclosed and claimed in the ’552 Patent. For at least the same reasons, Sourcefire does not disclose making “a determination” that “at least a portion of the plurality of HTTPS packets[] have one or more application-header-field values corresponding to one or more TLS-version values.” In particular, rather than determining that an HTTPS packet includes the recited application-header-field value, Sourcefire discloses invoking the SSL preprocessor, which previously extracted the SSL version information for that session. Sourcefire at 597.

For at least the foregoing reasons, Petitioner has not demonstrated that Sourcefire discloses “a determination by the at least one processor that one or more packets, of the at least a portion of the plurality of HTTPS packets, have one or more application-header-field values corresponding to one or more TLS-version values,” as recited in each independent claim of the ’552 Patent. Accordingly, Patent Owner respectfully requests that the Board decline to institute *inter partes* review of the ’552 Patent.

3. Petitioner Has Not Demonstrated that Sourcefire in view of the Knowledge of a POSA Discloses applying an operator specifying “at least one packet-transformation function specified by the operator” (claims 1-21)

Petitioner fails to identify any teaching in Sourcefire that suggests a “at least one packet-transformation function specified by the operator” as is its burden under 37 C.F.R. § 42.104(b)(4). Rather, Petitioner conclusorily states that “a POSA understood that Sourcefire discloses that the TLS version value for a packet could be used to apply a packet transformation function (block or drop) to block the packet from continuing toward its destination.” Petition at 44.

Even taking this conclusory and unsupported assertion as true, Petitioner falls short of asserting—let alone demonstrating—that Sourcefire discloses that the alleged “packet-transformation function” is “specified by the operator,” as recited in the independent claims of the '552 Patent. Compounding this failure to “specify where each element of the claim is found in the prior art patents or printed publications relied upon” pursuant to 37 C.F.R. § 42.104(b)(4) is the fact that Petitioner did not identify which teachings in Sourcefire allegedly correspond to the claimed “operator.” See § V.B.1, *supra*.

For at least the foregoing reasons, Patent Owner respectfully requests that the Board decline to institute *inter partes* review of the '552 Patent.

4. Petitioner Has Not Demonstrated That a POSA Would Have Modified Sourcefire to Reach the Invention Claimed in the '552 Patent

As noted above, Petitioner has not demonstrated that Sourcefire discloses “applying, by the at least one processor, an operator specified by the packet-filtering rule to the at least a portion of the plurality of HTTPS packets, wherein the operator specifies one or more application-header-field-value criteria identifying one or more transport layer security (TLS)-version values for which packets should be blocked from continuing toward their respective destinations.” See § VI.B.1, *supra*.

(a) *Petitioner Describes No Motivation to Modify Sourcefire to Prevent Exfiltrations*

As a preliminary matter, and as Petitioner explicitly notes, the '552 Patent is also capable of preventing “a category of cyber-attacks known as exfiltrations”:

The '552 Patent is directed to systems and methods for filtering data packets leaving a network to ensure that the packets are not improperly transferring sensitive data from the network, a category of cyber-attacks known as exfiltrations. For example, exfiltrations are often facilitated by using popular network data transfer protocols, such as the Hypertext Transfer Protocol (HTTP) used by the Web. Such attacks may appear to an insufficiently careful observer as normal network behavior; older network trust models (e.g., those used by firewalls that do not interpret application-layer data like HTTP) may interpret exfiltrations as trusted operations.

Petition at 4–5 (citing CS-1001 1:15-37) (emphasis added). On the other hand, the Sourcefire reference is the user manual for an “Intrusion Prevention System,” or “IPS”:

Sourcefire IPS™ (Intrusion Prevention System) provides best-in-class intrusion detection and prevention by harnessing the power of the industry-standard Snort® rules-based detection engine. Backed by the acclaimed Sourcefire Vulnerability Research Team™ (VRT), Sourcefire delivers its customers unrivaled protection against known and unknown threats.

Sourcefire at 33 (highlighting added). The entirety of the 2123-page Sourcefire reference fails to mention the term exfiltration and explicitly touts itself as “provid[ing] best-in-class **intrusion** detection and prevention.” *Id.* (emphasis added).

Petitioner makes no attempt to explain why a POSA would have modified the Sourcefire Intrusion Prevention System to protect against exfiltrations. Thus, the Petition fails to demonstrate sufficient motivation to modify Sourcefire. *See Broadcom Corp. v. Emulex Corp.*, 732 F. 3d 1325, 1334 (Fed. Cir. 2013) (finding invention nonobvious when an allegedly invalidating prior art reference did not disclose all elements of the claims and addressed a different problem from the challenged patent)..

(b) *Petitioner Describes No Motivation to Modify Sourcefire to Practice the “operator” Element of the Challenged Claims*

After mischaracterizing the Sourcefire reference in an attempt to demonstrate that Sourcefire discloses applying the recited operator, Petitioner essentially admits that Sourcefire does not disclose the subject matter of the claims, stating instead that this feature would be obvious because a “POSA understood that Sourcefire taught the use of ssl_version as a keyword, and thus it **could be** used as an application-layer header field value in a packet-filtering rule.” Petition at 40–41 (emphasis added). This argument fails on its face and as a matter of law.

First, Petitioner asserts without evidence that because “Sourcefire taught the use of ssl_version as a keyword... it could be used as an application-layer header field value in a packet-filtering rule.” Petition at 40–41. This is untrue. As previously noted, the ssl_version keyword does not perform a match against an application-layer header field, but rather invokes the SSL preprocessor, which returns the version number that was extracted from previously-received handshake packets for the session. Sourcefire at 598. There is no evidence cited in the Petition to suggest that the ssl_version keyword could operate in any other way. In fact, the evidence demonstrates the opposite, that the ssl_version keyword can **only** operate by invoking the SSL preprocessor to glean version information from previously-received handshake packets associated with a session:

Note that the SSL preprocessor must be enabled to allow processing of rules using the `ssl_version` keyword. When the SSL preprocessor is disabled and you enable rules that use this keyword, you are prompted whether to enable the preprocessor when you save the policy. See Automatically Enabling Advanced Settings on page 490.

Sourcefire at 827 (highlighting added).

Second, Petitioner's argument fails as a matter of law because the question is not whether a POSA could have modified Sourcefire to practice the invention claimed in the '552 Patent—which Petitioner has not satisfied—but whether a POSA would have been motivated to make the asserted modification. *See Belden Inc. v. Berk-Tek LLC*, 805 F.3d 1064, 1073 (Fed. Cir. 2015) (“[O]bviousness concerns whether a skilled artisan not only could have made but would have been motivated to make the combinations or modifications of prior art to arrive at the claimed invention.”); *see also Orexo AB v. Actavis Elizabeth LLC*, 903 F.3d 1265, 1272-73 (Fed. Cir. 2018) (citing *In re Gordon*, 733 F.2d 900, 902 (Fed. Cir. 1984)) (“The mere fact that the prior art could be so modified would not have made the modification obvious unless the prior art suggested the desirability of the modification.”). Petitioner makes no pretense at explaining why a POSA would be motivated to make such a modification, and thus its argument fails as a matter of law.

Third, Petitioner's argument fails as a matter of law because it improperly relies on “knowledge gleaned only from applicant's disclosure.” *In re*

McLaughlin, 443 F.2d 1392, 1395 (CCPA 1971). That is, Petitioner cites only to the '552 Patent for the allegation that a "a POSA understood that the SSL/TLS version in a Record Protocol packet is not encrypted." Petition at 41. This is classic hindsight and should be rejected as a matter of law. "Defining the problem in terms of its solution reveals improper hindsight in the selection of the prior art relevant to obviousness." *Insite Vision Inc. v. Sandoz, Inc.*, 783 F. 3d 853, 859 (Fed. Cir. 2015) (citing *Monarch Knitting Mach. Corp. v. Sulzer Morat GmbH*, 139 F.3d 877, 881 (Fed.Cir.1998)).

For at least the foregoing reasons, Petitioner has not met its burden to demonstrate that a POSA would have modified Sourcefire to reach the claims of the '552 Patent.

(c) *Petitioner Describes No Motivation to Modify Sourcefire to Practice the Blocking Element of the Challenged Claims*

After mischaracterizing the Sourcefire reference in an attempt to demonstrate that Sourcefire discloses "the operator specifies one or more application-header-field-value criteria identifying one or more transport layer security (TLS)-version values for which packets should be blocked from continuing toward their respective destinations," Petitioner again essentially admits that Sourcefire does not disclose the subject matter of the claims. See Petition at 41–42 ("Sourcefire teaches that SSLv2 may have vulnerabilities associated with it

and thus a POSA would be motivated to identify and block packets including handshake Records that proposed using SSLv2 for a SSL session.”).

However, as noted above Sourcefire vaguely references identifying, but **not** blocking, traffic using SSL version 2:

For example, if you know there is a buffer overflow vulnerability in SSL version 2, you could use the `ssl_version` keyword with the `ssl_v2` argument to **identify** traffic using that version of SSL.

Sourcefire at 827. Thus, Petitioner's reliance on the Sourcefire reference for the proposition that “a POSA would be motivated to identify and block packets including *handshake* Records that proposed using SSLv2 for a SSL session” lacks any evidentiary basis. Petition at 41–42 (emphasis original).⁵

Petitioner cites to the Staniford Declaration for the proposition that “because security vulnerabilities associated with TLS 1.0 were widely known, a POSA would be motivated to identify and block packets including HTTP application-data Records that were encrypted using TLS 1.0 and packets including handshake Records that proposed using (or did use) TLS 1.0 for a TLS session.” Petition at 42 (citing Staniford Decl., ¶ 153). The citation to Dr. Staniford's Declaration is

⁵ Petitioner improperly cites to, but does not otherwise explain, 12 pages of the Sourcefire reference and a paragraph of the Stanford Declaration to support this conclusory and factually incorrect argument.

unavailing in this regard because merely repeats the argument from the Petition.

See Kinetic Technologies, Inc. v. Skyworks Solutions, Inc., IPR2014-00529, Paper 8 at 14–15 (PTAB Sept. 23, 2014) (“Merely repeating an argument from the Petition in the declaration of a proposed expert does not give that argument enhanced probative value.”).

Notably, Dr. Staniford's declaration at ¶ 153 differs slightly from the Petition in that it cites ***without explanation*** or pincites (in five cases) to ***six*** additional references, each of which is omitted from the Petition itself. *See* Staniford Decl., ¶ 153 (citing CS-1037; CS-1038; CS-1039 p. 4; CS-1013; CS-1014; CS-1015). And in the only reference that actually refers to something called “BEAST,” the reference explicitly counsels **not** to block TLS 1.0 traffic “but to rely on browser counter-measures in the meantime.” CS-1037. Accordingly Petitioner's argument fails ***at least*** because (1) Petitioner's argument to modify Sourcefire lacks any evidentiary basis from the Sourcefire reference, (2) the alleged evidentiary basis is cited in only in the Staniford Declaration and cannot be incorporated by reference into the Petition, and (3) the alleged evidentiary basis does not support the proposition for which it is cited.

For at least the foregoing reasons, a POSA would not have modified Sourcefire as suggested in the Petition to arrive at the claimed subject matter of the

'552 Patent. Accordingly, Patent Owner requests that the Board decline to institute *inter partes* review of the '552 Patent.

VI. CONCLUSION

Petitioner has not established a reasonable likelihood that it will prevail in establishing that claims 1-21 of the '552 Patent are invalid. Centripetal accordingly requests that the Board deny institution of *inter partes* review of the '552 Patent.

Respectfully submitted,

Dated: October 26, 2018

/James Hannah/

James Hannah (Reg. No. 56,369)
Michael Lee (Reg. No. 63,941)
Kramer Levin Naftalis & Frankel LLP
990 Marsh Road
Menlo Park, CA 94025
Tel: 650.752.1700 Fax: 212.715.8000

Jeffrey H. Price (Reg. No. 69,141)
Kramer Levin Naftalis & Frankel LLP
1177 Avenue of the Americas
New York, NY 10036
Tel: 212.715.7502 Fax: 212.715.8302

Bradley C. Wright (Reg. No. 38,061)
Banner & Witcoff, Ltd.
1100 13th Street NW, Suite 1200,
Washington, DC 20005.
Tel: 202.824.3160. Fax: 202.824.3000.
Email: bwright@bannerwitcoff.com

(Case No. IPR2018-01436) *Attorneys for Patent Owner*

Appendix A Challenged Claims

1. A method, comprising:
at a computing device comprising at least one processor, a memory,
and a communication interface:
receiving, via the communication interface, a plurality of
hypertext transfer protocol secure (HTTPS) packets;
responsive to a determination by the at least one processor that
at least a portion of the plurality of HTTPS packets have packet-header-field
values corresponding to a packet-filtering rule stored in the memory, applying, by
the at least one processor, an operator specified by the packet-filtering rule to the at
least a portion of the plurality of HTTPS packets, wherein the operator specifies
one or more application-header-field-value criteria identifying one or more
transport layer security (TLS)-version values for which packets should be blocked
from continuing toward their respective destinations; and
responsive to a determination by the at least one processor that
one or more packets, of the at least a portion of the plurality of HTTPS packets,
have one or more application-header-field values corresponding to one or more
TLS-version values of the one or more TLS-version values for which packets
should be blocked from continuing toward their respective destinations, applying,
by the at least one processor, at least one packet-transformation function specified
by the operator to the one or more packets to block each packet of the one or more
packets from continuing toward its respective destination.
2. The method of claim 1, wherein the one or more application-
header-field-value criteria identify one or more TLS-version values for which
packets should be allowed to continue toward their respective destinations, the

method comprising, responsive to a determination by the at least one processor that one or more other packets, of the at least a portion of the plurality of HTTPS packets, have one or more application-header-field values corresponding to one or more TLS-version values of the one or more TLS-version values for which packets should be allowed to continue toward their respective destinations, applying, by the at least one processor, a packet-transformation function specified by the operator to the one or more other packets to allow each packet of the one or more other packets to continue toward its respective destination.

3. The method of claim 1, wherein the one or more application-header-field-value criteria identify one or more hypertext transfer protocol (HTTP) methods for which packets should be allowed to continue toward their respective destinations, the method comprising, responsive to a determination by the at least one processor that one or more other packets, of the at least a portion of the plurality of HTTPS packets, have one or more application-header-field values corresponding to one or more HTTP methods of the one or more HTTP methods for which packets should be allowed to continue toward their respective destinations, applying, by the at least one processor, a packet-transformation function specified by the operator to the one or more other packets to allow each packet of the one or more other packets to continue toward its respective destination.

4. The method of claim 1, wherein the one or more application-header-field-value criteria identify one or more hypertext transfer protocol (HTTP) methods for which packets should be blocked from continuing toward their respective destinations, the method comprising, responsive to a determination by the at least one processor that one or more other packets, of the at least a portion of the plurality of HTTPS packets, have one or more application-header-field values

corresponding to one or more HTTP methods of the one or more HTTP methods for which packets should be blocked from continuing toward their respective destinations, applying, by the at least one processor, a packet-transformation function specified by the operator to the one or more other packets to block each packet of the one or more other packets from continuing toward its respective destination.

5. The method of claim 1, wherein the one or more application-header-field-value criteria identify one or more hypertext transfer protocol (HTTP) methods, and wherein the at least a portion of the plurality of HTTPS packets comprises a first portion of packets and a second portion of packets, each packet of the first portion comprising at least one GET method value, and each packet of the second portion comprising at least one of a PUT method value, a POST method value, or a CONNECT method value, the method comprising:

allowing each packet of the first portion of packets to continue toward its respective destination; and

blocking each packet of the second portion of packets from continuing toward its respective destination.

6. The method of claim 1, comprising, for each packet of the at least a portion of the plurality of HTTPS packets:

comparing a protocol of a data section of the packet to a protocol specified by the packet-filtering rule;

comparing a source Internet Protocol (IP) address of the packet to one or more source IP addresses specified by the packet-filtering rule;

comparing a source port of the packet to one or more source ports specified by the packet-filtering rule;

comparing a destination IP address of the packet to one or more

destination IP addresses specified by the packet-filtering rule; and

comparing a destination port of the packet to one or more destination ports specified by the packet-filtering rule.

7. The method of claim 1, comprising:

receiving, via the communication interface, a second plurality of HTTPS packets; and

responsive to a determination by the at least one processor that the second plurality of HTTPS packets does not have packet-header-field values corresponding to the packet-filtering rule, blocking, by the computing device and without applying the operator to the second plurality of HTTPS packets, each packet of the second plurality of HTTPS packets from continuing toward its respective destination.

8. An apparatus, comprising:

at least one processor; and

a memory storing instructions that when executed by the at least one processor cause the apparatus to:

receive a plurality of hypertext transfer protocol secure (HTTPS) packets;

responsive to a determination by the at least one processor that at least a portion of the plurality of HTTPS packets have packet-header-field values corresponding to a packet-filtering rule, apply an operator specified by the packet-filtering rule to the at least a portion of the plurality of HTTPS packets, wherein the operator specifies one or more application-header-field-value criteria identifying one or more transport layer security (TLS)-version values for which packets should be blocked from continuing toward their respective destinations; and

responsive to a determination by the at least one processor that one or more packets, of the at least a portion of the plurality of HTTPS packets, have one or more application-header-field values corresponding to one or more TLS-version values of the one or more TLS-version values for which packets should be blocked from continuing toward their respective destinations, apply at least one packet-transformation function specified by the operator to the one or more packets to block each packet of the one or more packets from continuing toward its respective destination.

9. The apparatus of claim 8, wherein the one or more application-header-field-value criteria identify one or more TLS-version values for which packets should be allowed to continue toward their respective destinations, and wherein the instructions, when executed by the at least one processor, cause the apparatus to, responsive to a determination by the at least one processor that one or more other packets, of the at least a portion of the plurality of HTTPS packets, have one or more application-header-field values corresponding to one or more TLS-version values of the one or more TLS-version values for which packets should be allowed to continue toward their respective destinations, apply a packet-transformation function specified by the operator to the one or more other packets to allow each packet of the one or more other packets to continue toward its respective destination.

10. The apparatus of claim 8, wherein the one or more application-header-field-value criteria identify one or more hypertext transfer protocol (HTTP) methods for which packets should be allowed to continue toward their respective destinations, and wherein the instructions, when executed by the at least one processor, cause the apparatus to, responsive to a determination by the at least one processor that one or more other packets, of the at least a portion of the plurality of

HTTPS packets, have one or more application-header-field values corresponding to one or more HTTP methods of the one or more HTTP methods for which packets should be allowed to continue toward their respective destinations, apply a packet-transformation function specified by the operator to the one or more other packets to allow each packet of the one or more other packets to continue toward its respective destination.

11. The apparatus of claim 8, wherein the one or more application-header-field-value criteria identify one or more hypertext transfer protocol (HTTP) methods for which packets should be blocked from continuing toward their respective destinations, and wherein the instructions, when executed by the at least one processor, cause the apparatus to, responsive to a determination by the at least one processor that one or more other packets, of the at least a portion of the plurality of HTTPS packets, have one or more application-header-field values corresponding to one or more HTTP methods of the one or more HTTP methods for which packets should be blocked from continuing toward their respective destinations, apply a packet-transformation function specified by the operator to the one or more other packets to block each packet of the one or more other packets from continuing toward its respective destination.

12. The apparatus of claim 8, wherein the one or more application-header-field-value criteria identify one or more hypertext transfer protocol (HTTP) methods, wherein the at least a portion of the plurality of HTTPS packets comprises a first portion of packets and a second portion of packets, each packet of the first portion comprising at least one GET method value, and each packet of the second portion comprising at least one of a PUT method value, a POST method value, or a CONNECT method value, and wherein the instructions, when executed by the at least one processor, cause the apparatus to:

allow each packet of the first portion to continue toward its respective destination; and

block each packet of the second portion from continuing toward its respective destination.

13. The apparatus of claim 8, wherein the instructions, when executed by the at least one processor, cause the apparatus to, for each packet of the at least a portion of the plurality of HTTPS packets:

compare a protocol of a data section of the packet to a protocol specified by the packet-filtering rule;

compare a source Internet Protocol (IP) address of the packet to one or more source IP addresses specified by the packet-filtering rule;

compare a source port of the packet to one or more source ports specified by the packet-filtering rule;

compare a destination IP address of the packet to one or more destination IP addresses specified by the packet-filtering rule; and

compare a destination port of the packet to one or more destination ports specified by the packet-filtering rule.

14. The apparatus of claim 8, wherein the instructions, when executed by the at least one processor, cause the apparatus to:

receive a second plurality of HTTPS packets; and

responsive to a determination by the at least one processor that the second plurality of HTTPS packets does not have packet-header-field values corresponding to the packet-filtering rule, block, without applying the operator to the second plurality of HTTPS packets, each packet of the second plurality of HTTPS packets from continuing toward its respective destination.

15. One or more non-transitory computer-readable media having instructions stored thereon that when executed by one or more computing devices cause the one or more computing devices to:

receive a plurality of hypertext transfer protocol secure (HTTPS) packets;

responsive to a determination by the one or more computing devices that at least a portion of the plurality of HTTPS packets have packet-header-field values corresponding to a packet-filtering rule, apply an operator specified by the packet-filtering rule to the at least a portion of the plurality of HTTPS packets, wherein the operator specifies one or more application-header-field-value criteria identifying one or more transport layer security (TLS)-version values for which packets should be blocked from continuing toward their respective destinations; and

responsive to a determination by the one or more computing devices that one or more packets, of the at least a portion of the plurality of HTTPS packets have one or more application-header-field values corresponding to one or more TLS-version values of the one or more TLS-version values for which packets should be blocked from continuing toward their respective destinations, apply at least one packet-transformation function specified by the operator to the one or more packets to block each packet of the one or more packets from continuing toward its respective destination.

16. The one or more non-transitory computer-readable media of claim 15, wherein the one or more application-header-field-value criteria identify one or more TLS-version values for which packets should be allowed to continue toward their respective destinations, and wherein the instructions, when executed by the one or more computing devices, cause the one or more computing devices to,

responsive to a determination by the one or more computing devices that one or more other packets, of the at least a portion of the plurality of HTTPS packets, have one or more application-header-field values corresponding to one or more TLS-version values of the one or more TLS-version values for which packets should be allowed to continue toward their respective destinations, apply a packet-transformation function specified by the operator to the one or more other packets to allow each packet of the one or more other packets to continue toward its respective destination.

17. The one or more non-transitory computer-readable media of claim 15, wherein the one or more application-header-field-value criteria identify one or more hypertext transfer protocol (HTTP) methods for which packets should be allowed to continue toward their respective destinations, and wherein the instructions, when executed by the one or more computing devices, cause the one or more computing devices to, responsive to a determination by the one or more computing devices that one or more other packets, of the at least a portion of the plurality of HTTPS packets, have one or more application-header-field values corresponding to one or more HTTP methods of the one or more HTTP methods for which packets should be allowed to continue toward their respective destinations, apply a packet-transformation function specified by the operator to the one or more other packets to allow each packet of the one or more other packets to continue toward its respective destination.

18. The one or more non-transitory computer-readable media of claim 15, wherein the one or more application-header-field-value criteria identify one or more hypertext transfer protocol (HTTP) methods for which packets should be blocked from continuing toward their respective destinations, and wherein the instructions, when executed by the one or more computing devices, cause the one

or more computing devices to, responsive to a determination by the one or more computing devices that one or more other packets, of the at least a portion of the plurality of HTTPS packets, have one or more application-header-field values corresponding to one or more HTTP methods of the one or more HTTP methods for which packets should be blocked from continuing toward their respective destinations, apply a packet-transformation function specified by the operator to the one or more other packets to block each packet of the one or more other packets from continuing toward its respective destination.

19. The one or more non-transitory computer-readable media of claim 15, wherein the one or more application-header-field-value criteria identify one or more hypertext transfer protocol (HTTP) methods, wherein the at least a portion of the plurality of HTTPS packets comprises a first portion of packets and a second portion of packets, each packet of the first portion comprising at least one GET method value, and each packet of the second portion comprising at least one of a PUT method value, a POST method value, or a CONNECT method value, and wherein the instructions, when executed by the one or more computing devices, cause the one or more computing devices to:

allow each packet of the first portion to continue toward its respective destination; and

block each packet of the second portion from continuing toward its respective destination.

20. The one or more non-transitory computer-readable media of claim 15, wherein the instructions, when executed by the one or more computing devices, cause the one or more computing devices to, for each packet of the at least a portion of the plurality of HTTPS packets:

compare a protocol of a data section of the packet to a protocol

specified by the packet-filtering rule;

compare a source Internet Protocol (IP) address of the packet to one or more source IP addresses specified by the packet-filtering rule;

compare a source port of the packet to one or more source ports specified by the packet-filtering rule;

compare a destination IP address of the packet to one or more destination IP addresses specified by the packet-filtering rule; and

compare a destination port of the packet to one or more destination ports specified by the packet-filtering rule.

21. The one or more non-transitory computer-readable media of claim 15, wherein the instructions, when executed by the one or more computing devices, cause the one or more computing devices to:

receive a second plurality of HTTPS packets; and

responsive to a determination by the one or more computing devices that the second plurality of HTTPS packets does not have packet-header-field values corresponding to the packet-filtering rule, block, without applying the operator to the second plurality of HTTPS packets, each packet of the second plurality of HTTPS packets from continuing toward its respective destination.

Patent Owner's Preliminary Response
IPR2019-01436 (U.S. Patent No. 9,124,552)

CERTIFICATE OF COMPLIANCE WITH 37 C.F.R. § 42.24

This paper complies with the type-volume limitation of 37 C.F.R. § 42.24.

The paper includes 6,507 words, excluding the parts of the paper exempted by § 42.24(a).

This paper also complies with the typeface requirements of 37 C.F.R. § 42.6(a)(ii) and the type style requirements of § 42.6(a)(iii)&(iv).

/James Hannah/
James Hannah (Reg. No. 56,369)
Kramer Levin Naftalis & Frankel LLP
990 Marsh Road,
Menlo Park, CA 94025
(650) 752-1700

Patent Owner's Preliminary Response
IPR2018-01436 (U.S. Patent No. 9,124,552)

CERTIFICATE OF SERVICE

The undersigned certifies, in accordance with 37 C.F.R. § 42.6(e), that service was made on the Petitioner as detailed below.

Date of service October 26, 2018

Manner of service Electronic Mail
(PDMcPherson@duanemorris.com;
CJTyson@duanemorris.com;
JAPowers@duanemorris.com)

Documents served PATENT OWNER'S PRELIMINARY RESPONSE

Persons Served (PDMcPherson@duanemorris.com;
CJTyson@duanemorris.com;
JAPowers@duanemorris.com)

/James Hannah/
James Hannah
Registration No. 56,369
Counsel for Patent Owner