

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

CISCO SYSTEMS INC.,

Petitioner

IPR2025-00836

U.S. Patent No. 11,816,249

**DECLARATION OF NADER F. MIR, PH.D.,
UNDER 37 C.F.R. § 1.68 IN SUPPORT OF PETITION
FOR *INTER PARTES* REVIEW**

TABLE OF CONTENTS

I.	INTRODUCTION	4
II.	QUALIFICATIONS AND PROFESSIONAL EXPERIENCE	7
III.	LEVEL OF ORDINARY SKILL IN THE ART	11
IV.	RELEVANT LEGAL STANDARDS	12
V.	BACKGROUND	13
A.	TCP/IP and OSI Models.....	13
B.	Packet Encapsulation.....	18
C.	Encryption	21
D.	Intrusion Detection System (IDS).....	22
VI.	OVERVIEW OF THE '249 PATENT	24
VII.	CLAIM CONSTRUCTION	26
VIII.	IDENTIFICATION OF HOW THE CLAIMS ARE UNPATENTABLE....	27
A.	Summary of Burns.....	27
B.	Summary of Yang	30
C.	Reasons to Combine Burns and Yang.....	32
D.	Summary of Guruswamy	33
E.	Reasons to Combine Burns, Yang, and Guruswamy	35
F.	Detailed Analysis of Claims.....	36
1.	Claim 1	36
2.	Claim 2	56
3.	Claim 3	62
4.	Claim 4.....	63
5.	Claim 5.....	75
6.	Claim 6.....	76
7.	Claim 7.....	77
8.	Claim 8.....	81

9.	Claim 9.....	81
10.	Claim 10.....	86
11.	Claim 11.....	86
12.	Claim 12.....	89
13.	Claim 13.....	91
14.	Claim 14.....	91
15.	Claim 15.....	92
16.	Claim 16.....	92
17.	Claim 17.....	93
18.	Claim 18.....	93
19.	Claim 19.....	94
20.	Claim 20.....	101
21.	Claim 21.....	102
22.	Claim 22.....	102
23.	Claim 23.....	103
24.	Claim 24.....	104
25.	Claim 25.....	105
26.	Claim 26.....	105
27.	Claim 27.....	105
28.	Claim 28.....	106
29.	Claim 29.....	106
30.	Claim 30.....	106
IX.	CONCLUSION.....	110

I, Nader F. Mir, Ph.D., do hereby declare as follows:

I. INTRODUCTION

1. I am making this declaration at the request of Cisco Systems, Inc. in the matter of the *Inter Partes* Review of U.S. Patent No. 11,816,249 (“the ’249 patent”) to Koren and Wasserman.

2. I am being compensated for my work in this matter at my standard hourly rate. I am also being reimbursed for reasonable and customary expenses associated with my work and testimony in this proceeding. My compensation is not contingent on the outcome of this matter or the specifics of my testimony.

3. I have been asked to provide my opinions regarding whether the subject matter of claims 1-30 (“the Challenged Claims”) of the ’249 patent would have been obvious to a person having ordinary skill in the art (“POSITA”) at the time of the alleged invention, in light of the prior art. It is my opinion that the Challenged Claims would have been obvious to a POSITA.

4. In the preparation of this declaration, I have studied:

- a. The ’249 patent, Ex.1001;
- b. The prosecution history of the ’249 patent (“’249 File History”), Ex.1002;
- c. U.S. Patent No. 8,341,724 to Burns and Sukhanov (“Burns”), Ex.1005;
- d. U.S. Patent No. 8,291,495 to Burns and Yang (“Yang”), Ex.1006;
- e. U.S. Patent No. 8,209,756 to Guruswamy et al. (“Guruswamy”), Ex.1007

5. In forming the opinions expressed below, I have considered: the documents listed above; the relevant legal standards, including the standard for obviousness; and my own knowledge and experience based upon my work in the field of network communications and security as described below, as well as portions of the following additional materials:

- a. Transmission Control Protocol, RFC 793 (Sept. 1981), Ex.1011;
- b. Andrew Tanenbaum, “Computer Networks,” 3rd ed. (1996) (“Tanenbaum”), Ex.1012;
- c. U.S. Patent Appl. No. 2006/0215695 to Olderdissen (“Olderdissen”), Ex.1013;
- d. Perlman, Radia, “Interconnections: Bridges, Routers, Switches, and Internetworking Protocols,” 2d ed. (2000), Ex.1014;
- e. Kozierok, Charles M., “The TCP/IP Guide: A Comprehensive, Illustrated Internet Protocols Reference,” (2005), Ex.1016;
- f. Nayak, Umesha; Hodeghatta Rao, Umesh, “The InfoSec Handbook: An Introduction to Information Security,” (2014), Ex.1017;
- g. U.S. Patent No. 7,406,522 to Riddle, Ex.1018;
- h. U.S. Patent Publication No. 2011/0106710 to Reed et al., Ex.1019;
- i. U.S. Patent Publication No. 2011/0022835 to Schibuk et al., Ex.1020;

- j. U.S. Patent No. 6,944,762 to Garrison, Ex.1021;
- k. U.S. Patent Publication No. 2005/0281192 to Nadeau et al., Ex.1022;
- l. U.S. Patent Publication No. 2011/0110328 to Pradeep et al., Ex.1023;
- m. U.S. Patent Publication No. 2013/0329578 to Groves et al., Ex.1024;
- n. U.S. Patent Publication No. 2002/0046264 to Dillon et al., Ex.1025;
- o. U.S. Patent Publication No. 2002/0059435 to Border et al., Ex.1026;
- p. U.S. Patent No. 6,826,620 to Mawhinney et al., Ex.1027;
- q. U.S. Patent Publication No. 2017/0230065 to Saraswathyama et al., Ex.1028;
- r. U.S. Patent Publication No. 2014/0351106 to Furr et al., Ex.1029;
- s. U.S. Patent Publication No. 2012/0230208 to Pyatkovskiy, Ex.1030;
- t. U.S. Patent Publication No. 2006/0248582 to Panjwani et al., Ex.1031;
- u. U.S. Patent Publication No. 2004/0013112 to Goldberg et al., Ex.1032;
- v. Transmission Control Protocol, RFC 1700 (1994), Ex.1033;
- w. Transmission Control Protocol, RFC 791 (1981), Ex.1034;
- x. Transmission Control Protocol, RFC 790 (1981), Ex.1035;
- y. Proctor, Paul E., “The Practical Intrusion Detection Handbook,” (2001),
Ex.1036;
- z. U.S. Patent Publication No. 2008/0159146 to Claudatos et al., Ex.1037;
- aa. U.S. Patent Publication No. 2007/0088845 to Memon et al., Ex.1038;
- bb. U.S. Patent Publication No. 2005/0047449 to Adolph et al., Ex.1039;

- cc. Transmission Control Protocol, RFC 5246 (2008), Ex.1040;
- dd. U.S. Patent Publication No. 2003/0038791 to Chou, Ex.1041;
- ee. U.S. Patent No. 5,289,589 to Bingham et al., Ex.1042;
- ff. U.S. Patent Publication No. 2006/0040650 to Schepers et al., Ex.1043;
- gg. U.S. Patent Publication No. 2003/0014624 to Maturana et al., Ex.1044;
- hh. U.S. Patent Publication No. 2006/0209858 to Blum, Ex.1045;
- ii. Transmission Control Protocol, RFC 4253 (2006), Ex.1046

6. Unless otherwise noted, all emphasis in any quoted material has been added. Claim terms are italicized.

II. QUALIFICATIONS AND PROFESSIONAL EXPERIENCE

7. My complete qualifications and professional experience are described in my *Curriculum Vitae*, a copy of which can be found in Exhibit 1004. The following is a brief summary of my relevant qualifications and professional experience.

8. I am a tenured professor in the Electrical Engineering Department of San Jose State University, California, where I also held the position of Associate Chair from 2006 through 2008. I have been a faculty member at San Jose State University since 2001.

9. In 1995, I received a Ph.D. in electrical engineering from Washington University, St. Louis. I received an M.S. degree in electrical engineering in 1990,

also from Washington University. In 1985, I received a B.S. in electrical engineering from Polytechnic University.

10. I have over 30 years of experience with computer networks and communication systems, including both industry and academic experience. Before my current position at San Jose State University, I was an Assistant Professor in the Department of Electrical and Computer Engineering at the University of Kentucky. I joined San Jose State University in 2001 as an Associate Professor. Since 2005, I have been a Full Professor in the Electrical Engineering Department at San Jose State University, where I perform research and teach in the areas of computer networks, networking devices, switches, routers, virtualization, virtual switches, cloud computing, software-defined networking in clouds, and cloud data centers.

11. I held several technical editorial positions for various journals, including IEEE Communication Magazine. As a Technical Editor of IEEE Communications Standards Magazine for Cloud and Edge Computing, I was responsible for accepting or rejecting scientific articles submitted to the journal in the areas of computer networking focusing on cloud systems. From 2005 through 2020, I held the technical editorial position for IEEE Communications Magazine. I am a senior member of the IEEE and have served as a member of the technical program committees and the steering committees for a number of major IEEE communications and networking conferences.

12. For more than 30 years, I have conducted research in the areas of computer and communication networks; networking devices; protocols including (but not limited to) packet-switched networks, cloud and edge computing, software-defined networking in cloud systems, integrated voice, video, data center networks, TCP/IP, client/server operations, content delivery networks, multimedia and streaming, client/server, networking traffic flow analysis, wireless/mobile networks, networking devices, network security, and network virtualization, among others.

13. I have received a number of university awards as well as national and international awards. In particular, I have received a number of research grants from private, state, and governmental funding agencies for conducting research in the fields of computers and communication networks. I am also the recipient of several university teaching recognition awards from San Jose State University; several research excellence awards; and the school's published book award for the year.

14. Before my positions in academia, I worked in various positions in the telecommunication and computer network industries. From 1985 to 1988, I was a Telecomm R&D Engineer at the Telecommunication Research & Development Center (TRDC) in Surrey, England. In particular, I headed engineering projects in the area of telecommunication engineering. I also participated in the design of a high-speed switch for digital Private Branch Exchange (PBX) and SS7 networks. In 1991, I worked as an Electrical Engineering Consultant for Bussmann Company,

where I instructed engineers on electrical engineering topics and prepared them for the National Principles and Practice of Engineering (PE) exam. From 1989 to 1995, I worked as a Computer/Communications Chip-Design Engineer at the Computer & Communications Research Center in St. Louis, Missouri. In that position, I designed a VLSI gigabit switch control plane chip for a high-speed computer network using CMOS technology. From 1995 to 1996, I was a Research Scientist Engineer at Advanced Telecommunications Institute, Stevens Institute of Technology in Hoboken, New Jersey. There I headed analyzing communications and high-speed computer networking projects.

15. Since obtaining my Ph.D., I have served in various consulting positions in the area of computer networking technology and communication engineering, including for CITS Inc. in San Jose, California (2012-2013), Morrison & Foerster LLP in San Francisco, California (2015-2016), and Butler and Snow in Jackson, Mississippi (2017-2018).

16. I have authored or co-authored over 100 articles in peer-reviewed journals, conference proceedings, texts, and monographs in the aforementioned areas of computer networking, communications, and related fields. I have published two textbooks, one of which is entitled “Computer Communication Networks.” This textbook has been adopted by numerous universities worldwide.

17. I am a named inventor on one U.S. patent, which relates to a packet-switching network and method of routing packets.

18. I have served as an expert witness and technical consultant in litigation and *inter partes* review matters concerning computer networks, wireless networks at both the protocol and hardware levels, and telecommunications.

III. LEVEL OF ORDINARY SKILL IN THE ART

19. I understand there are multiple factors relevant to determining the level of ordinary skill in the pertinent art, including (1) the levels of education and experience of persons working in the field at the time of the invention; (2) the sophistication of the technology; (3) the types of problems encountered in the field; and (4) the prior art solutions to those problems.

20. A POSITA in the field of the '249 patent, as of its priority date of April 9, 2017, would have been someone knowledgeable and familiar with the data communications art that is pertinent to the '249 patent, including packet-based computer networking. *See* '249 patent, 6:24-27. The '249 patent refers to a variety of internet networking terms and protocols acronyms without explanation, indicating that a POSITA would have been familiar with internet networking topics such as Transmission Control Protocol / Internet Protocol (TCP/IP). *See* '249 patent, 9:3-13, 10:4. Such a POSITA would have a bachelor's degree in computer science, computer engineering, electrical engineering, or equivalent training, and approximately three

years of experience relating to packet-based network communications. Additional work experience can substitute for specific educational background, and vice versa.

21. For purposes of this Declaration, in general, and unless otherwise noted, my statements and opinions, such as those regarding my own experience and what a POSITA would have understood or known generally (and specifically related to the references I consulted herein), reflect the knowledge that existed in the relevant field as of the priority date of the '249 patent.

IV. RELEVANT LEGAL STANDARDS

22. I am not an attorney. In preparing and expressing my opinions and considering the subject matter of the '249 patent, I am relying on certain basic legal principles that Cisco's counsel has explained to me.

23. I understand that prior art to the '249 patent includes patents and printed publications in the relevant art that predate the priority date of the '249 patent. For purposes of this Declaration, I am applying April 9, 2017, as the priority date of the '249 patent.

24. I have been informed by Cisco's counsel that a claimed invention is unpatentable under 35 U.S.C. §103 if the differences between the claimed invention and the prior art are such that the subject matter as a whole would have been obvious to a POSITA at the time of the invention. I have also been informed by Cisco's counsel that the obviousness analysis considers factual inquiries, including the level

of ordinary skill in the art, the scope and content of the prior art, and the differences between the prior art and the claimed subject matter.

25. I have been further informed by Cisco’s counsel that there are several recognized rationales for combining references or modifying a reference to show obviousness. These rationales include: (a) combining prior art elements according to known methods to yield predictable results; (b) simple substitution of one known element for another to obtain predictable results; (c) use of a known technique to improve a similar device (method, or product) in the same way; (d) applying a known technique to a known device (method, or product) ready for improvement to yield predictable results; (e) choosing from a finite number of identified, predictable solutions, with a reasonable expectation of success; and (f) some teaching, suggestion, or motivation in the prior art that would have led a POSITA to modify the prior art or to combine prior art teachings to arrive at the claimed invention.

V. BACKGROUND

A. TCP/IP and OSI Models

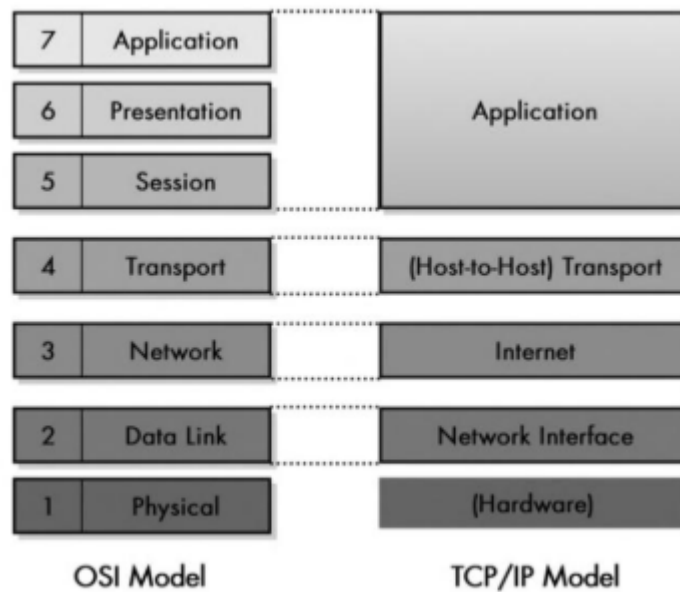
26. Computer networking is a complex task, one that is made easier by partitioning it “into smaller subtasks, traditionally by dividing the problem into layers.” Perlman, 1. The idea behind layering is that “each layer is responsible for providing a service to the layer above by using the services of the layer below.” Perlman, 1.

27. One of the most common sets of networking protocol layers is the Open Systems Interconnection (“OSI”) model. It was the original standard description for how packets should be transmitted between any two points. Tanenbaum, 28, 36. The OSI model has seven layers: physical, data link, network, transport, session, presentation, and application. Kozierok, 129.

28. The physical layer defines electrical aspects of activating and maintaining physical links in networks. The physical layer represents cables, connectors, transceivers, and hardware. Kozierok, 102. The data link layer is where many wired and wireless LAN technologies function and specifies how packets access links and are attached to different frames when entering a new network environment. Kozierok, 103-04; Olderdisen, [0012]. The network layer specifies networking aspects, and handles the way that packets are addressed and forwarded. Olderdisen, [0012]. The transport layer handles the details of data transmission such as segmentation, packaging, reassembly, and flow control. Kozierok, 108. The session layer sets up and coordinates sessions between different machines and the applications at each end. Tanenbaum, 50. Another basic task of this layer is “dialog control.” Tanenbaum, 50. The presentation layer converts incoming and outgoing data from one presentation format to another for various reasons such as data compression, data encryption, and data translation. Tanenbaum, 51. This layer of the protocol facilitates the interoperability between the different encoding methods.

Tanenbaum, 51. The application layer is the layer used to implement the functions performed by user applications and other high-level functions. Kozierok, 111-12.

29. The basic structure of communication networks is represented by the Transmission Control Protocol/Internet Protocol (TCP/IP) model. Tanenbaum, 28. This model is structured in five layers: hardware, network interface, internet, transport, and application. Kozierok, 129. The session, presentation, and application layers of the OSI model are condensed into the application layer of the TCP/IP model. Kozierok, 129. Other layers of the TCP/IP model track onto the OSI model as shown in Figure 8-2, Kozierok. The hardware layer of the TCP/IP model is the underlying network hardware. Kozierok, 128. The data link layer of the TCP/IP model is equivalent to the link layer and is used to interface with the network hardware. Kozierok, 128-29.



Kozierok, Fig. 8-2

30. TCP/IP has two primary protocols at the transport layer: TCP, which requires a connection between two devices, and UDP, which is connectionless. Kozierok, 17. Numerous protocols exist at the application layer including Hypertext Transfer Protocol (HTTP), and Simple Mail Transfer Protocol (SMTP) for users, as well as Simple Network Management Protocol (SNMP), and Domain Name Service (DNS) for administrators. Kozierok, 130.

31. Each layer (irrespective of model) at a source node communicates with its peer layer at a destination node using a protocol. Perlman, 1. It is common for a single node to support several protocols and for a single layer to support several protocols as well. Perlman, 3.

32. In the connection-based TCP protocol, a connection is established between two devices that wish to communicate, vis-a-vis a communication session. Kozierok, 752. After the communication session has concluded, the connection between the two devices may also be terminated, e.g., closed. Kozierok, 761. In a standard scenario, the device that wishes to close the connection sends a packet to the other device that contains the FIN TCP header flag, and the other device responds by sending an ACK, followed by its own FIN message. Kozierok, 763, Fig. 47-5. *See also* Nayak, 752, 760-761, 763, Fig. 47-5; U.S. Patent Publication No. 2006/0248582, [0095] (“[A] TCP endpoint that desires to initiate session closing operations sends a FIN command to the other endpoint. A FIN is represented by a TCP header flag.”); U.S. Patent Publication No. 2004/0013112, [0109] (“A TCP session is closed by both sides sending packets comprising the FIN flag indicating that the sender has no more data to send.”); RFC 793, 12 (“The clearing of a connection also involves the exchange of segments, in this case carrying the FIN control flag.”), 23 (Fig. 6 showing “CLOSE” actions accomplished by “s[e]nd FIN”), 38 (connection closing is initiated “by sending a FIN control signal”), 75 (if a packet received where “the FIN bit is set, signal the user ‘connection closing’”).

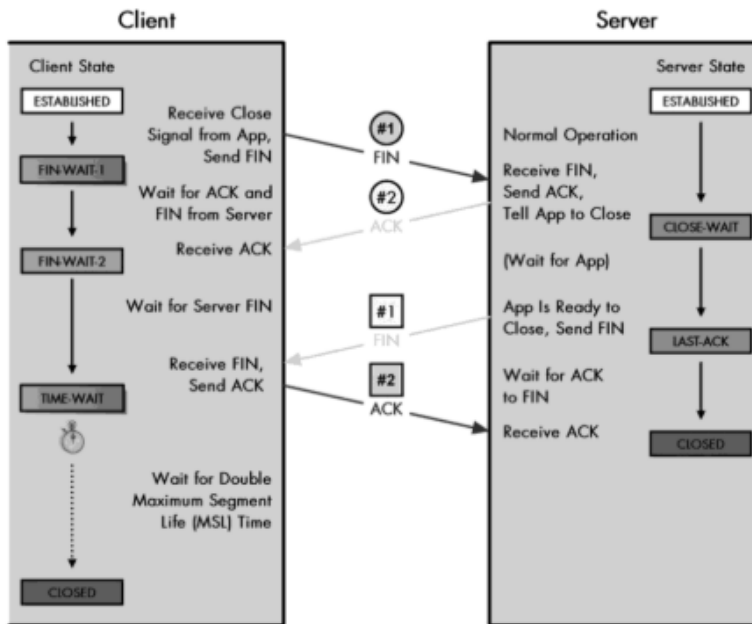


Fig. 47-5, Kozierok, 765.

33. In a situation where one device wishes to end the connection, but the other device is unresponsive, the device attempting to close the connection may send the other device a packet that contains the RST flag in the TCP header to force the connection to close. Kozierok, 760.

B. Packet Encapsulation

34. In communications networks, data is formatted differently according to the protocol utilized by the network, but most protocols have a relatively uniform structure. A given “message” that is sent across a network on links will generally consist of a header, the data, and the footer. Kozierok, 19, Fig. 1-3.



Figure 1-3, Kozierok.

35. For example, a layer protocol may receive a packet and add a header to the information it received in a process called encapsulation. Perlman, 4. The header is like an envelope for a letter, with information such as a source and destination addresses, which help deliver the letter to its destination. Perlman, 4. As the packet is passed down layer by layer, protocols operating within those lower layers may add their own headers to the packet. Tanenbaum, 19 (“A message, *M*, is produced by an application process running in layer 5 and given to layer 4 for transmission. Layer 4 puts a header in front of the message to identify the message and passes the result to layer 3. The header includes control information, such as sequence numbers, to allow layer 4 on the destination machine to deliver messages in the right order if the lower layers do not maintain sequence.”). Information contained in the packet header, such as a TCP port number or IP address is considered packet metadata. U.S. Patent Publication No. 2017/0230065, [0030] (“traffic metadata, such as MAC address, IP address, TCP port, UDP port, etc.”); U.S. Patent Publication No. 2014/0351106, [0033], (“a port

number (e.g., a Transmission Control Protocol (TCP) port number) may also be included in the metadata record”); U.S. Patent Publication No. 2012/0230208, [0057] (“metadata extracted from each outgoing TCP packet may include source and destination IP addresses, source and destination TCP ports”).

36. The actual data to be transmitted is called the “payload.” A packet’s “footer” is information that is placed after the payload and can be used in a similar fashion to the header in certain protocols. Kozierok, 19.

37. TCP is responsible for ensuring both that message data is divided into packets by the sending party, and that it is reassembled by the receiving party. U.S. Patent Publication No. 2003/0014624, [0003]. TCP/IP is widely used for transmitting information as packets. U.S. Patent Publication No. 2006/024858, [0015]; U.S. Patent No. 7,406,522, 1:59-60; Perlman, 4; Tanenbaum, 19; Kozierok, 19. A router or network device would contain a forwarding plane that would receive information such as source and destination IP addresses and protocol type of incoming packets, and forward packets with minimal delay. U.S. Patent Publication No. 2005/0281192, [0038] (“excessive delay” is a “forwarding plane failure”); U.S. Patent Publication No. 2011/0110328, [0030] (“achieving lower latency” is an advantage of a forwarding plane); U.S. Patent Publication No. 2013/0329578, [0029] (describing forwarding plane architectures “to reduce latency and other delays”).

C. Encryption

38. If information is transmitted in plain text, confidential or sensitive information could be read by someone other than the intended recipient, which might include an entity with a nefarious intent. Nayak, 163. Encryption is one method of disguising data to make it less likely to be read by a third-party using cryptography. Nayak, 163. At a relatively high level, encryption of data is performed using a “key” that is used to scramble the data, thereby rendering the data unreadable to those who do not have the key. Nayak, 164. The corresponding key that can decrypt encrypted data might be the same or different than the key used to encrypt data in the first place, depending on the specific encryption mechanism being employed. Nayak, 165. Typically, only the endpoints to the communication would typically have the decryption key. For example, the decryption key may be a “public key” which is known to both the sender and receiver, or a “private key” which is known to only one party. Nayak, 167-69. Keys may also be used for authentication—a packet could be encrypted with a private key unique to the sender, so the packet’s origin can be verified. Nayak, 171. *See also* Nayak, 243-247; U.S. Patent Publication No. 2011/0106710, [0067] (“As should be understood, because the PIN is encrypted by the terminal using a key that is not know[n] by the merchant or acquirer, the merchant or acquirer is unable to decrypt the PIN.”); U.S. Patent Publication No. 2011/0022835, [0056] (“if an unauthorized client device 110 receives this message,

it is unable to decrypt message 178 because it does not have possession of decryption key 112.”); U.S. Patent No. 6,944,476, 1:25-36.

39. Encryption can be performed at different layers in the protocol stack. Kozierok, 111. For example, the Secure Sockets Layer (SSL) protocol is usually associated with (and implemented within) the presentation layer. Kozierok, 111. Internet Protocol Security (IPsec) is a set of protocols that may similarly be implemented at lower layers in the protocol stack. Kozierok, 111. Encapsulating Security Payload (ESP) is another example of an encryption IPsec protocol that can be used to encrypt the payload of the packet, while the header and footer of the packet are left unencrypted. Kozierok, 466-71. Therefore, even without a key to decrypt the packet’s payload, a receiving or routing device could still authenticate a packet based on the header and footer information. Kozierok, 470.

D. Intrusion Detection System (IDS)

40. An IDS is used to detect and block malicious activity. Nayak, 225. Generally, an IDS will inspect both the packet header and the payload. Nayak, 225.

41. There are various detection mechanisms that an IDS may utilize, including signature-based detection, anomaly-based detection, or stateful protocol analysis. Nayak, 229. In most modern systems, an IDS is paired with an Intrusion Prevention System (IPS) in a combined system sometimes called an IDPS that identifies the attack (using the IDS functionality) and takes action to drop the packet,

deny entry to the packet, or block the connection (using the IPS functionality). Nayak, 228. An IDPS analyzes network behavior and prevents attacks in real-time with active monitoring. Nayak, 229.

42. In signature-based detection, the IDS compares the traffic with known signatures for possible attacks, to determine if the incoming traffic matches a known attack signature. Nayak, 230. In anomaly-based detection, the IDS protects a system from unknown threats by establishing a “baseline” of normal traffic through studying the traffic over a given time period. Once the baseline is established, the IDS can compare the baseline traffic with detected traffic and identify abnormalities or deviations from the baseline as a possible threat. Nayak, 230. These anomalies may be anomalies in the protocol format and behavior, such as unusual TCP segmentation, incorrect IP fragmentation and reassembly, incorrect source and destination port numbers, or flags in the TCP or IP header. Nayak, 232. Stateful protocol analysis involves creating predetermined profiles for permissible network activity that corresponds with protocols from standard bodies. Nayak, 233. An IDPS that performs stateful protocol analysis can track protocol states at both the network and application layers to track if a protocol is within the standards for acceptable protocol behavior. Nayak, 233.

43. When a packet’s payload is encrypted, an IDS cannot base its analysis and determination on the payload, making it more difficult to effectively identify

and block intrusion attempts. Nayak, 242. However, the IDS can still analyze flow information and the packet header. Nayak, 232-37.

44. One way to mitigate potential attacks based on the information available in a packet's header is by throttling a communication session by manipulating the TCP congestion window size. U.S. Patent Publication No. 2002/0046264, [0079] ("by regulating the advertized window size of each system user, each user's bandwidth can be controlled"); U.S. Patent Publication No. 2002/0059435, [0123] ("flow control can be applied... by shrinking the TCP windows being advertised"); U.S. Patent No. 6,826,620, 11:30-45 (an exemplary rate control technique where a "FRAU [frame relay access unit] controls... the TCP window size within the acknowledge packets.").

VI. OVERVIEW OF THE '249 PATENT

45. The '249 patent relates to management of private data during communication between a remote server and a user's device. '249 patent, Abstract. The '249 patent describes that the user's device receives a request from the remote server for retrieval of at least one data packet from the user's device. '249 patent, 2:10-12. The content, or "private data," within these data packets may be encrypted. '249 patent, 12:45. The user's device is "configured to provide a response corresponding to the received request." '249 patent, 2:13-15.

46. Characteristics of a data packet may be used to identify the packet's "data type" and the content of the data packet. '249 patent, 9:13-17. The characteristics of a data packet include "header, footer, version number, request by hostname/IP, sent to hostname/IP, server hostname/IP, IP address, file extension, standard title, non-standard title, encryption yes/no, encryption method, encoding yes/no, encoding method (e.g., 'UTF8'), whole file, chunked file, keywords, API version, SDK version, Social plug-in version, driver, communication protocol, action verbs, HTTP/HTTPS, URL Part Names, and/or any other specific character T within the at least one data packet." '249 patent, 9:5-13.

47. Similarly, at least one "dynamic feature" of a data packet may be used to identify the packet's "data pattern." '249 patent, 9:65-10:9. Dynamic features of a data packet include "frequency (e.g., appearance(s) per second), size (e.g., in kilobytes), speed (in kilobytes per second), count (e.g., numbers), ratio (e.g., compared to other datatype flow), repeat (e.g., cyclic order), order, server reference (e.g., website reference such as "example.com/feature") and/or any combination thereof and/or other pattern behavior 'm' within the at least one data packet." '249 patent, 9:67-10:9.

48. The user's device may have a "privacy preference," which contains a list of allowed data packet types or allowed data packet patterns for sharing during communication. '249 patent, 2:18-19, 32-33.

49. Whether a data packet is shared depends on whether the data packet matches the data types or data patterns specified in the privacy preference. '249 patent, 11:15-19. If the privacy preference dictates that a certain “data packet is forbidden for sharing” due to the data type and/or data pattern being incompatible with the privacy preferences, the content of the data packet can be modified. '249 patent, 10:52-56. Alternatively, if the privacy preferences do not recognize the packet type, the packet can be blocked. '249 patent, 13:12-16.

50. As I will explain below, these concepts were well known as of the priority date of the '249 patent.

VII. CLAIM CONSTRUCTION

51. It is my understanding that to properly evaluate the '249 patent, the terms of the claims must first be interpreted. I have been informed by Cisco's counsel that for the purposes of this *inter partes* review, the claims are to be construed under the so-called *Phillips* standard, under which claim terms are given their ordinary and customary meaning as would have been understood by a POSITA in light of the specification and prosecution history, unless the inventor has set forth a special meaning for a term. I have also been informed that claim terms only need to be construed to the extent necessary to resolve the obviousness inquiry.

52. I have reviewed the entirety of the '249 patent, as well as its prosecution history. It is my opinion that for purposes of applying the asserted prior art to

evaluate the patentability of the Challenged Claims, no terms require express construction.

VIII. IDENTIFICATION OF HOW THE CLAIMS ARE UNPATENTABLE

53. The discussion in this Declaration provides a detailed analysis of how the asserted prior art references teach each limitation of the Challenged Claims.

54. As part of my analysis, I have considered, and discuss in detail, the scope and content of the prior art and any differences between the alleged invention and the prior art.

55. It is my opinion that the alleged invention recited in the Challenged Claims would have been obvious in view of the teachings of the asserted prior art and the knowledge of a POSITA.

A. Summary of Burns

56. Burns is directed to identifying whether an encrypted packet is associated with a network attack, and either responding to the attack by, for example, blocking the packet, or forwarding the packet. Burns, Abstract. Burns's IDS identifies if an application is associated with a packet, determines if the packet is encrypted, and determines if a packet is associated with a network attack. Burns, Abstract, 2:51-3:33. If a packet is not associated with a network attack, IDS may forward the packet. Burns, Abstract. If a packet is associated with a network attack,

the IDS takes action and can either drop the packet, terminate the session, or block future sessions from the computing device. Burns, 5:61-63.

57. To determine if a packet is encrypted, Burns's IDS can employ multiple techniques via an encryption detection module. Burns, 9:8. The encryption detection module may extract the application-layer header and the application-layer payload from the packet. Burns, 12:10-16. The encryption detection module analyzes the packet header and the payload to determine if the randomness within the data indicates that encryption is present.¹ Burns, 12:20-25. Burns describes multiple methods for determining the randomness of a packet via the Wald-Wolfowitz runs test algorithm, or through additional tests and filters that can be added, removed, or edited. Burns, 9:54-59.

58. Burns describes that potentially malicious applications may encrypt all or portions of an application-layer header and/or application-layer data (payload). Burns calls applications where both the header and payload are encrypted "fully encrypted." Burns, 5:1-4. When fully encrypted packets are detected, Burns describes that the packets can be dropped, the traffic may be throttled, details about

¹Randomness strengthens cryptographic encryption—if an encryption algorithm has high randomness, it becomes more difficult for a third-party to "break" the encryption and decipher the encrypted data. *See* Nayak, 172-180.

the communication session can be logged, or an alert message can be sent. Burns, 6:49-56.

59. If a packet is fully encrypted, the IDS can also determine if the packet includes key exchange information. Burns, 13:40-42. If key exchange information is provided, the IDS can determine that the packet, although it may be fully encrypted, is not associated with an invalid or unwanted application. Burns, 13:43-46.

60. In contrast, legitimate applications following well-known protocols generally do not encrypt the application-layer header, the TCP header, or the IP header. Burns, 5:7-13. If a packet is not fully encrypted, Burns describes forwarding these packets or sending these packets to an attack detection module. Burns, 11:25-29.

61. Burns and the '249 patent are both directed towards dynamically managing network communications. '249 patent, Claim 1, 1:19-21, 17:34-3; Burns, 2:47-49, 5:53-6:8, 6:46-56. Both Burns and the '249 patent describe a method by which a user or administrator may selectively block (or modify) packets that do not correspond with their settings. *See* '249 patent, 3:10-12 (“wherein the privacy preference includes a list of allowed data patterns for sharing during communication”), 3:27-29 (“wherein the privacy preference includes a list of allowed data packet types for sharing during communication”), 12:13-19 (“may

perform at least one of sharing, blocking and/or modifying... according to the corresponding privacy preference”); Burns, 5:30-34 (“a system administrator may configure IDS 10 to explicitly allow all identifiable applications, allow all applications except for a specified list of identifiable applications, or prevent all communications...”), 6:50-51 (“the user interface may permit the administrator to drop all fully encrypted packets...drop the packets only if a key exchange has not been previously identified for the communication session...”). Burns is also directed to addressing the same problem of controlling the transmission of a user’s private data, including by restricting the use of encrypted communications. ’249 patent, 1:25-32, 2:42-54; Burns, 1:21-35, 2:21-26.

B. Summary of Yang

62. Yang, like Burns, describes an IDS that can determine whether packet flow represents a network attack. Yang, Abstract. The IDS consists of a flow analysis module, a stateful inspection engine, and an application identification module. Yang, 6:29-36.

63. The flow analysis module receives inbound traffic and uses source media access control (MAC) address, destination MAC address, source port, destination port, and IP addresses to identify network flows. Yang, 6:60-7:2.

64. The stateful inspection engine buffers a copy of the packet flow and reassembles it to determine whether the packet flow is associated with an identifiable

application. Yang, 7:16-20. The stateful inspection engine can determine if a packet is encrypted. Yang, 7:27-28. If a packet is encrypted, it can determine whether the packet flow is associated with an unwanted application, and whether key exchange has occurred. Yang, 7:23-32. If an unwanted application is detected, the stateful inspection engine can block a communication, output an alert, or take on different actions. Yang, 7:45-54. If no security risk is detected, the stateful inspection engine directs the forwarding component to forward the packet flows. Yang, 7:51-54.

65. Once a communication session has been interrupted by the stateful inspection engine, the application identification module can attempt to identify an application associated with an intercepted session. Yang, 8:11-13. The application identification module sends data from the communication session to a protocol decoder, which utilizes an application-layer header of a packet to identify a protocol being used by the communication session. Yang, 8:24-44.

66. Yang, like Burns and the '249 patent, is directed towards dynamically managing network communications. '249 patent, Claim 1, 1:19-21, 17:34-35; Yang, 1:6-42, 2:47-49, 5:54-6:8, 6:46-56. Yang and the '249 patent describe a method by which a user or administrator may selectively block (or modify) packets that do not correspond with their settings. *See* '249 patent, 3:10-12 (“wherein the privacy preference includes a list of allowed data patterns for sharing during communication”), 3:27-29 (“wherein the privacy preference includes a list of

allowed data packet types for sharing during communication”), 12:13-19 (“may perform at least one of sharing, blocking and/or modifying... according to the corresponding privacy preference”); Yang, 5:53-65 (“administrator may specify a compound network attack... IDS 10 may take one or more programmed actions, such as automatically dropping packet flows”). Yang is also directed to addressing the same problem of controlling the transmission of private data, such as by restricting communications to and from a private computing devices in a private enterprise computing network. ’249 patent, 1:25-32, 2:42-54; Yang, 4:17-45; 5:53-65.

C. Reasons to Combine Burns and Yang

67. A POSITA would have combined Burns and Yang because Burns refers to and incorporates Yang by reference.

Exemplary techniques for identifying specific applications and protocols are described in greater detail in U.S. patent application Ser. No. 11/835,923, Burns et al., “Identifying Applications for Intrusion Detection Systems,” filed Aug. 8, 2007, assigned to the assignee of the present application, which is hereby incorporated by reference in its entirety.

Burns, 4:39-45.

68. Yang issued from the 11/835,923 application referenced above. Burns therefore encourages the combination of its disclosure with Yang and states that Yang provides “exemplary techniques for identifying specific applications and

protocols.” Burns, 4:39-45. Burns describes an IDS that is responsible for “monitor[ing] traffic into and out of an enterprise network.” Burns, 2:52-54. Yang’s disclosure is directed to “techniques for detecting and preventing network attacks ... network viruses or other malicious activity.” Yang, 1:45-49. Yang adds crucial implementation details that Burns expects but does not expressly address. In fact, implementing Burns by itself would result in an IDS that can identify a threat, but does not have capability alone to address it. A POSITA would have seen this to be a shortcoming and would have looked to a reference like Yang to improve Burns’ IDS to provide it with the functionality to respond to a threat, not just identify it. To ensure that an IDS would detect and prevent attacks or other malicious activity, it would have been obvious to incorporate the disclosure of Yang into the disclosure of Burns.

D. Summary of Guruswamy

69. Guruswamy, like Yang and Burns, describes a system for detecting whether packet flow represents a network attack. The system in Guruswamy is called an “intrusion detection and prevention” (IDP) device. Guruswamy, Abstract. The IDP analyzes network traffic to detect network attacks. Guruswamy, 2:46-47. Guruswamy describes an “attack detection module” as part of the IDP, which applies “compound attack definitions 33 to application-layer elements 36 and protocol anomaly data 38.” Guruswamy, 6:7-9.

70. The “compound attack definitions” of Guruswamy enable an administrator to create a compound attack definition by specifying the protocol, the scope, and patterns associated with an attack. Guruswamy, 7:36-37. For example, an administrator may identify “HTTP” as the protocol of the compound attack definition. Guruswamy, 7:41-43. The administrator may also specify whether the attack definition should be confined to a singular event (“transaction”) or applied over the lifetime of a session. Guruswamy, 7:54-59. Finally, the administrator may define patterns for the compound attack, for example, in the form of regular expressions. Guruswamy, 7:65-67.

71. Guruswamy and the ’249 patent are both related to the same field of dynamically managing network communications. ’249 patent, 1:19-21, 17:34-35; Guruswamy, 1:36-52, 5:23-40, 6:7-38, 6:54-7:22 (e.g., selecting applicable compound attack definitions based on a detected protocol). Guruswamy is also directed to addressing the same problem of controlling the transmission of private data, such as by restricting communications to and from a private computing devices in a private enterprise computing network. ’249 patent, 1:25-32, 2:42-54; Guruswamy, 2:27-35. And like the ’249 patent, Guruswamy contemplates providing “limited access” to “semi-private” resources. Guruswamy, 2:36-42; ’249 patent, 1:62-66 (discussing “partial” blocking of data collection).

E. Reasons to Combine Burns, Yang, and Guruswamy

72. As addressed above, Burns refers to and incorporates Yang by reference in its entirety. Burns, 4:39:45. Yang refers to and incorporates Guruswamy in its entirety:

Further details on application of attack definitions, e.g. compound attack definitions, may be found within U.S. patent application Ser. No. 11/045,572, Guruswamy et al., “Compound Attack Detection in a Computer Network,” filed Jan. 27, 2005, assigned to the assignee of the current application, which is incorporated herein by reference in its entirety.

Yang, 4:52-58.

73. Guruswamy issued from the 11/045,572 patent application described above. Yang encourages the combination of its disclosure with Guruswamy and refers to Guruswamy’s disclosure of “details on application of attack definitions, e.g. compound attack definitions.” Yang, 4:52-58. Guruswamy provides details that Burns and Yang do not explicitly address, such as details on how to implement compound attack definitions, for which an administrator may specify per Burns. Burns, 6:40-45; Yang 4:50. A POSITA would have seen this to be a shortcoming and would have looked to a reference like Guruswamy to provide details on how to implement a compound attack definition. Guruswamy enables an administrator to define a possible attack through combined aspects such as protocol, scope, and patterns. Guruswamy, 7:45-67.

F. Detailed Analysis of Claims

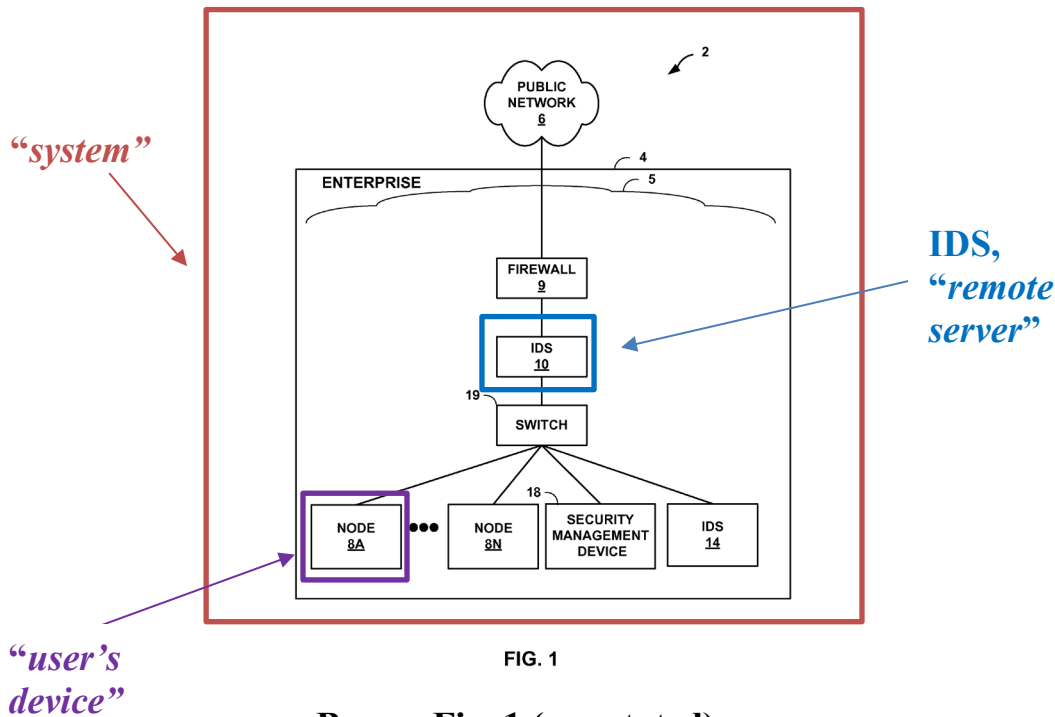
1. Claim 1

- a. [1.0] *A method of dynamic management of encrypted data during communication between a remote server and a user's device, the method comprising:*

74. To the extent that the preamble is limiting, Burns discloses it.

75. Burns describes “*a method*” of “receiving a network packet” in an “unidentified encrypted communication session[,]” and “tak[ing] a response” to receiving the packet. Burns, 3:6, Abstract; *see also* Burns, 9:60-63, Fig. 4 (“flowchart illustrating an exemplary operation of an IDS [intrusion detection system]”). Burns further discloses that “all or portions of an application-layer header within each packet” may be encrypted. Burns, 5:3-5; *see also* Burns, 11:30-38. In the Fig. 4 method, Burns’s IDS provides “*dynamic management of encrypted data*” by managing encrypted communication sessions, for example identifying and terminating unpermitted sessions). A POSITA would have recognized that encryption of the payload or the application-layer header of a packet would correspond to the claimed “*encrypted data*.” As Burns discloses identifying and terminating a session in response to the receipt of an unidentified encrypted data, a POSITA would have understood Burns to teach “*a method of dynamic management of encrypted data*.” Burns, 11:30-38.

76. In Fig. 1, Burns shows a system wherein an IDS “attempts to identify applications and protocols for each communication session between computing nodes 8 and other computing devices in public network 6.” Burns, 4:36-39.



Burns, Fig. 1 (annotated)

77. Part of the system includes “internal computing nodes” which represent any “private computing device within enterprise network 5, including workstations.” Burns, 4:32-34. It would have been obvious for a computer in an enterprise network, such as a workstation, to be used by a user. A given node (e.g., node 8A), computing device or workstation corresponds to the claimed “a user’s device.” Alternatively, Burns describes configuring the IDS via security management device 18. Burns, 6:11-13. It would have been obvious for security management device 18 to be a

workstation computer similar to node 8A that is used by an administrative user. *See* Burns, 6:5-8, 6:37-38. Thus, security management device 18 also corresponds to “*a user’s device.*”

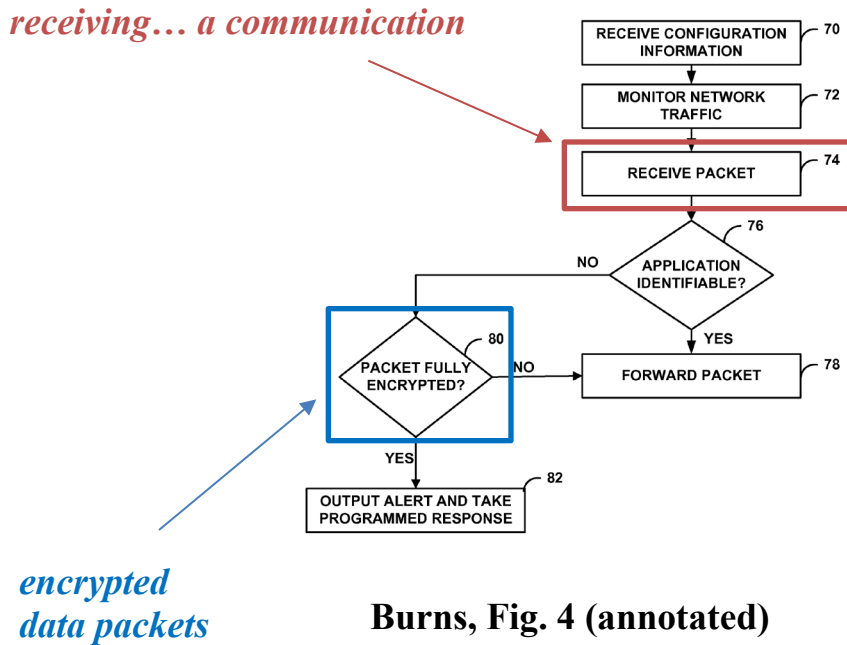
78. The IDS of Fig. 1 is shown to be separated from the node 8A (*user’s device*) and security management device 18 by the switch 19. Burns explains that the IDS may be implemented within a “network device,” thereby making it a *server*. Because the IDS and node are separate, a POSITA would have understood the IDS to correspond to a “*remote server.*” *See, e.g.,* Nayak 238-42 (showing different configurations of a switch separating an IDS from devices.)

79. Burns further explains that its IDS “attempts to identify applications and protocols for each communication session between computing nodes 8 and other computing devices.” Burns, 4:37-38. Therefore, a POSITA would have appreciated that Burns teaches “*communication*” between the IDS (*remote server*) and the nodes (*user’s device*), rendering obvious “[*a method of dynamic management of encrypted data during communication between a remote server and a user's device.*” Alternatively, since Burns contemplates the IDS presenting a user interface to an administrative user at security management device 18 (Burns, 6:5-8, 6:11-13, 6:37-38), it would have been obvious for there to be “*communication between*” security management device 18 and the IDS.

b. [1.1] *receiving, by the remote server, a communication comprising encrypted data packets;*

80. Burns's IDS (*remote server*) includes the "stateful inspection engine 28 of forwarding plane 22." Burns, 9:29-36, Fig.2.

81. The stateful inspection engine "monitor[s] network traffic," and "receive[s] all packets of a particular communication session," i.e., "*receiving...a communication.*" Burns, 10:18-26. This is performed by "stateful inspection engine 28 of forwarding plane 22," which is part of the IDS (*the remote server*). Burns, 6:29-36, Fig.2. As discussed above in the analysis of [1.0], Burns deals with routing "*encrypted data*" within network packets. This configuration is depicted in Fig. 4, below:



82. Once a packet is received by the IDS, (*remote server*) it determines whether the “packets of the communication session are fully encrypted.” Burns, 9:9-25, 11:3-18, Fig.4 (80). This disclosure teaches receiving packets within a “communication” that “compris[es] encrypted data packets.”

- c. [1.2] *determining, by the remote server, a content of at least one data packet of the communication in accordance with characteristics of the at least one data packet, and*

83. Burns’s IDS (*remote server*) identifies, i.e., *determines*, applications and protocols for communication sessions that include “*at least one data packet*”:

In one embodiment, IDS 10 attempts to identify applications and protocols for each communication session between computing nodes 8 and other computing devices in public network 6.

Burns, 4:36-39.

84. Burns references Yang's disclosure for greater detail regarding "[e]xemplary techniques for identifying specific applications and protocols." Burns, 4:36-42.

85. Yang describes identifying the "type of application and protocol associated with the packet flow" through static port mapping with an "application identification module." Yang, 9:59-66. Yang's application identification module uses a packet's TCP port number to identify certain applications and/or protocols. Yang, 9:59-10:29, Table 1. This technique employs "static port mapping" that associates exemplary TCP port numbers with exemplary applications and protocols. Yang's "port number" corresponds to a "*characteristic*" of the packet.

TABLE I	
PORT	APPLICATION
20	FTP
22	SSH
23	Telnet
25	SMTP
43	WHOIS
53	DNS
67	BOOTP or DHCP
70	Gopher
79	Finger
80	HTTP
109	POP
110	POP3
113	ident/IRC
118	SQL
119	NNTP
194	IRC
443	HTTPS
445	SMB
564	RTSP

Table 1, Yang

86. A POSITA would have known that Yang’s “static port mapping” approach allows for an association between the port number (a *characteristic*) and certain types of “*content*,” without having to decrypt the content itself. *See* Nayak, 232-27. For example, for the port value 80, HTTP corresponds to “Hypertext Transfer Protocol (World Wide Web)” and indicates that a packet contains website data. Kozierok, 709. For 443, HTTPS is the encrypted and therefore “secure” HTTP. For 109, POP corresponds to “Post Office Protocol” and indicate that a packet contains email data. Kozierok, 709. *See also* Kozierok, Table 43-1 (showing well-known port numbers and their corresponding applications); and RFC 1700, (listing port numbers for thousands of known applications.)

Declaration of Nader F. Mir, Ph.D.
Inter Partes Review of U.S. 11,816,249

Table 43-1: Common TCP/IP Well-Known Port Numbers and Applications

Port #	TCP/UDP	Keyword	Protocol Abbreviation	Application or Protocol Name/Comments
7	TCP + UDP	echo	—	Echo Protocol
9	TCP + UDP	discard	—	Discard Protocol
11	TCP + UDP	systat	—	Active Users Protocol
13	TCP + UDP	daytime	—	Daytime Protocol
17	TCP + UDP	qotd	QOTD	Quote of the Day Protocol
19	TCP + UDP	chargen	—	Character Generator Protocol
20	TCP	ftp-data	FTP (data)	File Transfer Protocol (default data port)
21	TCP	ftp	FTP (control)	File Transfer Protocol (control/commands)
23	TCP	telnet	—	Telnet Protocol
25	TCP	smtp	SMTP	Simple Mail Transfer Protocol
37	TCP + UDP	time	—	Time Protocol
43	TCP	nicname	—	Whois Protocol (also called Nicname)
53	TCP + UDP	domain	DNS	Domain Name Server (Domain Name System)
67	UDP	bootps	BOOTP/DHCP	Bootstrap Protocol/Dynamic Host Configuration Protocol (server)
68	UDP	bootpc	BOOTP/DHCP	Bootstrap Protocol/Dynamic Host Configuration Protocol (client)

Table 43-1: Common TCP/IP Well-Known Port Numbers and Applications (continued)

Port #	TCP/UDP	Keyword	Protocol Abbreviation	Application or Protocol Name/Comments
69	UDP	ftpp	TFTP	Trivial File Transfer Protocol
70	TCP	gopher	—	Gopher Protocol
79	TCP	finger	—	Finger User Information Protocol
80	TCP	http	HTTP	Hypertext Transfer Protocol (World Wide Web)
110	TCP	pop3	POP	Post Office Protocol (version 3)
119	TCP	nntp	NNTP	Network News Transfer Protocol
123	UDP	ntp	NTP	Network Time Protocol
137	TCP + UDP	netbios-ns	—	NetBIOS (Name Service)
138	UDP	netbios-dgm	—	NetBIOS (Datagram Service)
139	TCP	netbios-ssn	—	NetBIOS (Session Service)
143	TCP	imap	IMAP	Internet Message Access Protocol
161	UDP	snmp	SNMP	Simple Network Management Protocol
162	UDP	snmptrap	SNMP	Simple Network Management Protocol (Trap)
179	TCP	bgp	BGP	Border Gateway Protocol
194	TCP	irc	IRC	Internet Relay Chat
443	TCP	https	HTTP over SSL	Hypertext Transfer Protocol over Secure Sockets Layer
500	UDP	isakmp	IKE	IPsec Internet Key Exchange
520	UDP	router	RIP	Routing Information Protocol (RIP-1 and RIP-2)
521	UDP	ripng	RIPing	Routing Information Protocol - Next Generation

Kozierok, Table 43-1

87. A POSITA would have been familiar with the format of TCP and IP packet headers and would have recognized that identifying an application or protocol with Yang's technique would use multiple packet header values (*characteristics*). U.S. Patent No. 7406522, 1:59-60; U.S. Patent Publication No. 2006/0209858, [0015]; U.S. Patent Publication No. 2003/0014624, [0003]; Perlman, 4; Tanenbaum, 19; Kozierok, 19. First, Yang's technique requires identifying that an IP packet contains a TCP packet via the IP packet header's "protocol" field. Yang, 5:16-18, 6:24-27; *see* RFC 791, 14 ("Protocol:... This field indicates the next level protocol used in the data portion of the internet datagram."). Second, Yang's technique identifies an application or protocol via the source and destination port numbers in the TCP packer's header. Yang, 6:54-58, 17:3-5; *see* RFC793, 15 (showing "Source Port" and "Destination Port" in the "TCP Header Format").

88. In the combination, Burns's IDS (*remote server*) would receive a packet. Yang's application identification module would use static port mapping to identify a port. The port number would correspond to an application, which would be used by Burns's IDS to determine how the packet would be routed in subsequent steps. Because Burns's IDS (*remote server*) identifies an application or protocol and thus the packet's *content* based on mapping port numbers from TCP packet headers (*characteristics*), the combination of Burns and Yang render obvious "*determining*,

by the remote server, a content of the at least one data packet in accordance with the characteristics of the at least one data packet.”

d. [1.3] *wherein the content of the at least one data packet is not decrypted by the remote server, and*

89. Burns explains that packets received by its IDS may be encrypted. Burns 5:17-20. It was common knowledge that encrypted content, including within packets, cannot be decrypted without an encryption key. Nayak, 163-167; U.S. Patent Publication No. 2011/0106710, [0067] (“As should be understood, because the PIN is encrypted by the terminal using a key that is not know[n] by the merchant or acquirer, the merchant or acquirer is unable to decrypt the PIN.”); U.S. Patent Publication No. 2011/0022835, [0056] (“if an unauthorized client device 110 receives this message, it is unable to decrypt message 178 because it does not have possession of decryption key 112.”); U.S. Patent No. 6944762, 1:25-36. A POSITA would have also known that an IDS would generally not have access to the encryption key needed to decrypt packets it receives. *See, e.g.*, Nayak, 243-247. As Burns explains, packets are “encrypted... by one party and decrypted by the other party to a communication session.” Burns, 2:1-3. Since Burns’s IDS is not a “party” to the communication session itself, it would have been obvious to a POSITA for the IDS not to decrypt packets passing through it. Therefore, Burns’s IDS (*remote server*) would not “*decrypt[]*” the “*content of the data packet,*” rendering obvious this limitation.

- e. **[1.4] *the determination of the content is performed by the remote server in real time during a communication session between the remote server and the user's device;***

90. Burns's IDS is an example of an IDPS, as upon identification, it can "drop[] the packet" or "block[] future communication sessions." Burns, 5:61-63. It would have been obvious to a POSITA that the IDS would perform its double function in *real-time*. This is consistent with a POSITA's knowledge that most modern IDSs have IDPS functionality that allows them to not only identify the attack, but also to take action to drop the packet, deny entry, or block the connection in real-time. Nayak, 228-29.

91. As further support, Burns's IDS also utilizes Yang's "exemplary techniques for identifying specific applications and protocols." Burns, 4:36-42. Yang's techniques "analyze the traffic in real-time prior to forwarding." Yang, 11:63-12:3. Furthermore, Burns and Yang describe an IDS that "includes 20 a forwarding plane 22 that transparently monitors inbound network traffic 24 and forwards the network traffic as outbound network traffic 26." Burns, 6:30-34.

92. For the IDS to monitor traffic "transparently," it would have been obvious to a POSITA for the IDS to perform its content determinations "*in real time*" because delaying traffic would interfere with communication sessions. In other words, for the IDS to operate "transparently," it would need to make content determinations "*in real time*."

93. Burns and Yang describe identifying an application associated with a packet with an application identification module 51 within stateful inspection engine 28. Burns, 8:11-23, Fig. 3; Yang, 9:27-49. Thus, application identification module 51 is part of the IDS's "forwarding plane" that operates transparently. As part of the forwarding plane, it would have been obvious for the application identification module 51 to operate "*in real time*" as claimed. The forwarding plane of a network device is responsible for forwarding packets toward their destination with minimal delay. U.S. Patent Publication No. 2005/0281192, [0038] ("excessive delay" is a "forwarding plane failure"); U.S. Patent Publication No. 2011/011, [0030] ("achieving lower latency" is a forwarding plane advantage); U.S. Patent Publication No. 2013/0329578, [0029] (describing forwarding plane architectures "to reduce latency and other delays"). It would have been obvious to a POSITA that an IDS functioning in a "transparent" manner is undetectable to the user and is thus occurring in *real-time*, as, "[m]ost commercial network intrusion detection systems run in real-time." Proctor, 38.

94. As explained in the analysis of [1.1] and [1.2], Burns's IDS (*remote server*) receives a communication (*data packet*) that is sent from node 8 (*user's device*). Burns's IDS is on the data path between the *remote server* and the *user's device*, where it monitors this traffic and determines an application or protocol corresponding to the traffic (*a content*) of the communication. Therefore, Burns and

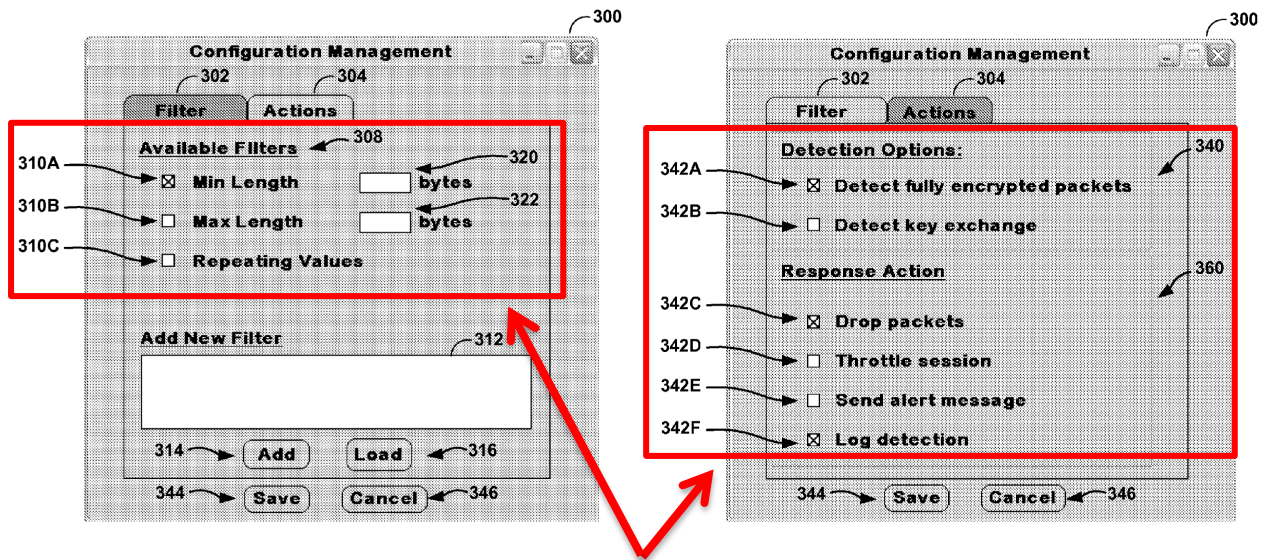
Yang render obvious the “*determination of the content... during a communication session.*”

f. [1.5] *storing, by the remote server, a preference list;*

95. Burns’s IDS can be tailored to an administrator’s desired configuration information. Burns, 1:21-35. Burns allows a system administrator to “configure IDS 10 to explicitly allow all identifiable applications, allow all applications except for a specified list of identifiable applications, or prevent all communications.” Burns, 5:30-34. Furthermore, an administrator can specify attack definitions and desired responses:

Initially, security management module 44 receives configuration information from administrator 42 and, in response, configures IDS 20 to monitor a network or portions thereof (subnets) of interest... configuration manager 44 presents a user interface by which **administrator 42 specifies patterns or other attack definitions 33.** For example, administrator 42 **may configure IDS 20 to block all packets deemed to be fully encrypted and to log information about the communication session associated with the blocked packets.** Administrator 42 **may also configure IDS 20 to throttle down (i.e., bandwidth limit) the communication session** associated with the packets to minimize the bandwidth used by the communication session. Burns, 9:64-10:9.

96. Burns also illustrates in Figs. 7A-7B (below) various options for an administrator to selectively enable or disable, such as whether to detect repeating values or an encryption key exchange. Burns, 17:58-20:67; *see also infra*, claim 4.



Configurable preference list items

Burns, Figs. 7A-7B (annotated)

97. The administrator's configuration information allows the administrator to explicitly permit certain applications, deny others, and provide an "attack definition." A POSITA would have appreciated that, because the configuration information allows administrator to tailor the IDS to their preferences, the configuration information specified by the administrator corresponds to a "preference list."

98. Burns explains that configuration information is sent to a security management module 44 within the IDS, which then controls other IDS components to operate as configured:

Initially, security management module 44 receives configuration information from administrator 42 and, in response, configures IDS 20 to monitor a network or portions thereof (subnets) of interest (70).

Burns, 9:64-67; *see also id.* 19:7-17, 19:22-33, 6:37-59.

99. The administrator transmits configuration to the security management module 44 by clicking the “Save” button shown in Figs. 7A-7B. *See, e.g.,* Burns, 19:7-17. Burns expressly discloses that security management module 44 can store information in “a database, a text file, or any appropriate data structure.” Burns, 20:65-67. Thus, it would have been obvious for the configuration information to be saved (*stor[ed]*) by security management module 44 to facilitate instructing other IDS components to operate in accordance with the administrator-provided configuration information.

100. A POSITA would have found it obvious for the configuration information to be stored by the IDS, as the IDS accesses and uses the configuration information to respond accordingly to incoming packets. Burns’s administrator configuration information renders obvious, “*storing, by the remote server, a preference list.*”

- g. [1.6] *determining, by the remote server, based on a comparison of the determined content, whether to modify the at least one data packet, and*

101. As explained in the analysis of [1.2], an application or protocol in Burns corresponds to “*content*.” As explained in the analysis of [1.5], the IDS (*remote server*) stores patterns or attack definitions an administrator may have specified:

For example, a system administrator may configure IDS 10 to explicitly **allow all identifiable applications**, allow all applications except for a specified list of identifiable applications, or **prevent all communications except for communications from a specified list** of permitted software applications.

Burns, 5:29-34.

102. More specifically, in Burns, the administrator configures a “specified list of permitted software applications,” and the IDS “*compar[es]*” the packet’s identified application (*the determined content*) to the applications on the administrator’s list (*the preference list*). Burns, 5:34. This disclosure is consistent with common knowledge that an IDS compares administrator-specified patterns or other known attacks with the packet’s identified application (*the determined content*) to determine which actions to take with the packet. *See, e.g.*, Nayak, 229-33.

103. Once Burns’s IDS compares the packet’s identified application with the applications in the administrator-specified list, the IDS acts accordingly by, for example, dropping the packet, closing the communication session, or throttling the related communication session:

In addition, stateful inspection engine 28 may take additional actions, such as **dropping the packets associated with the communication session, automatically closing the communication session**, or other actions.

Burns, 7:47-51.

Administrator 42 may also configure IDS 20 to **throttle down (i.e., bandwidth limit) the communication session** associated with the packets to minimize the bandwidth used by the communication session.

Burns, 10:6-9.

104. Deciding whether to perform any one of these three actions renders obvious the claimed “*determining... whether to modify*” step.

105. A POSITA would have appreciated that closing the session as Burns describes would require “*modify[ing]*” the packet. This is because a POSITA would have known that one way to close a connection is to modify the packet’s header to add a FIN flag. Kozierok, 763. Alternatively, as Burns describes, the packet could be dropped. Burns, 7:49-50. This is consistent with the ’249 patent that explains that “blocking the packet” is accomplished by modifying it.’249 patent, 18:14-16.

106. A POSITA would have been familiar with techniques for closing a communication session, such as the TCP communication sessions contemplated by Burns. Burns, 7:20-22, 9:16-25, 9:32-33 (“the TCP/IP payload of a packet of the communication session”). As discussed further at [1.7], it would have been obvious for the IDS to close a TCP communication session by modifying a packet header.

Thus, by deciding to close a TCP communication session, Burns's IDS is "*determining... whether to modify the at least one data packet.*"

107. A POSITA also would have been familiar with techniques for throttling a communication session, such as by manipulating the TCP congestion window size. As discussed further at [1.7], it would have been obvious for the IDS to throttle a TCP session by modifying a packet header. Thus, by deciding to throttle a TCP communication session, Burns's IDS is "*determining... whether to modify the at least one data packet.*"

h. [1.7] if so, modifying the at least one data packet; and

108. As discussed in the analysis of [1.6], after comparing a packet's identified application with the applications in the administrator-specified list, Burns's IDS may drop a packet or close the communication session. Burns, 7:47-51. A POSITA would have understood that dropping a packet is akin to "blocking" a packet. This understanding is also consistent with the '249 patent, which states that "blocking" is a form of "modification." '249 patent, 18:14-16.

109. A POSITA would have understood TCP protocols and would have known that in order to close a TCP session, a packet is "*modif[ied]*." To close a TCP connection, the device that wishes to close the connection will send a FIN flag in the packet header. The other device will respond by sending an ACK, followed by its own FIN message. Kozierok, 763. In a situation where one device wishes to end the

connection, but the other is unresponsive, the device may send an RST flag in the packet header. Kozierok, 760. Such modifications were well known to a POSITA:

[A] TCP endpoint that desires to initiate session closing operations sends a FIN command to the other endpoint. A FIN is represented by a TCP header flag.

U.S. Patent Publication No. 2006/0248582, [0095].

A TCP session is closed by both sides sending packets comprising the FIN flag indicating that the sender has no more data to send.

U.S. Patent Publication No. 2004/0013112, [0109].

Here, if an attack packet is detected, the IPS transmits an RST (Reset) packet of the TCP (Transfer Control Protocol) to an attack source (the source of the attack packet), which is made to forcibly disconnect a TCP connection.

U.S. Patent Publication No. 2008/0059596, [0006].

110. A POSITA would have known that such a modification would terminate a session, as this has been the case since the release of RFC 793 in 1981. *See* RFC793, 12 (“The clearing of a connection also involves the exchange of segments, in this case carrying the FIN control flag”), 23 (Fig. 6 showing “CLOSE” actions accomplished by “s[e]nd FIN”), 38 (“remote TCP initiates by sending a FIN control signal”), 75 (“[i]f the FIN bit is set, signal the user ‘connection closing’”). Burns, therefore, renders obvious *modifying the at least one data packet* by closing the session or dropping the packet.

111. A POSITA also would have been familiar with techniques for throttling a communication session, such as by manipulating the TCP congestion window size. U.S. Patent Publication No. 2002/0046264, [0079] (“by regulating the advertized window size of each system user, each user's bandwidth can be controlled”); U.S. Patent Publication No. 2002/0059435, [0123] (“flow control can be applied... by shrinking the TCP windows being advertised”); U.S. Patent No. 6826620, 11:30-45 (an exemplary rate control technique where a “FRAU [frame relay access unit] controls... the TCP window size within the acknowledge packets.”). Burns describes that the IDS may “throttle down the communication session associated with the packet.” Burns, 11:53-55; *see* Burns, 10:6-9. The TCP reception window size is a value in a TCP packet header. Tanenbaum 4, 537-538 & Fig. 6-29 (describing the “Window size” field), 547-549; RFC 791, 15-16 (“Window:... The number of data octets... which the sender of this segment is willing to accept.”). It would have been obvious to a POSITA for throttling a communication session to involve modifying (e.g., reducing) the window size in a TCP header, which is “*modifying*.” Kozeriok, 805-06.

112. In summary, Burns’s IDS selectively drops or modifies packets, which renders obvious this limitation.

i. **[1.8] *sharing, by the remote server, the modified communication.***

113. As discussed at [1.4], Burns's IDS operates transparently. Burns's IDS includes the stateful inspection engine that "instructs forwarding component 31 to forward the packet flows." Burns, 7:53-55. Thus, it would have been obvious for the IDS to transmit the packet modified as discussed at [1.6]-[1.7].

114. As discussed in the analysis of [1.7], Burns's disclosure of "dropping a packet" corresponds with "*if so, modifying the at least one packet.*" When the IDS (*remote server*) terminates a communication session, the packet would still be "shared," because the packet containing the modified header with either a FIN or RST flag is sent by the IDS to indicate that the IDS wishes to end the connection.

115. Alternatively, when the IDS (*remote server*) drops the packet, the packet is still "*shared*" because the IDS "log[s] information about the communication session associated with the packet." Burns, 11:51-52. A POSITA would have known that this packet information would need to be shared to storage, as it would enable the IDS to identify and prevent associated abnormalities from being sent in the future. Nayak, 230.

2. Claim 2

a. **[2.0] *The method of claim 1,***

116. Claim 2 is dependent from claim 1 and would have been obvious for the reasons addressed in the analysis of [1.0].

- b. [2.1] *wherein determining a content of the at least one data packet comprises determining at least one data type of the at least one data packet of the communication in accordance with characteristics of the at least one data packet, and*

117. As explained in the analysis of [1.2], Burns's IDS determines that a packet's port value (*characteristic*) corresponds to an application or protocol, which indicates the packet's *content*. Claim 2 further limits claim 1 to state that the determination of a "*content*" further corresponds to the determination of a "*data type*." In the '249 patent, the terms "*data type*" and "*content*" are used analogously numerous times. *See e.g.*, '249 patent, 10:46 ("each data type (content)"), 11:13 ("which data types (content)"), Fig.3A ("DATA TYPE (CONTENT)"), Fig.3b ("DATA TYPE (CONTENT)").

118. As discussed in the analysis of claim 1, Burns' IDS determines a packet "*content*," based on its corresponding application or protocol (as discussed in Yang). It would have been obvious for any of these "*content[s]*" to correspond to a "*data type*" as recited in claim 2. Thus, claim 2 is obvious for the same reasons discussed for claim 1, where a packet's "*data type*" corresponds to its application or protocol.

119. Burns's IDS determines whether "a packet is associated with an identifiable network application," "using an application-layer header of the packet." Burns, 3:6-10. The application identification module sends data from the packets to protocol decoders, which might identify a "message sent by the server to the client"

by “inspecting the application layer packet header.” Burns, 13:5-8, Fig. 3. Different types of information in the application layer packet header, results in the packet being sent to different modules. A POSITA would have appreciated that the application-specific layer header was a “*data type*,” as it indicates to the IDS how to categorize and treat the associated packet. See Fig. 3:

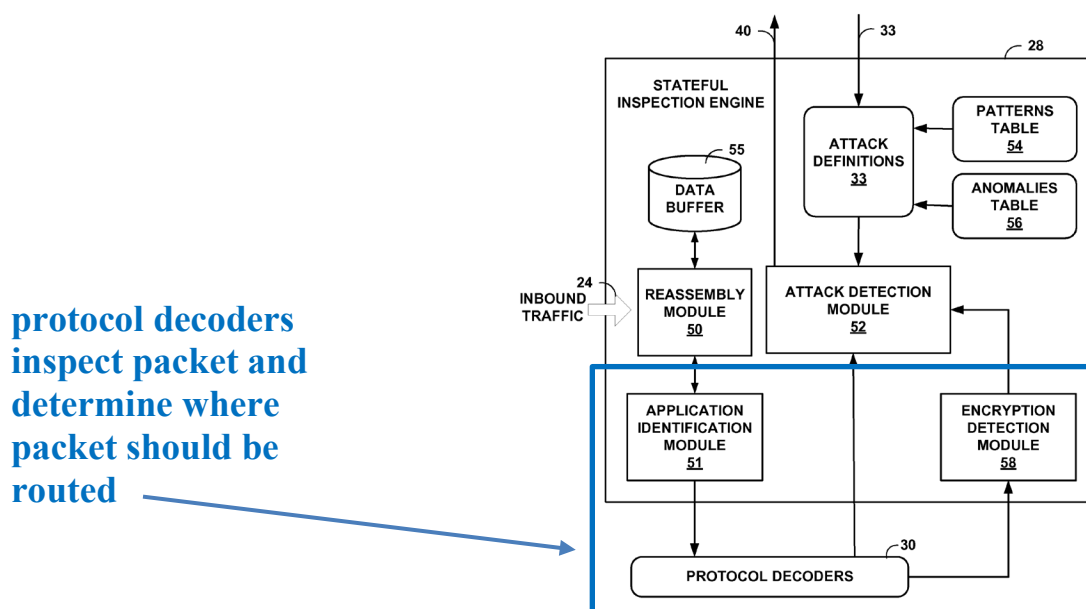
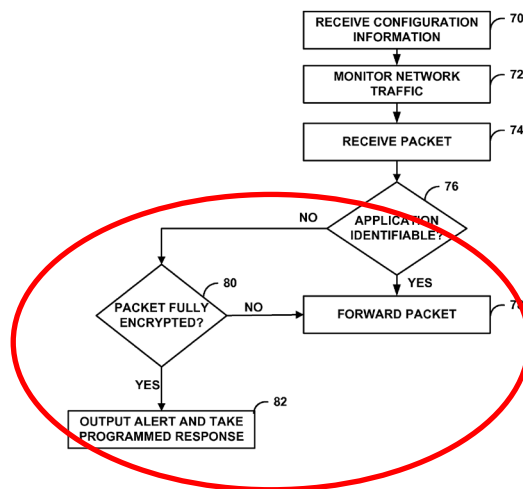


FIG. 3

Burns, Fig.3 (annotated)

120. Similarly, as depicted in Fig. 4, the encryption detection module of Burns’s IDS “examines the packet, including the application-layer header, to determine whether the packet, including the application-layer header, is encrypted.” Burns, 11:7-10. Burns describes that the encryption detection module may send a

packet down the “NO” branch at 80 if it determines that the packet is “not encrypted at all or because the packet is part of a valid encryption scheme.” Burns, 11:19-22, Fig. 4. The encryption detection module may send a packet down the “YES” branch at 80 and trigger a response if the “encryption detection module determines that the packet is fully encrypted.” Burns, 11:30-33, Fig. 4. In this way, Burns discloses “characteristics of...[the] data packets.”



Burns, Fig. 4 (annotated)

121. Burns’s encryption detection module operates consistent with the ’249 patent. The ’249 patent states that “characteristics vectors 304 having at least one identical characteristic may correspond to a particular data type,” and uses “encryption yes/no” as an example of such a characteristic. ’249 patent, 9:6-8. Since one matching characteristic can identify a data type, “encryption yes/no” is an

example of *data type* in the '249 patent. A POSITA would have recognized that the disclosure in Burns of “determining whether a packet is encrypted” corresponds to the “*characteristic*” of the '249 patent “encryption yes/no.” As one *characteristic* can correspond to a *data type*, Burns’s disclosure of the characteristic “whether a packet is encrypted” also corresponds to a “*data type*.”

122. Additionally, Yang describes identifying that a packet’s corresponding application is “unknown,” which is a “*data type*” in the context of the '249 patent. Yang, 8:14-18; *see* 249 patent, 9:13-17, 9:25-30 (“unknown data type content”).

123. Alternatively, the IDS determines whether a packet’s contents correspond to a data type of “encrypted” or “plain text.” It would have been obvious to a POSITA that the encrypted-ness of a packet corresponds to “*at least one data type*.” It was common in the art to refer to generically refer to a packet’s data type as being “encrypted.” *See* U.S. Patent Publication No. 2008/0159146, [0024] (referring to “encrypted” as a “high entropy data type[]”); U.S. Patent Publication No. 2007/0088845, [0010] (“data types like... encrypted data”); U.S. Patent Publication No. 2005/0047449, [0031] (referring to “encrypted multimedia data” as a “data type[]”).

c. [2.2] *wherein determining whether to modify the at least one data packet is based on the determined data type.*

124. As addressed at [1.6]-[1.7], the combination of Burns and Yang render obvious the determination to modify a packet based on its “*determined content*.” As

addressed in the analysis of [2.1], “*the determined content*” of a packet corresponds to the packet’s “*data type*.”

125. Burns describes determining whether a packet is encrypted. Whether a packet is encrypted or not informs the IDS whether to modify the packet:

[W]hen the data in the packet is determined to be encrypted, execute a response, and, when the data in the packet is determined not to be encrypted, forwarding the packet.

Burns, 3:46-49, *see also* Fig. 4.

IDS 10 may permit a communication session that is unencrypted to continue, despite being unable to identify a particular application that is using the communication session. In one embodiment, where IDS 10 determines that a communication session, for which an application cannot be identified, is encrypted, IDS 10 may terminate the communication session.

Burns, 5:55-61; *see also* 9:40-44.

126. By permitting an unencrypted communication session to continue (e.g., by deciding not to modify a packet), Burns’s IDS renders obvious “*determining whether to modify the at least one data packet... based on the determined data type*.” As discussed at [1.7], it would have been obvious for terminating a prohibited encrypted communication session to involve modifying a packet, such as by setting the FIN flag or congestion window size in a TCP header. This further renders obvious “*determining whether to modify the at least one data packet... based on the determined data type*.”

127. By modifying packets that are encrypted, Burns renders obvious this limitation.

3. Claim 3

- a. [3.0] *The method of claim 2, wherein the preference list includes a list of forbidden and/or allowed data packet types for sharing during communication.*

128. As discussed in the analysis of [1.5], Burns's IDS can be configured to match an administrator's desired configuration information such that packets are forwarded or dropped based on their application, protocol, and/or encryption:

Where IDS 10 is able to identify the application using a particular communication session, IDS 10 may **either permit or prevent** the communication session from continuing. For example, a system administrator may configure IDS 10 to explicitly allow all identifiable applications, allow all applications except for a **specified list of identifiable applications**, or prevent all communications except for communications from a **specified list of permitted software applications**.

Burns, 5:27-34.

[A]dministrator 42 may configure IDS 20 to take particular actions when a fully encrypted packet is identified. For example, the user interface may permit the administrator to **drop all fully encrypted packets**, log details about the communication session associated with the fully encrypted packets, drop the packets only if a key exchange has not been previously identified for the communication session, throttle down the communication session with fully encrypted packets to

reduce the bandwidth usage of that communication session, or other actions.

Burns, 6:47-56; *see also* Burns, 12:7-26 & 13:9-57.

129. As addressed in the analysis of [2.2], certain applications or “*data types*” are forwarded, while others are modified. The administrator-provided “specified list of identifiable applications” that are either allowed or prevented renders obvious “*the list of forbidden and/or allowed data packet types for sharing during communication.*” Similarly, the administrator’s choice to “drop all fully encrypted packets” is an “*list of forbidden... data packet types for sharing during communication.*”

4. Claim 4

a. [4.0] *The method of claim 1, further comprising:*

130. Claim 4 is dependent on the “*method*” disclosed in claim 1 and would have been obvious for the reasons addressed in the analysis of [1.0].

b. [4.1] *determining, by the remote server, at least one communication data pattern in accordance with a behavior range of the data packet, and*

131. As discussed in the analysis of [1.0], Burns’s IDS corresponds to the “*remote server.*” The IDS consists of multiple components, such as the “forwarding plane 22 that transparently monitors inbound network traffic 24 and forwards the network traffic as out-bound network traffic.” Burns, 6:29-32. The “forwarding plane” includes “flow analysis module 25, stateful inspection engine 28, protocol

decoders 30, forwarding component 31 and security management module 44.”
Burns, 6:33-36. The stateful inspection engine includes the application identification module 51, attack detection module 52, and encryption detection module 58. Burns, 7:61-62, 8:9-25, Fig. 3.

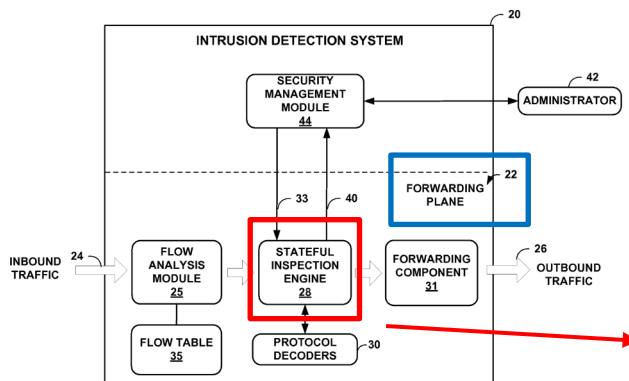


FIG. 2

enlarged view of stateful inspection engine

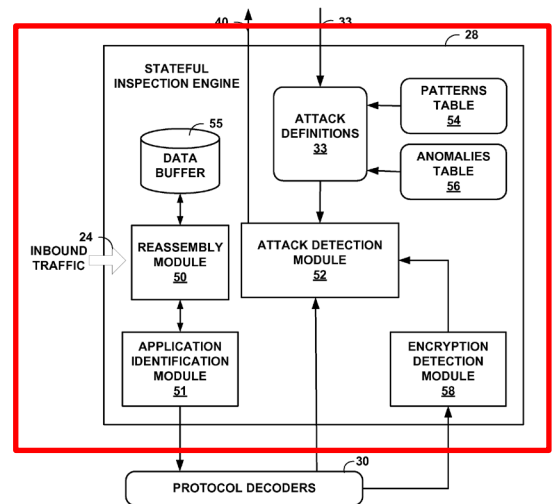


FIG. 3

Burns, Figs. 2-3, (annotated)

132. The IDS can identify a pattern or other “repeating sequence[s]” in a packet. Burns, 18:36-38. The encryption detection module works to identify whether a packet is encrypted and can send “a message to attack detection module 52 that the communication session is being conducted by an undesirable application.” Burns, 9:37-40. The attack detection module can then determine whether a packet is associated with a known network attack:

Another exemplary method for determining whether a packet is encrypted includes **identifying repeating patterns** of varying lengths, e.g., one byte, two bytes, four bytes, or other within a packet.

Burns, 14:32-35.

In one embodiment, encryption detection module 58 compares data in the packet to other data in the packet to find a repeating sequence. In one embodiment, encryption detection module 58 stores a sequence of data from each packet in a communication session and determines whether there exists **a repeated value in each packet of the communication session**; in this case, the repeated value may indicate a standard header of an unknown protocol.

Burns, 18:30-44.

Each of attack definitions **33 specifies a combination of one or more patterns** specified within patterns table 54 and one or more protocol-specific anomalies specified within anomalies table 56.

Burns, 8:6-10.

133. A repeating sequence is a “*communication data pattern in accordance with a behavior range of the data packet.*” See ’249 patent, 9:65-10:6.

134. Yang provides further detail about “protocol-specific anomaly analysis,” including using anomalies called “attack definitions” in order “to detect sophisticated attack **behaviors.**” Yang, 11:49-51. Identifying whether a packet is encrypted or whether a packet’s behavior might be indicative of an attack corresponds to a “*communication data pattern in accordance with a behavior range of the data packet.*” See ’249 patent, 9:65-10:6.

135. Burns describes that a packet's size may be one such "pattern" used to determine how to classify a packet. Burns, 2:34-35 ("If the average size and frequency of data transmission matches known characteristics for a malicious or unwanted application, the IDS may block further communication of that session,"), 14:17-19 ("Another exemplary method for identifying encrypted packets includes tracking two classes for bytes of a packet and comparing the sizes of the classes,"), 14:42-43 ("Encryption detection module 58 may then determine the size N of the TCP/IP payload of the received packet.") Using size as a "*pattern*" to classify a data packet is consistent with the '249 patent, which cites size as an example of a data pattern:

Dynamic **features of a data pattern include** frequency (e.g., appearance(s) per second), **size (e.g., in kilobytes)**, speed (in kilobytes per second), count (e.g., numbers), ratio (e.g., compared to other datatype flow), repeat (e.g., cyclic order), order, server reference (e.g., website reference such as "example.com/feature") and/or any combination thereof and/or other pattern behavior 'm' within the at least one data packet.

'249 patent, 9:65-10:6.

136. Burns also describes analyzing packets to determine whether an encryption key has been exchanged:

In particular, when "Detect key exchange" check box 342A is selected, administrator user interface 300 sends a message to security

management module 44 to detect key exchanges for monitored communication sessions, or to operate in a detect key exchange mode. Accordingly, security management module 44 instructs stateful inspection engine 28 to **analyze incoming packets to determine whether those packets represent a key exchange for their corresponding communication session**. Stateful inspection engine 28 then begins to determine whether incoming packets represent a key exchange as part of the communication session. In one embodiment, stateful inspection engine 28 may only determine whether a first set of packets of a communication session represents a key exchange.

Burns, 19:25-38.

137. A POSITA would have recognized that different protocols utilize different key exchanges and have associated patterns. For example, the SSH key exchange involves SSH_MSG_KEXINIT and SSH_MSG_NEWKEYS. RFC 4253, 17, 21. Burns uses SSL as an example of an encryption key exchange:

Protocol decoders 30 may further inspect the packet to determine if the packet is part of a key exchange of the communication session....As an example, the communication session may be in accordance with the secure socket layer (SSL) protocol. Although application identification module 51 may not be able to determine the specific application associated with the communication, protocol decoders 30 may determine that the communication session is using the SSL protocol.... protocol decoders 30 may recognize a packet that includes key exchange information that is exchanged as part of the SSL protocol.

Burns, 12:42-13:2; *see also* 12:52-13:8 (describing SSL key exchange that requires a specific order of packets sent with ClientHello, ClientMasterKey, and ServerVerify messages to exchange keys).

138. Determining that packets represent a key exchange renders obvious determining a “*communication data pattern in accordance with a behavior range of the data packet.*”

- c. **[4.2]** *wherein the content of the at least one data packet is not decrypted by the remote server, and*

139. As addressed in the analysis of [1.3], Burns’s IDS does not decrypt packets. The same analysis applies here.

- d. **[4.3]** *the determination of the at least one communication data pattern is performed by the remote server in real time during a communication session between the remote server and the user’s device;*

140. As addressed in the analysis of [4.1], the patterns and anomalies taught by Burns correspond to the “*at least one communication data pattern.*”

141. As discussed at [4.1], both the encryption detection module and the attack detection module are part of the stateful inspection engine which is in the forwarding plane. Burns, Figs. 2-3, 7:58-64, 6:33-36. As addressed in the analysis of [1.4], the IDS has IDPS functionality and operates in “real-time” via the forwarding plane so that it is “transparent” to the user. *See* Nayak, 228-29, Burns, 6:30-34. As discussed at [4.1], it would have been obvious for known anomalies to be identified by attack detection module 52, which is also part of the forwarding

plane. Burns, Figs. 2-3, 7:58-64, 6:33-36. Since the forwarding plane operates in real time, a POSITA would have recognized that components of the forwarding plane, including the stateful inspection engine, would function in real time. Therefore, the prior art renders obvious “*determination of the at least one communication data pattern is performed by the remote server in real time during a communication session between the remote server and the user’s device.*”

- e. **[4.4] wherein the determining whether to modify the at least one data packet, by the remote server, is further based on a comparison of the determined data pattern and the preference list,**

142. Burns’s IDS (*remote server*) relies on patterns, such as repeated data, or randomness, to determine whether to close the connection, block a packet, or take other action:

Repeated data in a packet may be an indication that the packet is not random, or that the packet is associated with a protocol that should not be blocked.

Burns, 18:33-36.

[E]ncryption detection module 58 stores a sequence of data...and determines whether there exists a repeated value in each packet of the communication session; in this case, the repeated value may indicate the standard header of an unknown protocol.

Burns, 18:38-44.

When the randomness value of a particular TCP/IP payload of the communication session exceeds the **randomness threshold**,

encryption detection module 58 determines that the communication session is encrypted.

Burns, 9:33-37.

Accordingly, encryption detection module 58 sends a message to attack detection module 52 that the communication session is being conducted by an undesirable application. Attack detection module 52, therefore, may perform one or more of a variety of actions. For example, attack detection module 52 may terminate the communication session, log the communication session, generate a message for, e.g., administrator 42, or take other actions.

Burns, 9:37-44.

143. Therefore, Burns renders obvious using detection of repeated data (a “*determined data pattern*”) to identify whether a packet is encrypted. As addressed at [1.6]-[1.7], whether a packet is encrypted is one criteria Burns’s IDS uses to determine whether to modify a packet. *See, e.g.* Burns, 5:44-49. By using repetition and randomness to determine whether a packet is modified, and administrator preferences about which applications are permitted to employ encryption, Burns renders obvious the limitation of [4.4].

f. [4.5] *whether to modify the at least one data packet,*

144. To the extent that [4.5] is limiting, Burns renders it obvious. As addressed in the analysis of [1.7] and [4.6], Burns’ IDS determines whether to modify a packet based on a *data pattern*, such as repeated data or randomness.

g. [4.6] *if so, modifying the at least one data packet; and*

145. As discussed in the analysis of [1.7] and [4.1]-[4.4], Burns and Yang render obvious determining whether to modify a packet and subsequently modifying the packet.

h. [4.7] *wherein the preference list includes a list of forbidden and/or allowed data packet types and communication data patterns for sharing during communication.*

146. As discussed at [1.5]-[1.6], Burns and Yang disclose that an administrator's desired configuration information can be used to selectively allow certain packets to be forwarded, while other packets are blocked. It would have been obvious to a POSITA that forwarded packets correspond to "*allowed*" packet types, and blocked packets correspond to "*forbidden*" packet types. This configuration information corresponds to a "*preference list*."

147. Burns illustrates in Fig. 5 that the IDS may be configured to make determinations based on multiple factors, including whether the application or protocol is known (step 126) and whether a packet is encrypted (step 128). As discussed at [1.5], Burns's IDS is configurable to act in accordance with an administrator's preferences. Thus, it would have been obvious for the configuration information to include preferences relating to both applications and protocols (*data packet types*) and whether a packet contains repeated values indicative of not being encrypted (*communication data patterns*). See, e.g., Burns, 5:29-53, 14:32-38.

148. Burns illustrates in Fig. 5 that the IDS may be configured to make determinations based on multiple factors, including whether the application or protocol is known (step 126) and whether a key exchange is observed (step 130). As discussed at [1.5], Burns's IDS is configurable to act in accordance with an administrator's preferences. Thus, it would have been obvious for the configuration information to include preferences relating to both applications and protocols (*data packet types*) and whether a key exchange is observed (*communication data patterns*). See, e.g., Burns, 5:29-53.

149. As discussed at [4.4] and [1.5], the administrator's desired configuration information may also apply to "patterns or other attack definitions":

During this process, configuration manager 44 presents a user interface by which **administrator 42 specifies patterns or other attack definitions** 33.

...

In one embodiment, an administrator, such as **administrator 42, may add, edit, or remove filters of encryption detection module 58**. A filter instructs encryption detection module 58 to determine that a packet is random or not random based on certain filtering characteristics. The **filtering characteristics may include, for example**, the number of bytes in the packet, examining bytes in the packet for **a certain pattern**, determining whether a minimum number of byte values above or below the mean occur, or other characteristics of a random or nonrandom packet.

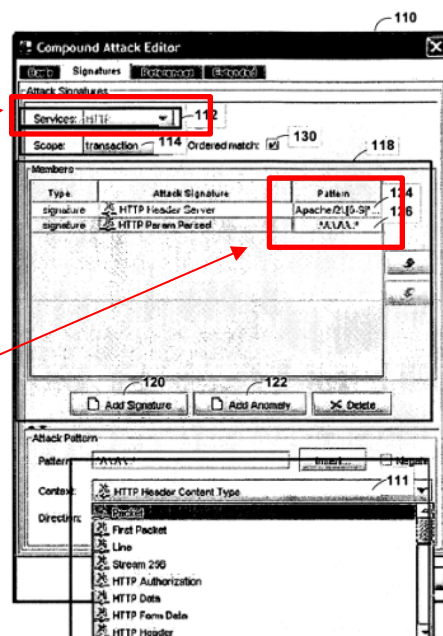
Burns, 16:43-52.

150. Both Burns and Yang describe filtering traffic with “compound attack definitions.” Burns, 6:44-45; Yang, 12:55-67. Yang details that “compound attack definitions” can utilize “specified patterns” and “protocol anomalies,” which can be specified to occur “in a required order.” Yang, 12:55-67. A POSITA would have appreciated that the “compound attack definitions” could be part of the configuration information specified by the administrator.

151. Yang further refers to Guruswamy for details on compound attack definitions. Yang, 4:52-58. In Figs. 6-7, Guruswamy shows a user interface for a compound attack definition. In one field, a use can “specify a protocol to which the compound attack definition pertains,” for example, “HTTP.” Guruswamy, 7:39-41. In another field, the user provides “constituent members of the compound attack definition,” such as “patterns... defined in the form of regular expressions.” Guruswamy, 7:60-67. The specified protocol of a compound attack definition corresponds to a “*data packet type*,” and the specified patterns correspond to “*communication data patterns*.” When a “match” to a compound attack definition is detected, the system can act to “drop[] the packets” or “automatically clos[e] the communication session.” Guruswamy, 7:13-19; *see also* Burns, 7:47-51.

user specifies
that attack
pertains to
certain
protocol, e.g.
HTTP
(data type)

user specifies
that attack
pertains to
certain
pattern(s)
(data pattern)



Guruswamy, Fig. 7, (annotated)

152. It would have been obvious to a POSITA to implement the techniques taught by Yang and Guruswamy to develop “compound attack” definitions, because Burns expressly incorporates the contents of Yang, and Yang expressly incorporates the contents of Guruswamy. Furthermore, it would have been obvious for Burns to utilize compound attack definitions that specified both a protocol or application (*data packet type*) as well as a pattern (*communication data pattern*), as shown in Guruswamy Fig. 7. As addressed in the analysis of [1.5], it would have been obvious for the compound attack definitions to identify protocols as part of the administrator’s configuration information (*preference list*) of the IDS, and for these protocols to be “*forbidden*,” or “*allowed*,” thus rendering obvious this limitation. See

Burns, 6:40-44 (“administrator 42 may specify attack definitions 33”); Guruswamy, 7:23-25.

5. Claim 5

- a. **[5.0] *The method of claim 1, wherein the modification of the communication is selected from the group consisting of:***

153. Claim 1 does not recite “*the modification of the communication.*” For the purposes of interpretation, this limitation will be construed as referring to the step “*modifying the at least one data packet,*” of [1.7].

- b. **[5.1] *blocking the at least one data packet that is not compatible with the preference list, data nullification, data randomization, content modification, change of encoding, change of file template, change of header, change of footer, addition of a predetermined data packet, and encryption.***

154. I understand from Cisco’s counsel that a recitation of each element listed after the colon is not necessary to be shown in the prior art. Nevertheless, my analysis below demonstrates multiple components listed in this claim.

155. As discussed in the analysis of [1.7], Burns describes “dropping” packets based on the administrator’s configuration of the IDS. Burns, 7:47-51. Dropping packets corresponds to “*blocking the at least one data packet that is not compatible with the preference list.*”

156. Similarly, Burns describes “automatically closing the communication session” as a potential action to take per the administrator’s configuration. Burns,

7:47-51, 10:6-9. To close a communication session would entail the modification of a packet header by adding a FIN flag as addressed above in [1.6]. Kozierok, 765.

157. Burns describes that the IDS may “throttle down the communication session associated with the packet.” Burns, 11:53-55. A POSITA would have recognized that one way to throttle a session would be to adjust the TCP window size by modifying the TCP header. Kozierok, 805-06.

158. Therefore, Burns renders obvious “*wherein the modification of the communication is selected from the group consisting of: blocking the at least one data packet that is not compatible with the preference list.... change of header.*”

6. Claim 6

- a. [6.0] *The method of claim 2, wherein the data type of the at least one data packet is determined based at least in part on metadata of at least one data packet of the communication.***

159. Claim 2 is a dependent claim of claim 1. As discussed in the analysis of [1.2] and [2.1], Burns describes a packet’s *data type* (application or protocol) may be identified based on the TCP port value. Kozierok, 708-09; *see* discussion at [1.2], (for example, a port value 80 indicates the HTTP port, which would indicate that a packet contains website data.) The TCP port value in a packet header is “*metadata.*”²

² A TCP port number was commonly considered to be a form of “metadata” about a packet. *See* U.S. Patent Application No. 2017/0230065, [0030] (“traffic

7. Claim 7

a. [7.0] *The method of claim 4,*

160. Claim 7 is dependent from claim 4 and limitation [7.0] would have been obvious for the reasons addressed in the analysis of [4.0].

b. [7.1] *wherein the data pattern is determined based on communication behavior,*

161. As discussed in the analysis of [4.1], Burns's IDS (the *remote server*) can determine the "*data pattern*" of a packet based on "identifying repeating patterns." Burns, 14:32-35. Burns describes that the IDS can determine "attack definitions" that are composed of one or more patterns. Burns, 8:6-10.

162. The encryption detection module of the IDS "stores a sequence of data from each packet in a communication session and determines whether there exists a repeated value in each packet of the communication session." Burns, 18:30-44.

metadata, such as MAC address, IP address, TCP port, UDP port, etc."); U.S. Patent Application No. 2014/0351106, [0033], ("a port number (e.g., a Transmission Control Protocol (TCP) port number) may also be included in the metadata record"); U.S. Patent Application No. 2012/0230208, [0057] ("metadata extracted from each outgoing TCP packet may include source and destination IP addresses, source and destination TCP ports").

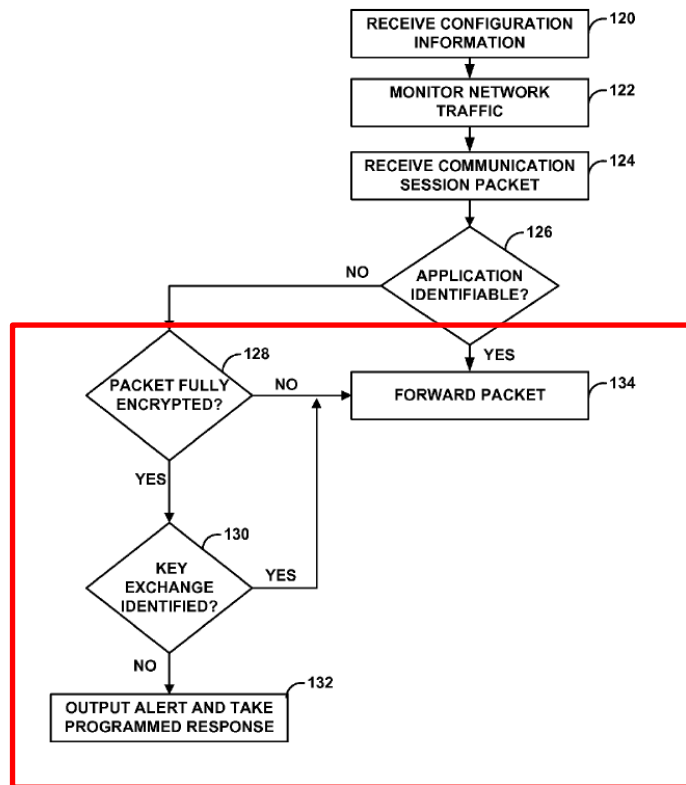
Burns renders obvious determining a “*data pattern*” “*based on communication behavior*.”

163. Burns also discloses detection of a key exchange from an encrypted session. Burns, 5:64-6:4, 7:32-36, Fig. 5.

[T]he default protocol decoder attempts to analyze the bi-directional communications for the communication session to identify a key exchange for cryptography. As another example, the default protocol decoder attempts to identify regularly occurring patterns that correspond to protocol headers in the packets.

Burns, 8:59-64.

164. Determining whether to exchange keys or bypass the key exchange is a “*communication behavior*.”



Burns, FIG. 5, (annotated)

165. Guruswamy describes identifying an attack with an “attack definition” when “specified patterns and protocol anomalies are satisfied in a required order.” Guruswamy, 7:9-12. It would have been obvious for such patterns and anomalies to be satisfied over the course of multiple packets, and hence, for the attack definition to require packets arrive in a particular order, which is a “*communication behavior*.”

- c. [7.2] *wherein the communication behavior consists of data packet frequency, data packet size, data packet speed, data packet count, data packet ratio compared to*

other data type flow, data packet repetition, or data packet order.

166. As discussed above at [7.1], Burns describes that a repeating pattern can be identified across multiple packets that comprise a communication session. Identifying patterns across multiple packets corresponds to “*data packet repetition*,” which is a form of “*communication behavior*” disclosed in limitation [7.2].

167. I understand from Cisco’s counsel that a recitation of each element listed after the colon is not necessary to be shown in the prior art.

168. It would have been obvious that the “key exchange” in Burns would require for packets to be exchanged in a “*data packet order*,” as encryption keys must be exchanged in a specific order to complete the handshake and identify legitimate encrypted traffic. *See, e.g.* RFC 5246, 34; Kozierok, 162-65, 172-177.

169. As discussed at [4.7], Guruswamy’s compound attack definitions can contain specified protocols as part of the compound attack definition. Guruswamy describes that the stateful inspection engine “may determine whether the specified patterns and protocol anomalies are satisfied in a required order.” Guruswamy, 7:10-12. Guruswamy further renders obvious a “*data packet order*.”

170. A POSITA would have understood that an IDS could also look to “the average size and frequency of data transmission” to determine whether it “matches known characteristics for a malicious or unwanted application.” Burns, 2:33-36. Matching size or frequency of a transmission corresponds to “*data packet*

frequency” and “*data packet size*,” which are forms of “*communication behavior*” disclosed in limitation [7.2].

171. Therefore, the prior art renders obvious “*wherein the communication behavior consists of data packet frequency, data packet size...data packet repetition, or data packet order.*”

8. Claim 8

- a. **[8.0]** *The method of claim 1, further comprising, the remote server maintaining communication between the remote server and the user’s device by sharing of the modified communication.*

172. As discussed in the analysis of [1.4] and [1.8], Burns and Yang describe an IDS that “transparently monitors...and forwards the network traffic.” Burns, 6:30-34; Yang 7:17-19. It would have been obvious to a POSITA that for the IDS to function “transparently” as it shares network traffic, this would require “*maintaining communication between the remote server and the user’s device by sharing of the modified communication.*”

9. Claim 9

- a. **[9.0]** *The method of claim 1, wherein the data type is determined based on data packet characteristics selected from a group consisting of: data packet header, data packet footer, version number, IP (Internet Protocol) address, HTTPS (HyperText Transfer Protocol Secure),*

***file extension, encryption method, encoding method,
keywords, driver, and communication protocol.***

173. As discussed at [1.2], both Burns and Yang determine the application or protocol by comparing a TCP port value in the header to a list of known TCP port values that correspond to certain applications and protocols. *See* Burns, 4:36-42; Yang, 9:59-66, Table 1. A POSITA would have understood that a TCP port value corresponds to the claimed “*data packet characteristic*,” because it defines how the packet is treated by the receiving device. For example, a packet with the TCP port value 80 corresponds to the HTTP protocol. Kozierok, 709. It would have been obvious that a packet on port 80 would be formatted according to HTTP protocol, thus, the protocol type corresponds to the to the “*data type*” of the packet. It would have also been obvious that the “*content*” of an HTTP packet would be web data.

174. I understand from Cisco’s counsel that a recitation of each element listed after the colon is not necessary to be shown in the prior art. Nevertheless, my analysis below demonstrates multiple components listed in this claim.

175. The TCP port value (*data packet characteristic*) is a field of the TCP header (*data packet header*). *See* RFC793, 15 (showing “Source Port” and “Destination Port” in the “TCP Header Format”). Therefore, using the TCP port value to determine the protocol renders obvious “*the data type is determined based on data packet characteristics selected from a group consisting of: data packet header....*”

176. Just as the port number 80 would have been understood to correspond to a “*data packet characteristic*” indicating that the “*data type*” of the packet is the HTTP protocol, a POSITA would have recognized that other port numbers could be used to determine other applications (*data type[s]*). Kozierok, 709. For example, as described in Yang at Table 1, the port number 443 corresponds to HTTPS. Yang, Table 1. Therefore, using the TCP port value 443 to determine that the corresponding protocol is HTTPS renders obvious “*the data type is determined based on data packet characteristics selected from a group consisting of: ... HTTPS (HyperText Transfer Protocol Secure)....*”

177. It would have been obvious to a POSITA that other fields of a packet header could be used to determine a “*data type*.” A POSITA would have recognized that “version information” is included in the packet header for an HTTP packet. Kozierok, 1346-48. As addressed at [1.6], the IDS of Burns and Yang may be configured to “prevent unwanted applications.” Burns, 4:1. Yang describes that the IDS may look for anomalies, such as missing HTTP version information, in the packet header to determine that the application is unwanted. Yang, 5:35-40 (“Other anomalies refer to protocol events...that may warrant a heightened level of scrutiny... Example anomalies for the HTTP protocol include missing HTTP version information...”). It would have been obvious that “version information” would have corresponded to a “*characteristic*” that would identify the application (*data type*),

and that “missing version information” would correspond to an unwanted application. Therefore, determining that a packet is of an unwanted type based on version information in the packet header renders obvious “*the data type is determined based on data packet characteristics selected from a group consisting of: ...version information.*”

178. A POSITA would have known that IP address is another field of a packet header. Kozierok, 330-33. As addressed at [1.6], the IDS of Burns and Yang can be configured to allow or prevent certain packets. Burns, 5:29-34. As addressed at [1.8], the IDS can “log information about the communication session associated with a packet” that has been dropped to keep track of unwanted packets. Burns, 11:51-52. Burns describes that “information” logged about a packet may include “source IP address” and “destination IP address.” Burns, 20:60-62. A POSITA would have recognized that an IP address corresponds to a “*data packet characteristic*” that is used to determine the “*data type*” of the corresponding application, rendering obvious “*wherein the data type is determined based on data packet characteristics selected from a group consisting of: ...IP (Internet Protocol) address.*”

179. Similarly, it would have been obvious that the protocol field of an IP header corresponds to a “*data packet characteristic*” that can be used to identify whether a packet’s protocol (*data type*) is TCP or UDP. See RFC791, 14

(“Protocol:... This field indicates the next level protocol used in the data portion of the internet datagram.”); RFC 790, 6 (showing TCP assigned Internet Protocol Number 6); Kozierok, 457 (“in transport mode, the protocol protects the messaged passed down to IP from the transport layer...the appropriate headers are added in front of the transport (UDP or TCP) header. The IP header is then added in front of that by IP.”) Therefore, “*wherein the data type is determined based on data packet characteristics selected from a group consisting of:...*” By determining whether a packet corresponds to the UDP or TCP communication protocol, this renders obvious “*the data type is determined based on data packet characteristics selected from a group consisting of:... communication protocol.*”

180. A POSITA would have understood that like a packet header, a packet footer may also contain “*data packet characteristics*” that identify the packet as pertaining to a certain protocol (*data type*.) For example, redundancy data such as cyclic redundancy check (CRC) is used in the packet footer in protocols such as Asynchronous Transfer Mode (ATM). Kozierok, 17-20; 103-105. The use of CRC (*a characteristic*) in a packet footer (*data packet footer*) would have been used to identify ATM traffic (*data type*).

181. Therefore, the prior art renders obvious “*wherein the data type is determined based on data packet characteristics selected from a group consisting of: data packet header, data packet footer, version number, IP (Internet Protocol)*

address, HTTPS (HyperText Transfer Protocol Secure)... and communication protocol.”

10. Claim 10

- a. [10.0] *The method of claim 1, wherein the remote server is a gatekeeper server.***

182. The IDS in Burns “intercepts packet flows” and “filters unwanted or malicious packet flows.” Burns, 4:20-25. Burns’s IDS also monitors traffic in a bidirectional sense when determining whether a key exchange has occurred. Burns, 7:8-15, 8:59-61 (“the default protocol decoder attempts to analyze the bi-directional communications for the communication session...”). By monitoring both incoming traffic and outflow communication, Burns’s IDS is akin to the “exit gatekeeper” described in the ’249 patent. *See* ’249 patent, 16:31-34 (“a data management system...may be implemented as an exit gatekeeper to monitor outflow communication.”) Therefore, Burns’s IDS renders obvious “*wherein the remote server is a gatekeeper server.*”

11. Claim 11

- a. [11.0] *The method of claim 2, further comprising, the remote server issuing an alert to the user device based on the comparison of the determined data type and the preference list.***

183. Burns’s IDS can be configured to modify or drop a packet by determining the application or protocol (*data type*) via the port number, and by comparing the application to the administrator’s configuration. *See supra*, [1.6]-[1.7]

and [2.1-2.2]. Burns describes that the IDS can take other actions based on the packet's *data type* and the administrator's configuration (*preference list*):

When stateful inspection engine 28 determines that the packet flow is encrypted, stateful inspection engine 28 further determines whether characteristics of the communication session indicates that the packet flow is associated with an unwanted software application.... [W]hen stateful inspection engine 28 determines that a key exchange has not occurred, stateful inspection engine 28 takes a programmed response, e.g., to block the communication session, block future communication sessions from a peer of the communication session, **output an alert**, or other actions.

In the event a security risk is detected, stateful inspection engine 28 outputs an alert 40 to security management module 44 for logging and further analysis.

Burns, 7:28-47.

When encryption detection module determines that the packet is fully encrypted, however, ("YES" branch of 80), encryption detection module 58 may **output an alert and take a programmed response** (82).

Burns, 11:30-33.

When administrator 42 selects send alert message check box 342E, and then selects save button 344, administrator user interface 300 sends a corresponding message to security management module 44. In particular, when "Send alert message" check box 342E is selected, administrator user interface 300 sends a message to security

management module 44 to send an alert message to, e.g., administrator 42. Accordingly, when IDS 20 detects a fully encrypted packet, security management module 44 **sends an alert to administrator 42**. The alert may be any sort of an alert, e.g., a pop-up window, an e-mail, a text message, a page, an audible alert, or any other sort of an alert to inform administrator 42 that IDS 20 has detected, e.g., a fully encrypted packet. Burns, 20:41-53.

184. As explained above in [1.0], Burns explains that the system includes “internal computing nodes” that correspond to a “*user device*”. A POSITA would have understood, in view of Burns’s background discussion, that these “internal computing nodes” would include “end user computers.” Burns, 1:8-16. Burns further explains that an administrator may be a user, and illustrates administrator 42 in Fig. 2, showing that the administrator is outside of the IDS. Burns, 6:5-8 (“a user, such as a system administrator”), 9:56 (“a user, such as administrator”). Burns does not show administrator 42 in Fig. 1, which illustrates node 8 and security management device 18, either of which correspond to a “*user’s device*.” *See supra*, [1.0]. It would have been obvious for administrator 42 to configure IDS 10 from inside the enterprise network, and thus, for administrator 42 to use node 8 or security management device 18. Thus, to send an alert to administrator 42 as describe by Burns, the IDS would send the alert to the node 8 or device 18 used by the administrator (*issue an alert to the user device*). Therefore, based on the combined

disclosure, a POSITA would have understood that an “administrator” would use an internal computing node as a user.

12. Claim 12

- a. **[12.0] *A method of dynamic management of encrypted data during communication between a remote server and a user's device, the method comprising:***

185. Claim limitation [12.0] is the same as [1.0] and would have been obvious for the reasons discussed above.

- b. **[12.1] *receiving, by the remote server, a communication comprising encrypted data packets;***

186. Claim limitation [12.1] is the same as [1.1] and would have been obvious for the reasons discussed above.

- c. **[12.2] *determining, by the remote server, at least one communication data pattern of at least one data packet of the communication in accordance with a behavior range of the data packet, and***

187. Claim limitation [1.2] recites “*determining... a content of at least one data packet... in accordance with characteristics,*” whereas limitation [12.2] recites “*determining... at least one communication data pattern of at least one data packet... in accordance with a behavior range.*” The main differences are that [12.2] substitutes “*the content*” of [1.2] for “*at least one data pattern*,” and “*characteristics*” for “*a behavior range*.” Both claims involve classifying the data packet based on a subset of its features, and therefore, the same analysis as applied in [1.2] applies.

- d. **[12.3] *wherein content of the at least one data packet is not decrypted by the remote server, and***

188. Claim limitation [12.3] is the same as [1.3] and would have been obvious for the reasons discussed above.

- e. **[12.4] *the determination of the at least one data pattern is performed by the remote server in real time during a communication session between the remote server and the user's device;***

189. Claim limitation [12.4] is the same as [1.4], except “*the content*” is substituted for the phrase “*communication data pattern.*” The same analysis at [1.4] applies.

- f. **[12.5] *storing, by the remote server, a preference list;***

190. Claim limitation [12.5] is the same as [1.5] and would have been obvious for the reasons discussed above.

- g. **[12.6] *determining, by the remote server, based on a comparison of the determined data pattern and the preference list, whether to modify the at least one data packet, and***

191. As discussed at [4.1]-[4.3], Burns renders obvious determining a “*communication data pattern*” based on a “*behavior range.*” As discussed at [4.4-4.6], the “*determined pattern*” is used to determine whether to modify a packet. Therefore, “*determining...whether to modify*” based on “*the determined pattern*” would have been obvious for substantially the same reasons as discussed in the analysis of claims 1 and 4.

h. [12.7] *if so, modifying the at least one data packet; and*

192. Claim limitation [12.7] is the same as [1.7] and would have been obvious for the reasons discussed above.

i. [12.8] *sharing, by the remote server, the modified communication.*

193. Claim limitation [12.8] is the same as [1.8] and would have been obvious for the reasons discussed above.

13. Claim 13

a. [13.0] *The method of claim 12, wherein the preference list includes a list of forbidden and/or allowed communication data patterns for sharing during communication.*

194. Claim 13 recites the same features as limitation [4.7] and would have been obvious for the reasons discussed above.

14. Claim 14

a. [14.0] *The method of claim 12, wherein the modification of the communication is selected from the group consisting of: blocking the at least one data packet that is not compatible with the preference list, data nullification, data randomization, content modification, change of encoding, change of file template, change of header, change of footer, addition of a predetermined data packet, and encryption.*

195. Claim 14 recites the same group of modifications as [5.1] and would have been obvious for the reasons discussed above.

15. Claim 15

a. [15.0] *The method of claim 12,*

196. The “*method*” of claim 12 is obvious for the same reasons addressed in the analysis of claim 1, therefore, [15.0] is obvious for the same reasons.

b. [15.1] *wherein the communication data pattern of the at least one data packet is determined based on communication behavior,*

197. Claim [15.1] recites substantially the same features as [7.1]. The only difference is that [7.1] states that “*the data pattern is determined based on communication behavior,*” whereas [15.1] states that “*wherein the communication data pattern of the at least one data packet is determined based on communication behavior.*” A “*communication data pattern*” is substantially the same as a “*data pattern.*” See *supra* [4.1], [12.2]. Similarly, that a “*data pattern*” pertains to “*at least one data packet*” would have been obvious to a POSITA. See *supra* [4.1], [12.2].

c. [15.2] *wherein the communication behavior consists of data packet frequency, data packet size, data packet speed, data packet count, data packet ratio compared to other data type flow, data packet repetition, or data packet order.*

198. Claim limitation [15.2] is the same as [7.2] and would have been obvious for the reasons discussed above.

16. Claim 16

a. [16.0] *The method of claim 12, further comprising, the remote server maintaining communication between the*

remote server and the user's device by sharing of the modified communication.

199. Claim 16 recites substantially the same features as claim 8 and would have been obvious for the reasons discussed above.

17. Claim 17

- a. [17.0] *The method of claim 12, wherein the remote server is a gatekeeper server.***

200. Claim 17 recites the same “gatekeeper server” limitation of claim 10 and would have been obvious for the reasons discussed above.

18. Claim 18

- a. [18.0] *The method of claim 12, further comprising, the remote server issuing an alert to the user device based on the comparison of the determined data pattern and the preference list.***

201. Claim 18 is similar to claim 11 but describes “*issuing an alert...based on the comparison of the determined data pattern*” rather than “*issuing an alert...based on the determined data type.*” As discussed at [4.1], Burns renders obvious analyzing packets in order to determine whether an encryption key has been exchanged (*data pattern*). In Fig. 5, 132, Burns shows that an alert may be sent based on whether an encryption key has been exchanged. In Fig.7B, “Detect key exchange” and “Send alert message” are part of the administrator’s configuration (*preference list*, as addressed at [1.5]) interface for the IDS. Burns, 20:41-53. Therefore, in addition to the analysis of [11.0], Burns renders obvious issuing an

alert based on administrator's configuration (*preference list*) indicating to issue an alert if a key exchange is detected (e.g., a detected *data pattern*).

19. Claim 19

- a. [19.0] *A system for dynamic management of encrypted data during communication between a remote server and a user's device, the system comprising:***

202. To the extent this preamble is limiting, Burns renders it obvious. This preamble is similar to [1.0], except the claim is a “*system*” rather than a “*method*.” Because Burns describes an “exemplary system” and describes the techniques therein as part of a system (Fig. 1), it would have been obvious that Burns describes management of encrypted data using “systems.” Burns, 1-6:8, 4:20-25. Additional features of the preamble would have been obvious for the reasons discussed in the analysis of [1.0].

- b. [19.1] *a memory;***

203. The system in Burns comprises different types of memory, including “random access memory (RAM),” “a hard drive, a network drive, [or] a flash memory stick.” Burns, 8:4-6, 18:51-53; *see also id.*, 21:24-31.

- c. [19.2] *a communication data type database comprising at least one communication data type corresponding to sharing of at least one data packet with the user's device;***

204. As addressed at [1.2], Burns's IDS identifies, i.e., *determines*, applications and protocols for communication sessions, “IDS 10 attempts to identify applications and protocols for each communication session between computing

nodes 8 and other computing devices in public network 6.” Burns, 4:36-39. Burns references Yang’s disclosure for greater detail regarding “[e]xemplary techniques for identifying specific applications and protocols.” Burns, 4:36-42. Yang’s application identification module uses a packet’s TCP port number to identify certain applications and/or protocols. Yang, 9:59-10:24, Table 1.

TABLE I

PORT	APPLICATION
20	FTP
22	SSH
23	Telnet
25	SMTP
43	WHOIS
53	DNS
67	BOOTP or DHCP
70	Gopher
79	Finger
80	HTTP
109	POP
110	POP3
113	ident/IRC
118	SQL
119	NNTP
194	IRC
443	HTTPS
445	SMB
564	RTSP

205. A POSITA would have known that Yang’s “static port mapping” approach allows for an association between the port number (a *characteristic*) and certain “*data types*,” which indicate the *content* of the packet. For example, port number 80 corresponds to the *data type*, HTTP, which indicates that a packet contains website data. Kozierok, 709. Port number 443 corresponds to the *data type* HTTPS and indicates that the content is encrypted or “secure” HTTP. Port 109

corresponds to the *data type* POP “Post Office Protocol,” and indicates that a packet contains email data. Kozierok, 709; *see supra* Kozierok, Table 43-1.

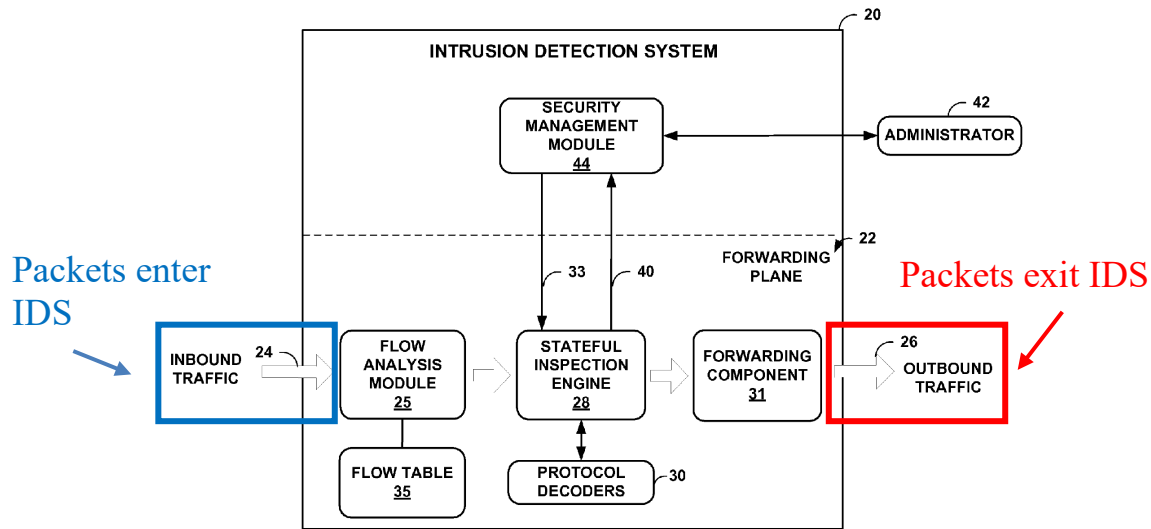
206. Burns discusses analyzing traffic associated with the applications and protocols in Yang’s table: “[f]or example, port 80 is typically associated with HTTP traffic, while port 22 is typically associated with SSH traffic.” Burns, 10:32-45. Therefore, the list of protocols or applications as identified by Yang’s static port mapping approach corresponds to “*a communication type database comprising at least one communication data type corresponding to sharing of at least one data packet with the user's device.*”

d. [19.3] *a preference list database,*

207. Claim limitation [19.3] is substantially similar to claim limitation [1.5] and would have been obvious for the same reasons.

e. [19.4] *a communication module configured to allow communication between the remote server and the at least one user's device, and*

208. In Fig. 2, Burns shows that network traffic flows into and out of the IDS, wherein “each network flow represents a flow of packets.” Burns, 6:61-63. The IDS contains elements that would be part of a forwarding plane, such as the “forwarding component 31”, thus, the IDS corresponds to the claimed “*communication module.*”



Burns, Fig. 2 (annotated)

209. As discussed at [1.0], Burns shows in Fig. 1 that the IDS manages communications “between computing nodes 8 and other computing devices in public network 6.” Burns, 4:36-39. Thus, it would have been obvious for the IDS’s network interface(s) to “allow communication between the remote server [i.e., the IDS itself] and the at least one user’s device.”

f. [19.5] a processor, coupled to the preference list database,

210. Burns describes a “programmable processor” that receives a network packet. Burns, 3:37. Burns further describes that the IDS may be implemented via a “general-purpose processor”:

Methods described herein may be performed in hardware, software, or any combination thereof within a network device. For example, methods described herein may be performed by an application specific

integrated circuit (ASIC) or a general-purpose processor. Methods described herein may also be embodied in a computer readable medium containing instructions. Instructions embedded in a computer readable medium may cause a programmable processor, or other processor, to perform the method, e.g. when the instructions are executed.

Burns, 21:14-23.

211. The IDS operates in accordance with the administrator's configuration, as addressed at [19.3]. To facilitate this (i.e., for the IDS to operate as the administrator has configured it), it would have been obvious for the processor to access and use the configuration information provided by the administrator.

g. [19.6] *wherein the processor is configured to instruct the remote server to:*

212. As addressed above at [19.5], Burns's system involves a processor configured to work with the IDS. As addressed at [1.0], Burns's IDS corresponds with the "*remote server*." It would have been obvious for the processor to instruct the IDS, because a processor is the "brain" of a computer and is responsible for directing its operations. *See, e.g.*, U.S. Patent Publication No. 2003/0038791, [0025] ("central processing unit 22 is a brain of computer system" that will "direct the operation of the internal processor components."); U.S. Patent No. 5289589, 1:17 ("Processors are the brain of computer systems..."); U.S. Patent Publication No. 20060040650, [0031] ("Executed by a device's processor(s), these instructions direct the operation of the device. These instructions can be stored in a memory of the

computer. Instructions can invoke other instructions.”). Burns discloses that “[i]nstructions embedded in a computer readable medium may cause a programmable processor, or other processor to perform the method,” wherein the method is the identification of and response to an encrypted packet. Burns, 21:20-22.

213. It would have been obvious to a POSITA that the “*processor*” would be a component within the “*remote server*,” an interpretation that is consistent with the ’249 specification. *See* ’249 patent, 13:41-42 (“In some embodiments, processor 402 may be embedded in server 401.”) Thus, Burns renders obvious the claimed “*processor*.”

h. [19.7]: *determine a content of at least one data packet of the communication in accordance with characteristics of the at least one data packet, and*

214. Claim limitation [19.7] is substantially the same as [1.2], except for [1.2] recites “*determining, by the remote server, a content...*” It would have been obvious to a POSITA that the IDS (*remote server*) would “*determine a content*.” [19.7] would have been obvious for the reasons addressed in [1.2] above.

i. [19.8]: *wherein the content of the at least one data packet is not decrypted by the remote server,*

215. Claim limitation [19.8] is substantially the same as [1.3] and would have been obvious for the reasons discussed above.

- j. **[19.9]: *and the determination of the content is performed by the remote server in real time during a communication session between the remote server and the user's device,***

216. Claim limitation [19.9] is substantially the same as [1.4] and would have been obvious for the reasons discussed above.

- k. **[19.10]: *based on a comparison of the determined content and the preference list,***

217. Claim limitation [19.10] recites substantially the same features of [1.6], which recites “*determining, by the remote server, based on a comparison of the determined content...*”. As addressed at [19.7], it would have been obvious that the remote server is performing the function. [1.6] does not explicitly recite that the “*determined content*” is being compared to the “*preference list,*” however, it would have been obvious to a POSITA to compare the determined content to the preference list, as addressed in the analysis of [1.6.] Therefore, [19.10] would have been obvious.

- l. **[19.11]: *determine whether to modify the at least one data packet, and***

218. Claim limitation [19.11] recites substantially the same features of [1.6]. As addressed at [1.6], would have been obvious to compare the “*determined content*” to the “*preference list.*”

m. [19.12]: *if so, modify the at least one data packet; and*

219. Claim limitation [19.12] recites substantially the same features as [1.7]. As addressed at [1.7], it would have been obvious after determining whether to modify the packet to modify the packet.

n. [19.13]: *share the modified communication.*

220. Claim limitation [19.13] recites substantially the same features as [1.8]. As addressed at [1.8], it would have been obvious to share the modified communication.

20. Claim 20

a. [20.0] *The system of claim 19,*

221. The “*system*” of [20.0] would have been obvious for the reasons discussed at [19.0] in view of the analysis of the methods described at [1.0].

b. [20.1] *wherein determining a content of the at least one data packet comprises determining at least one data type of the at least one data packet of the communication in accordance with characteristics of the at least one data packet, and*

222. Claim limitation [20.1] is substantially the same as [2.1] and would have been obvious for the reasons discussed above.

c. [20.2] *wherein determining whether to modify the at least one data packet is based on the determined data type.*

223. Claim limitation [20.2] is substantially the same as [2.2] and would have been obvious for the reasons discussed above.

21. Claim 21

- a. **[21.0] *The system of claim 20, wherein the preference list database comprises a list of forbidden and/or allowed data packet types for sharing during communication.***

224. Claim limitation [21.0] is substantially identical to [3.0] and [13.0] and would have been obvious for the reasons discussed above.

22. Claim 22

- a. **[22.0] *The system of claim 19,***

225. The “*system*” of [22.0] would have been obvious for the reasons discussed at [19.0].

- b. **[22.1] *wherein, the processor is further configured to determine at least one communication data pattern in accordance with a behavior range of the data packet, and***

226. Claim limitation [22.1] is substantially similar to [4.1], except [22.1] recites that the “*processor is further configured to determine....*” The processor instructing the remote server to determine the “*communication data pattern*” is obvious for the reasons discussed at [19.5]-[19.6]. Therefore, [22.1] would have been obvious for the reasons discussed above at [4.1].

- c. **[22.2] *wherein the content of the at least one data packet is not decrypted by the remote server,***

227. Claim limitation [22.2] is substantially the same as [4.2] and would have been obvious for the reasons discussed above.

- d. **[22.3] *and the determination of the at least one communication data pattern is performed by the remote***

server in real time during a communication session between the remote server and the user's device; and

228. Claim limitation [22.3] is substantially the same as [4.3] and would have been obvious for the reasons discussed above.

e. *[22.4] wherein the determining whether to modify the at least one data packet, by the remote server, is further based on a comparison of the determined data pattern and the preference list,*

229. Claim limitation [22.4] is substantially the same as [4.4] and would have been obvious for the reasons discussed above.

f. *[22.5] whether to modify the at least one data packet, and*

230. Claim limitation [22.5] is the same as [4.5] and would have been obvious for the reasons discussed above.

g. *[22.6] if so, modifying the at least one data packet; and*

231. Claim limitation [22.6] is substantially the same as [4.6] and would have been obvious for the reasons discussed above.

h. *[22.7] wherein the preference list includes a list of forbidden and/or allowed data packet types and communication data patterns for sharing during communication.*

232. Claim limitation [22.7] is substantially the same as [4.7] and would have been obvious for the reasons discussed above.

23. Claim 23

a. **[23.0] The system of claim 20, wherein the data type of the at least one data packet is determined based at least**

in part on metadata of at least one data packet of the communication.

233. Dependent claim 23 is substantially identical to claim 6 and would have been obvious for the reasons discussed above.

24. Claim 24

a. [24.0] *The system of claim 22,*

234. The “system” of [24.0] would have been obvious for the reasons discussed at [19.0].

b. [24.1] *wherein the at least one communication data pattern is determined in accordance with a behavior range of the data packet,*

235. Claim limitation [24.1] recites substantially the same features as [7.1] and would have been obvious for the reasons discussed above at [7.1].

c. [24.2] *wherein the behavior range is selected from a group consisting of: data packet frequency, data packet size, data packet speed, data packet count, data packet ratio compared to other data type flow, data packet repetition, and data packet order.*

236. Claim limitation [24.2] recites substantially the same features as [7.2], except it refers to the “behavior range” of the data packet as in [4.1]. The list of potential “behavior range[s]” listed here in limitation [24.2], however, is the same list of potential “communication behavior[s]” listed in limitation [7.2]. Thus, the claims themselves provide that any of the enumerated items is both a “behavior

range” and a “*communication behavior*.” Therefore, [24.2] would have been obvious for the reasons discussed above at [7.2].

25. Claim 25

- a. [25.0] *The system of claim 19, wherein the modification of the communication is selected from the group consisting of*

237. The “*system*” of [25.0] is substantially similar to claim [5.0] and would have been obvious for the reasons discussed above.

- b. [25.1] *blocking the at least one data packet that is not compatible with the preference list, data nullification, data randomization, content modification, change of encoding, change of file template, change of header, change of footer, addition of a predetermined data packet and encryption.*

238. Claim limitation [25.1] is substantially the same as [5.1] and [14.0], and would have been obvious for the reasons discussed above.

26. Claim 26

- a. [26.0] *The system of claim 19, wherein the remote server is configured to maintain communication between the remote server and the user's device by sharing of the modified communication.*

239. Claim limitation [26.0] is substantially identical to [8.0] and [16.0], and would have been obvious for the reasons discussed above.

27. Claim 27

- a. [27.0] *The system of claim 20, wherein the data type is determined based on data packet characteristics selected from a group consisting of: data packet header, data*

packet footer, version number, IP (Internet Protocol) address, HTTPS (HyperText Transfer Protocol Secure), file extension, encryption method, encoding method, keywords, driver, and communication protocol.

240. Dependent claim 27 is substantially identical to claim 9 and would have been obvious for the reasons discussed above.

28. Claim 28

- a. [28.0] *The system of claim 19, wherein the remote server is a gatekeeper server.*

241. Dependent claim 28 is substantially identical to claim 10 and would have been obvious for the reasons discussed above.

29. Claim 29

- a. [29.0] *The system of claim 20, wherein the processor is configured to issue an alert to the user device based on the comparison of the determined data type and the preference list.*

242. Dependent claim 29 is substantially identical to claim 11 and would have been obvious for the reasons discussed above.

30. Claim 30

- a. [30.0] *The system of claim 22,*

243. The “system” of claim 22 would have been obvious for the reasons discussed at [19.0].

- b. [30.1] *wherein the preference list includes a list of forbidden and/or allowed data packet types and*

communication data patterns for sharing during communication, and

244. Claim limitation [30.1] is the same as [3.0] and [4.7] and would have been obvious for the reasons discussed above.

- c. ***[30.2] wherein, if the at least one data type of the at least one data packet is included in the list of forbidden and/or allowed data packets, the processor is configured to determine whether to modify the at least one data packet***

245. Claims 19 and 22 do not provide antecedent basis for “*the at least one data type of the at least one data packet.*” For the purposes of interpretation, this limitation will be construed as referring to “*content*” as synonymous with “*data type*” as per the ‘249 patent’s own practice (*see supra* [2.1]). Therefore, the claim limitation will be analyzed here as if it referred to the “*content of at least one data packet*” introduced in [19.7]. Claim 22 does not provide antecedent basis for “*the list of forbidden and/or allowed data packets.*” Therefore, the claim limitation will be analyzed here as if it referred to the “*list of forbidden and/or allowed data packet types*” introduced in [30.1].

246. As discussed in the analysis of [1.6], [2.2], and [4.4], the prior art renders obvious determining whether a packet should be modified. Because Burns discloses whether a packet should be modified or dropped based on the packet’s contents, it would have been obvious that Burns’s IDS would modify or drop a

packet if the data packet was “*included in the list of forbidden and/or allowed data packet [type]s.*”

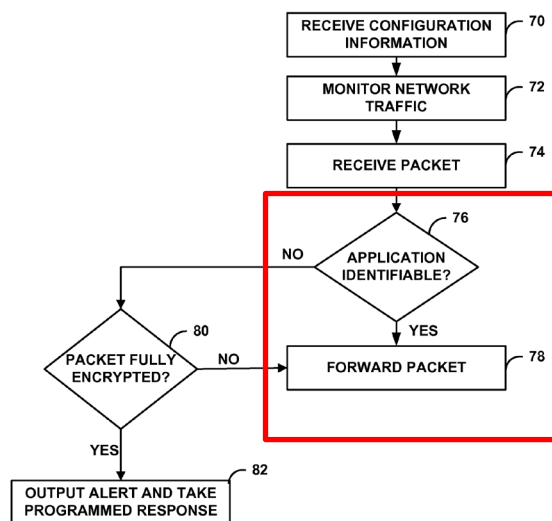
- d. [30.3] *based on the determined data type and the preference list without performing a comparison of a determined data pattern and the preference list.*

247. Claims 19 and 22 do not provide antecedent basis for “*the determined data type.*” For the purposes of interpretation, this limitation will be construed here as referring to “*content*” as synonymous with “*data type*” as per the ’249 patent’s own practice (*see supra* [2.1]). Therefore, the limitation will be analyzed as if it referred to “*the determined content*” of claim limitations [19.7] and [19.10].

248. As addressed previously in the discussion of [1.6], [2.2], and [4.4], Burns renders obvious determining whether to modify a packet based on a determined “*content*” or “*data type.*” Burns describes that a packet may be forwarded without determining if the packet should be modified when the associated application (*data type*) is identifiable:

When application identification module 51 is able to determine the application associated with the communication session ("YES" branch of 76), application identification module 51 forwards the packet (78).

Burns, 10:46-49; *see also* Fig. 4.



Burns, Fig. 4 (annotated)

249. Burns describes that the administrator’s configuration of the IDS can be used to override whether a determination to modify based on repeated values (*data pattern*) is made. Burns, Fig.7A (checkbox 310C), 18:1-7, 18:30-44 (“IDS 20 may utilize a repeating values filter.”) Similarly, the administrator can disable whether the system should detect if an encryption key has been exchanged. Burns, Fig.7A (checkbox 342B), 19:22-42.

250. Therefore, Burns describes scenarios in which the IDS determines whether to modify a packet based solely on the packet’s “*data type*,” without considering repeated values or encryption key exchange (*data pattern*). Burns therefore renders obvious determining “*whether to modify the at least one data*

packet based on the determined data type and the preference list without performing a comparison of a determined data pattern and the preference list.”

IX. CONCLUSION

251. I declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true, and that these statements were made with knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under section 1001 of Title 18 of the United States Code.

Dated: 4/9/2025

Nader F. Mir
Nader F. Mir, Ph.D.