Filed on behalf of: GoSecure, Inc.

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

CROWDSTRIKE, INC.,
CrowdStrike,

v.

GOSECURE, INC.,
Patent Owner.

Case IPR2025-00070
Patent 9,954,872

DECLARATION OF ANGELOS STAVROU
IN SUPPORT OF
PATENT OWNER'S RESPONSE

GoSecure, Inc.
Exhibit 2031
CrowdStrike v. GoSecure
IPR2025-00070

Table of Contents

I. Introduction and Qualifications 3

II. Summary of Materials Reviewed and Considered 5

III. Legal Standards 7

IV. Level of Skill in the Art 9

V. Overview of the '872 Patent 10

VI. Overview of the Asserted Prior Art References 22

A. Capalik (Ex.1004) 22
B. Goel (Ex.1005) 25
C. King (Ex.1006) 25
D. Pike (Ex.1007) 26

VII. Claim Construction 27

A. "Computer System" 27

VIII. The Claims of the '872 Patent Would Not Have Been Obvious in View of CrowdStrike's Prior Art 36

A. Ground 1: Capalik Does Not Render Any of Claims 1-6, 12, 15, or 20-21 Obvious 36
1. Claim 1 36
2. Claim 2 43
3. Claims 3-6, 12, 15, and 20-21 49
B. Ground 2: Capalik and Goel Do Not Render Any of Claims 1-6, 12, 14, 15, 20, or 21 Obvious 49
1. A POSITA Would Not Have Been Motivated to Combine Capalik and Goel 49
2. Claims 1-6, 12, 14, 15, and 20-21 53
C. Ground 3: Capalik, Goel, and King Do Not Render Any of Claims 7-11, 14, 16, 18, or 19 Obvious 54
1. Claim 8 54
2. Claim 16 55
3. Claim 19 58
4. Claims 7, 9-11, 14, 16, and 18 59

D. Ground 4: Capalik, Goel, and Pike Do Not Render Claim 13 Obvious 59
E. Ground 5: Capalik, Goel, and Farley Do Not Render Claim 17 Obvious 60

IX. CrowdStrike's Materials From Around 2013 Show That Its Product At That Time Practiced At Least Claims 1 And 2 The '872 Patent 61

A. 1[pre]: "A computer implemented method of identifying unauthorized activities on a first computer system attached to a computer network, wherein the first computer system comprises one or more processors and memory" 61
B. 1[A]: "monitoring activity on the first computer system" 62
C. 1[B]: "identifying a plurality of activities being performed at the first computer system, wherein each of the activities includes an activity source, an activity target, and an association between the activity source and the activity target" 62
D. 1[C]: "storing in the memory a data structure that identifies the activity sources, the activity targets, and the associations for the plurality of activities" 63
E. 1[D]: "transmitting to one or more computer systems other than the first computer system information identifying one or more of the activity sources, the activity targets, and the associations for preventing future attacks, to the one or more computer systems, associated with the one or more of the activity sources, the activity targets, and the associations" ... 63
F. Claim 2: "The method of claim 1, wherein the monitoring further comprises monitoring all activity on the first computer system." 64

XI. Conclusion 66

I. Introduction and Qualifications

1. I, Angelos Stavrou, submit this declaration on behalf of GoSecure, Inc. ("GoSecure" or "Patent Owner") to provide my opinions concerning the validity of the claims of U.S. Patent No. 9,954,872 (the "'872 patent," Ex.1001) in support of GoSecure's Patent Owner Response in IPR2025-00070.

2. I am more than eighteen years of age and am competent to testify on the matters set forth in this Declaration. I am a citizen of the United States and currently reside in the Commonwealth of Virginia.

3. In formulating my opinions in this matter, I have relied upon my education, training, and experience in the relevant technical fields, as well as the materials cited in Section II below. My qualifications are set forth in detail in my curriculum vitae (attached as Ex.2032). A brief summary of my relevant qualifications is as follows.

4. I received my B.Sc. (Physics, with distinction) from the University of Patras (Greece, 1997); my M.Sc. (Theoretical Computer Science) from the National University of Athens (2001); and my M.S., M.Phil., and Ph.D. in Computer and Systems Engineering from Columbia University (2007). From 2012 to 2020 I was a faculty member in the Computer Science department at George Mason University. I am a founding Professor at the Virginia Tech Innovation Campus and the Entrepreneurship activities lead. I am also a member of the

Bradley Department of Electrical & Computer Engineering at Virginia Tech. From 2011 to 2020, I was an Associate Researcher in the Computer Security Division at the National Institute of Standards and Technology ("NIST"). In addition, I co-founded and served as CEO of Kryptowire, Inc. (now Quokka, Inc.), a mobile security startup. I am also the founder of A2Labs LLC, Aether Argus, and Impedyme Inc. I have served as a principal investigator on research awards from NSF, DARPA, IARPA, DHS, AFOSR, ARO, ARL, and ONR.

5. Furthermore, I am an inventor of nineteen issued patents and several pending patent applications. I am an Associate Editor of IEEE Transactions on Computers, IEEE Security & Privacy, and IEEE Internet Computing magazine and part of the IEEE Blockchain initiative. I am a senior member of the ACM, USENIX, and IEEE. My research and professional experience focus on large-scale systems security, intrusion detection, cyber-defense techniques, and related areas, which are highly relevant to the subject matter of the '872 patent.

6. I received the GMU Department of Computer Science Outstanding Research Award in 2010, 2016, and 2018. Also, I was awarded the 2012 George Mason Emerging Researcher, Scholar, Creator Award, a university-wide award. In 2013, I received the IEEE Reliability Society Engineer of the Year award, and in IEEE Reliability Society Lifetime Award 2024. My team at Kryptowire was awarded the DHS Cyber Security Division's "Significant Government Impact

Award" in 2017 and the "Bang for the Buck Award" in 2019. Currently, I am the primary Principal Investigator (PI) in two DARPA awards focusing on Cyber Security for cloud and wireless systems, namely the DARPA OPS-5G and CASTLE efforts.

II. Summary of Materials Reviewed and Considered

7. All of the opinions contained in this Declaration are based on the documents I reviewed and my knowledge and professional judgment. In forming the opinions expressed in this Declaration, while drawing on my experience in the field of cybersecurity, I reviewed the following documents:

Ex.	Description
1001	U.S. Patent No. 9,954,872 (the "'872 patent")
1002	File History for U.S. Patent 9,954,872 (the "'872 File History")
1003	Markus Jakobsson Declaration
1004	U.S. Application Publication No. 2008/0016570 to Capalik ("Capalik")
1005	Ashvin Goel et al. "Forensix: A Robust, High-Performance Reconstruction System," Proceedings of the 25th International Conference on Distributed Computing Systems Workshops, 2005 ("Goel")
1006	Samuel T. King, "Analyzing Intrusions Using Operating System Level Information Flow," Dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in The University of Michigan, 2006 ("King")
1007	U.S. Patent No. 8,819,822 to Pike et al. ("Pike")
1008	U.S. Patent No. 7,089,428 to Farley et al. ("Farley")

Ex.	Description
2015	Wayback Machine archive of *CrowdStrike*, "Technology," archived July 7, 2013, https://web.archive.org/web/20130707085243/http://www.crowdstrike.com/technology/index.html (last accessed June 20, 2025)
2016	Wayback Machine archive of *CrowdStrike*, "Next-Generation Threat Protection," archived November 17, 2013, https://web.archive.org/web/20131127114229/http://www.crowdstrike.com/falcon-host/index.html (last accessed June 16, 2025)
2017	*YouTube*, "CrowdStrike Falcon Use Case: True Endpoint Visibility," Oct. 13, 2015, https://www.youtube.com/watch?v=pHxb6EyjhPw (last accessed June 20, 2025)
2033	Markus Jakobsson Deposition Transcript taken on November 11, 2025
2038	US 2012/0331553 (Aziz)
2039	US 2006/0136720 (Armstrong)
2040	US 7,356,845 (Jakobsson)
2041	Bart Lenaerts-Bergmans, "What is a Backdoor Attack?" July 17, 2023, available at: https://www.crowdstrike.com/en-us/cybersecurity-101/cyberattacks/backdoor-attack/ (last accessed November 26, 2025)
2042	Arfan Sharif, "What is CVE? Common Vulnerabilities & Exposures," December 7, 2022, available at: https://www.crowdstrike.com/en-us/cybersecurity-101/exposure-management/common-vulnerabilities-and-exposures-cve/ (last accessed November 26, 2025)
2043	Bart Lenaerts-Bergmans, "What is Hacktivism?" July 7, 2022, available at: https://www.crowdstrike.com/en-us/cybersecurity-101/threat-intelligence/hacktivism/ (last accessed November 26, 2025)
2044	Thuy Nguyen, "What are Tailgating and Piggybacking Attacks?" June 25, 2025, available at: https://www.crowdstrike.com/en-us/cybersecurity-101/cyberattacks/tailgating-piggybacking-attack/ (last accessed November 26, 2025)

Ex.	Description
2045	Cybersecurity 101: The Fundamentals of Cybersecurity, available at: https://www.crowdstrike.com/en-us/cybersecurity-101/ (last accessed November 26, 2025)
2046	David Rojas and Mark Robinson, "Hiding in Plain Sight: Remediating "Hidden" Malware with Real Time Response," December 16, 2020, available at: https://www.crowdstrike.com/en-us/blog/how-to-remediate-hidden-malware-real-time-response/ (last accessed November 26, 2025)
2047	Kurt Baker, "Bootkit: Definition, Prevention, and Removal," January 4, 2023, available at: https://www.crowdstrike.com/en-us/cybersecurity-101/malware/bootkits/ (last accessed November 26, 2025)
2048	M. Churchill, "Parsing Sysmon Events for IR Indicators," February 23, 2015, available at: https://www.crowdstrike.com/en-us/blog/sysmon-2/ (last accessed November 26, 2025)
2049	Microsoft Computer Dictionary, p. 105 (4th ed. 1999)
2050	Heng Yin, Zhenkai Liang and Dawn Song, "HookFinder: Identifying and Understanding Malware Hooking Behaviors," October 17, 2007, CyLab, CMU
2051	Video: Introduction to CrowdStrike, available at: https://www.youtube.com/watch?v=n2LkiwbAYCI (last accessed December 11, 2025)

8. My opinions are additionally guided by my appreciation of how a person of ordinary skill in the art would have understood the claims of the '872 patent at the time of the alleged invention, which I have been asked to assume is June 24, 2010.

III. Legal Standards

9. I have been instructed to assume that the effective priority (filing) date for the '872 patent is June 24, 2010, and my analysis of the prior art is conducted

with respect to what would have been known by a person of ordinary skill on that date. I am not an attorney, but I have been informed by counsel of the legal standards governing patent validity and obviousness in inter partes review, and I have applied those standards in formulating my opinions.

10. I understand that the patent claims define the scope of the patent owner's rights. Claim terms are generally given their ordinary and customary meaning as understood by a person of ordinary skill in the art at the time of invention.

11. Claim construction is conducted primarily by reference to the intrinsic evidence (claim language, the specification, and the prosecution history), with the patent specification being the best guide to the meaning of disputed terms. Extrinsic evidence (such as technical dictionaries or expert testimony) may be considered if needed, but the intrinsic record is accorded greater weight.

12. I understand that under 35 U.S.C. § 103, a claim is unpatentable as obvious if the differences between the claimed invention and the prior art would have been such that the claimed invention would have been obvious to a person of ordinary skill in the art at the time the invention was made. In assessing obviousness, it is appropriate to consider the scope and content of the prior art, the differences between the prior art and the claimed invention, and the level of ordinary skill in the art.

13. I also understand that secondary considerations (sometimes called objective indicia of non-obviousness) may be relevant. These include evidence of commercial success, long-felt but unsolved needs, failure of others, copying by others, unexpected results, and similar factors, all of which may provide evidence of non-obviousness if supported by the facts. The burden of proving that a claim is unpatentable in an IPR is on the Petitioner (here, CrowdStrike) by a preponderance of the evidence.

14. I understand that a claim can be found obvious if a skilled artisan would have been motivated to combine elements or teachings from multiple prior art references, and that combination yields no more than what the skilled artisan would have expected from the ordinary use of those elements. In particular, I understand that a claim cannot be held patentable if the claimed subject matter is nothing more than the predictable use of prior art elements according to their established functions. In determining obviousness, I understand that one must take into account whether the claimed invention provides more than a predictable variation of what was already known.

IV. Level of Skill in the Art

15. I understand that the hypothetical person of ordinary skill in the art is defined by factors such as education, experience, and knowledge of relevant problems in the field. I have reviewed the definition of a person of ordinary skill

in the art used by Dr. Jakobsson in his declaration (Ex. 1003). A person of ordinary skill in the field of the '872 patent (i.e., computer security and intrusion detection) as of June 24, 2010 would have had at least a bachelor's degree in computer science or computer engineering (or equivalent) and roughly two years of relevant industry or research experience, with practical knowledge of software security analysis and dynamic malware analysis. Dr. Jakobsson also states that such a person would have possessed "a working knowledge of software security analysis and dynamic malware analysis" and would have had the educational and practical background described above. I agree.

16. Under Dr. Jakobsson's definition, I personally qualified as a person of ordinary skill in the art as of June 24, 2010, by which point in time I had already earned my Ph.D. and had held faculty appointments. I am likewise familiar with the education, experience, and knowledge that such a person would have had. Accordingly, I have tailored all of my opinions to the perspective of that ordinary-skill individual: when I state that something would or would not have been obvious, I mean that it would or would not have been obvious to the person of ordinary skill described above as of June 24, 2010.

V. Overview of the '872 Patent

17. The '872 patent is titled "System and method for identifying unauthorized activities on a computer system using a data structure model." In a

section labeled "Technical Field," the '872 patent explains that the claimed invention "relates generally to systems and methods for protecting computer networks, including but not limited to systems and methods for analyzing malicious activities on a computer system in order to better protect the computer system from future malicious activity." '872 patent at 1:44-48.

18. The "Background" section of the '872 patent states the self-evident principle that computers that are "connected to computer networks (e.g., the Internet)" are "vulnerable to attacks." '872 patent at 1:52-57. At the time of the invention, "existing network security systems" included firewalls and intrusion detection/prevention systems (referred to as IDS or IPS, respectively). '872 patent at 1:58-65. Whereas firewalls prevented access to the network based on predetermined security rules, an IDS/IPS typically relied on a "library of malware fingerprints," which could be compared to "low-level IP data packets" to see if they matched. '872 patent at 1:58-2:8.[1]

19. These "existing network security systems" remained vulnerable to "zero-day attacks," particularly those that were not readily detectable at the

[1] A "low-level IP data packet" refers to a data packet that is transmitted using the Internet Protocol that includes a header and the data payload. The header portion contains routing information, such as a source address and/or destination address.

network boundary or that executed malicious activity within the host after bypassing perimeter defenses. '872 patent at 2:9-19. As the number of network devices skyrocketed and malware became more sophisticated, techniques for quickly and efficiently generating fingerprint data for novel attacks became more important. '872 patent at 2:20-24.

20. To solve these problems, the '872 patent shifted the focus of malware detection from the network perimeter to activity occurring within a given computer system. This shift, which is reflected throughout the specification and in the claims (*see, e.g.*, Claim 1, which recites "monitoring activity on the first computer system") represented a paradigm shift and serves as a key distinction between the '872 patent and the prior art referenced in the Petition, including Capalik (as explained further below).

21. The '872 patent discloses and claims techniques for identifying "unauthorized activities by monitoring and identifying associations between activity sources and activity targets without using pre-determined fingerprints" and, in turn, using the collected data to generate fingerprints of zero-day attacks. '872 patent at 3:11-16. Thus, rather than relying on detecting unauthorized access at the network level, the claimed system provides for monitoring activity occurring directly within the operating system environment, such as processes, files, users, threads, and network sockets. '872 patent at 4:15-

21, 6:35-43, 15:34-47. Importantly, this monitoring is not conditioned on detecting a network connection or intrusion event. Instead, the system continuously observes activity occurring inside the endpoint, enabling the identification of malicious behaviors undetected by prior systems. *See, e.g.*, '872 patent at 10:8-10 ("In some embodiments, the introspection module monitors all activity (e.g., system calls, library calls, kernel calls, etc.) on virtualized decoy operating system 112.").

22. Figure 3A illustrates an example in which a data packet stream 304 is detected at a socket 306 of virtual machine 113-1. '872 patent at 11:23-44. In the example of Figure 3A, the packet stream 304 is treated as unauthorized activity—e.g., virtual machine 113-1 may be configured as a decoy system, which does not typically send and receive data. *Id.* 11:29-36. Accordingly, any data received at socket 306 is assumed to be unauthorized (e.g., an attempt by a hacker to probe a security system). *Id.*

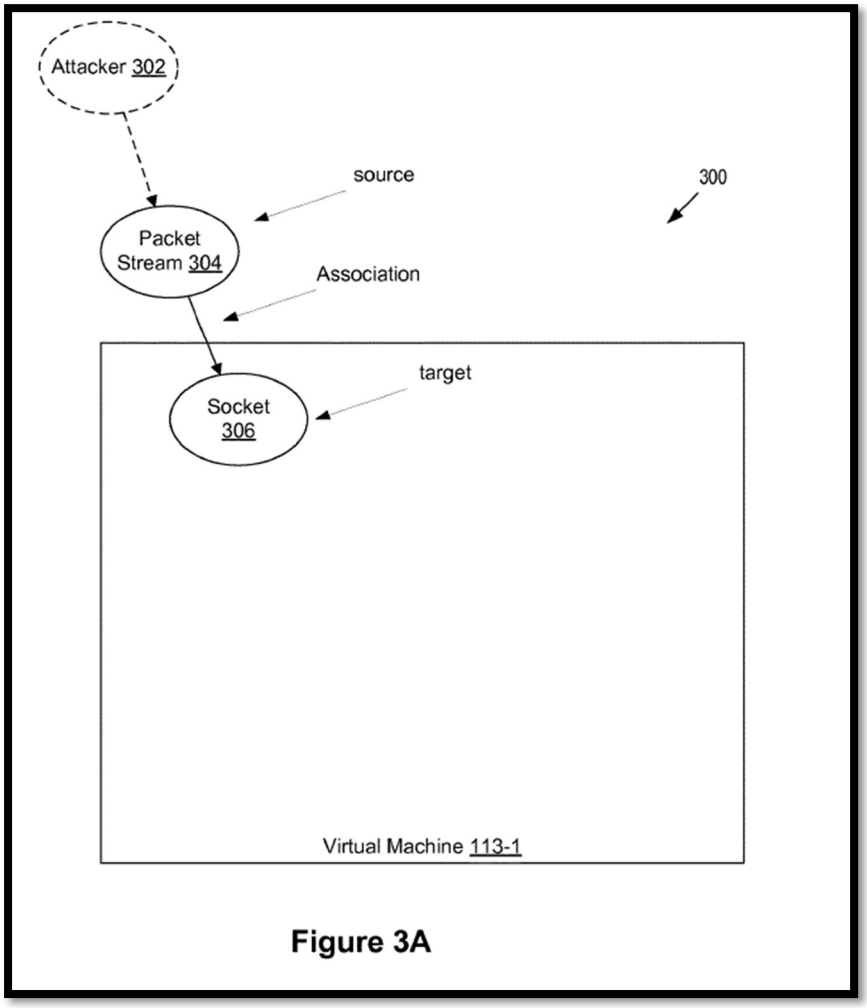
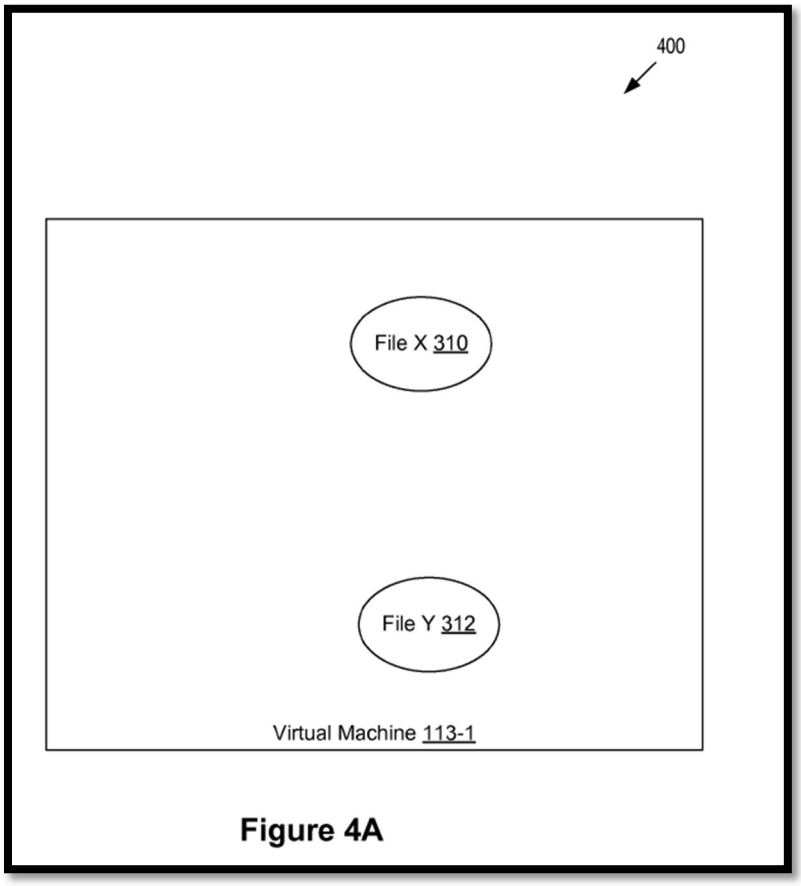


Figure 3A

23. "Once the packet stream 304 and the socket 306 are identified as affected by unauthorized activities, related unauthorized activities are identified," as illustrated in Figures 3B, 3C, and 3D. '872 patent at 11:45-55. "By analyzing these activities, the fingerprint of the unauthorized activities can be identified" and used to improve the security of the "protected network devices." '872 patent at 11:56-65.

24. But the techniques disclosed in the '872 patent are not restricted to generating fingerprints based on, for example, detecting unauthorized activity at a

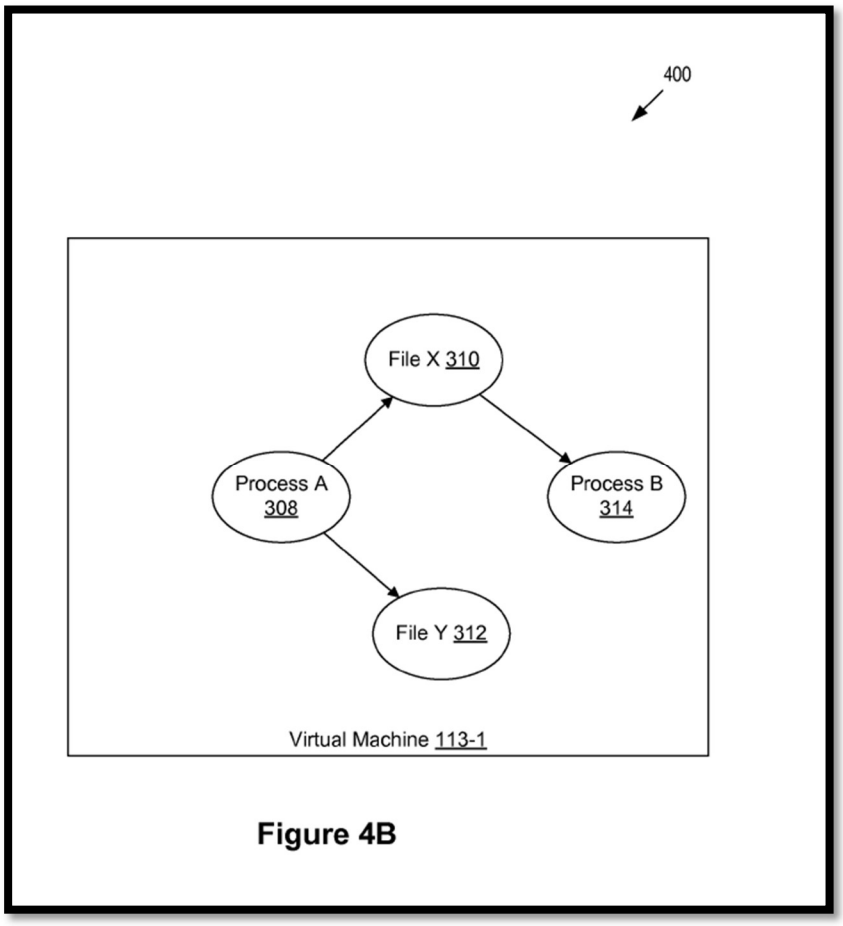
network socket. As another example, Figures 4A through 4D illustrate how the claimed techniques enable detection of unauthorized activity starting from activity inside the computer system.



'872 Patent Fig. 4A

25. In Figure 4A, the system detects unauthorized modifications to File X 310 and File Y 312 "by using checksum values," where a different checksum indicates that a corresponding file has been modified by an unauthorized process. '872 patent at 12:2-8. The techniques disclosed with respect to Figure 4A do not depend on the system having detected malware when it first attempted

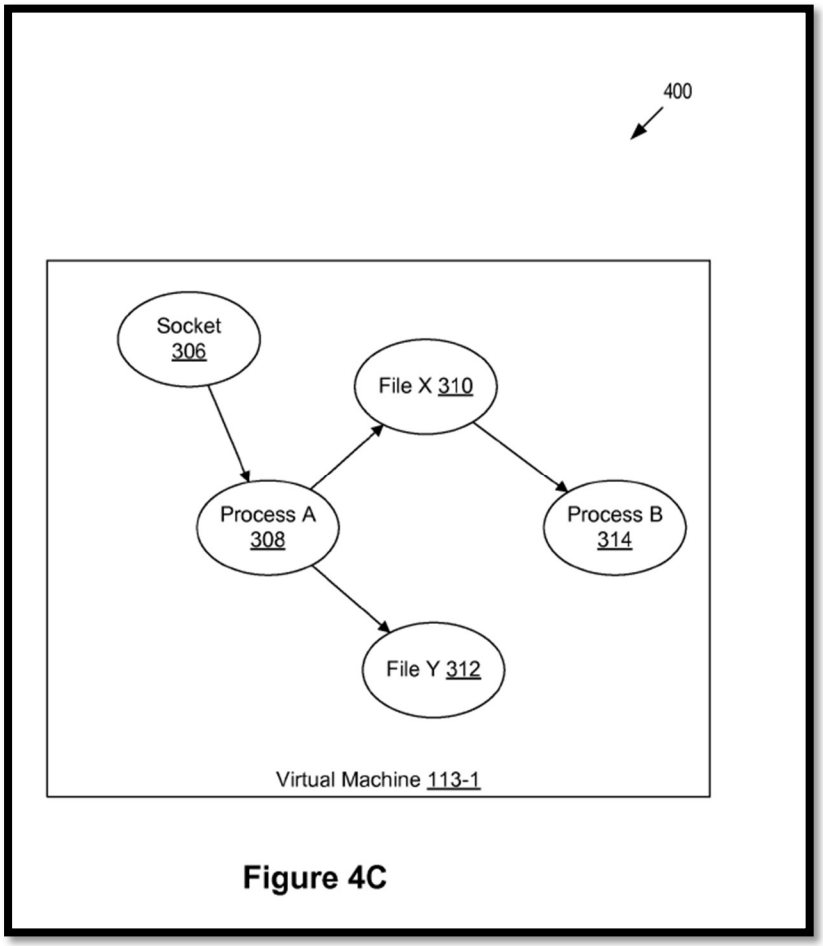
to access the computer system or network and they don't require a network event to begin with. Instead, the triggering event illustrated in Figure 4A is based exclusively on monitoring activity happening within the computer system (i.e., Virtual Machine 113-1).



'872 Patent Fig. 4B

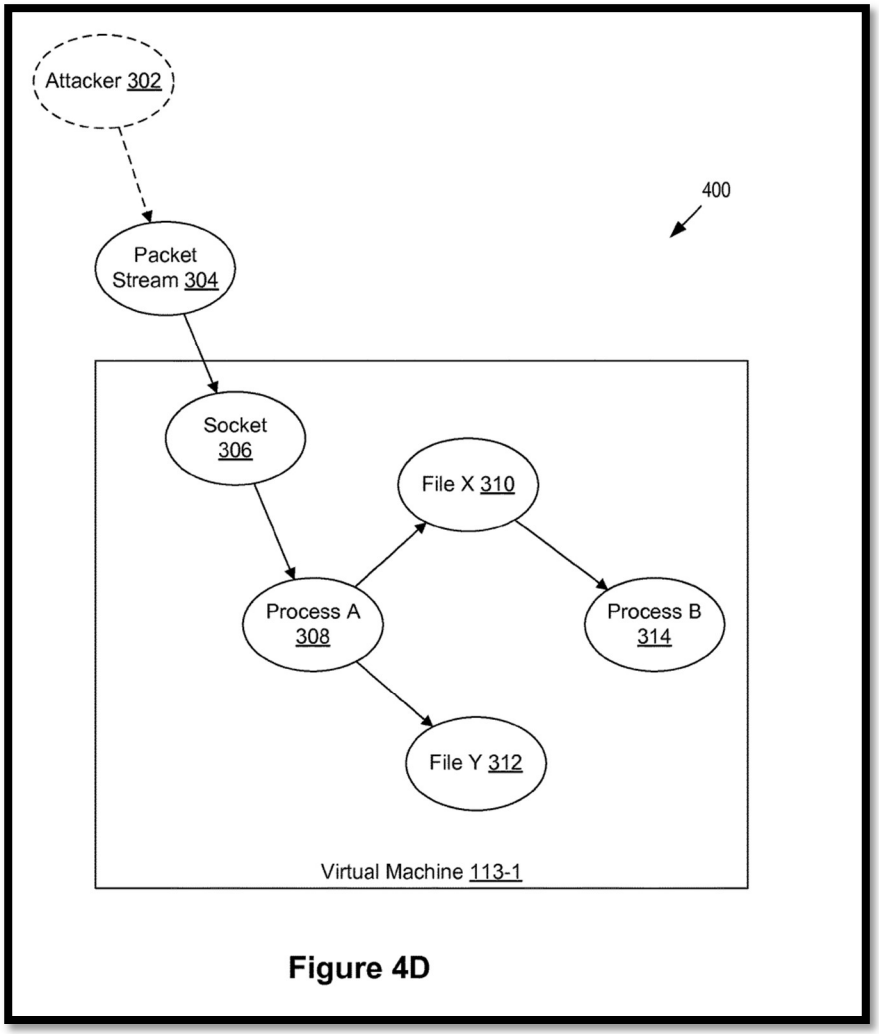
26. Figure 4B shows how the system traces the unauthorized activity to other components of the computer system. In this example, process A 308 accesses both file X 310 and file Y 312 and is therefore "treated as affected by unauthorized activities." '872 patent at 12:15-17. Similarly, because file X 310 initiates process

B 314, process B is also treated as affected by unauthorized activities. '872 patent at 12:17-19. This illustrates how the system propagates an indication of unauthorized activity from files to processes based on their interactions, thereby building a chain of potentially malicious behavior entirely within the computer system.



'872 Patent Fig. 4C

27. Figure 4C extends that causal chain to network sockets. As shown, data from socket 306 is directed to process A 308, and the socket is therefore treated as affected by unauthorized activities. '872 patent at 12:20–22.



'872 Patent Fig. 4D

28. Finally, in Figure 4D, packet stream 304 received over socket 306 is treated as affected by unauthorized activities, which may allow identification of attacker 302 who sent the packet stream, for example, by its IP address. '872 patent at 12:22-26. Only at this last stage does external network access come into view. Note that this packet stream was not identified as suspicious on its own and therefore would have evaded an IDS/IPS.

29. Prior art systems (such as Capalik) that only monitored for unauthorized access or intrusions at the network boundary would not have monitored for or detected internal changes to files X and Y, the initiation of process B, or the propagation of unauthorized behavior to process A. That represents a key technical distinction between the '872 patent and prior art approaches.

30. Figure 5B illustrates how a stored data structure (e.g., as illustrated in Figure 5A) can be used to generate a fingerprint that describes a behavioral pattern of unauthorized activities within the computer system, including activity sources, activity targets, and associations (along with other optional information like timestamps, association types, and an activity status). '872 patent at 13:64-14:4.

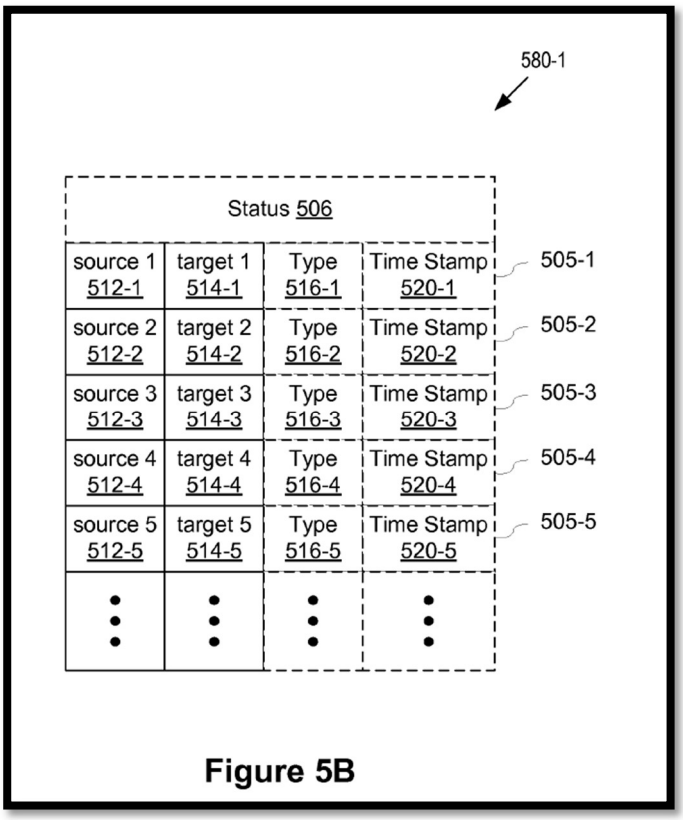


Figure 5B

'872 Patent Fig. 5B

31. A unifying feature of both examples discussed above (i.e., those discussed with respect to Figures 3A-3D and Figures 4A-4D) is the fact that the fingerprint is used to provide "detailed forensic analysis of the character and nature of the attack and its impact ***on the target operating system***," and thus can be used to generate triggers that deny would-be malicious activity on the computer system. '872 patent at 10:49-55 (emphasis added). For example, the fingerprint might be used to "prevent future attacks on the computer system or other computer systems by modifying the activity targets 514 (e.g., increasing privileges, changing the file/instruction name, changing the file/instruction structure, moving the file/instruction to a different location, blocking a particular port, encrypting the file, etc.) to protect the activity targets 514 or blocking the same or similar (e.g., at least partially matching) pattern or sequence of activities (e.g., if the fingerprint of the unauthorized activities starts with sending packets A, B, and C in order, blocking the receipt of the packet C after receiving the packets A and B)." '872 patent at 14:5-16.

32. Unlike IDS/IPS signatures that are essentially byte patterns or protocol rules for matching network packets, these fingerprints encode relationships between internal objects—files, processes, sockets, and so forth—and track how unauthorized behavior propagated among them. Because the '872

patent fingerprints capture activity sources, targets, and associations rather than just packet content, they provide a richer set of information regarding how an attack interacts with computer systems and are thus particularly valuable for detecting and responding to zero-day attacks that go beyond network signatures and can be applied at endpoint computing devices.

33. An important aspect of the invention is the transmission of the generated fingerprint to one or more other computer systems so that those systems can protect themselves against future attacks involving similar unauthorized activities. '872 patent at 14:5-16, 18:44-47. The claims expressly recite transmitting information "to one or more computer systems other than the first computer system" for "preventing future attacks" on those computer systems.

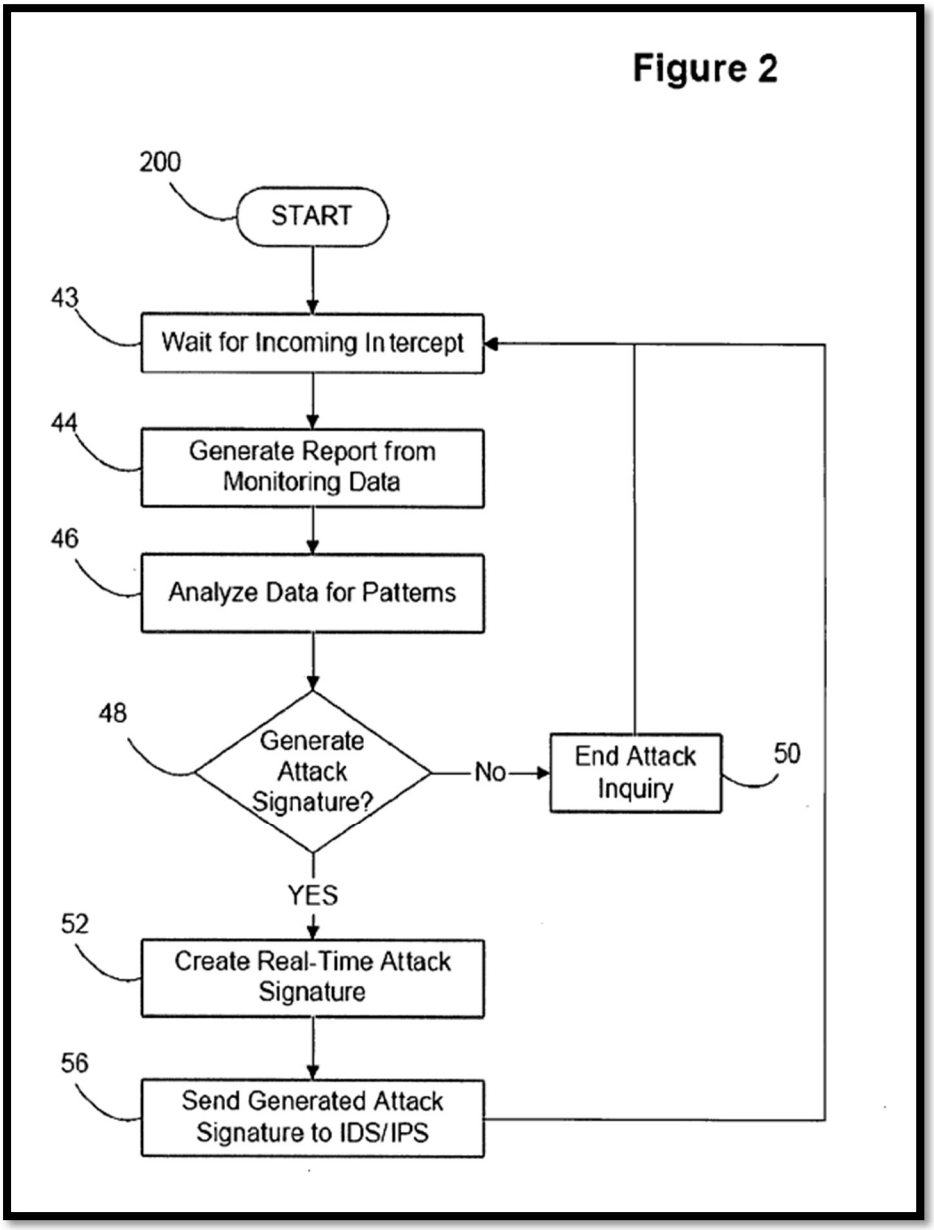
34. The patent distinguishes this claimed embodiment from the separate scenario in which information may be transmitted to an IDS/IPS system as an alternative. '872 patent at 18:47-51 (discussing how network device may transmit the fingerprint to the IDS/IPS in "some embodiments"), 10:45-49 ("It should be appreciated that although the signature of the attack may be used to generate an attack signature suitable for use in security applications (e.g., IPS, AV, etc.), the signature of the attack is distinct from the attack signature.").

VI. Overview of the Asserted Prior Art References

A. Capalik (Ex.1004)

35. U.S. Patent Application Publication No. 2008/0016570 to Capalik ("Capalik") is "directed to decoy network technology with automatic signature generation for intrusion detection and intrusion prevention systems." Capalik [0002]. As reflected in its title, Capalik is focused on analyzing "unauthorized intrusion into a computer network." Capalik Cover. In other words, Capalik is concerned with network-level intrusions and analyzes activity that occurs after an attacker attempts to access a decoy system over the network.

36. Figure 2 of Capalik (reproduced below) presents a flowchart in which the decoy waits for an incoming intercept at step 43, receives and records data once an attacker connects, and then decides whether to create an attack signature. Capalik [0055]. Specifically, Capalik discloses that the decoy system waits until "an attacker connects to a port, and begins interacting with decoy operating system 32." Capalik [0027]. Only when such a [network] connection is established does the system begin to monitor and capture information from the connection, including port numbers, data streams, keystrokes, file uploads, and other data transfers. Capalik [0027], [0049]. Thus, Capalik's monitoring is triggered by the detection of a network-level connection to a port and is focused on the data transferred through that socket.



Capalik, Fig. 2

37. When an attack signature is generated, Capalik does so by "analyzing all the data passed through the socket." Capalik [0013]. This includes analyzing the "attack payload including keystrokes or transferred bytes and any files uploaded to the client operating system of an ASCII or binary nature." Capalik

[0013]. Thus, Capalik generates signatures based on network activity that the separate IDS/IPS can monitor, as shown, for example, in Figure 4, below. This signature provides, for example, the "protocol" (e.g., TCP) for network data and "ServicePorts."



***Capalik*, Fig. 4**

38. After the attack signature is generated, Capalik describes sending that signature to an IDS/IPS signature library. Capalik [0036]-[0040], [0055], [0059]. The IDS/IPS then uses this signature when examining "low-level IP data packets" and can issue alerts when incoming packets match previously stored signatures. Capalik [0004].

B. Goel (Ex.1005)

39. Goel describes "a robust, high-precision reconstruction and analysis system for supporting the computer equivalent of 'TiVo.'" Goel at Abstract. As such, Goel is focused on the "capture, analysis, and reconstruction of system activities in order to determine, post-facto, how or whether a machine was compromised." Goel at 1 (Section I, Introduction).

40. Goel's system includes three components: (1) "an auditing module on the target system," (2) "a receiver daemon and a database injector running on the backend," and (3) "database queries and scripts that allow replay of system activities for forensic purposes." Goel at 4 (Section 5, Implementation). The auditing module "traps system calls and logs data in a kernel buffer." Goel at 4 (Section 5.1, Target System). The receiver daemon "reads data from the network and stores it to human-readable, tab-separated log files." Goel at 5 (Section 5.2, Backend System). Human analysts can then use the queries to interrogate the recorded data in order to support "post-facto analysis" of a crash. Goel at 5 (Section 5.3, Queries).

C. King (Ex.1006)

41. King is a thesis dissertation that discusses the use of "operating-system-level (OS-level) information-flow graphs to highlight the activities of an attacker." King at 11. King's system includes two basic components:

EventLogger, which "logs information-flow events in real time," and *GraphGen*, which "processes EventLogger logs to generate the information-flow graphs [used] to analyze intrusions." King at 12 (Chapter 3, Information-Flow Graphs). Due to the significant processing overhead involved in King's system, "GraphGen is typically run off-line, i.e. after the attack." King at 17 (Section 3.3, Dependency Graphs).

D. Pike (Ex.1007)

42. U.S. Patent No. 7,380,264 to Pike et al. ("Pike") is directed to "security techniques for identifying and (in some cases) interdicting information system intrusions." Pike at 1:8-11. Pike describes exploits that seek to introduce data into a target system "that can or will be interpreted by the target system in a way that facilitates the attack." Pike at 1:25-27. Pike provides an example of a common exploit called a buffer overflow attack where "an exploit causes vulnerable code to write data to memory in such a way that locations beyond the ostensible write target are updated." Pike at 1:27-31. "If those memory locations include the stack or a pointer used by privileged code, an attacker may successfully alter an execution path of privileged code" or otherwise "modify data upon which privileged code relies." Pike at 1:39-44. Accordingly, Pike is directed to techniques for "scanning an input string for subsequences contained therein and configuring the computational system to generate a fault (or other triggered event)

coincident with access to a memory location corresponding to one or more possible interpretations of data contained in the input string[.]" Pike at 2:8-15.

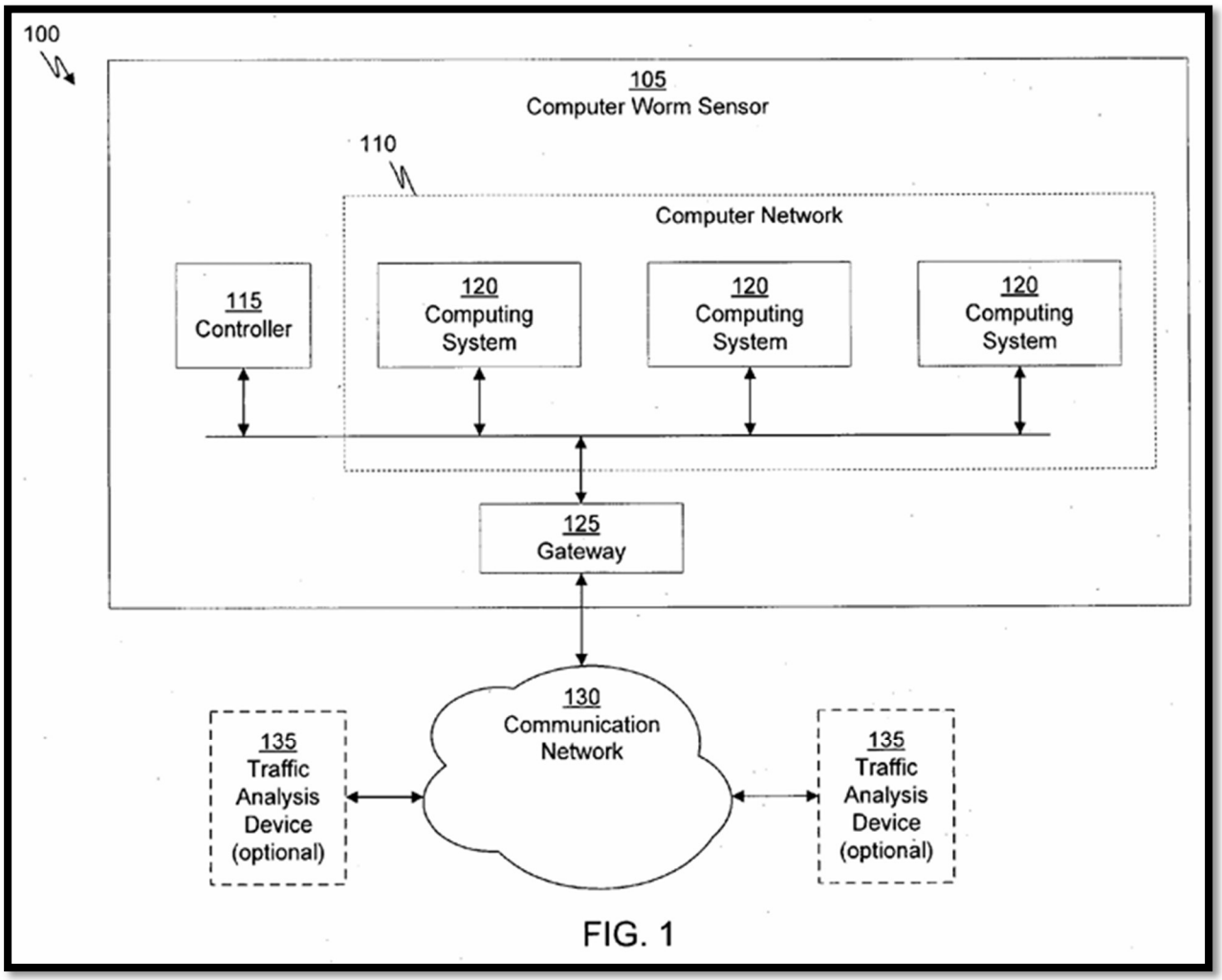
VII. Claim Construction

A. "Computer System"

43. I have been informed that Petitioner has proposed construing "computer system" to encompass a network. I also understand Dr. Jakobsson proposed that "computer system" means one or more computational devices. Ex.2033 53:5-7. I disagree. The ordinary and customary meaning of "computer system" in the context of the '872 patent is a single computing device. And although a computer system can encompass multiple components that work together as one unit, that does not mean it covers any collection or network of computers. As such, it does not include a "network." It also does not encompass any number of (seemingly distinct and unrelated) computational devices as Dr. Jakobsson asserts. *See* Ex.2033 57:9-13 (testifying that "every computer on earth" together form a "computer system").

44. I have been informed that, during prosecution of the '872 patent, the examiner cited prior art such as Aziz and Armstrong in evaluating the claims of the '872 patent. Ex.1002 76-77 (citing Aziz), 140 (citing Armstrong); '872 patent, Cover (including Aziz and Armstrong in list of references cited). In those references, "computing systems" or "computer systems" are described as

individual devices that are connected to a network, not as the network itself. For example, Aziz refers to "computing systems" as computing devices typically found in a computer network and illustrates multiple computing systems forming a network. Ex.2038 ("Aziz") [0048], Fig. 1.



***Aziz* Fig. 1**

45. Armstrong similarly describes "each computer system in the network or array" as a physically separate entity with its own network attachment, CPU(s), operating system, and so forth. Ex.2039 ("Armstrong") [0005].

46. These disclosures mirror the '872 patent's usage, where a "computer system" has processors and memory and is attached to a computer network, and where multiple such systems form or are connected to networks. '872 patent at 1:52-55, 14:61-67. Supporting that interpretation, the '872 patent consistently uses "**device**" and "**computer system**" interchangeably. For example, it provides some embodiments where components "are part of or stored within the decoy network **device**," and it offers other embodiments where they "are implemented using separate **computer systems**." '872 patent at 8:29-34; *see also* '872 patent at 18:52-53 ("one or more other **computer systems** (e.g., protected network **devices** 136) on the network"). As another example, the flowcharts of Figures 6A-6C represent "a method 600 of identifying unauthorized activities on a **computer system** (e.g., the virtual machine 113-1) attached to a computer network."[2] '872 patent 14:61-

[2] I note that in this passage and others that use the language "a computer system (e.g., the virtual machine 113-1)," the '872 patent is noting that the virtual machine is the component of the computer system where the described activity occurs and is not equating computer system and virtual machine. To the contrary, the '872 patent defines a VM as emulating computer systems. '872 patent 7:59-61. Dr. Jakobsson provided a similar definition. Ex.1003 ¶55 (citing Ex.1020 [0059]).

64. The method illustrated in these flowcharts is "performed at a decoy network **device** (e.g., 106 in FIG. 1B) having one or more processors and memory storing a virtual machine." '872 patent 14:64-67.

47. By contrast, the '872 patent never uses the term "computer system" to refer to a network, such as the three "networks" illustrated in Figure 1: communications network 148, decoy computer network 102, and protected computer network 104.

48. Extrinsic evidence is also consistent with this understanding. In one of his own earlier patents, Dr. Jakobsson referred to "computer system 100" as comprising a processor, memory, and network I/O for communication with data networks, and also referred to that same entity simply as "computer 100." Ex.2040 3:10-33, Figure 1. He described such "computer systems" as "well known in the art," indicating that he was using the term in its conventional sense, i.e., as a computing device with its own processor and memory. Ex.2040 3:11-12. He used the term interchangeably with "computer." Ex.2040 4:27.

49. Additionally, materials from CrowdStrike's own website consistently distinguish between "computer systems" and networks, using phrases such as

---

Regardless, this passage in no way supports a computer system encompassing a network.

"computer system or network" and "network or computer system," which again treats "computer system" as distinct from the network. Ex.2041 ("computer system *or* network"); Ex.2042 (same); Ex.2043 ("network *or* computer system"); Ex.2044 ("networks *or* computer systems"); Ex.2045 (same); Ex.2046 ("…connects an infected drive to a computer system"); Ex.2047 (equating "computer system" with a PC); Ex. 2048 (equating monitoring a computer system to endpoint monitoring)..

50. A well-known computer dictionary likewise defines "computer system" as "[t]he configuration that includes all functional components of a computer and its associated hardware." Ex.2049 (Microsoft Computer Dictionary) at 105. In my view, these sources reflect the ordinary and customary use of "computer system" as a single computing device, consistent with the '872 patent. Although some authors may define "computer system" to include a network, that is not the ordinary and customary use of the term in light of the intrinsic record.

51. In his deposition in this matter, Dr. Jakobsson analogized "computer system" to a ball of "dough" that can be shaped and cut in "many, many ways," including as a single computer, a network, an arbitrary subset of computers, or even "every computer on earth," depending on how one chooses to define the boundary in a given context. Dr. Jakobsson further indicated that, when mapping the claims to the prior art, he effectively looked at the claim language first and then

carved up the prior-art arrangement into whatever "computer systems" were needed for his mapping. Ex.2033 ("Jakobsson Dep.") at 57:3-59:14. For ease of reference, I have included this exchange below and have emphasized certain portions.

> **Q.** Okay. If you draw a box around both 504 and 502, one big box, does that change the operation of 502?
>
> **MR. HART:** Object to form.
>
> **THE WITNESS:**
>
> It's really a matter of identifying and naming things to understand what the claim is saying. ***So think of computer system as dough. You're baking a bread, right? So computer system could be all of the dough. It's, you know, every computer on earth, and the internet and all the routers of the internet.***
>
> You could also cut off a piece of your dough, you know, form a bun. That's the computer system that comprises all the devices in your home. Computer system is a little bit like this dough. You could cut it in different places and say identify this portion. This, of course, is dough. This is a computer system. It comprises one or more computational devices. Now, some of these cuts make it kind of hard to explain, you know. So, you know, you can cut it in many, many ways, but you choose to cut it in a way that you can express an opinion or a functionality. Now, claim one expresses a functionality where you say this piece of dough or one piece of dough, and you say another piece of dough. Now you're, I think, are asking about are there different ways of cutting this dough. Yes, there are.
>
> ***But if you want to map the claim to the situation, then the only relevant way of doing it is a way that you could say well, the claim we talk about this part doing that to this part. So I'm saying let's cut the dough in this way.*** The decoy computer network, of course, is a computer system. It comprises one or more computational devices.

> The protected computer network 504 is another computer system. It comprises one or more computational devices. Now I know from claim one that one computer system sends information to another computer system. In the context of Capalik in Figure 5, that would correspond to the decoy computer network 502 sends information to the computer system that corresponds to the upper portion of the figure.
>
> So the future protected computer network 504, which include the three protected network devices 536, the IDS IPS system 542, and also the 534 component, which is the part of – it's where the IDS IPS system stores information. Stores –
>
> **BY MR. PETERS:**
> **Q.** Okay.
>
> **A.** -- I should say.
>
> **Q.** Sorry. I think I cut you off, the last part. I'm sorry. You said stores some information. Is that what you said?
>
> **A.** Yeah. There might be other information stored by the IDS IPS system. The library of signatures is where some of the information is stored.

52. From a technical perspective, this approach does not align with how "computer system" is used in the '872 patent or in the technical literature. If "computer system" can be defined as arbitrarily as suggested—ranging from a single device to an entire distributed environment including all networks and appliances—then the term ceases to have a stable meaning. Under such a view, the same physical configuration could be labeled in one way to satisfy a claim limitation and labeled differently to avoid that limitation, depending solely on how

one chooses to "cut the dough," rather than on any underlying technical distinction. That is not how a person of skill in the art would use the term "computer system," and it is not how the '872 patent uses it.

53. The '872 patent's description of "distributed computer system 100" underscores this point. Figure 1 and the corresponding text describe distributed computer system 100 as including decoy computer network 102, protected computer network 104, communications network 148, firewall/router 198, and clients 101. '872 patent at Fig. 1, 4:5-9, 6:1-2. This is the entire collection of networks, computer systems, and appliances discussed in the patent, including the Internet. The patent never refers to distributed computer system 100 as a "computer system" in the sense used in the claims, and there is no disclosure of any system or component outside this distributed computer system let alone transmissions outside of it. As a result, if one were to treat the entire distributed computer system 100 as the claimed "computer system," there would be no "one or more computer systems other than the first computer system" to which information could be transmitted, which would make the "transmitting" limitation of claim 1 meaningless. '872 patent 2:44-45, 18:44-55. As annotated in Figure 1 below, all transmissions (orange arrows) are within the distributed computer system (purple box).

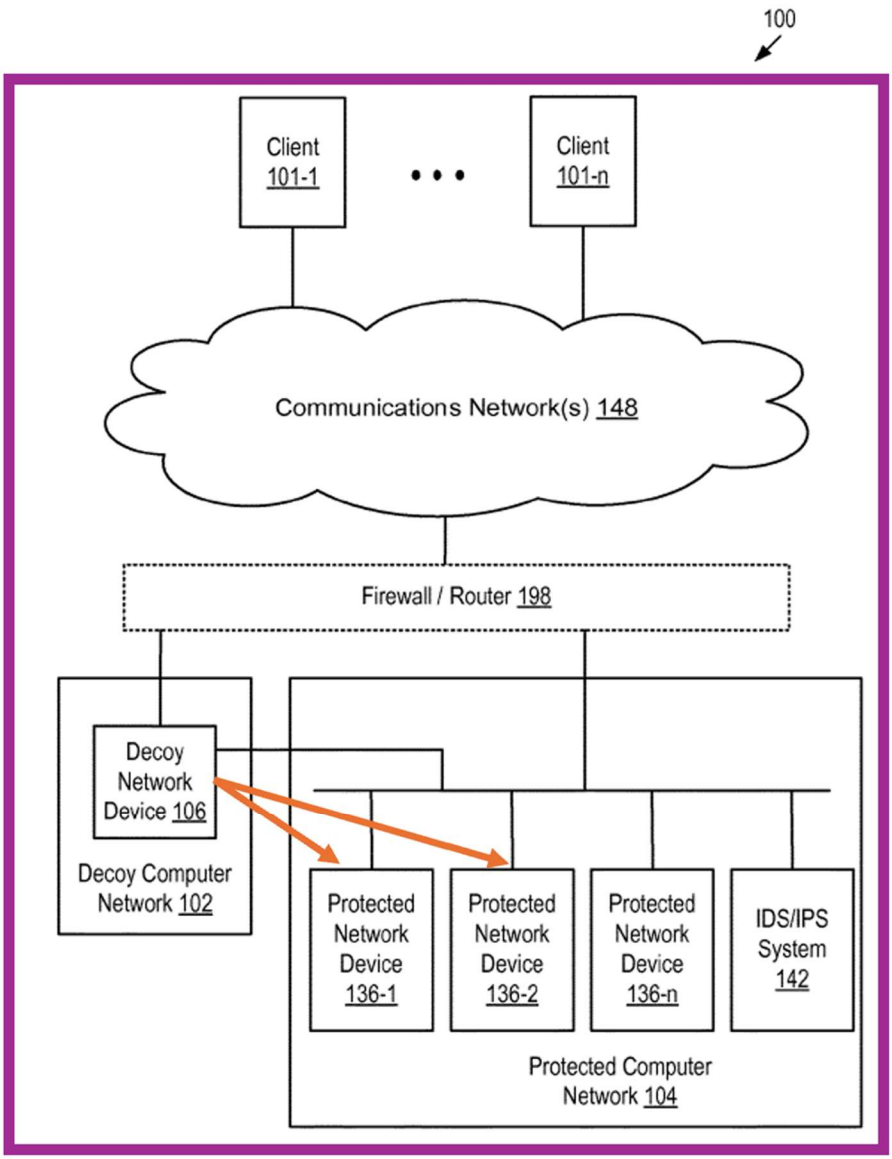


Figure 1

'872 Patent Fig. 1 (annotated)

54. This distinction illustrates why treating "computer system" as "dough" is not technically sensible in the context of the '872 patent. Under such a construction, the claim language about a "first computer system" attached to a

network and "one or more computer systems other than the first computer system" loses its meaning.

55. By contrast, interpreting "computer system" as a single computing device—consistent with the specification, prosecution history, and extrinsic evidence—results in a coherent and technically meaningful reading: a first computer system (such as a laptop) monitors its own internal activity, identifies unauthorized behavior, constructs a behavioral fingerprint, and transmits that fingerprint to one or more other computer systems on the network so that those receiving devices can protect themselves from similar future attacks.

VIII. The Claims of the '872 Patent Would Not Have Been Obvious in View of CrowdStrike's Prior Art

56. CrowdStrike and Dr. Jakobsson challenge the claims of the '872 patent based on five asserted obviousness combinations, each of which depends on the pairing of Capalik with the disclosure of Goel. For the reasons described below, it is my opinion that none of the asserted combinations discloses or suggests the limitations of the challenged claims.

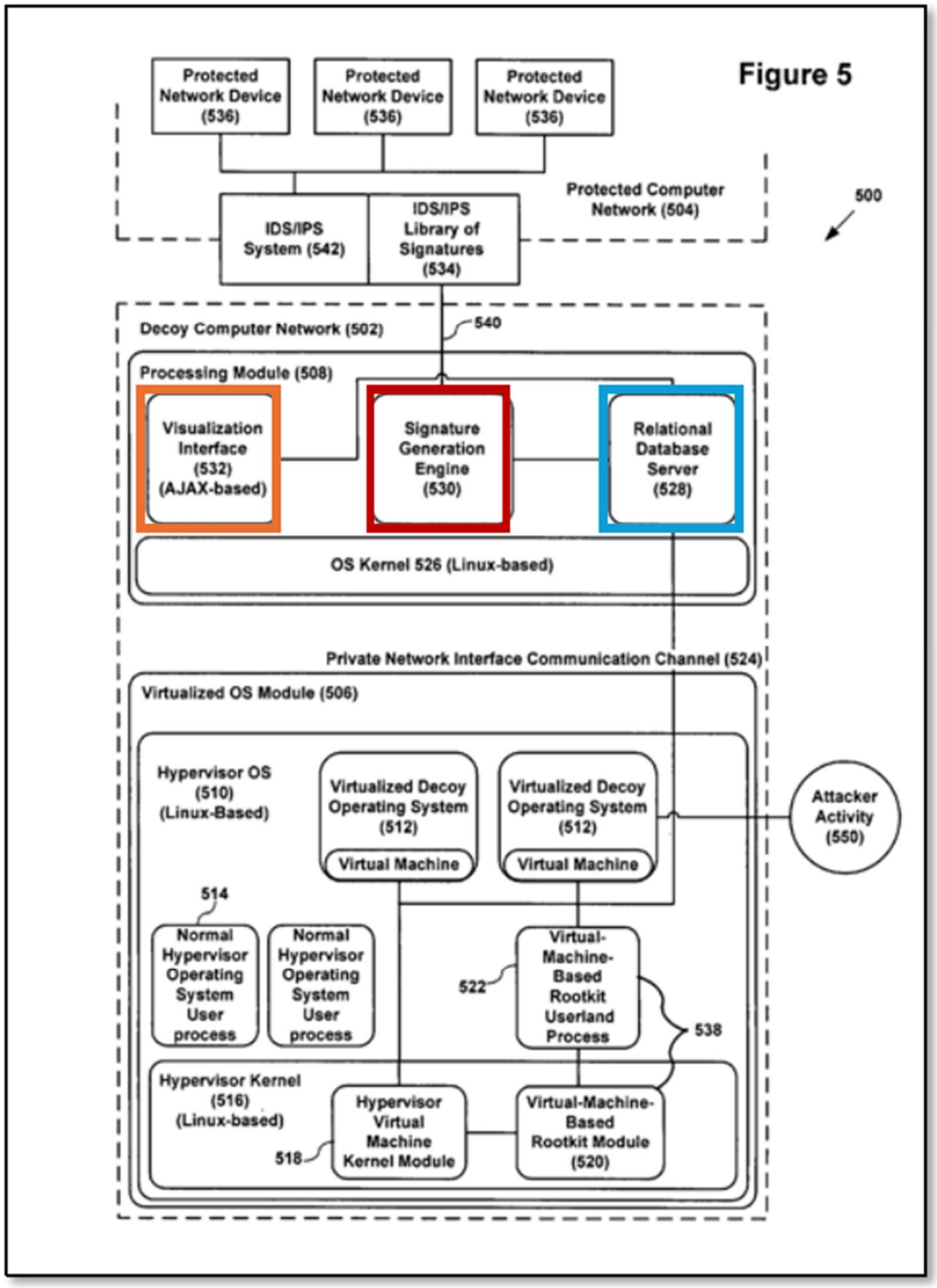
A. Ground 1: Capalik Does Not Render Any of Claims 1-6, 12, 15, or 20-21 Obvious

1. Claim 1

a. Limitation 1[C]: *Capalik* does not teach or suggest, or render obvious storing in memory a data structure that identifies the activity sources, the

activity targets, and the associations for the plurality of activities.

57. A person of ordinary skill in the art would not understand Capalik to disclose or render obvious any "data structure" or any organization of the information it stores. Capalik refers only to a relational database server that stores unspecified "forensic data," without describing the contents, fields, schema, relationships, or how any stored information is structured. Capalik [0052]. CrowdStrike relies on an annotated Figure 5 to highlight the relational database server (528):



***Capalik*, Fig. 5 (as annotated at Pet. 28)**

58. Nothing in Capalik identifies, for any activity, a source, a target, and an association between them.

59. A POSITA would not have understood Capalik to disclose or suggest or render obvious a structured representation identifying sources, targets, and associations simply because Capalik generates a signature for use by an IDS/IPS or a report for review by an administrator. Capalik's reports and signatures are generated from raw data without any discussion of organizing the data in any structure identifying associations between activity sources and targets, as claimed in the '872 patent. Capalik [0013], [0027], [0049]. Reports and signatures of this kind can be generated from unstructured or semi-structured data using intermediate processing, and neither the resulting reports nor the act of generating them retroactively imposes or implies a particular stored data structure that identifies activity sources, activity targets, or associations between them. The existence of an XML-formatted signature suited to packet inspection—focusing on protocol, service ports, severity, and pattern elements—further illustrates that Capalik aims to produce network-matching artifacts, not to maintain a stored model of intra-host activity relationships. Capalik, Fig. 4. And Capalik's database server can store raw text logs or in Capalik's case, raw forensic data. Capalik monitors only data that passes through a network socket after an attacker connects to the decoy, not internal activities such as file access, process interactions, or propagation across

system components. Capalik [0027], [0049]. That socket-centric perspective is consistent with capturing and later processing network-session events, not with building or storing a structured representation of activity relationships among files, processes, threads, data streams, and sockets. A POSITA would see nothing in Capalik to suggest or render obvious that it stores information in a structured form that identifies sources, targets, or associations among internal activities.

60. The '872 patent stands in sharp contrast, as it expressly describes and illustrates data structures, including Figures 5A-5C, that identify for each of a plurality of activities, their sources, their targets, and the associations between them. '872 patent 12:45-48, 13:65-66, 14:21-22, Figs. 3A-3C, 4A-4E, 5A-5C. In other words, for each activity or event, the data structure must identify a source, target, and the associations between the source and target. Capalik provides no comparable disclosure, CrowdStrike does not provide any rationale for adding such structure to Capalik's network-based signature-generation approach.

b. Limitation 1[D]: *Capalik* does not teach or suggest transmitting activity information to other computer systems to prevent future attacks on those systems.

61. CrowdStrike also relies exclusively on Capalik to satisfy the "transmitting" limitation of claim 1, which requires transmitting information identifying activity sources, targets, and associations to "one or more computer systems other than the first computer system" for preventing future attacks on those

computer systems. A person of ordinary skill in art would not have read Capalik to disclose or suggest this limitation.

62. As explained above, Capalik generates a signature by analyzing the data "passed through the socket" during an attacker's interaction with a decoy. Capalik [0013], [0055], [0059]. That signature is then transmitted to an IDS/IPS signature library, which uses it to examine low-level IP packets destined for protected network devices. Capalik [0004], [0059].

63. Capalik does not disclose transmitting information corresponding to internal activities on a computer system to other computer systems in order to prevent future attacks against those other computer systems, as required by claim 1. Under the proper construction of "computer system" and the "transmitting" limitation, Capalik plainly does not disclose this limitation. Even if Capalik's IDS/IPS is a "computer system" in the abstract, transmitting signatures to it would not satisfy the claim because the IDS/IPS is not itself the subject of future attacks. Instead, the IDS/IPS uses the signatures to protect ***other*** computer systems, which Dr. Jakobsson agreed with in his deposition. Jakobsson Dep. at 47:12-48:5 ("…So given my understanding that the IDS/IPS system 142 is connected to the protected network devices 136, which is supported by Figure 5, I do say that it's – ***the fingerprint is sent to the IDS/IPS to protect the other network devices***….").

64. To the extent CrowdStrike intended to argue that a POSITA would have understood the IDS to forward the signatures to the protected network devices, I disagree. Petition at 33-34. As an initial matter, I note that CrowdStrike has not cited to any passage in Capalik suggesting such transmission, nor any evidence that it happens. To the contrary, the signature is sent to and stored in the IDS/IPS signature library separate from the protected network devices. Capalik [0054]. Capalik's "attack signature" is initiated by network activity. In addition, it is derived from the network data that traversed the socket, such as keystrokes and uploaded files, and is intended to allow the IDS/IPS to match low-level IP packets in the future. Capalik [0004], [0013], [0022], [0055]. It is for use by the IDS, not by the protected devices.

65. Additionally, the claim does not encompass transmission to an IDS for protection of other devices. Limitation 1[D] requires transmission to a computer system for protection of that same computer system. The specification supports that this is not intended to cover mere transmission to an IDS. The '872 patent distinguishes between the claimed transmission of information to computer systems for their own protection and transmitting to an IDS/IPS. The latter is not encompassed by the claim. Specifically, in one embodiment, the "decoy network device transmits [a] fingerprint to ***one or more protected network devices 136*** to prevent future attacks." *Id.* at 18:44-47. In other embodiments, the "decoy

network device transmits the fingerprint ***to the IDS/IPS system*** 142 to prevent future attacks that comprise the same or similar activities on one or more protected network devices 136." *Id*. at 18:47-51. This delineation between the claimed transmission and transmission to an IDS supports that a POSITA would have understood the claim is intended to cover the first embodiment and not the second.

66. This is further supports by the nature of the fingerprint in the '872 patent versus the attack signature of Capalik. Namely, as I described above, the fingerprint contains behavioral data about a computer system's internal activities, which are most effective to share with other computer systems, not with a network-based IDS/IPS, because the IDS/IPS does not have any visibility into the internal operation of the computer system. Further, the IDS/IPS is never described as part of a computer system with protected network devices.

2. Claim 2

67. Claim 2 recites "monitoring all activity on the first computer system," where the "first computer system" is the system on which unauthorized activities are being identified. As discussed above, Capalik monitors only activity associated with a network connection through a socket and only after an attacker connects to the decoy. Capalik [0027], [0049], [0059]. It does not monitor all activity across the decoy computer, including in processing module 508, which has its own operating system and performs substantial work such as storing attack information,

creating signatures, and transmitting signatures to the IDS/IPS. Capalik [0051]-[0052]. CrowdStrike does not identify any disclosure in Capalik of any monitoring of that processing module. To the contrary, the processing module 508 is merely a trust part of the decoy and not monitored by the introspection module. Capalik [0051] (processing module is "part of a secure and separate administrative network" from the introspection module 538).

68. Capalik assumes that the monitored system is uncompromised until an attacker connects to a port and starts interacting with the decoy. Any monitoring in Capalik is only triggered by such a network intrusion. Capalik's decoy-based system for updating an IDS/IPS signature library would be ineffective against an attacker that was able to access protected devices without interacting with the honeypot. For example, Capalik begins collecting data only after the attacker opens a socket connection to a decoy and transmits information such as port numbers, keystrokes, uploaded files, or other data streams. Capalik [0027], [0049]. Capalik does not disclose or suggest monitoring internal system-level activity, such as changes to local files, unauthorized process behavior, or movement across virtual machines or system components, that occurs before any network-based intrusion is detected.

69. By contrast, the '872 patent explicitly discloses monitoring various types of internal activities, such as file modifications, user actions, instructions or

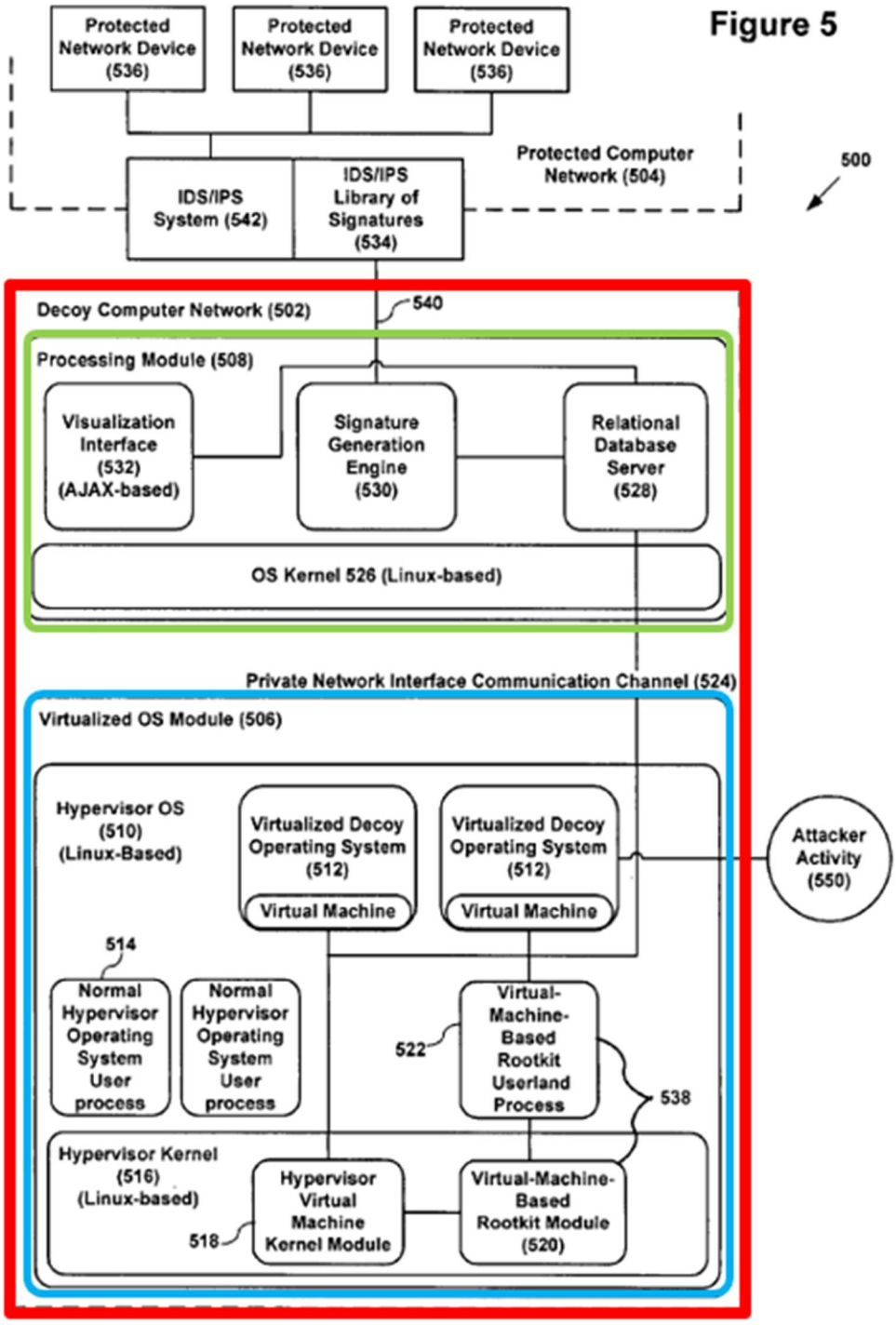
processes, threads, data streams, and socket connections. '872 patent 15:41-47. As illustrated in Figures 4A-4D, the system can detect unauthorized activity in files X and Y, propagate that activity to processes A and B, and only later identify a socket and incoming packet stream as affected by unauthorized activity. '872 patent 12:2-26. These internal events can be detected and modeled even before any suspicious network activity is recognized. A person of ordinary skill would have understood such a solution to be fundamentally different from the monitoring described in Capalik, which is based on monitoring data that flows through a socket.

70. Additionally, CrowdStrike argues that Capalik "teaches monitoring all activity on the computer system" through its disclosure that the processing module "may generate a signature whenever data is found to be transferred through the socket." Pet. at 34 (quoting Capalik [0059]). Again, that only indicates monitoring data passing through the socket—i.e., monitoring specific information from a specific event (receipt of data) at a specific component (a socket). *See, e.g.*, Capalik [0026] ("Sentinel kernel driver 34 monitors all data coming through the socket."). But monitoring all data that passes through a socket is not the same as monitoring all activity on the computer system as recited in claim 2. The fact that Capalik may capture "all data passed through the socket" does not mean it monitors, for example, internal file checksum changes or data execution prevention

(DEP) exceptions, or process launches that occur before any network interaction or happen without network activity like the '872 patent. 72 patent 12:4-14.

71. I disagree with CrowdStrike's argument with respect to claim 2 for an even more basic reason. CrowdStrike mapped the claimed "first computer system" to a "decoy computer 502." Pet. at 15 ("[A] POSITA would have understood Decoy Computer 502 is a first computer system ('computer system') attached to a computer network, as claimed."). As shown in the annotated Figure 5 below, Capalik's decoy system 502 (red) includes both a processing module 508 (green) and virtualized operating system module 506 (blue). CrowdStrike does not identify any discussion in Capalik of monitoring activity within processing module 508—which has its own operating system within the decoy 502. Capalik [0051]. In Capalik, the processing module 508 is "part of a secure and separate administrative network" from the introspection module 538 and is therefore trusted and not monitored by the introspection module. Capalik [0051]. Important and substantial unmonitored activity is performed by the processing module 508, including storage of the attack information in relational database server 528, creation of the signature by signature-generation engine 530, and transmission of the signature to the IDS/IPS system. Capalik [0052]. Additionally, I note that Dr. Jakobsson opines that "[t]o be an efficient decoy, the system would monitor all activity to prevent an accidental allowance of an attacker through." Ex.1003 ¶146.

He further asserts that "for exhaustive purposes, the honeypot was known to assume all activity at the honeypot was unauthorized." *Id.* Further, he asserts that "[b]y acting as a fake port, any activity that reaches the honeypot is assumed unauthorized…." *Id.* This last sentence indicates that Dr. Jakobsson is using the term "activity" to mean "network activity"—i.e., "activity that reaches the honeypot"—rather than activity that is occurring on the system without any network activity. To that end, Dr. Jakobsson does not appear to argue, nor does he provide any evidence or motivation, that honeypots were known to assume all non-network-originating activity was unauthorized—only network-originating activity.



Capalik Fig. 5 (annotated)

3. Claims 3-6, 12, 15, and 20-21

72. Claims 3-6, 12, and 15 all depend (directly or indirectly) from claim 1. Because CrowdStrike failed to meet its burden to show that claim 1 is obvious over Capalik, it follows that CrowdStrike also failed to meet its burden for claims 3-6, 12, and 15.

73. Independent claims 20 and 21 each recite the same "transmitting" limitation as discussed above with respect to claim 1. CrowdStrike did not separately address these limitations in its Petition, instead relying on its claim 1 analysis. *See* Pet. at 39-40. The same is true of Dr. Jakobsson's analysis. *See* Jakobsson Decl. ¶¶ 161 (incorporating discussion of limitation 1[D] into claim 20 by reference), 166 (same for claim 21). Claims 20 and 21 are patentable over Ground 1 for the same reasons as claim 1.

B. Ground 2: Capalik and Goel Do Not Render Any of Claims 1-6, 12, 14, 15, 20, or 21 Obvious

1. A POSITA Would Not Have Been Motivated to Combine Capalik and Goel

74. In Ground 2, CrowdStrike combines Capalik with Goel to challenge claims 1-6, 12, 14-15, 20, and 21. In my opinion, a person of ordinary skill in the art ("POSITA") would not have been motivated to combine the teachings attributed to Capalik and Goel. As noted above, Capalik describes an undetectable decoy system that records and analyzes attacker activity "without knowledge of the attacker" by running a rootkit in the hypervisor while leaving the decoy operating

system unmodified. Capalik [0024], [0045], [0047]. Undetectability is central to its design. Capalik criticizes prior art honeypots as "generally not useful against sophisticated attackers who can detect and effectively avoid the honeynet." Capalik [0006]. To avoid that problem, Capalik's rootkit "remain[s] hidden from the instigator of the attack, with no discoverable impact on the decoy operating systems' performance." Capalik [0047].

75. Goel, in contrast, is not a decoy system and is not concerned with hiding from attackers. *See* Goel at 2 ("Design Goals" not mentioning detectability). Goel modifies the operating system kernel to log every system call and forwards the resulting events to a backend database. Moreover, it does not use virtual machines or introspection, making the kernel-level instrumentation an obvious target for the malware to either disable or to prevent manifesting its true behavior. The authors report significant and measurable performance overhead, including a 7%–36% decrease in web server throughput, and present this as an acceptable trade-off for detailed forensic reconstruction. Goel at 6-7, Table 3. From a security practitioner's perspective, a system that introduces that level of overhead and requires kernel-level hooks is, by design, more likely to be detectable or be subverted by a sophisticated adversary. This point is consistent with other work showing that monitoring mechanisms placed inside or coupled to the guest OS can introduce observable effects and expose attack surfaces. *See, e.g.*, Ex.2050

("Yin") at 1 (discussing tools that detect kernel hooks and proposing methods to detect installation of "any hooks into the system"); Ex.2052 ("Jones"), at 1-2 (describing how rootkits target kernel-resident monitoring and how VMM-level techniques attempt to avoid detectability).

76. A person of ordinary skill would appreciate that combining Goel's kernel-level logging with Capalik's decoy system would undermine the very property that Capalik emphasizes: remaining hidden from the attacker. Adding pervasive kernel hooks and accepting noticeable performance degradation runs directly counter to Capalik's goal of "no discoverable impact" on the decoy OS. Capalik [0047]. A skilled artisan reading these references together would see them as reflecting two different design philosophies: one focused on stealthy decoys and one on exhaustive forensic logging, not as complementary components of a single system. In fact, combining them would render Capalik's decoy "not useful" in the same manner as the detectable honeynets it describes. Capalik [0006].

77. CrowdStrike nevertheless suggests that a person of ordinary skill would incorporate Goel's logging into Capalik's decoy, including into the VMM or hypervisor that hosts the decoy operating system. Pet. at 43-46. However, Goel's logging is implemented explicitly inside the kernel of the monitored system, not in a separate hypervisor. Goel describes that capturing events at the kernel provides a unified view of system calls and addresses limitations of

application-level logging. Goel at 3-4 ("[C]apturing activity at this point thus addresses the problem at a more fundamental, unified level."). Thus, there is no indication in Goel that the same level of detail and fidelity could be achieved from outside the OS kernel.

78. Even if one assumed that some aspects of Goel's design could be re-implemented in a hypervisor, a person of ordinary skill would not expect to achieve Goel's forensic "Design Goals" while also preserving the undetectability guarantees that Capalik requires. The combination would require non-trivial re-engineering, introduce additional attack surface and complexity, and significantly increase the risk of observable side effects on performance or system behavior. From an engineering standpoint, such a design would be at odds with Capalik's criticism of detectable honeypots and its insistence that the decoy OS remain unmodified and indistinguishable from a real system. Capalik [0006], [0047].

79. CrowdStrike also points to Goel's statement that the authors were "working on implementing the Forensix system on a VM" as evidence that a VM integration would have been straightforward. Pet. at 45. However, that brief statement provides no details about such an implementation, and there is no disclosure that the authors were attempting to build an undetectable decoy or that they considered the stringent stealth requirements that drive the design in Capalik.

From a technical standpoint, mentioning a VM deployment does not suggest that Goel is compatible with, or a natural fit for, a stealthy decoy architecture of the type described in Capalik, or that it would have been implemented as part of the protected hypervisor logic rather than as part of the deployed virtual machine. For the above reasons, a person of ordinary skill in the art would not have had a reasonable expectation of achieving *Goel's* level of logging, or its own Design Goals, by capturing activity from a hypervisor outside of the system.

80. For these reasons, a person of ordinary skill in the art would not have been motivated to combine Capalik and Goel in the manner proposed in Ground 2. The combination conflicts with Capalik's core design goals and renders Capalik's decoy "not useful." Because a POSITA would not have been motivated combine Capalik and Goel, Ground 2 fails with respect to all challenged claims—including claim 14, which was not challenged in Ground 1.

2. Claims 1-6, 12, 14, 15, and 20-21

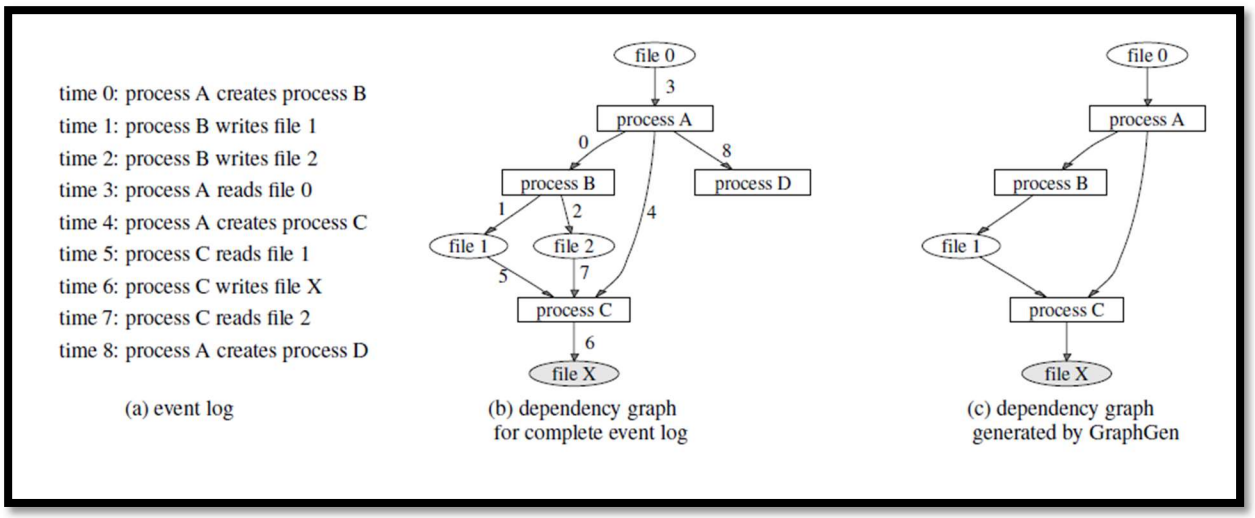
81. CrowdStrike does not rely on Goel to cure any of the deficiencies identified with respect to Capalik in Ground 1. Even if a POSITA would have been motivated to combine Goel with Capalik, CrowdStrike still fails to establish that any of claims 1-6, 12, 14 (which is added to Ground 2 but also depends from claim 1), 15, 20, or 21 is unpatentable.

C. Ground 3: Capalik, Goel, and King Do Not Render Any of Claims 7-11, 14, 16, 18, or 19 Obvious

1. Claim 8

82. Claim 7 requires "graphically displaying at least a subset of the stored activities," and claim 8 (which depends from claim 7) adds the requirement that "the displaying includes visually distinguishing at least a subset of the associations."

83. CrowdStrike relies on Figure 3.2 of King as allegedly disclosing the subject matter of claim 8. Pet. at 54-55. For reference, I have excerpted that Figure below. According to CrowdStrike, "*King*'s dependency graphs visually distinguish a subset of the associations by depicting only malignant associations in a filtered dependency graph." Pet. at 55. Simply displaying or not displaying associations is different from visually distinguishing, as discussed in the context of the '872 patent. *See, e.g.*, '872 patent at 12:32-41 (discussing how a bold arrow can represent a malignant association and a regular arrow can represent a benign association). If "visually distinguishing" entailed only displaying a subset, then claim 8 would add nothing to claim 7 (which already recites displaying a subset of the activities).



King, Fig. 3.2

84. Additionally, the "complete event log" in King's Fig. 3.2 (b) (middle above) is never even displayed in the first place. As shown in the annotations under the diagrams above, Fig. 3.2(b) is the "dependency graph for [the] complete event log," while Fig. 3.2(c) is the "dependency graph generated by GraphGen" that is actually "displayed to a system administrator." Pet. at 51 (citing King at 26-28).

2. Claim 16

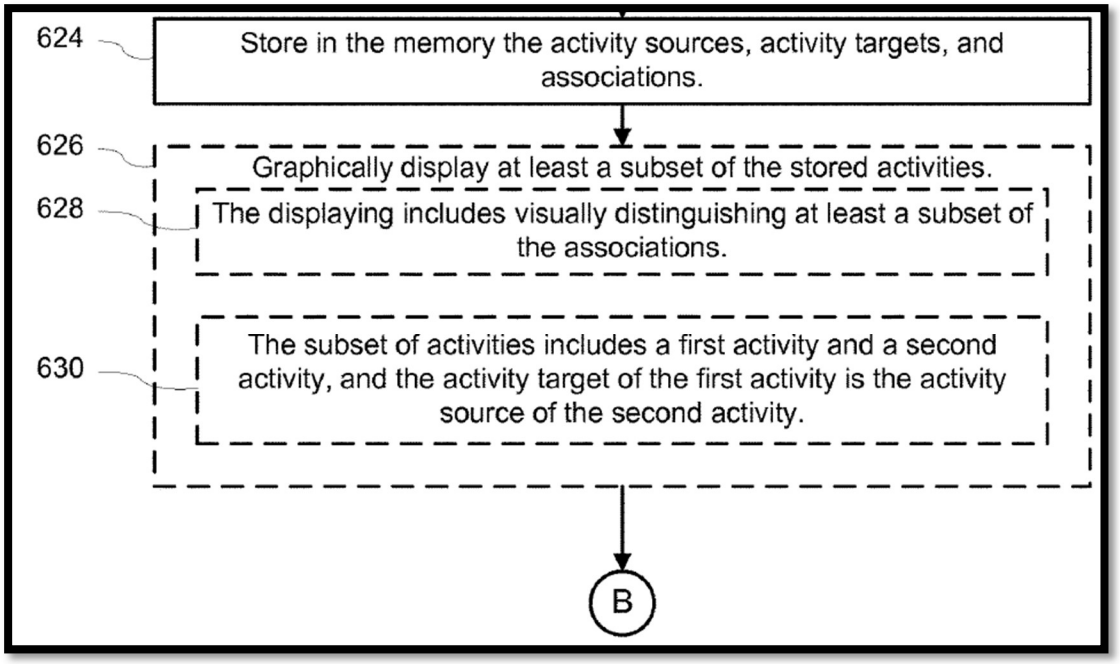
85. Claim 16 depends from claim 1 and recites "determining an activity status for the stored events, the activity status comprising one of: a complete status… and an incomplete status…." The complete status represents that "a source of unauthorized activities is identified" and "each activity target in the stored activities is associated with the source through one or more associations." On the other hand, an incomplete status represents that the source of unauthorized

activities is not identified, or *that* at least one activity target is not associated with the source through one or more associations.

86. CrowdStrike points to King's displaying of the pruned dependency graph in Fig. 3.2(c) (discussed immediately above with respect to claim 8) corresponds to a "complete status." Pet. at 63-64. But CrowdStrike makes no attempt to show any determination of an activity status. CrowdStrike does not point to any disclosure in King that determines that both: (1) "a source of unauthorized activities is identified," and (2) "each activity target in the stored activities is associated with the source through one or more associations," as required for determination of a complete status. Pet. at 63-64.

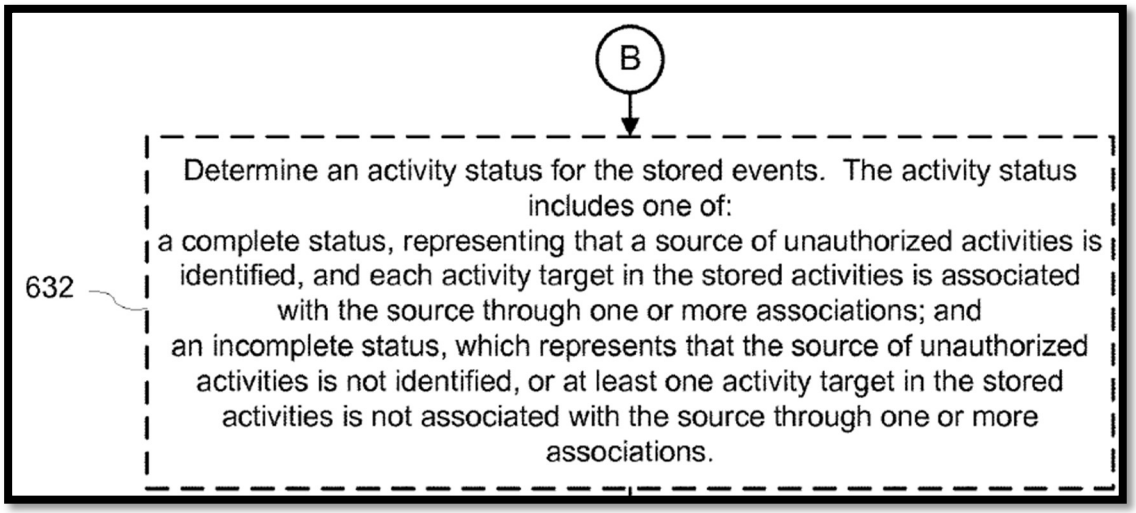
87. King's GraphGen reconstructs dependency graphs based around a specific "detection point, which is file X at time 10." King at 18-19. King's "filtered" graph is the result of omitting irrelevant data from the graph data; it is not a determination whether each target in the stored activities are associated to a source as required by claim 16.

88. The '872 patent specifically describes determining an activity status as a separate step from graphical display. For example, in Figure 6B, step 626 is "[g]raphically display at least a subset of the stored activities."



'872 Patent, Fig. 6B (excerpted)

89. The next step of this computer-implemented method (step 632 in Figure 6C) is "[d]etermine an activity status for the stored events."



'872 Patent, Fig. 6C (excerpted)

90. In short, Claim 16 requires an affirmative step of determining a status for the stored events. The claim does not recite displaying a graph from which a user may infer that the activity status is complete. Accordingly, a POSITA would not have considered King to disclose or render obvious claim 16.

3. Claim 19

91. Claim 19 depends from claim 1 and recites "wherein the association includes a type that represents whether or not the activity source has modified the activity target to perform the unauthorized activities."

92. In arguing that King discloses this limitation, CrowdStrike relies on the fact that King's filtered dependency graph that does not include "any associations that did not contributed … to the unauthorized activity." Pet. at 65. Based solely on this filtered dependency graph, CrowdStrike says that "[a] POSITA would have understood that the associations remaining after this filtering are of a type that represents a source modifying a target to perform unauthorized activities either directly… or indirectly." Pet. at 65-66.

93. I disagree with CrowdStrike's argument. Claim 1 (from which claim 19 depends) requires storing a data structure that identifies associations, and claim 19 adds that the associations must include a type. Taken together, a POSITA would have understood claim 19 to require the type to be stored in the data structure. For example, the highlighted column of Figure 5A below includes a "Type":

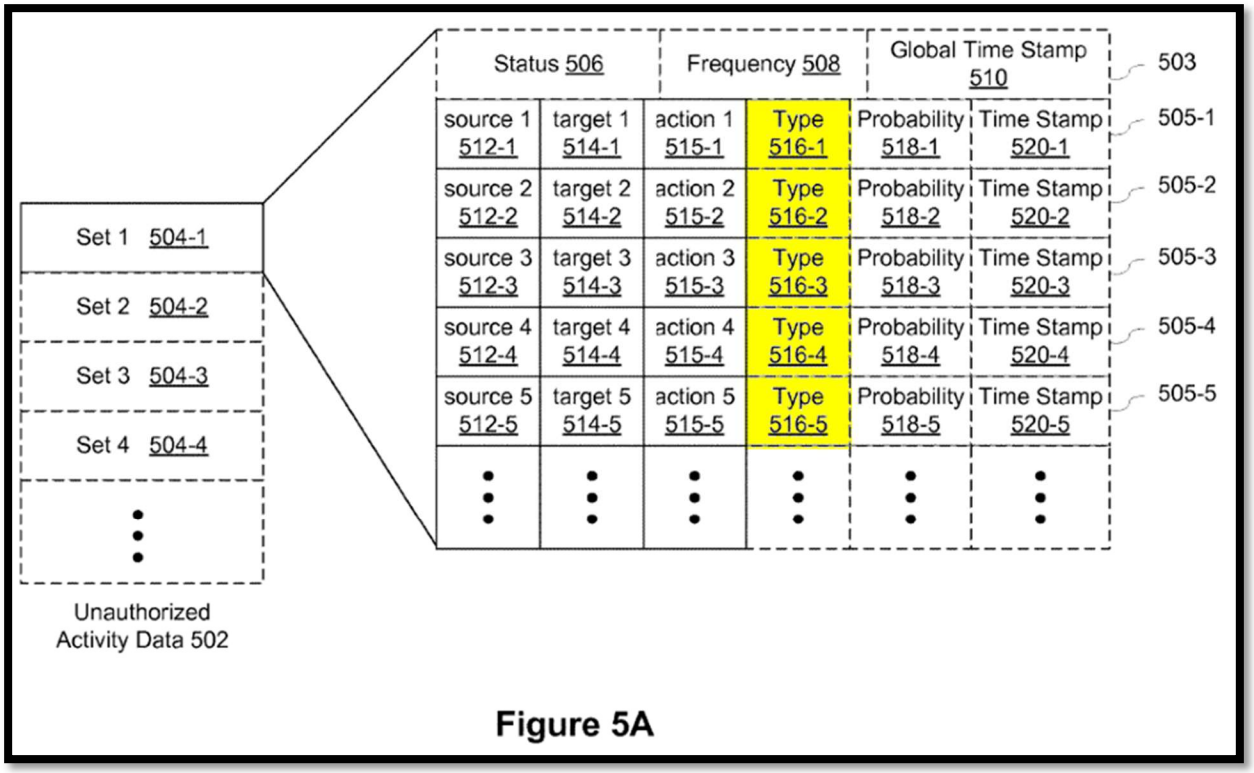


Figure 5A

'872 Patent, Figure 5A

4. Claims 7, 9-11, 14, 16, and 18

94. Claim 7, 9-11, 14, 16, and 18 all depend from claim 1. CrowdStrike does not address claim 1 with respect to Ground 3. Because each of these claims depends from claim 1, CrowdStrike has failed to establish that any of these claims are unpatentable for at least the reasons discussed with respect to Ground 1 above, which I incorporate by reference.

D. Ground 4: Capalik, Goel, and Pike Do Not Render Claim 13 Obvious

95. Claim 13 depends from claim 12, which says "identifying activity sources that are affected by unauthorized activities." Claim 13 further states

"wherein the identifying activity sources… includes identifying activity sources that **request** to access a portion of memory that is set as non-executable." CrowdStrike cites to Pike for claim 13 (Pet. 66-68), but Pike does not disclose the features of claim 13.

96. Pike discloses techniques whereby accesses to certain addresses trigger exceptions, and Pike then analyzes exceptions after they occur. Pike is only concerned with scenarios where "the potentially dangerous address is **accessed**." Pike at 8:65-67. Pike does not describe monitoring a **request** to access the non-executable portion of memory before the access is completed. Instead, Pike observes the system after the fact when an exception has already been raised. While Pike might be useful in identifying unauthorized access after it occurs , Pike would not be able to detect mere requests for access.

E. Ground 5: Capalik, Goel, and Farley Do Not Render Claim 17 Obvious

97. Claim 17 depends from claim 1. Farley does not cure the shortcomings of Capalik or the Capalik-Goel combination above with respect to claim 1. Claim 17 is thus patentable over Ground 5 for at least the same reasons as discussed above with respect to Grounds 1 and 2.

IX. CrowdStrike's Materials From Around 2013 Show That Its Product At That Time Practiced At Least Claims 1 And 2 The '872 Patent

98. I have been asked to consider certain materials which I have been informed were created and published by CrowdStrike in approximately 2013. These materials are Exhibits 2015-2017 and 2051. I will refer to these materials as "the Falcon Platform Materials." In my opinion, these materials show that CrowdStrike's Falcon Platform in 2013 practiced at least claims 1 and 2 of the '872 patent.

A. 1[pre]: "A computer implemented method of identifying unauthorized activities on a first computer system attached to a computer network, wherein the first computer system comprises one or more processors and memory"

99. The Falcon Platform Materials describe CrowdStrike's Falcon Platform, which identifies unauthorized activities on a computer system attached to a computer network and having one or more processors and memory. For example, Exhibit 2015 describes that the Falcon Platform "[i]dentifies malware and provides attribution of the attackers." Ex.2015 at 1. Exhibit 2016 similarly describes that "Falcon Host leverages a lightweight host-based sensor that shadows, captures, and correlates low-level operating system events to instantly identify the adversary tradecraft and activity through our patent-pending Execution Profiling Technology." Ex.2016 at 1. Falcon Host is described as an "on-premise

host-based sensor." *Id.* Accordingly, these materials show that the Falcon Platform practiced the preamble of claim 1.

B. 1[A]: "monitoring activity on the first computer system"

100. The Falcon Platform Materials describe monitoring activity on the first computer system. For example, Exhibit 2017 describes that Falcon sensors capture "over 100 security-related events, including things like process creation, drivers being loaded, registry modifications…, etc." Ex.2017 at 1:42–2:11. Additionally, Exhibit 2016 describes that Falcon Host "shadows, captures, and correlates low-level operating system events…." Ex.2016 at 1. This provides "highly efficient real-time monitoring of all system activity." Ex.2016 at 1. Accordingly, these materials show that the Falcon Platform practiced 1[A].

C. 1[B]: "identifying a plurality of activities being performed at the first computer system, wherein each of the activities includes an activity source, an activity target, and an association between the activity source and the activity target"

101. The Falcon Platform Materials describe identifying the plurality of activities recited in 1[B]. For example, Exhibit 2017 describes that Falcon sensors capture "over 100 security-related events, including things like process creation, drivers being loaded, registry modifications…, etc." Ex.2017 at 1:42–2:11. That video further explains that "[W]e need more than just this detail, we need the context as well. The Falcon Platform has a situational model that keeps track of all the relationships and contexts between all these different processes in a graph

database." *Id.* Additionally, Exhibit 2016 describes that Falcon Host "provides real-time detection by focusing on what the attacker is doing, as opposed to looking for a specific, easily changeable indicator used in an attack." Ex.2016 at 1. Accordingly, these materials show that the Falcon Platform practiced 1[B].

D. 1[C]: "storing in the memory a data structure that identifies the activity sources, the activity targets, and the associations for the plurality of activities"

102. The Falcon Platform Materials describe storing in the memory the data structure recited in 1[C]. For example, Exhibit 2017 describes that Falcon sensors capture "over 100 security-related events, including things like process creation, drivers being loaded, registry modifications…, etc." Ex.2017 at 1:42–2:11. That video further explains that "[W]e need more than just this detail, we need the context as well. The Falcon Platform has a situational model that keeps track of all the relationships and contexts between all these different processes in a ***graph database***." *Id.* The "relationships and contexts" correspond to the recited associations between sources and targets. Further, storing the data in a graph database identifies the sources and targets (e.g., the nodes in the graph) and the associations (e.g., the relationships and contexts between nodes). *See* Ex.2017 at 1:42-2:11. Accordingly, these materials show that the Falcon Platform practiced 1[C].

E. 1[D]: "transmitting to one or more computer systems other than the first computer system information identifying one or more of

the activity sources, the activity targets, and the associations for preventing future attacks, to the one or more computer systems, associated with the one or more of the activity sources, the activity targets, and the associations"

103. The Falcon Platform Materials describe the transmission recited in 1[D]. For example, Exhibit 2015 describes that the Falcon Platform "[s]hares information across an entire community of like-minded trusted defenders." Ex.2015 at 1. Similarly, Exhibit 2051 describes that "the product is a game changer primarily because it focuses on the concept of the community helping each other…. [People] need to basically share indicators… of what the bad guys are doing, so as the bad guys and the adversaries use these techniques… we can actually protect ourselves." Ex.2051 at 0:02-0:42. These materials indicate that the Falcon Platform transmits to other computer systems "indicators"—the claimed "information identifying one or more of the activity sources, the activity targets, and the associations"—to "protect ourselves," meaning to prevent future attacks to the other computer systems associated with the identified sources, targets, and associations. Accordingly, these materials show that the Falcon Platform practiced 1[D].

F. Claim 2: "The method of claim 1, wherein the monitoring further comprises monitoring all activity on the first computer system."

104. The Falcon Platform Materials describe monitoring all activity on the first computer system as recited in claim 2. Falcon Host is described as tracking

"all execution on all hosts at all times" and providing "maximum visibility into all adversary activities." Ex.2016 at 1. Exhibit 2017 explains that visibility is provided by "very efficiently record[ing] everything that goes on in the environment." Ex.2017 at 0:45-0:52. Accordingly, these materials show that the Falcon Platform practiced claim 2.

XI. Conclusion

All statements made herein of my own knowledge are true, and all statements made on information and belief are believed to be true. Further, I am aware that these statements are made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under 18 U.S.C. § 1001. I declare under penalty of perjury that the foregoing is true and correct.

Executed on December, 12th, 2025, in _Fairfax, Virginia.

/Angelos Stavrou/

Angelos Stavrou, Ph.D.