

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

CROWDSTRIKE, INC.,
Petitioner
v.

GOSECURE, INC.,
Patent Owner

Case No. IPR2025-00068
Patent No. 9,954,872

**PETITION FOR *INTER PARTES* REVIEW
OF U.S. PATENT NO. 9,954,872**

TABLE OF CONTENTS

I.	INTRODUCTION.....	1
II.	THE '872 PATENT	1
	A. THE '872 PATENT'S ALLEGED INVENTION	1
	B. THE '872 PATENT'S PROSECUTION	3
	C. CLAIM CONSTRUCTION	4
	D. A PERSON OF ORDINARY SKILL IN THE ART.....	7
III.	REQUIREMENTS UNDER 37 C.F.R. § 42.104.....	7
	A. STANDING UNDER 37 C.F.R. § 42.104(A).....	7
	B. IDENTIFICATION OF CHALLENGE UNDER 37 C.F.R. § 42.104(B)	8
IV.	PRIOR ART	8
	A. CAPALIK	8
	B. KING.....	11
	C. PIKE.....	14
	D. FARLEY.....	16
V.	GROUND 1: <i>CAPALIK</i> IN VIEW OF <i>KING</i> RENDERS CLAIMS 1-12, 14-16, 18-21 OBVIOUS.....	18
	A. CLAIM [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, THE METHOD COMPRISING:”	18
	B. CLAIM [1(A)]: “MONITORING ACTIVITY ON THE FIRST COMPUTER SYSTEM;”	21
	C. CLAIM [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;”	27
	D. CLAIM [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; AND”	40
	E. CLAIM [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.” ..44
- F. CLAIM 2: “THE METHOD OF CLAIM 1, WHEREIN THE MONITORING FURTHER COMPRISES MONITORING ALL ACTIVITY ON THE FIRST COMPUTER SYSTEM.” 49
- G. CLAIM 3: “THE METHOD OF CLAIM 1, FURTHER COMPRISING: CREATING, FROM THE STORED INFORMATION, A FINGERPRINT INDICATIVE OF THE ACTIVITY ON THE FIRST COMPUTER SYSTEM.” 50
- H. CLAIM 4: “THE METHOD OF CLAIM 1, WHEREIN THE ACTIVITIES COMPRISE TWO OR MORE OF: A FILE INITIATING AN EXECUTION OF AN INSTRUCTION, AN INSTRUCTION READING A FILE, AN INSTRUCTION WRITING TO A FILE, AND RECEIVING DATA THROUGH THE NETWORK.” 50
- I. CLAIM 5: “THE METHOD OF CLAIM 1, WHEREIN A RESPECTIVE ACTIVITY TARGET COMPRISES ONE OF: A FILE, A USER, AN INSTRUCTION, A THREAD, DATA STREAM, AND NETWORK SOCKET CONNECTIONS.” 51
- J. CLAIM 6: “THE METHOD OF CLAIM 1, WHEREIN A RESPECTIVE ACTIVITY SOURCE COMPRISES ONE OF: A FILE, A USER, AN INSTRUCTION, A THREAD, DATA STREAM, AND NETWORK SOCKET CONNECTIONS.” 52
- K. CLAIM 7: “THE METHOD OF CLAIM 1, FURTHER COMPRISING: GRAPHICALLY DISPLAYING AT LEAST A SUBSET OF THE STORED ACTIVITIES.” 53
- L. CLAIM 8: “THE METHOD OF CLAIM 7, WHEREIN THE DISPLAYING INCLUDES VISUALLY DISTINGUISHING AT LEAST A SUBSET OF THE ASSOCIATIONS.” .54
- M. CLAIM 9: “THE METHOD OF CLAIM 1, WHEREIN AN ACTIVITY TARGET OF A FIRST ACTIVITY OF THE PLURALITY OF ACTIVITIES IS AN ACTIVITY SOURCE OF A SECOND ACTIVITY OF THE PLURALITY OF ACTIVITIES.” 56
- N. CLAIM 10: “THE METHOD OF CLAIM 1, FURTHER COMPRISING: IDENTIFYING A RESPECTIVE ASSOCIATION BETWEEN A FIRST ACTIVITY SOURCE AND A FIRST ACTIVITY TARGET AS UNAUTHORIZED, WHEREIN THE FIRST ACTIVITY TARGET, WHEN ASSOCIATED AS A SECOND ACTIVITY SOURCE WITH A SECOND ACTIVITY TARGET, CAUSES UNAUTHORIZED ACTIVITIES ON THE SECOND ACTIVITY TARGET.” 59
- O. CLAIM 11: “THE METHOD OF CLAIM 10, FURTHER COMPRISING: TRACKING A TRANSFER OF MALICIOUS ASSOCIATIONS OVER TIME.” 60
- P. CLAIM 12: “THE METHOD OF CLAIM 1, FURTHER COMPRISING: IDENTIFYING ACTIVITY SOURCES THAT ARE AFFECTED BY UNAUTHORIZED ACTIVITIES.” 61
- Q. CLAIM 14: “THE METHOD OF CLAIM 12, WHEREIN: INSTRUCTIONS EXECUTED BY THE ONE OR MORE PROCESSORS HAVE RESPECTIVE PRIVILEGE LEVELS, RESPECTIVE ACTIVITY SOURCES HAVE RESPECTIVE PRIVILEGE LEVELS; AND THE IDENTIFYING ACTIVITY SOURCES THAT ARE

	AFFECTED BY UNAUTHORIZED ACTIVITIES INCLUDES IDENTIFYING ACTIVITY SOURCES THAT REQUEST TO EXECUTE UNDERPRIVILEGED INSTRUCTIONS.”	62
R.	CLAIM 15: “THE METHOD OF CLAIM 12, WHEREIN THE IDENTIFYING ACTIVITY SOURCES THAT ARE AFFECTED BY UNAUTHORIZED ACTIVITIES INCLUDES IDENTIFYING DATA STREAM INCOMING FROM OUTSIDE THE FIRST COMPUTER SYSTEM.”	62
S.	CLAIM 16: “THE METHOD OF CLAIM 1, FURTHER COMPRISING: DETERMINING AN ACTIVITY STATUS FOR THE STORED EVENTS, THE ACTIVITY STATUS COMPRISING ONE OF: A COMPLETE STATUS, REPRESENTING 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; AND AN INCOMPLETE STATUS, WHICH REPRESENTS THAT THE SOURCE OF UNAUTHORIZED ACTIVITIES IS NOT IDENTIFIED, OR AT LEAST ONE ACTIVITY TARGET IN THE STORED ACTIVITIES IS NOT ASSOCIATED WITH THE SOURCE THROUGH ONE OR MORE ASSOCIATIONS.”	63
T.	CLAIM 18: “THE METHOD OF CLAIM 1, FURTHER COMPRISING: IDENTIFYING A RESPECTIVE ACTIVITY AS UNAUTHORIZED BASED ON A DETERMINATION THAT THE RESPECTIVE ACTIVITY IS ASSOCIATED WITH AN UNAUTHORIZED ACTIVITY THAT IS DISTINCT FROM THE RESPECTIVE ACTIVITY.”	65
U.	CLAIM 19: “THE METHOD OF CLAIM 1, WHEREIN THE ASSOCIATION INCLUDES A TYPE THAT REPRESENTS WHETHER OR NOT THE ACTIVITY SOURCE HAS MODIFIED THE ACTIVITY TARGET TO PERFORM THE UNAUTHORIZED ACTIVITIES.”	65
V.	CLAIM [20(PRE)]: “A FIRST COMPUTER SYSTEM, COMPRISING: ONE OR MORE PROCESSORS; AND MEMORY STORING ONE OR MORE PROGRAMS, THE ONE OR MORE PROGRAMS INCLUDING INSTRUCTIONS FOR:”	67
W.	CLAIM [20(A)]: “MONITORING ACTIVITY ON THE FIRST COMPUTER SYSTEM;”	67
X.	CLAIM [20(B)]: “IDENTIFYING A PLURALITY OF ACTIVITIES BEING PERFORMED AT THE FIRST COMPUTER SYSTEM, WHERE EACH OF THE ACTIVITIES INCLUDES AN ACTIVITY SOURCE, AN ACTIVITY TARGET, AND AN ASSOCIATION BETWEEN THE ACTIVITY SOURCE AND THE ACTIVITY TARGET;”	68
Y.	CLAIM [20(C)]: “STORING IN THE MEMORY A DATA STRUCTURE THAT IDENTIFIES THE ACTIVITY SOURCES, THE ACTIVITY TARGETS, AND THE ASSOCIATIONS FOR THE PLURALITY OF ACTIVITIES; AND”	68
Z.	CLAIM [20(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.” ..68	
AA. CLAIM [21(PRE)]: “A COMPUTER READABLE STORAGE MEDIUM, INCLUDING ONE OR MORE PROGRAMS FOR EXECUTION BY ONE OR MORE PROCESSORS OF A FIRST COMPUTER SYSTEM, THE ONE OR MORE PROGRAMS INCLUDING INSTRUCTIONS FOR:”	68
BB. CLAIM [21(A)]: “MONITORING ACTIVITY ON THE FIRST COMPUTER SYSTEM;”	68
CC. CLAIM [21(B)]: “IDENTIFYING A PLURALITY OF ACTIVITIES BEING PERFORMED AT THE FIRST COMPUTER SYSTEM, WHERE EACH OF THE ACTIVITIES INCLUDES AN ACTIVITY SOURCE, AN ACTIVITY TARGET, AND AN ASSOCIATION BETWEEN THE ACTIVITY SOURCE AND THE ACTIVITY TARGET;”	69
DD. CLAIM [21(C)]: “STORING IN MEMORY OF THE FIRST COMPUTER SYSTEM A DATA STRUCTURE THAT IDENTIFIES THE ACTIVITY SOURCES, THE ACTIVITY TARGETS, AND THE ASSOCIATIONS FOR THE PLURALITY OF ACTIVITIES; AND	69
EE. CLAIM [21(D)]: “TRANSMITTING TO ONE OR MORE COMPUTER SYSTEMS OTHER THAN THE FIRST COMPUTER SYSTEM INFORMATION IDENTIFYING ONE OR MORE OF THE ACTIVITY SOURCES, ACTIVITY TARGETS, AND 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.” ..	69
VI. GROUND 2: <i>CAPALIK</i> IN VIEW OF <i>KING</i> , AND IN FURTHER VIEW OF <i>PIKE</i> RENDERS CLAIMS 13 OBVIOUS.....	69
A. CLAIM 13: “THE METHOD OF CLAIM 12, WHEREIN THE IDENTIFYING ACTIVITY SOURCES THAT ARE AFFECTED BY UNAUTHORIZED ACTIVITIES INCLUDES IDENTIFYING ACTIVITY SOURCES THAT REQUEST TO ACCESS A PORTION OF MEMORY THAT IS SET AS NON-EXECUTABLE.”	69
VII. GROUND 3: <i>CAPALIK</i> IN VIEW OF <i>KING</i> , AND IN FURTHER VIEW OF <i>FARLEY</i> RENDERS CLAIM 17 OBVIOUS.....	73
A. CLAIM 17: “THE METHOD OF CLAIM 1, FURTHER COMPRISING: DETERMINING AN ACTIVITY LEVEL REPRESENTING A FREQUENCY OF UNAUTHORIZED ACTIVITIES ON THE FIRST COMPUTER SYSTEM DURING A PREDEFINED TIME INTERVAL.”	73
VIII. DISCRETION UNDER 35 U.S.C. §325(d).....	77

IX.	DISCRETIONARY DENIAL IS NOT WARRANTED	79
A.	THE <i>FINTIV</i> FACTORS FAVOR INSTITUTION.....	79
X.	CONCLUSION	80
XI.	MANDATORY NOTICES UNDER 37 C.F.R. § 42.8.....	81
A.	REAL PARTIES-IN-INTEREST UNDER 37 C.F.R. § 42.8(B)(1)	81
B.	RELATED MATTERS UNDER 37 C.F.R. § 42.8(B)(2)	81
C.	LEAD AND BACK-UP COUNSEL UNDER 37 C.F.R. § 42.8(B)(3)	81

I. INTRODUCTION

CrowdStrike, Inc. (“Petitioner”) respectfully requests *inter partes* review (“IPR”) of all claims (“Challenged Claims”) of U.S. Patent No. 9,954,872 (“’872 Patent”).

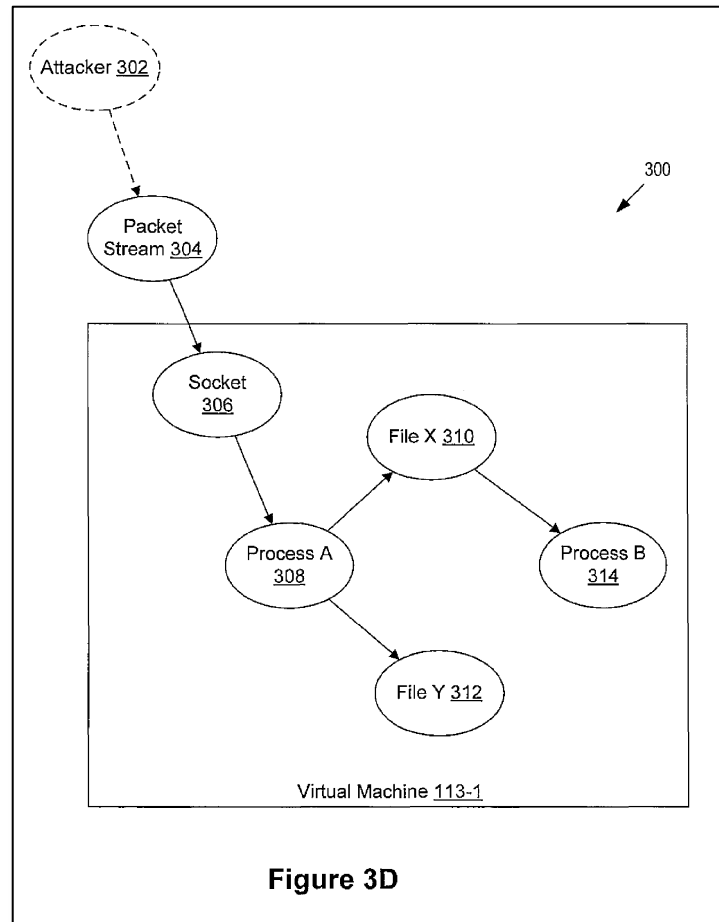
II. THE ’872 PATENT

A. The ’872 Patent’s Alleged Invention

The ’872 Patent describes “methods for analyzing malicious activities on a computer system in order to better protect the computer system from future malicious activity.” ’872 *Patent*, 1:44-48. More specifically, the ’872 Patent describes a method for using a virtual machine (VM) as a decoy computer system to monitor unauthorized activities and to generate a fingerprint indicative of those unauthorized activities. *Id.*, 2:27-48, Fig. 1. The fingerprint can then be used to protect the remainder of the computer network against similar unauthorized activities. *Id.*, 2:42-48. By generating fingerprints from monitored activities in real-time, the ’872 Patent purports to protect against so-called “zero-day attack[s]”—security vulnerabilities previously unknown to software developers or system operators. *Id.*, 2:9-11.

Figs. 3A-3D and 4A-4E illustrate exemplary patterns of attack monitored on a VM. *Id.*, 11:15-22. The network diagram of the pattern of attack depicted in Figs. 3A-3D starts with the detection of an unauthorized activity—Socket 306 receiving data Packet Stream 304. *Id.*, 11:26-36. Once the first detected activity is deemed to

be “affected by unauthorized activities,” related unauthorized activities are identified, providing a detailed picture of the compromise. *Id.*, 11:45-47, Figs. 3B-3D.



Id., Fig. 3D, 11:23-12:26.

The '872 Patent explains that by analyzing the activities depicted in Figures 3A-3D and 4A-4D, “the fingerprint of the unauthorized activities can be identified[.]” *Id.*, 11:56-57. “By using this information, the security of the protected network devices 136 can be improved (e.g., the socket 306 can be closed; the process A 308

can be modified to not respond to data similar to the packet stream 304; etc.).” *Id.*, 11:61-65.

B. The ’872 Patent’s Prosecution

The ’872 Patent was filed on August 11, 2015, claiming priority to Provisional Application No. 61/358,367 filed June 24, 2010. Petitioner does not concede that Patent Owner can establish priority based on the provisional application, but the Challenged Claims are unpatentable even if the earlier date is applied.

A first office action issued on August 30, 2016, rejecting all originally presented claims 1-20. ’872 *Patent File History* (Ex. 1002), 127-39. In response, Applicant amended the claims to require storing “a data structure that identifies the activity sources, the activity targets, and the associations for the plurality of activities; and transmitting one or more of the activity sources, the activity targets, and the associations to prevent future attacks associated with the one or more of the activity sources, the activity targets, and the associations.” *Id.*, 112-15 (amended language underlined). Applicant removed claim elements directed to a virtual machine. *Id.*

A second office action issued on June 6, 2017, rejecting all amended claims 1-20. *Id.*, 70-84. In response, Applicant submitted a Request for Continued Examination (RCE) on November 6, 2017. *Id.*, 49-59. Along with the RCE, Applicant amended the claims to require “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.” *Id.*, 55-58 (amended language underlined). A Notice of Allowance issued thereafter. *Id.*, 10-16.

C. Claim Construction

In this proceeding, claims are interpreted under the same standard applied by Article III courts (i.e., the *Phillips* standard). 37 C.F.R. § 42.100(b); 83 Fed. Reg. 197 (Oct. 11, 2018); *Phillips v. AWH Corp.*, 415 F.3d 1303, 1312 (Fed. Cir. 2005) (*en banc*). For all terms not expressly discussed below, Petitioner applies the ordinary and customary meaning of the claim terms as understood by a POSITA.¹

1. “Association”

The Challenged Claims require identifying activities “wherein each of the activities includes an activity source, an activity target, and an **association**² between the activity source and the activity target;” “storing in the memory the activity sources, activity targets, and associations;” and creating a “fingerprint indicative of

¹ Petitioner’s challenge is not a waiver of any argument in litigation that any claim term in the ’872 Patent is indefinite or invalid. And Petitioner does not waive its rights to construe other claim terms in other proceedings, including in the pending district-court litigation.

² All emphases added unless otherwise noted.

the activity” “from the stored activities[.]” *'872 Patent*, Claims 1, 20-21. Based on the intrinsic record’s ambiguity, there are two possible interpretations of “association.” First, “association” could be interpreted to broadly mean the action, event, or operation between an activity’s source and target. *See, e.g., '872 Patent*, 13:34-38 (referring to the third mandatory component of an activity as an “action . . . represent[ing] an event or operation occurring between the activity source [] and the activity target”). Second, “association” could be interpreted to more narrowly require characterizing whether the action, event, or operation between an activity’s source and target is authorized or unauthorized (i.e., as benign or malignant). This is also referred to in the specification as an “association type[.]” which characterizes the relationship between a source and target as benign, malignant, or in remission (i.e., formerly but no longer malignant). *'872 Patent*, 13:34-56.

Properly interpreting “association” meaningfully impacts the claim scope. Under the first (broader) interpretation, storing “activity sources, the activity targets, and the associations” and creating a “fingerprint indicative of the activity on the virtual machine” from “the stored activities” does not require separately characterizing the individual activities comprising an attack as authorized or unauthorized (i.e., benign or malignant). Instead, the system may characterize attacks more generally and generate signatures without narrowly assessing whether each action comprising an attack is benign or malignant. For example, as described

in U.S. Patent App. Pub. No. 2008/0016570 to Capalik (“*Capalik*”), which is expressly incorporated by reference into the ’872 Patent, the system may decide whether or not to generate a fingerprint/signature of an attack based generally on whether any harmful operations are detected incident to the attack. *Capalik*, [0032-33], [0058]. A separate petition is being filed concurrently that advances this broader interpretation of “association.” See IPR2025-00070 (Ex. 1010).³ That petition includes a detailed claim construction analysis supporting the broader interpretation and includes grounds showing Capalik renders obvious the claims under this broader interpretation.

Under the second (narrower) interpretation, before creating a “fingerprint indicative of the activity on the virtual machine[,]” the system must separately characterize the individual actions as authorized or unauthorized (i.e., benign or malignant). This petition is presented in parallel with IPR2025-0070 to account for the possibility that Patent Owner interprets the claims narrowly, requiring the claimed “association” of an activity source and target to include some indication of whether the activity is authorized or unauthorized (i.e., benign or malignant). Primarily, the grounds herein establish that a POSITA would have been motivated to modify *Capalik*’s method pursuant to *King* such that individual activities are

³ A “ranking” paper is concurrently filed, explaining and prioritizing two petitions.

classified as benign or malignant prior to generating a fingerprint or signature of the attack.

As explained in the Ranking Paper submitted concurrently, if the Board determines that “association” should not be construed so narrowly, Petitioner asks that the Board prioritize its other petition (IPR2025-0070), which targets the broader read of “association.”

D. A Person of Ordinary Skill in the Art

As of June 24, 2010, a person of ordinary skill in the art (POSITA) would be a person having a bachelor’s degree in computer science, computer engineering, or an equivalent, as well as two years of industry experience. A POSITA would also have had a working knowledge of software security analysis and dynamic malware analysis. Additional graduate education could substitute for professional experience, or significant experience in the field could substitute for formal education. Dr. Jakobsson Declaration (“*Dec.*”) (Ex. 1003), ¶¶41-43.

III. REQUIREMENTS UNDER 37 C.F.R. § 42.104

A. Standing Under 37 C.F.R. § 42.104(a)

Petitioner certifies the ’872 Patent is eligible for IPR and Petitioner is not barred or estopped from requesting IPR challenging the claims of the ’872 Patent. Petitioner is not the owner of the ’872 Patent, has not filed a civil action challenging the validity of any claim of the ’872 Patent, and this Petition is filed less than one

year after Petitioner was served with a complaint alleging infringement of the '872 Patent.

B. Identification of Challenge Under 37 C.F.R. § 42.104(b)

Petitioner requests the Challenged Claims be found unpatentable on the following grounds:

Proposed Grounds of Unpatentability	Exhibits
<u>Ground 1:</u> Claims 1-12, 14-16, and 18-21 of the '872 Patent are obvious under §103(a) over US Application Publication 2008/0016570 to Capalik (" <i>Capalik</i> ") in view of " <i>Analyzing Intrusions Using Operating System Level Information Flow</i> " by King (" <i>King</i> ").	Ex. 1004 Ex. 1006
<u>Ground 2:</u> Claim 13 of the '872 Patent is obvious under §103(a) over <i>Capalik</i> in view of <i>King</i> , and in further view of U.S. Patent No. 8,819,822 to Pike et al. (" <i>Pike</i> ").	Ex. 1004 Ex. 1006 Ex. 1007
<u>Ground 3:</u> Claim 17 of the '872 Patent is obvious under §103(a) over <i>Capalik</i> in view of <i>King</i> , and in further view of US Patent No. 7,089,428 to Farley (" <i>Farley</i> ").	Ex. 1004 Ex. 1006 Ex. 1008

IV. PRIOR ART

A. Capalik

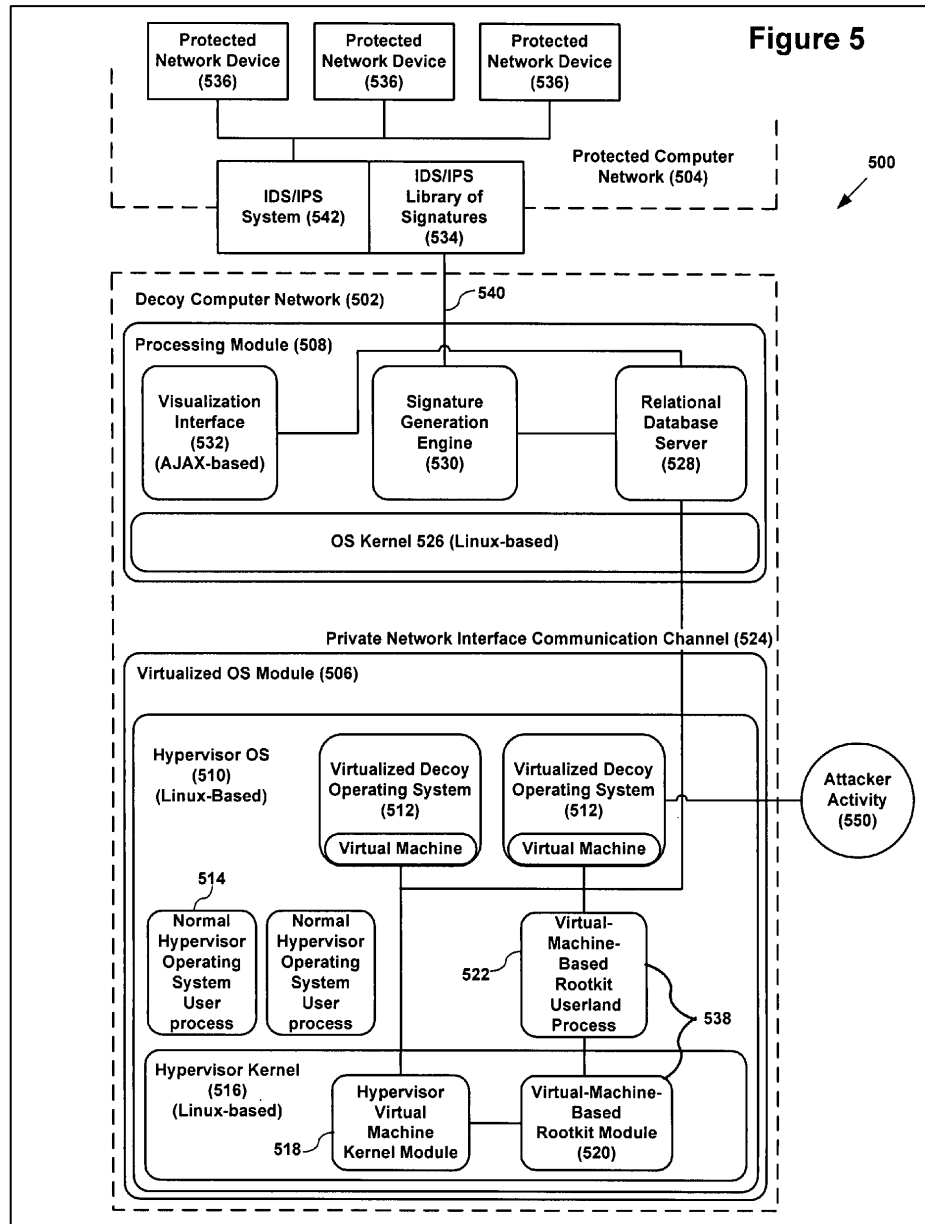
Capalik was filed April 20, 2007 and published January 17, 2008. *Capalik* (Ex. 1004). *Capalik* and the '872 Patent share one inventor, Alen Capalik. However, *Capalik* published January 17, 2008, more than one year prior to the earliest claimed

priority date of the '872 Patent (June 24, 2010). Therefore, *Capalik* qualifies as prior art to the '872 Patent under 35 U.S.C. § 102(b) (pre-AIA).

Capalik was incorporated by reference in the '872 Patent but was not discussed by the Examiner during prosecution. See '872 Patent, 1:30-40; see also generally '872 Patent File History. As set forth below in Section VIII, a discretionary denial under § 325(d) is not warranted on this record.

Capalik teaches using a virtual machine (VM) and virtual machine monitor (VMM) as a decoy computer system to monitor unauthorized activities on the VM, generate a signature indicative of the attack, and protect computer networks from new attacks (e.g., zero-day attacks). *Capalik*, Abstract, [0007]-[0011].

Figure 5 below depicts an exemplary system for analyzing and preventing unauthorized intrusion into a computer network:



Capalik, Fig. 5, [0037]. The described system includes decoy computer network 502 and a protected computer network 504. *Capalik*, [0038]. The decoy computer network 502 includes virtualized operating system (OS) module 506 for monitoring decoy network 502, and processing module 508 for obtaining, analyzing, and

responding to exploits. *Id.* *Capalik* explains that these decoy computer modules may be hosted on the same computing device. *Capalik*, [0039].

Capalik's decoy computer is intentionally kept vulnerable to lure an attacker. *Capalik*, [0044]. When an attacker begins interacting with a virtualized decoy OS, introspection module 538 monitors and captures attack-identifying information, which is stored as forensic data for later analysis. *Capalik*, [0049]-[0052]. Signature generation engine 530 uses this forensic data to generate an attack signature. *Capalik*, [0052]. The attack signature is then sent to the IDS/IPS system 542, which uses the signature to identify and prevent subsequent attacks. *Capalik*, [0054].

Capalik, like the '872 Patent, discloses systems and methods for providing computer security by analyzing event data from potential threat events. *Compare Capalik*, Abstract, [0002], [0015] *with '872 Patent*, 1:44-48, 2:32-48, 2:49-3:5. Thus, *Capalik* is in the same field of endeavor as the '872 Patent. *Dec.*, ¶¶82-88.

Capalik is also reasonably pertinent to certain problems addressed by the '872 Patent such as providing a forensic analysis of the impact of an attack on a target device. *Compare Capalik*, Abstract, [0002], [0003], [0007], [0015] *with '872 Patent*, 2:20-31, 10:30-55. *Dec.*, ¶¶82-88. Thus, *Capalik* is analogous art to the '872 Patent.

B. King

King was submitted to the University of Michigan library in partial fulfillment of the requirements for the degree of Doctor of Philosophy (Computer Science and

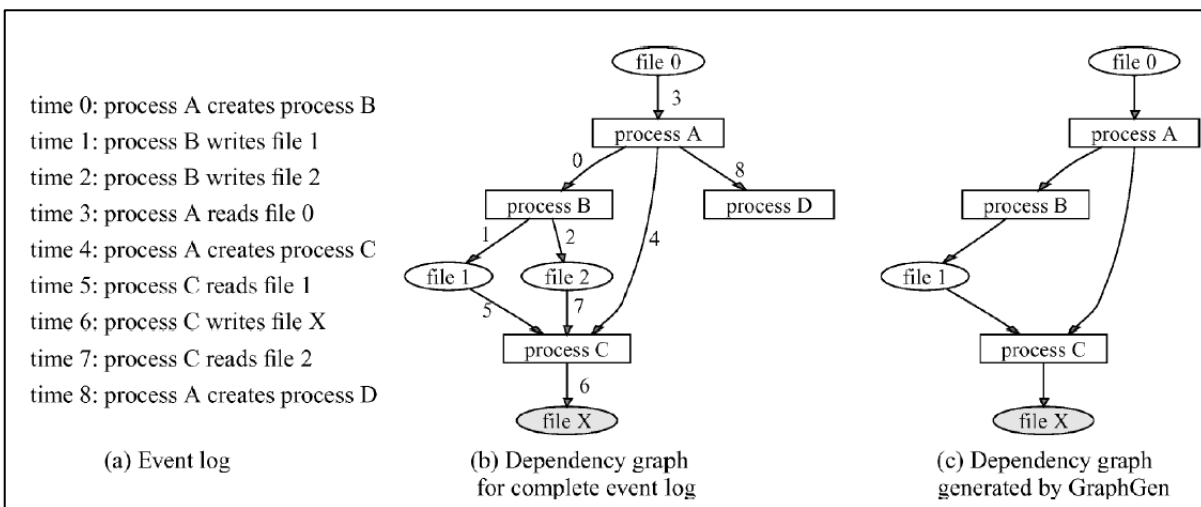
Engineering) of Samuel T. King. *See King* (Ex. 1006). The University of Michigan library first acquired *King* as of November 28, 2006, and it was made publicly available shortly after its initial acquisition date. *See Munford Dec.*, (Ex. 1009). Therefore, *King* qualifies as prior art under 35 U.S.C. § 102(b) (pre-AIA). *Id.*

King teaches systems and methods for analyzing unauthorized intrusions in a computer system to protect the system from future malicious attacks. *King*, 2-3, 64-65. Specifically, *King* discloses analyzing unauthorized activities using “operating-system-level (OS-level) information-flow graphs to highlight the activities of an attacker” being performed at a target system. *King*, 3.

King uses an “EventLogger” to log information-flow events in real time and “GraphGen” to process EventLogger logs to generate information-flow graphs used to analyze an intrusion. *King*, 12. Multiple methods for analyzing intrusions using information-flow graphs are described, including “BackTracker” and “ForwardTracker.” *King*, 4, 28-42. BackTracker provides information-flow graphs that provide an analysis of how an intruder broke into a system, and ForwardTracker provides information-flow graphs that provide an analysis of the attacker’s actions after the compromise. *Id.* In both methods, information-flow graphs start from a “detection point”—a point at which the administrator is made aware of the intrusion. *King*, 4, 17. From the detection point, *King*’s systems build a dependency graph of all objects and events that causally affect the state of the detection point (backward

information flow) or are causally affected by the state of the detection point (forward information flow). *King*, 17. Tracked objects include processes, files, filenames, and registry entries. *King*, 12-13.

An attack visualization is shown in Figure 3.2 below, created using *King's* Backtracker to create the Event Log in 3.2(a) and GraphGen to visually depict the all event activities in 3.2(b) and the subset of just the malignant activities in 3.2(c):



King, Fig. 3.2. *King's* BackTracker information-flow graph starts with detection of a suspicious object (i.e., file X having the wrong contents at time 10) and follows the attack back in time, using causal events, to highlight the sequence of events and objects that led to the suspicious state. *King*, 17-18 (noting the “detection point” for this Backtracker flow is time 10—the time at which “the administrator knows that file X has the wrong contents”).

As depicted above, GraphGen creates dependency graphs from the event logs. *King*, 17. Figure 3.2(b) depicts a complete dependency graph for the event, including

all the events GraphGen considers relevant based on time. *King*, 17-18. GraphGen then generates a filtered graph (Fig. 3.2(c)) by pruning away events and objects that do not affect file X by time 10. *King*, 18-19. That is, GraphGen classifies each individual action in an attack as related to unauthorized conduct (i.e., malignant) or unrelated to unauthorized conduct (i.e., benign). These filtered information-flow graphs can then be used as “signatures” of known bad behavior in computer systems to detect and prevent future intrusions. *King*, 64-65.

King, like the '872 Patent, discloses systems and methods for providing computer security by analyzing event data from potential threat events. *Compare King*, 2-3 with '872 Patent, 1:44-48, 2:32-3:5. Thus, *King* is in the same field of endeavor as the '872 Patent. *Dec.*, ¶¶89-95.

King is also reasonably pertinent to certain problems addressed by the '872 Patent such as providing a forensic analysis of the impact of an attack on a target device. *Compare King*, 2-5, with '872 Patent, 2:20-31, 10:30-55. *Dec.*, ¶¶89-95. Thus, *King* is analogous art to the '872 Patent.

C. Pike

Pike was filed March 30, 2006 and published August 26, 2014. *Pike* (Ex. 1007). Therefore, *Pike* qualifies as prior art to the '872 Patent under 35 U.S.C. § 102(e).

Pike discloses computational security techniques for identifying and preventing malware attacks. *Pike*, 2:7-8. Relevant to the Proposed Grounds, *Pike* discloses scanning “input strings” (e.g. data packets), identifying subsequences in the input strings that could be maliciously exploited, and configuring the target computer system to trigger a fault (or other triggered event) when certain execution conditions suggest use of data or code implicated by the identified subsequences. *Pike*, 3:12-19, 5:13-38, 8:65-67, 2:7-33. *Pike* discloses several techniques to trigger an action when memory of the target system is accessed by the potentially dangerous data in the input string. *Pike*, 8:65-9:5. In one technique, *Pike*’s system sets a relevant no-execute bit that triggers an exception, and an exception handler evaluates the likelihood that a particular exception was caused by a dangerous interpretation of a received input string. *Pike*, 8:65-9:12, 9:24-10:1, 12:58-13:22.

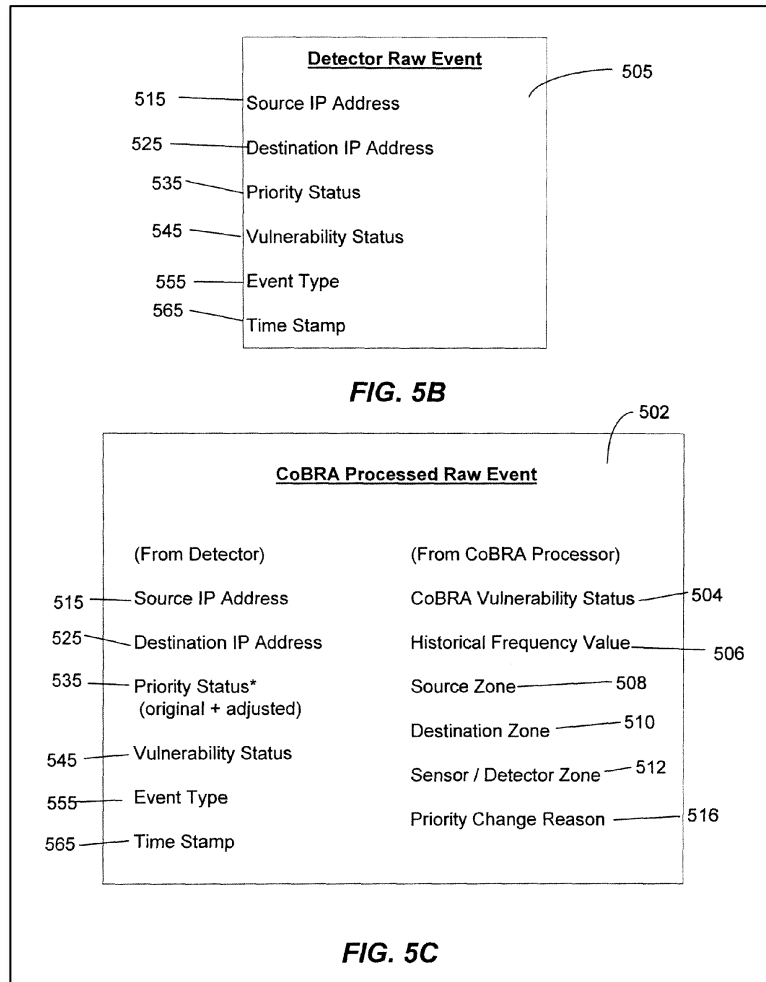
Pike, like the ’872 Patent, discloses systems and methods for providing computer security by analyzing event data from potential threat events. *Compare Pike*, 2:7-23, 3:10-22 *with* ’872 Patent, 1:44-48, 2:32-48, 2:49-3:5. Thus, *Pike* is in the same field of endeavor as the ’872 Patent. *Dec.*, ¶¶96-98.

Pike is also reasonably pertinent to certain problems addressed by the ’872 Patent such as providing a forensic analysis of the impact of an attack on a target device. *Compare Pike*, 2:34-47, 7:30-41 *with* ’872 Patent, 2:20-31, 10:30-55. *Dec.*, ¶¶96-98. Thus, *Pike* is analogous art to the ’872 Patent.

D. Farley

Farley was filed April 27, 2001 and published August 08, 2006. *Farley* (Ex. 1008). Therefore, *Farley* qualifies as prior art to the '872 Patent under 35 U.S.C. § 102(b) (pre-AIA).

Farley discloses a “computer security management system that can log, investigate, respond to, and track computer security incidents that can occur in a networked computer system. The invention can track suspicious computer activity or actual computer security threats.” *Farley*, 3:25-30. Relevant to the Proposed Grounds, *Farley* teaches assembling suspicious computer activity (“raw events”) from multiple data sources (e.g. IDS/IPS, firewalls, audit systems). *Farley*, 3:47-4:11. This data is analyzed to detect relationships between raw events that may indicate malicious behavior (“mature correlation event”), and an organized data presentation is generated, as depicted below:



Farley, Figs 5B-5C, 4:18-34. Figure 5B illustrates exemplary raw event 505 that an IDS detector generates. *Farley*, 11:65-67. Raw event 505 may comprise source IP address 515, destination IP address 525, priority status 535, detector assigned vulnerability status 545, event type parameter 555, and time stamp 565. *Farley*, 11:67-12:4.

The priority status of a raw event may be adjusted by a CoBRA (Context Based Risk Adjustment) processor by comparing the raw event to information stored in the context database such as historical event frequency values. *Farley*, 15:11-36,

Fig. 5C (depicting Historical Frequency Value 506). A historical frequency value “can be a number of events per unit time, such as events per day[.]” *Farley*, 24:27-39.

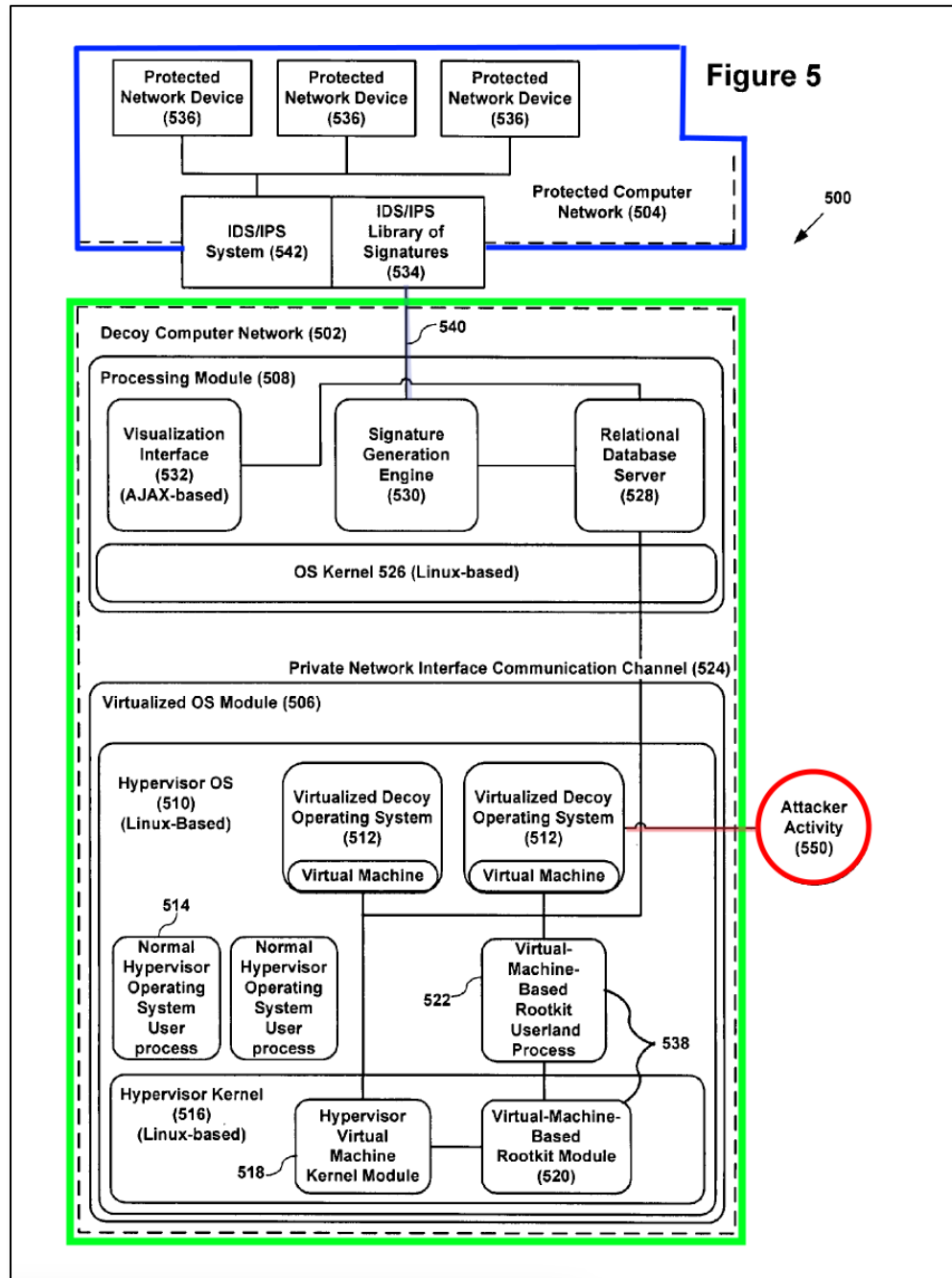
Farley, like the ’872 Patent, discloses systems and methods for providing computer security by analyzing event data from potential threat events. *Compare Farley*, 3:25-46, 4:1-34 with ’872 Patent, 1:44-48, 2:32-48, 2:49-3:5. Thus, *Farley* is in the same field of endeavor as the ’872 Patent. *Dec.*, ¶¶99-102.

Farley is also reasonably pertinent to certain problems addressed by the ’872 Patent such as providing a forensic analysis of the impact of an attack on a target device. *Compare Farley*, 3:25-46, with ’872 Patent, 2:20-31, 10:30-55. *Dec.*, ¶¶99-102. Thus, *Farley* is analogous art to the ’872 Patent.

V. GROUND 1: *CAPALIK* IN VIEW OF *KING* RENDERS CLAIMS 1-12, 14-16, 18-21 OBVIOUS

- A. Claim [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, the method comprising:”**

To the extent the Preamble is limiting, *Capalik* teaches it. *Capalik* describes “a system for analyzing and preventing unauthorized intrusion into a computer network” as shown in FIG. 5. *Capalik*, [0037]. Annotated green in Fig. 5 below, “decoy computer network 502” comprises multiple modules responsible for identifying unauthorized activities on a decoy computer system:



Capalik, Fig. 5 (annotated); [0038] (“The decoy computer network 502 includes a virtualized operating system module 506 for monitoring the decoy network 502, and a processing module 508 for obtaining, analyzing, and responding to exploits.”). *Capalik* teaches that “[t]hese modules may be hosted on the same computing device

or on separate computing devices.” *Capalik*, [0039]. To simplify the discussion, Petitioner focuses on an implementation in which the entirety of decoy computer network 502 is implemented on a single computing device and will refer to this device and the modules therein as *Capalik*’s “Decoy Computer 502.” Throughout the Proposed Grounds, *Capalik*’s Decoy Computer 502 is mapped to the claimed “first computer system.”

Capalik’s Decoy Computer 502 is attached to a computer network. Namely, *Capalik* teaches “attacker activity 550 is directed at the [D]ecoy [C]omputer [] 502 through one or more ports of each of the virtualized decoy operating systems 512 that are left open as a gateway for attacker activity 550.” *Capalik*, [0044]. This “attacker activity 550” is annotated red in Fig. 5 above. A POSITA would have understood that ports left open on Decoy Computer 502 through which attacker activity can be directed is a reference to network connectivity, facilitating attacker traffic to/from the decoy computer system. *Dec.*, ¶104. Accordingly, a POSITA would have understood that the Decoy Computer 502 is a first computer system (“computer system”) attached to a computer network, as claimed. *Dec.*, ¶104.

Additionally, *Capalik* teaches Decoy Computer 502 (i.e., computer system) is attached to “protected computer network 504” via connection 540. *Capalik*, [0054], [0038], Fig. 5; *Dec.*, ¶105.

Capalik's Decoy Computer 502 includes “one or more processors, input/output devices, communication circuitry, power sources, memory (both physical, e.g., RAM, and disks, e.g., hard disk drives), and any other physical hardware necessary for hosting and running the aforementioned modules.” *Capalik*, [0039]; *Dec.*, ¶106. Where decoy computer network 502 and all its modules are implemented on the same computing device, as is the focus of the Proposed Grounds, a POSITA would have understood that the virtualized operating system module 506 and a processing module 508—within Decoy Computer 502 (i.e. computer system)—run on a computing platform that includes at least a processor and memory, as claimed. *Dec.*, ¶106.

B. Claim [1(a)]: “monitoring activity on the first computer system;”

Proposing the same structure and using the same terminology as the '872 Patent, *Capalik* teaches monitoring activity on the computer system using a virtual machine monitor (i.e., Hypervisor module) that monitors activity on multiple virtual machines within the first computer system.⁴

⁴ Consistent with the '872 Patent's focus on monitoring virtual machines (i.e., decoy devices), the Proposed Grounds focus on *Capalik's* process of monitoring virtual machines within Decoy Computer 502. The Board should reject any argument that the claimed process of monitoring a “computer system” excludes monitoring virtual machines (or decoy devices). Although the '872 Patent contemplates *the possibility*

1. '872 Patent's monitoring activity on the computer system

Figure 2 of the '872 Patent depicts an exemplary decoy network device 106 including Virtual Machines (VMs) (annotated pink), and a Virtual Machine Monitor (VMM) (annotated orange). The VMs and VMM are contained in a Hypervisor OS block (annotated green) within the decoy computer, and an introspection kernel (annotated blue) is interposed between the VMM and VMs, as illustrated below:

of employing its methods on non-decoy systems, all detailed embodiments target decoy device monitoring. *See, e.g., '872 Patent*, 5:25-30 (“Although the methods described herein are described with respect to monitoring the decoy network device 106, . . . similar methods *may be used* for monitoring one or more non-decoy systems[.]”). The patent also teaches that virtual machines are an example of the described “computer system.” *Id.*, 11:15-18 (“FIGS. 3A-3D and 4A-4E illustrate exemplary network diagrams representing unauthorized activity on a **computer system** (e.g., the **virtual machine** 113-1)[.]”). Because the disclosed embodiments describe monitoring virtual machine “computer systems,” the claims cannot be construed to exclude virtual machine-based monitors. *Oatey Co. v. IPS Corp.*, 514 F.3d 1271, 1276 (Fed. Cir. 2008) (“We normally do not interpret claim terms in a way that excludes embodiments disclosed in the specification.”).

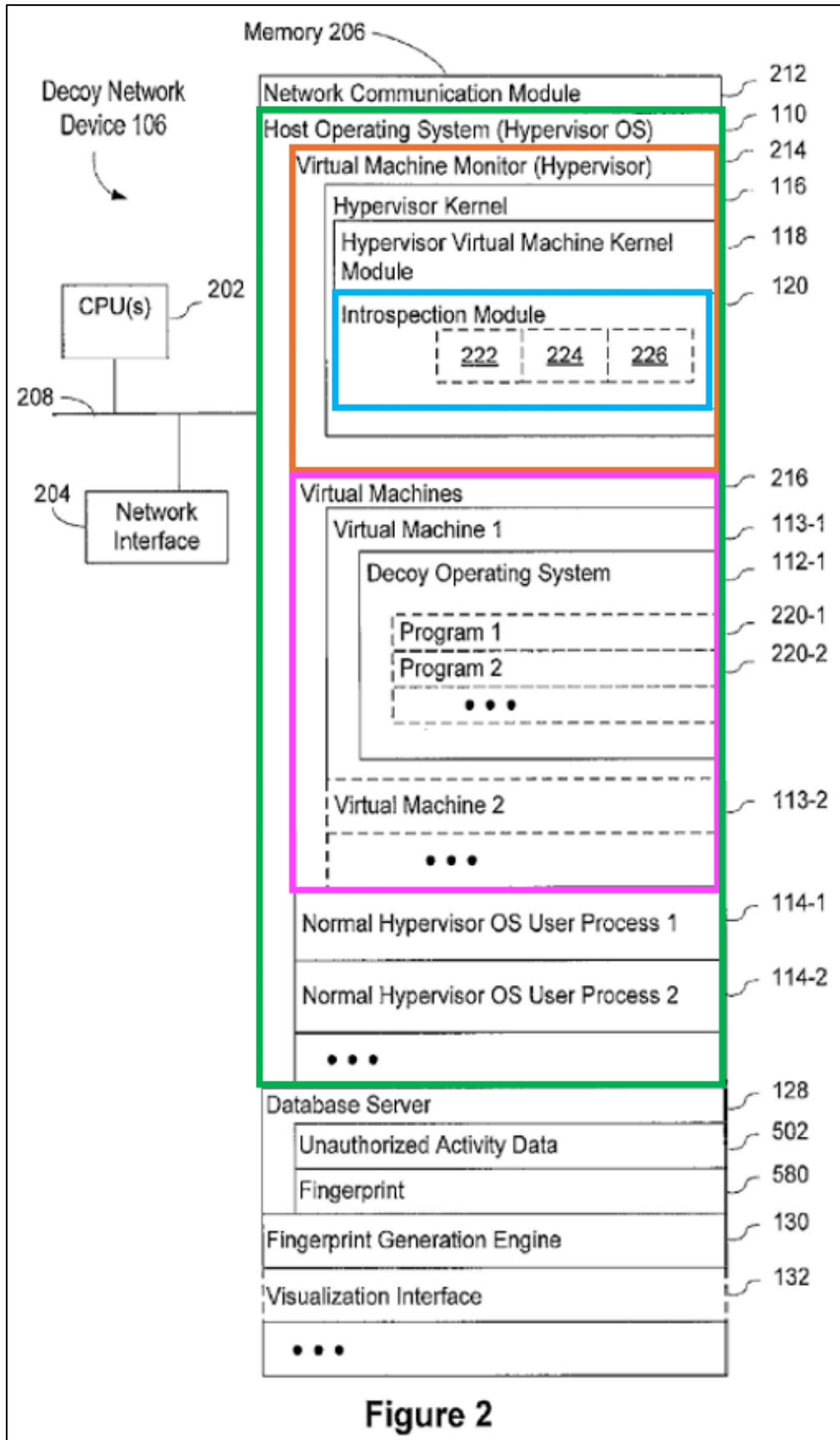
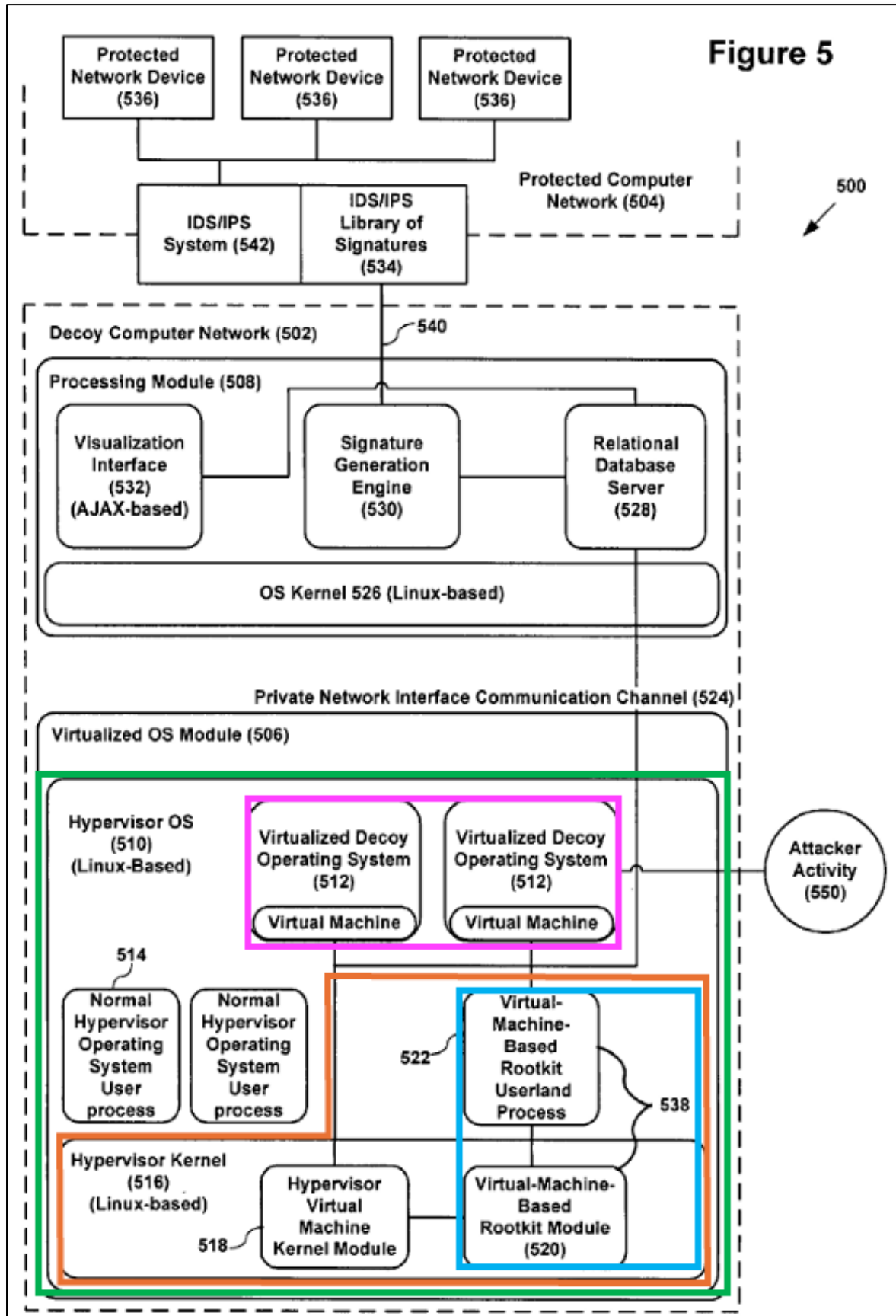


Figure 2

'872 *Patent*, Fig. 2 (annotated). The '872 Patent explains that the “virtual machine monitor 214 includes a hypervisor kernel 116 that resides in physical memory and provides the basic services to the virtual machine monitor 214.” *Id.*, 6:35-37. An introspection kernel (annotated blue), interposed between the VMM and VMs, “performs introspection (e.g., monitoring) into the physical memory segments assigned to each of the [VM operating systems].” *Id.*, 6:46-52.

2. *Capalik's* monitoring activity on the computer system

Capalik's Fig. 5 below depicts the same arrangement, using the same terminology:



Capalik, Fig. 5 (annotated).

Like the '872 Patent, *Capalik* describes Hypervisor OS 510 (annotated green), “also known as a virtual machine monitor operating system[.]” *Capalik*, [0040]. Within Hypervisor OS 510, multiple Virtual Machines (annotated pink) exist in the form of “virtualized operating systems 512 [and] act as decoy operating systems to attract attacker activity 550.” *Id.* Hypervisor kernel 516 (annotated orange) “is that part of the hypervisor operating system 510 that resides in physical memory at all times and provides the basic services to the hypervisor operating system 510.” *Capalik*, [0041]. “[T]he virtual machine-based rootkit module 520 and its associated userland processes 522 constitute an **introspection module 538** [(annotated blue)] (also known as a virtual machine-based memory introspection analysis tool) that **monitors and introspects into the virtualized decoy operating systems memory segments.**” *Capalik*, [0047].

In other words, with almost verbatim terminology and functionality, *Capalik* describes the same computer system as described and claimed in the '872 Patent. Both the '872 Patent and *Capalik* use the term “Hypervisor” interchangeably with virtual machine monitor. Compare '872 Patent, 6:26 (“virtual machine monitor 214 (also called a hypervisor)”), with *Capalik*, [0040] (“hypervisor operating system 510 (also known as a virtual machine monitor operating system)”), *Dec.*, ¶111. And, as

depicted above, both deploy “hypervisor kernels” to provide the basic VMM services, including “introspection modules” to monitor the VM memory segments.

Capalik teaches that Decoy Computer 502’s VMs and VMM are stored in memory. Namely, *Capalik* teaches that module 506—the Virtualized OS Module that includes the depicted VMs and VMM—is stored in memory and that the module is “present in physical memory [RAM] once the system has been booted and is operational.” *Capalik*, [0038]-[0039].

Accordingly, a POSITA would have understood that *Capalik*’s Decoy Computer 502 includes a VM and a VMM that supervises and monitors activity on the first computer system. *Dec.*, ¶¶109-113.

C. Claim [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;”

As discussed above in Section II.C, Petitioner presents the instant grounds to account for the possibility that Patent Owner argues the claimed *association* requires classifying or characterizing the individual activities as authorized or unauthorized (i.e., as benign or malignant).

As discussed below, *Capalik* teaches identifying a plurality of activities performed at the computer system and retaining detailed forensic data that can be used to generate signatures indicative of the activities. *See* Section V.C.2. However, *Capalik* does not expressly disclose classifying or characterizing the individual

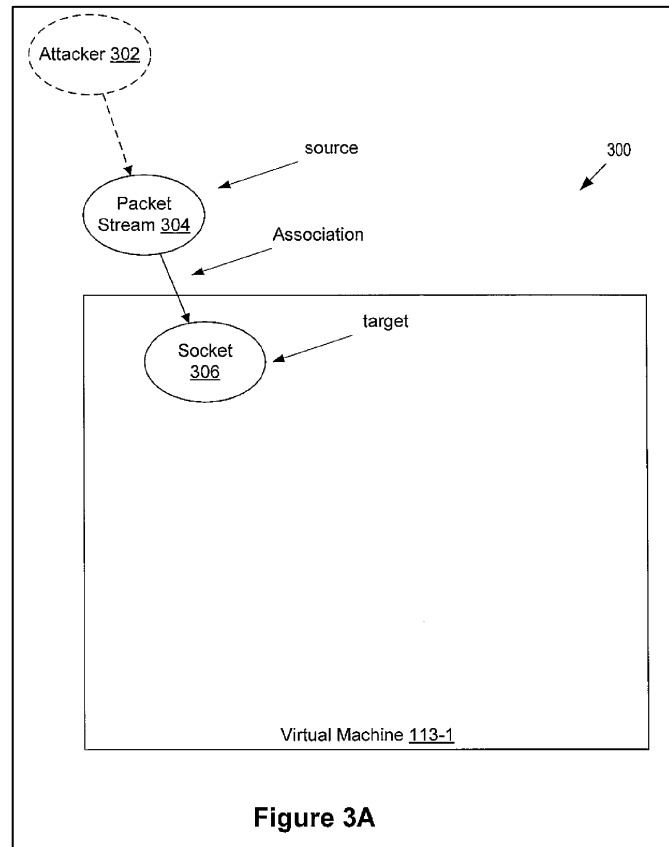
activities of an attack as benign or malignant. *King* teaches a computer monitoring process similar to *Capalik*'s, but additionally teaches that each individual activity related to an attack is classified as benign or malignant before generating a signature indicative of the attack. As set forth below, a POSITA would have been motivated to modify *Capalik* in view of *King* such that the individual activities related to an attack are classified as benign or malignant before generating a signature/fingerprint indicative of the attack. *See* Sections V.C.3-4.

1. The '872 Patent's description of "an activity source, an activity target, and an association between the activity source and the activity target"

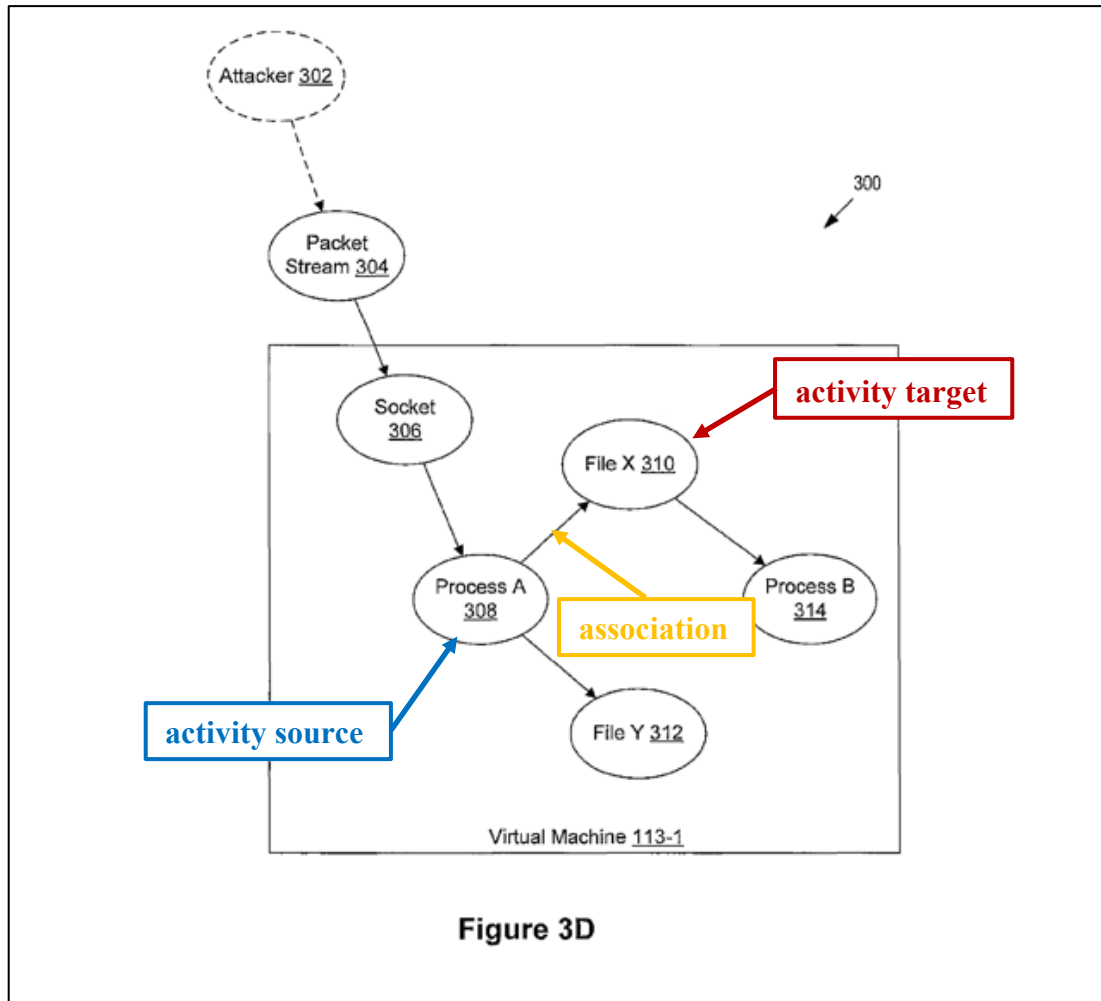
Figures 3A-3D and 4A-4D of the '872 Patent depict network diagrams illustrating a plurality of activities performed at the computer system wherein each activity includes an activity source, an activity target, and an association between the activity source and the activity target. '872 *Patent*, 11:15-22, 15:25-29. An activity source may be "access to a particular file, access from a particular user, an Internet Protocol address, an instruction, a message, data stream, a hardware component, and a network port[.]" *Id.*, 13:21-28. An activity target may be a "file, an Internet Protocol address, an instruction, a message, data stream, a hardware component, and a network port[.]" *Id.*, 13:29-33.

Figure 3A of the '872 Patent, shown below, depicts the start of a network diagram wherein the detected unauthorized activity is an incoming packet stream of

data 304 received by socket 306 (e.g., an attempt by hacker to probe a security system of the virtual machine), in a virtual machine:



Id., Fig. 3A, 11:26-36. Figure 3A of the '872 Patent identifies data packet stream 304 as an activity source, socket 306 as an activity target, and an association between them. A graph is built including all related “activities” that are causally affected by the delivery of data packet stream 304 to socket 306. The resulting network graph is shown in Fig. 3D below:



Id., Fig. 3D (annotated); *Dec.* ¶¶117-119. Relevant to the narrow interpretation of “association” advanced in this Petition, the ’872 Patent teaches that the individual associations can be classified as benign or malignant and this classification can be visually indicated on the network graph. ’872 *Patent*, 12:32-41 (“When the type of associations is identified (e.g., malignant, benign, etc. . .), [] at least a subset of the associations can be visually distinguished (e.g., a respective bold arrow representing a malignant association; a regular arrow representing a benign association . . .).”).

2. *Capalik* teaches “identifying a plurality of activities being performed at the first computer system”

Capalik discloses that when an attacker connects to a network port and begins interacting with virtualized decoy OS 512, *Capalik*'s introspection module 538, “monitors and captures information from the connection, including port numbers, data streams, file uploads, keystrokes, ASCII or binary files, malicious payloads, memory manipulation attempts, and any other data transfers or malicious attempts.” *Capalik*, [0049]. The captured information is then (1) sent to processing module 508, (2) stored in relational database server 528, and (3) used to generate an attack report. *Capalik*, [0050], [0052], [0055]. *Capalik* notes that “the report of the attack is written and then displayed via a visualization interface 532 and can include information about which sockets were accessed by the attack 550, what happened at a particular socket, the keystrokes entered or data transferred, what files were transferred, how the attack 550 was run, what happened on the virtualized operating system module 506, and how the virtualized decoy operating systems 512 [] and any related network services were affected.” *Capalik*, [0056]-[0057].

Capalik teaches that attack 550 may be categorized and analyzed by observing operations performed on decoy OS(s) 512 “to determine, for example, how an attack gained control of the decoy operating system(s) 512. Any method of analyzing the incoming data such as binary matching, neural-network matching, keyword matching, or any other method of matching attack identifying information is in

keeping with the spirit of the invention.” *Capalik*, [0057]. Thus, a POSITA would have understood that *Capalik* teaches identifying a plurality of activities being performed at the computer system. *Dec.*, ¶121.⁵

Capalik does not describe individually classifying the activities as benign or malignant. Instead, *Capalik*’s process more broadly assesses attacks, looking for “known attack patterns as well as non-standard patterns, such as repeating binary patterns, keystroke patterns, downloaded daemons, or errors (such as buffer

⁵ GoSecure has argued in the district court litigation that the claims of the ’872 Patent take “a behavior-based approach that, again, does not rely on matching attacks to fixed signatures[.]” GoSecure Opp’n to Mtn. to Dismiss (Ex. 1025), 7-8. However, none of the Challenged Claims actually recites the negative limitation of “not rely[ing] on matching attacks to fixed signatures[.]” To the extent that GoSecure argues here that the Challenged Claims must be construed to require this negative limitation, and that *Capalik* does not meet that limitation, that argument fails, as *Capalik* itself teaches away from using fixed (or static) signatures. *See Capalik*, [0005] (“The problem lies in the static nature of the conventional IDS and IPS signatures coupled with the ability of determined attackers to launch new undefined or zero-day automated attacks to gain access to the network. While an intrusion prevention system (IPS) equipped with behavioral signatures providing the ability to capture behavioral patterns offers a higher level of protection, these have similar drawbacks in that behavioral signatures are still static in nature and limited in their ability to stop zero-day attacks.”).

overflow attempts, malicious payloads attempting to execute arbitrary code on the system, memory overwriting attempts, stack attacks, and heap attacks).” *Capalik*, [0057]. Alternatively, *Capalik* teaches that all activity that originate at network addresses associated only with the monitored virtual machines may be assumed malicious. *Id.*, [0044].

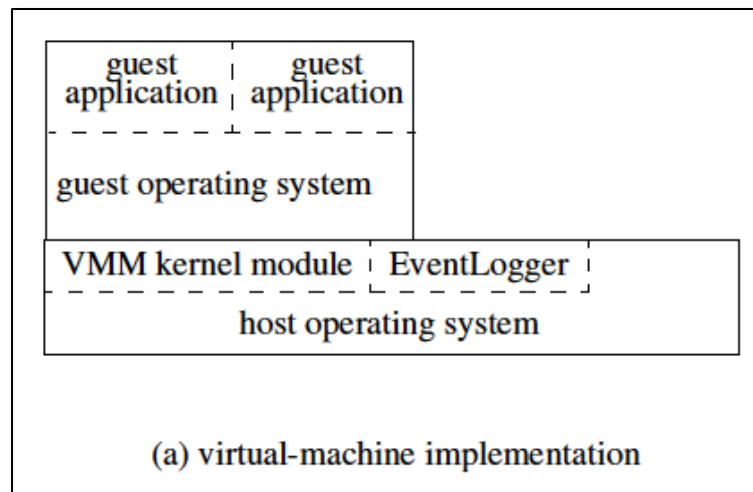
3. ***King* teaches identifying, storing in memory, and graphically displaying “an activity source, an activity target, and an association between the activity source and the activity target.” *King* teaches storing an activity source and activity target in memory.**

King discloses systems and methods for analyzing unauthorized activities using “operating-system-level (OS-level) information-flow graphs to highlight the activities of an attacker” inside a virtual machine. *King*, 3. *King*’s information-flow graphs provide a system-wide perspective of an attack by following the flow of information from one object to another, and the dependency-causing events between objects. *King*, 10, 13. Such objects include “processes, files, filenames, and registry entries.” *King*, 12. The dependency-causing events are based on the “source and sink objects for the dependency they induce: process/process, process/file, process/file-name, and process/registry.” *King*, 14; *Dec.*, ¶123. As discussed below, *King* teaches an EventLogger, which identifies and stores a plurality of activities including an activity source and target for each activity, and “GraphGen,” which processes

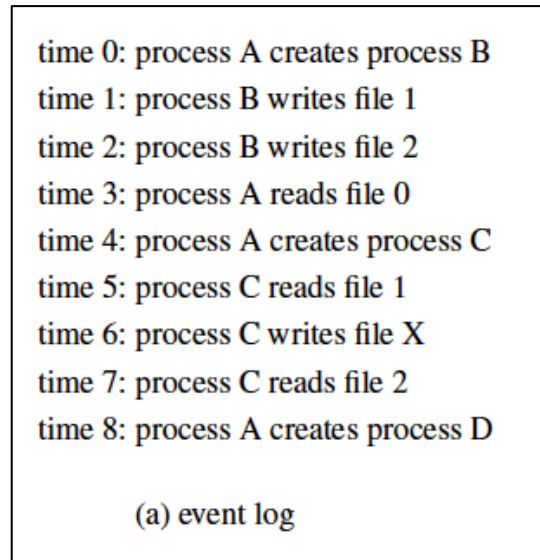
EventLogger logs to identify and graphically display information-flow graphs of an intrusion. *King*, 12.

(a) EventLogger – Identifying and Storing Activity Source and Target

King teaches an “EventLogger” that identifies and stores information-flow events in real-time. *King* discloses that the log written by EventLogger is stored in memory on the host operating system of a virtual machine using either a filesystem or a database. *King*, 23, 25. Fig. 3.3(a) below depicts *King*’s EventLogger implemented in a virtual-machine setup:



King, Fig. 3.3(a). Additionally, *King* teaches an exemplary event log in Fig. 3.2(a), where the event log includes times, sources (e.g., process A, process B, process C), and targets/sinks (e.g., process B, file 1, file 2).



King, Fig. 3.2(a).

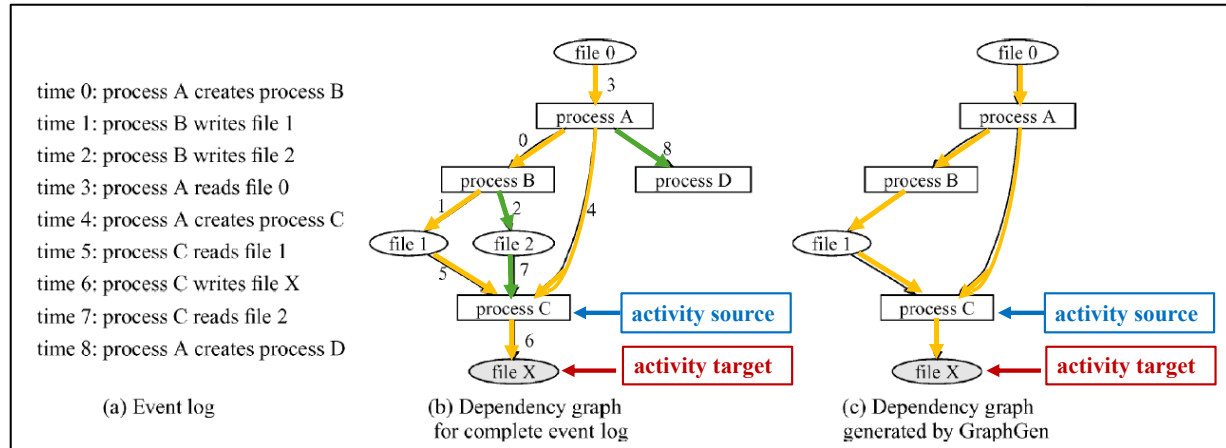
Finally, *King* teaches EventLogger logs events in which one object affects the state of another object (i.e., dependency relationships). *King*, 13. “A dependency relationship is specified by three parts: a source object, a sink object, and a time interval. For example, the reading of a file by a process causes that process (the sink object) to depend on that file (the source object).” *Id.* A POSITA would have understood that *King*’s “source objects” and “sink objects” are activity sources and activity targets, respectively. *Dec.*, ¶125; ’872 *Patent*, 13:21-33.

(b) GraphGen – Identifying and Graphically Displaying Activity Source, Target, and Associations

King teaches that GraphGen builds and graphically displays dependency graphs from an EventLogger log. *King*, 17. The dependency graph can “represent the flow of information either backward in time or forward in time[,]” depending on

which of *King*'s systems (e.g., Backtracker or ForwardTracker) was used to generate the graph. *King*, 17. See Section IV.B; *Dec.*, ¶¶126-130.

GraphGen identifies dependencies between an activity source and target and determines whether the association between the source and target is malicious or benign. As depicted below, the GraphGen of BackTracker then generates a “complete” dependency graph (Fig. 3.2(b)), applies filtering rules to remove activities unrelated to the ultimate unauthorized action, and generates a filtered dependency graph (Fig. 3.2(c)) that includes only the subset of activities that lead backwards from the unauthorized detection point. *King*, 18-19. Figure 3.2(c) below is similar to the '872 Patent's network diagram illustrated in Figs. 4A-4D:



King, Fig. 3.2 (annotated); *Dec.* ¶127.

As illustrated, GraphGen generates an information-flow graph (i.e., dependency graph) by detecting a suspicious object (e.g., file X having the wrong contents at time 10) and following the attack back in time from this suspicious

“detection point” (time 10), tracking the sequence of events and objects that led to the suspicious object. *King*, 17. *King*’s Figure 3.2(b) depicts the log of events as a complete dependency graph including all the events that GraphGen considers relevant to the affected file X. *King*, 17-18. As annotated above, process C represents one activity source that acts on file X the activity target. *Dec.*, ¶128 (noting that each arrow in the dependency graphs represents an activity source and target pair and the association between them). Figure 3.2(c) depicts a filtered graph, removing benign activities and objects that do not affect file X by time 10. *King*, 18-19.

A POSITA would have understood that *King*’s GraphGen graphically displays a plurality of activities performed at the computer system wherein each of the activities includes an activity source, activity target and an association between the activity source and activity target. *Dec.*, ¶129. A POSITA would have understood that each of the events depicted from time 0 to time 8 in Fig. 3.2(a) describe actions (e.g., reading/writing) between respective activity sources and targets. *Dec.*, ¶129. Further, a POSITA would have understood *King*’s Fig. 3.2(b) identifies and displays associations between activity sources and activity targets of each activity, wherein the associations that affect file X by time 10 are malignant (annotated yellow), and the associations that do not affect file X by time 10 are benign (annotated green). *Dec.*, ¶129. *King*’s system prunes away benign associations, resulting in a filtered

graph (Fig. 3.2(c)) depicting only sources and targets with malignant associations. *Dec.*, ¶129.

King's filtered graph may be displayed to a system administrator, allowing the administrator to more efficiently analyze intrusions in a computer system. *King*, 26-28; *Dec.*, ¶130.

4. Motivation to Combine *Capalik-King*

A POSITA would have been motivated to modify *Capalik* per *King's* teachings to identify, store, and graphically display attack activities including the activity sources, activity targets, and associations between them (i.e., benign and malignant activities). *Dec.*, ¶¶131-139.

Modifying *Capalik* as proposed would have resulted in an improved IDS. *Dec.*, ¶¶131-139. This modification applies the well-known computer security principle that gathering more detailed threat information provides improved threat analysis. *Dec.*, ¶132. *King* teaches its system has a “low overhead” for processing power needed to run and perform analysis of its decoy system. *King*, 8-11. Thus, a POSITA would have understood that even if there were a modest increase in processing power required to modify *Capalik* per *King's* teachings, a POSITA would have been motivated to do so to achieve security improvements at the expense of more processing overhead required. *Dec.*, ¶136.

Modifying *Capalik* according to *King*'s teachings would have been use of a known technique of identifying, storing, and graphically displaying activity source, target, and association and storing activity source and target to improve *Capalik*'s similar system in the same way as *King*'s system. *Dec.*, ¶137. Such a modification would have resulted in an improved analysis of unauthorized intrusions in a decoy system. *Dec.*, ¶¶132-137. *King* teaches its methods for identifying, storing and graphically displaying an attacker's unauthorized activities (including activity sources, targets, and associations) in a decoy system allow more efficiently determining which application contained the vulnerability that the attacker exploited and illustrating the pattern of related objects and events impacted by or part of the attack; further, *King* teaches preventing future intrusions by updating or disabling any vulnerable applications. *King*, 2; *Dec.*, ¶137. The modified *Capalik* would also have appreciated these benefits. *Dec.*, ¶137.

A POSITA would have had a reasonable expectation of success when modifying *Capalik* according to *King*'s teachings. *Dec.*, ¶¶138-139. Such a modification would have utilized *Capalik*'s VMM, relational database, and visualization interface which already have monitoring, storing, and graphically displaying capability of unauthorized activities. *Id.* *King* teaches implementing its system in a VMM, which can be achieved by adding programming instructions and requires no changes to the guest OS. *King*, 23-24 ("code for EventLogger is

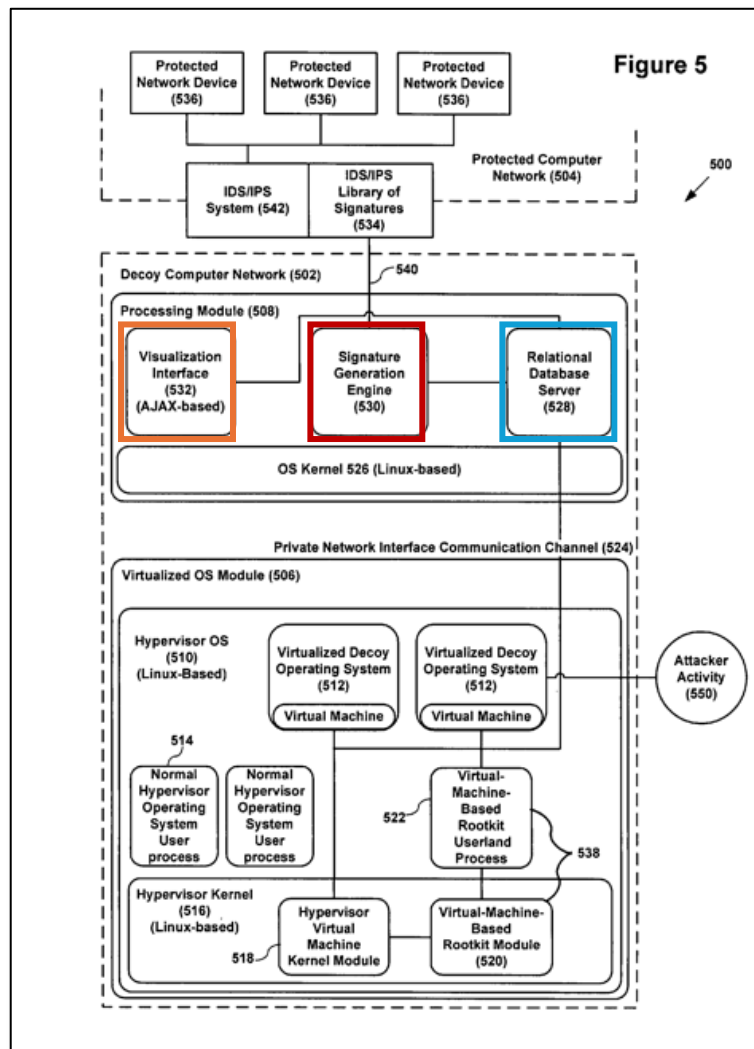
approximately 1300 lines, and we added 40 lines of code to the virtual-machine monitor to support EventLogger. We made no changes to the guest operating system”). Therefore, a POSITA would have understood that modifying *Capalik’s* system to additionally monitor, identify, store, and graphically display “activity sources, activity targets, and associations” would have required straightforward modifications to programs/code instructions, which would have been well within the skillset of a POSITA. *Dec.* ¶¶138-139. A POSITA would have also had a reasonable expectation of success based on the similarities between the decoy devices—both in structure and in function—described by *Capalik* and *King. Id.*

D. Claim [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; and”

As discussed above with respect to Claims [1(Pre)], [1(b)], *Capalik-King* teaches a method and system for storing in a database of a host operating system a data structure identifying a plurality of activities being performed at a computer system, wherein each activity includes an activity source and activity target. As discussed below, *Capalik-King* further teaches storing *in the memory* of the first computer system, a data structure that identifies the activity sources, activity targets, and corresponding associations.

1. *Capalik's Memory*

Capalik discloses storing “attack information” in relational database server 528 (shown in blue) in Decoy Computer 502, as depicted in Fig. 5.



Capalik, Fig. 5 (annotated). *Capalik* teaches virtualized operating system module 506 monitors decoy network 502, and processing module 508 obtains, analyzes, and responds to malicious exploits. *Capalik*, [0038]. Processing module 508 comprises Relational Database Server 528 (annotated blue), which stores “attack information” as forensic data for later analysis. *Capalik*, [0052]; *see also, id.* [0048] (“the

introspection module 538 examines the contents of the decoy operating systems' 512 memory segments in **an instrumented context that generates and retrieves forensic data** for analysis by the processing module 508.”). “The signature generation engine 530 [annotated red] then uses this forensic data to generate a signature of the attack.” *Id.* *Capalik's* “Visualization Interface 532 [annotated orange] which displays a report of the attack, may include information on the location of the instigator of the attack 550, and may be used to analyze, configure, and automate the system's response to attack activity 550.” *Capalik*, [0056]; *Dec.*, ¶141.

As explained in Claim [1(Pre)], the modules within *Capalik's* Decoy Computer 502 utilize memory (e.g., volatile memory (e.g., RAM) and non-volatile memory (e.g., hard disk drive)) to accomplish their respective tasks. *Capalik*, [0039]. A POSITA would have understood that “attack information” stored in Relational Database Server 528 uses system memory (e.g., RAM, hard disk drive), as claimed. *Dec.*, ¶¶143.

A POSITA would have understood or found obvious that *Capalik's* “forensic data” is stored in a data structure, as claimed, at least because the stored data is used by “[t]he signature generation engine 530 [annotated red] . . . to generate a signature of the attack.” *Capalik*, [0052]; *compare with '872 Patent*, 10:33-36; *Dec.* ¶¶144-146 (explaining that storing data such as forensic data in data structures was known

in the art). Indeed, the claims do not impose any specific structure of the stored data, generically recited as “data structure.” Given the breadth of this element and the fact that *Capalik*’s system processes the stored data to generate a signature of the attack, a POSITA would have understood that *Capalik* teaches or suggests storing its forensic data in a “data structure.” *Dec.* ¶¶144-146.

Moreover, *Capalik* teaches “processing module 508 generates a report of the attack that includes attack-identifying information[.]” *Capalik*, [0055]. A “report of the attack is written and then displayed via a visualization interface 532” *Capalik*, [0056]. As explained above at Claim [1(b)], *Capalik*’s attack report includes activity sources and targets, and the corresponding associations. Accordingly, a POSITA would have understood that the “attack information” or “attack-identifying information” stored in relational database server 528 as a forensic data structure, and used to create the attack report, includes activity sources, targets, and associations. *Dec.*, ¶146.

2. Motivation to Combine *Capalik-King* to Store an Association

As explained above in Claim [1(b)], a POSITA would have been motivated to modify *Capalik* per *King*’s teachings to identify, store, and graphically display attack activities including the activity sources, activity targets, and an association between them (i.e., an indication of whether each individual activity is benign or malignant). *See* Claim [1(b)]; *Dec.*, ¶147.

A POSITA would have been motivated to use *Capalik's* native storage infrastructure to store the additional activity data incorporated pursuant to King's teachings, e.g., Relational Database Server 528 (e.g., stored in *Capalik's* RAM or a hard disk drive). *Dec.*, ¶148. The additional storage requirements imposed by the proposed combination are minimal, leading a POSITA to conclude that no meaningful modifications to *Capalik's* native storage would be required. *Dec.*, ¶148.

Further, a POSITA would have had a reasonable expectation of success when modifying *Capalik* to store *King's* associations in memory for the same reasons discussed above. *See* Claim [1(b)]. Namely, such a modification would have merely required straightforward changes to software/programming instructions, which would have been well within the skillset of a POSITA. *Dec.*, ¶149.

- E. Claim [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.”**

As discussed above, *Capalik-King* teaches storing in the memory of the computer system data in a data structure, where the data identifies the activity sources, activity targets, and corresponding associations. *See* Claim [1(c)]. Satisfying this limitation, *Capalik* teaches creating a signature (i.e. fingerprint) from the stored forensic data, which contains information identifying one or more of the activity sources, targets and associations.

The '872 Patent describes generating a fingerprint⁶ indicative of attack activity that “can be used to prevent future attacks[,] by modifying the activity targets[,]” such as “blocking a particular port[.]”⁷ '872 Patent, 13:65-14:16; *see also id.*, 14:42-50 (teaching that fingerprints can also prevent attacks by “blocking activity sources” that match an attack on the virtual machine). The fingerprint can include “one or more of” an activity source, activity target, and other data descriptive of an identified attack. *Id.*, 14:1-4. The patent is clear that fingerprints can include both source and target data or even just source or target data. *Id.*, 14:17-53 (“In some embodiments, the fingerprint [] includes activity targets [] in lieu of, or in addition to, the activity sources[.]”).

1. Capalik’s native process creates the claimed fingerprint

As explained above in Claim [1(c)], *Capalik* teaches a decoy system comprising Relational Database Server 528, which stores “attack information” as forensic data. *Capalik*, [0052]; *see* Claim [1(c)]. Like the '872 Patent, *Capalik* uses a signature generation engine to create a “signature of the attack” from the stored forensic data in a relational database. *Compare Capalik*, [0052], with '872 Patent,

⁶ The '872 Patent also refers to a fingerprint as a signature. '872 Patent, 10:33-55.

⁷ Although *Capalik* and the '872 Patent include many similar examples, both expressly teach generating a fingerprint/signature of an attack that identifies the port (or socket) targeted by the attack. Petitioner highlights this parallel teaching merely as an illustrative example. The Proposed Grounds are not limited to this teaching.

6:10-14; *Dec.*, ¶¶150-154. *Capalik* discloses decoy system processing module 508 generates an attack report containing attack-identifying information, analyzes the information for attack patterns, and generates an attack signature. *Capalik*, [0055]. Like the '872 Patent, *Capalik*'s "signature of the attack" can include a wide variety of information, including "what happened at a *particular socket*, the keystrokes entered or data transferred, what files were transferred, how the attack 550 was run, what happened on the virtualized operating system module 506, and how the virtualized decoy operating systems 512 running on the hypervisor operating system 510 and any related network services were affected." *Capalik*, [0056]. *Capalik*'s Fig. 4 illustrates an "attack signature[,]" formatted in XML for transmission to an IDS/IPS system:

```

<item type="STRING.TCP"> ← 400
  <struct name="STRING.TCP">
    <var name="version" protected="true">4.0</var>
    <array name="signatures">
      <entry dontDelete="true" nda="false" sfr="85">
        <var name="SigName" default="RBOT.CBQ Worm Activity" protected="true" />
        <var name="SIGID" default="5571" protected="true" />
        <var name="SubSig" default="0" protected="true" />
        <var name="AlarmSeverity" default="high" />
        <var name="Enabled" default="True" />
        <var name="EventAction" default="alarm">alarm|reset</var>
        <var name="SigVersion" default="S185" />
        <var name="SigStringInfo" default="RBOT.CBQ" />
        <var name="AlarmThrottle" default="Summarize" />
        <var name="Direction" protected="true" default="ToService" />
        <var name="MinHits" default="1" />
        <var name="Protocol" default="TCP" />
        <var name="RegexString" protected="true"
default="\xb5\x71\xc4\x43\x1e\x34\xce\x3a\x18\x37\xef\xc3\x22\xa1\x20\x1f\x6c\x10\x8f\x6
8\x3d\xfd\xef\x51\x92\xfd\x4f\xbc\x46\xe9\x05\xdc" />
        <var name="ServicePorts" default="445" />
        <var name="StorageKey" default="STREAM" />
        <var name="SummaryKey" default="Axxx" />
        <var name="ThrottleInterval" default="15" />
      </entry>
    </array>
  </struct>
</item>

```

Capalik, Fig. 4; *Dec.*, ¶¶150-158. As illustrated, *Capalik*'s signature includes many details indicative of the activity on a computer system, including identifying the specific port on the computer system through which the attack occurred. *Dec.*, ¶156 (explaining that a POSITA would understand "service port" 445 refers to a TCP port, which is commonly targeted by malicious attacks). *Capalik* further teaches that "any ports, protocols, severity levels, alarm levels, signature name, or any other quantity, may be utilized to inform an IDS or IPS of an attack signature." *Id.*

A POSITA would have understood that *Capalik*'s native signature process creates a fingerprint indicative of the activity on the first computer system. *Dec.*, ¶¶150-158. A POSITA would also have understood that *Capalik*'s native signature

generation process is compatible with *Capalik* as modified by *King*. *Id.* Indeed, as set forth above with reference to Claims [1(b)] and [1(c)], the proposed combination supplements *Capalik*'s native process by additionally characterizing the individual activities comprising an attack. A POSITA would have understood that such supplemental information would have improved the effectiveness of *Capalik*'s signature process, allowing more precise attack definitions. *Dec.*, ¶¶131-139, 156.

Once the signature is created, *Capalik* discloses transmitting its signature (i.e. fingerprint) in a similar manner as the '872 Patent. The '872 Patent explains that “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.” '872 *Patent*, 18:47-51. *Capalik* similarly teaches that once the attack signature is generated, it can be sent “to the intrusion detection system (IDS) or intrusion prevention system (IPS) for the protected network 504.” *Capalik*, [0059], Fig. 5; *see also id.*, [0052] (teaching that the entire process may occur automatically, without human intervention). The attack signature is then applied to the IDS/IPS library of signatures, updating the library, to prevent an attacker from accessing protected network 504 in a future similar attack. *Capalik*, [0059], *see also id.*, [0053] (teaching that “protected computer network 504 includes an IDS/IPS library of signatures 534 and an IDS/IPS system 542 coupled to multiple protected network devices 536”), *Dec.* ¶¶157-158 (explaining that a POSITA would

have understood that signatures updated in the signature library are also transmitted to the multiple protected network devices 536 coupled to the IDS/IPS system, ensuring those network devices receive protection consistent with the most up to date signatures collected in the library). Accordingly, a POSITA would have understood that *Capalik* transmits the fingerprint, including information identifying one or more of the activity sources, the activity targets, and the associations to prevent future attacks that comprise the same or similar activities as indicated by the fingerprint. *Dec.*, ¶¶157-158. A POSITA would have further understood that *Capalik* transmits this information to a computer system other than computer decoy 502 since the signature is sent “through a standard network connection” to IDS/IPS 534 of protected network 504. *Capalik*, [0059], Fig. 5; *Dec.* ¶158.

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

As explained above, *Capalik* teaches monitoring activity on a first computer system. *See* Claim [1(b)]. *Capalik* further teaches monitoring all activity on the first computer system. *Capalik*, [0059] (“[T]he processing module 508 may generate a signature *whenever data is found to be transferred through the socket.*”), [0049] (“When an attacker connects to a network port and begins interacting with a virtualized decoy operating system 512, the introspection module 538 monitors and captures information from the connection, including port numbers, data streams, file uploads, keystrokes, ASCII or binary files, malicious payloads, memory

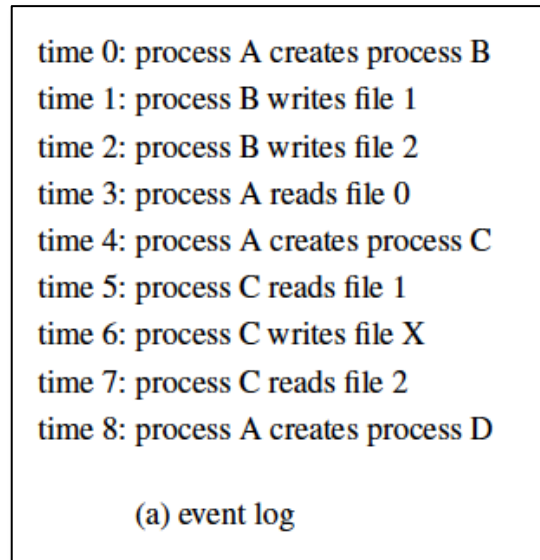
manipulation attempts, and any other data transfers or malicious attempts.”); *Dec.*, ¶¶159-160 (explaining that a POSITA would have understood *Capalik*’s detailed forensic data monitors all activity on the VM that begins with a connection through a VM network port). Accordingly, a POSITA would have understood that *Capalik* teaches monitoring all activity on the computer system. *Dec.*, ¶¶159-160.

- G. Claim 3: “The method of claim 1, further comprising: creating, from the stored information, a fingerprint indicative of the activity on the first computer system.”**

See Claim [1(d)].

- H. Claim 4: “The method of claim 1, wherein the activities comprise two or more of: a file initiating an execution of an instruction, an instruction reading a file, an instruction writing to a file, and receiving data through the network.”**

As explained above, *Capalik-King* teaches identifying a plurality of activities being performed in a computer system, wherein each of the activities includes an activity source, target and association. Claim [1(b)]. As also discussed above with reference to Claim [1(b)], *King* teaches an exemplary event log of the identified activities in Figure 3.2(a):

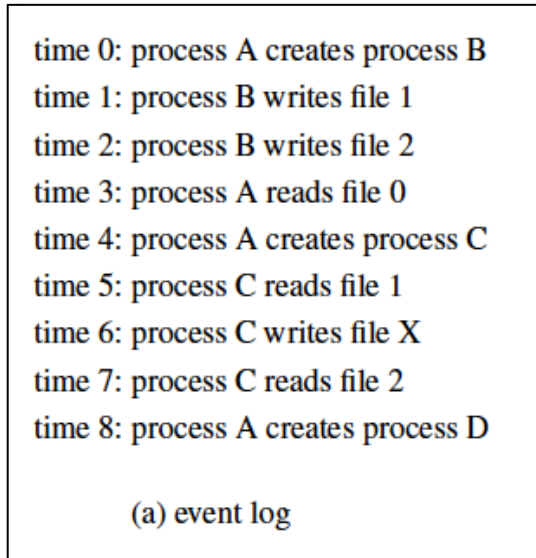


King, Fig. 3.2(a). The event log includes times, sources (e.g., process A, process B, process C), targets/sinks (e.g., process B, file 1, file 2), and dependency relationships between the sources and the targets (e.g., process B writes file 1, process C reads file 1). A POSITA would have understood that *Capalik-King*'s activities comprise at least an instruction reading a file (e.g., process C reads file 1), and an instruction writing to a file (e.g., process B writes to file 1). *Dec.*, ¶162.

I. Claim 5: “The method of claim 1, wherein a respective activity target comprises one of: a file, a user, an instruction, a thread, data stream, and network socket connections.”

See Claim [1(b)]. *Capalik* expressly teaches an activity target that is a network socket. *Capalik*, [0059]. Further, pursuant to *King*'s express teachings, the combination of *Capalik* and *King* teaches an activity target comprises at least a file or process (i.e., an instruction or a thread). *Dec.*, ¶163 (explaining that a POSITA

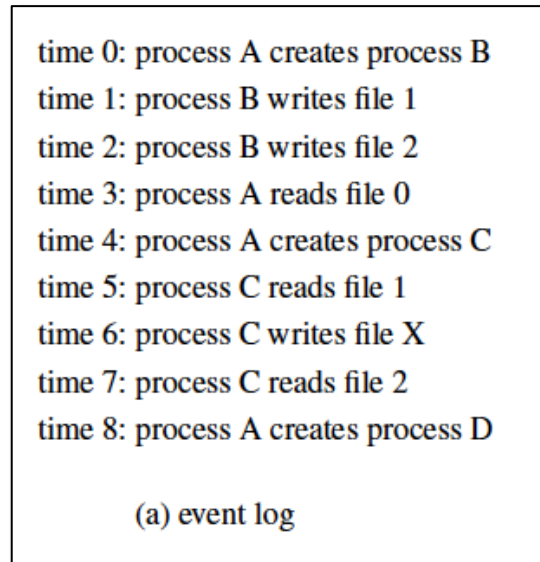
would have understood that an instruction is equivalent to a file, and a thread is equivalent to a process).



King, Fig. 3.2(a).

J. Claim 6: “The method of claim 1, wherein a respective activity source comprises one of: a file, a user, an instruction, a thread, data stream, and network socket connections.”

See *supra* Claim [1(b)]. *Capalik* expressly teaches an activity source that is a data stream. *Capalik*, [0049]. Further, pursuant to *King*’s express teachings, the combination of *Capalik* and *King* teaches an activity source comprises at least a file or process (i.e., an instruction or a thread). *Dec.*, ¶164.



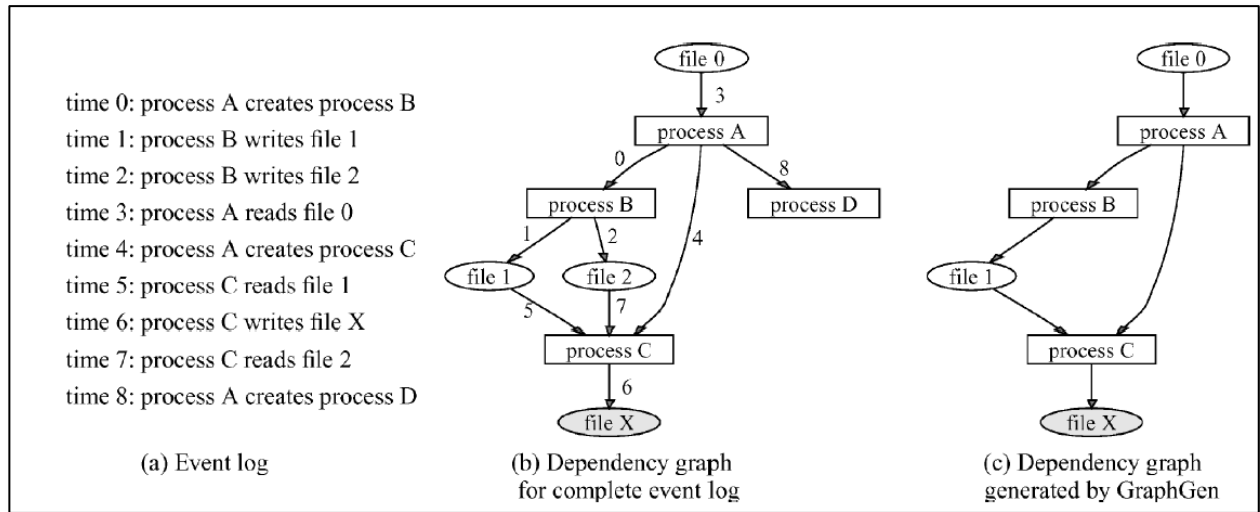
King, Fig. 3.2(a).

K. Claim 7: “The method of claim 1, further comprising: graphically displaying at least a subset of the stored activities.”

As explained above with respect to Claim [1(b)]-[1(c)], *Capalik-King* teaches identifying, storing, and graphically displaying a plurality of activities performed at the computer system, wherein each of the activities includes an activity source, target, and association. As explained below, *Capalik-King* further teaches graphically displaying at least a subset of the stored activities.

The '872 Patent explains that “the decoy network device graphically displays (626) at least a subset of the stored activities. For example, the graphical representation can be an activity diagram as shown in FIGS. 3A-3D and 4A-4E.” *'872 Patent*, 16:62-65.

As explained above with respect to Claim [1(b)], *King's* information flow graphs are similar to Figs. 3A-3D and 4A-4D of the '872 Patent.



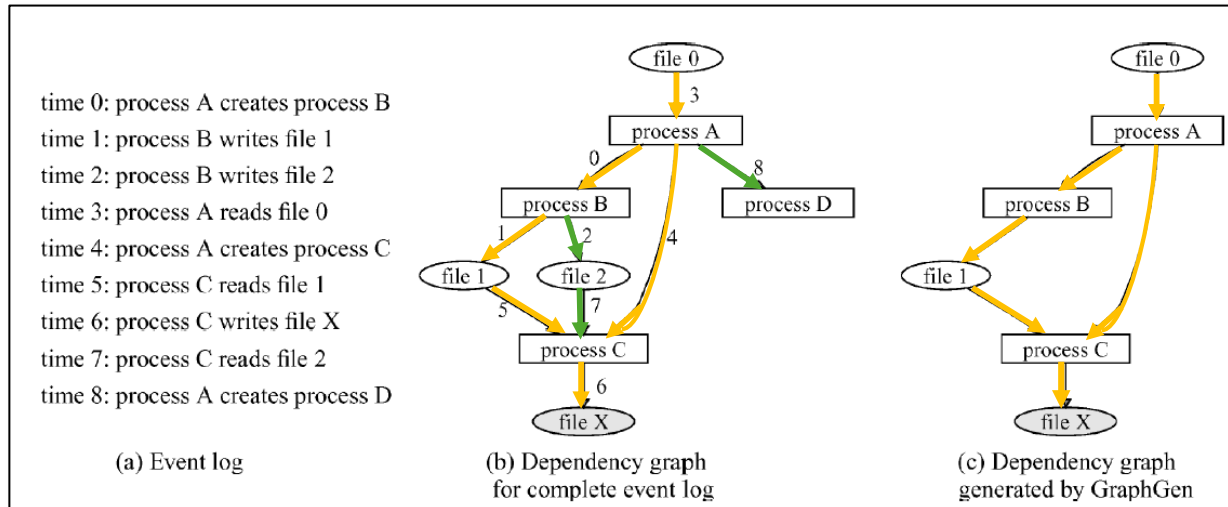
King, Fig. 3.2; *Dec.* ¶¶165-171. A POSITA would have understood that *King*'s dependency graphs, as depicted in Figure 3.2, graphically display at least a subset of the stored activities, including activity sources, targets and associations. *Dec.*, ¶¶165-171.

L. Claim 8: “The method of claim 7, wherein the displaying includes visually distinguishing at least a subset of the associations.”

As explained above in Claim 8, *Capalik-King* teaches dependency graphs that graphically display a subset of the stored activities. As explained below, *King*'s dependency graphs further visually distinguish at least a subset of the associations.

The '872 Patent explains that when the types of associations are identified (e.g., malignant, benign), at least a subset of the associations can be visually distinguished (e.g., a bold arrow representing a malignant association; a regular arrow representing a benign association), and that any graphical scheme can be used to visually distinguish the subset of the associations. '872 *Patent*, 12:32-41.

As explained in [1(b)], *King's* complete dependency graph (Fig. 3.2(b)) depicts malignant associations (annotated yellow) and benign associations (annotated green). And the filtered graph (Fig. 3.2(c)) depicts a subset of the associations, depicting only those associations classified as malignant:



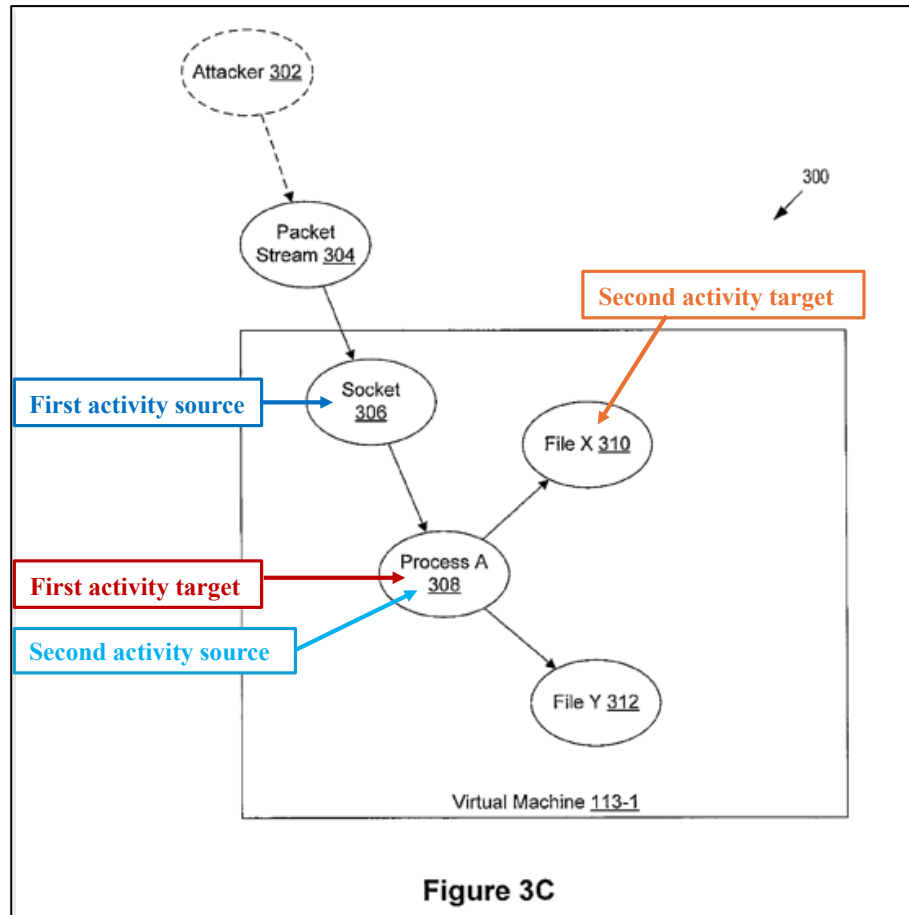
King, Fig. 3.2 (annotated); *Dec.*, ¶¶172-175. Accordingly, a POSITA would have understood that *King's* dependency graphs visually distinguish a subset of the associations by depicting only malignant associations in a filtered dependency graph. *Dec.*, ¶¶172-175. Accordingly, a POSITA would have understood that *Capalik-King* teaches graphically displaying at least a subset of the stored activities, wherein the displaying includes visually distinguishing at least a subset of the associations. *Dec.*, ¶¶172-175.

M. Claim 9: “The method of claim 1, wherein an activity target of a first activity of the plurality of activities is an activity source of a second activity of the plurality of activities.”

As explained above, *King*’s dependency graphs identify a subset of activities wherein each activity includes an activity source, activity target, and association. *See* Claim 7. As explained below, *Capalik-King* further teaches that the subset of activities includes a first activity and a second activity, and the activity target of the first activity is the activity source of the second activity.

1. ’872 Patent’s Teachings

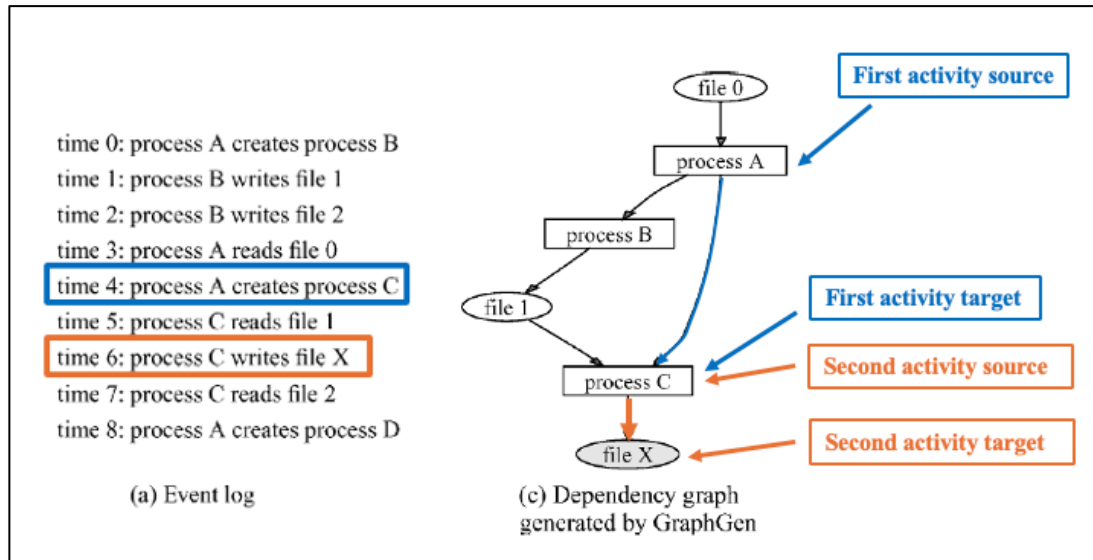
The ’872 Patent points to Figure 3C (below) as an exemplary embodiment of this Claim:



'872 Patent, 17:8-19, Fig. 3C (annotated); Dec. ¶177. The '872 Patent explains that figure 3C illustrates a first activity between the socket 306 and the process A 308 (i.e., data from socket 306 is directed to process A 308) and a second activity between the process A 308 and the file X 310 (i.e., Process A 308 accesses File X). The process A 308 is the activity target of the first activity (annotated in red) and the activity source of the second activity (annotated light blue). *Id.*, 17:11-15.

2. King's Teachings

Similar to the '872 Patent, *King* depicts a similar subset of activities in dependency graph 3.2(c):



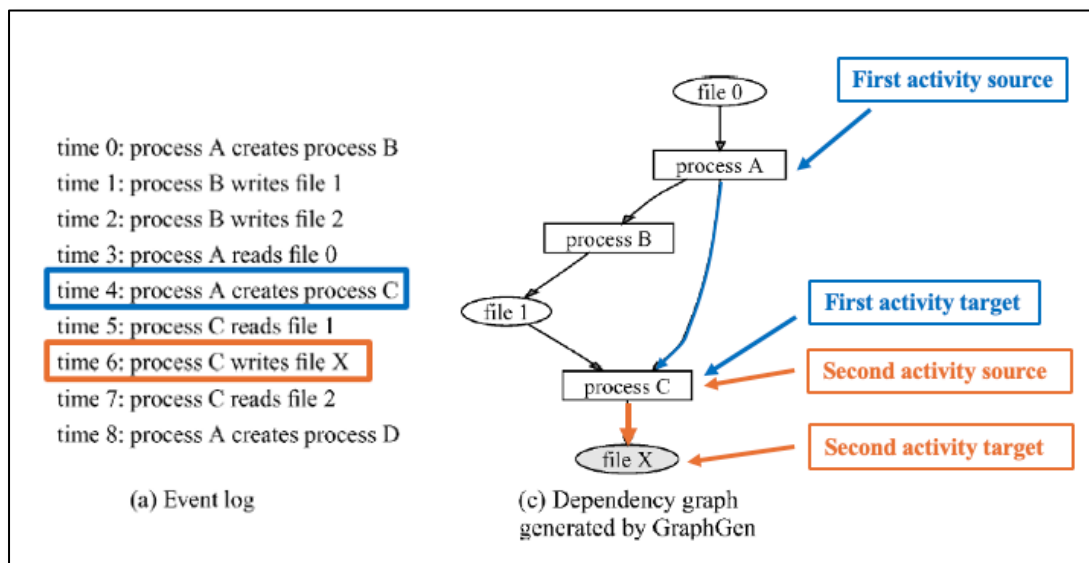
King, Fig. 3.2 (annotated); *Dec.*, ¶¶176-179. The annotations above highlight a first unauthorized association (blue) between a first activity source (process A) and a first activity target (process C). Namely, “process A creates process C” at time 4. The annotations above also highlight a second unauthorized association (orange) between a second activity source (process C) and a second activity target (file X). Namely, “process C write file X” at time 6. Accordingly, *King* teaches a first activity and a second activity where *process C* is the activity target of the first activity and the activity source of the second activity. *Dec.*, ¶¶176-179.

As discussed above, a POSITA would have been motivated to modify *Capalik* per *King*’s teachings and would have had a reasonable expectation of success. See Claim [1(b)]-[1(d)]. *Dec.*, ¶¶176-179.

- N. **Claim 10: “The method of claim 1, further comprising: identifying a respective association between a first activity source and a first activity target as unauthorized, wherein the first activity target, when associated as a second activity source with a second activity target, causes unauthorized activities on the second activity target.”**

As explained above, *Capalik-King* teaches identifying a plurality of activities performed at a first computer system, wherein each of the activities includes an activity source, activity target, and association. *See* Claim [1(b)]. As explained below, *Capalik-King* further teaches (i) identifying a respective association between a first activity source and a first activity target as unauthorized, and (ii) that unauthorized activity is caused between the second activity source and target when the first activity target is the second activity source.

King depicts a subset of activities and associations in dependency graph 3.2:



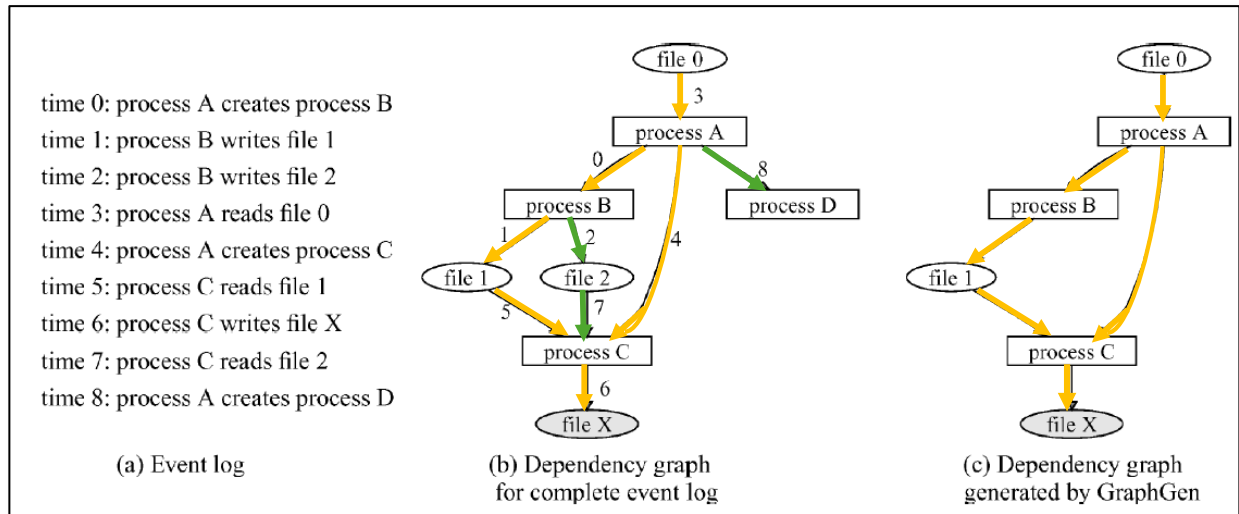
King, Fig. 3.2 (annotated); *Dec.*, ¶¶180-184. The annotations above highlight a first unauthorized association (blue) between a first activity source (process A) and a first

activity target (process C). Namely, “process A creates process C” at time 4. The annotations above also highlight a second unauthorized association (orange) between a second activity source (process C) and a second activity target (file X). Namely, “process C write file X” at time 6. As set forth above regarding Claim [1(c)], all associations depicted in King’s Fig. 3.2(c) are unauthorized/malignant. So, a POSITA would have understood that both associations annotated above are unauthorized. *Dec.*, ¶¶180-184.

Accordingly, *King* teaches (i) identifying a respective association between a first activity source (e.g., process A) and a first activity target (e.g., process C) as unauthorized, and (ii) that unauthorized activity is caused between the second activity source (e.g., process C) and target (e.g., file X). In at least the example discussed above, process C is both the first activity target and the second activity source, as claimed. *Dec.*, ¶¶180-184.

O. Claim 11: “The method of claim 10, further comprising: tracking a transfer of malicious associations over time.”

See Claim 10. As explained above, *Capalik-King* teaches identifying and tracking a plurality of activities over time, wherein each activity includes an activity source, activity target, and an association. *See* Claim [1(b)]. As explained in [1(b)], *King*’s dependency graphs distinguish between malignant and benign associations, and a filtered graph (Fig. 3.2(c)) depicts only malignant associations and the times at which they occur:



King, Fig. 3.2 (annotated), *Dec.*, ¶185. Accordingly, a POSITA would have understood *Capalik-King* teaches tracking a transfer of malicious associations over time. *Dec.* ¶185.

P. Claim 12: “The method of claim 1, further comprising: identifying activity sources that are affected by unauthorized activities.”

As explained above, *Capalik-King* teaches identifying a plurality of activities performed on a computer system, wherein each of the activities includes an activity source, target and association. *See* Claim [1(b)]. Because the activities include unauthorized activities, a POSITA would have understood that *Capalik-King* also teaches identifying activity sources that are affected by unauthorized activities. *Dec.*, ¶¶186-187; *see* Claim [1(Pre)], [1(b)].

- Q. Claim 14: “The method of claim 12, wherein: instructions executed by the one or more processors have respective privilege levels, respective activity sources have respective privilege levels; and the identifying activity sources that are affected by unauthorized activities includes identifying activity sources that request to execute underprivileged instructions.”**

See Claim 12. *Capalik-King* teaches one or more processors. *See* Claim 1. As discussed below, *Capalik-King*’s one or more processors execute instructions at various privilege levels (i.e., respective privilege levels). *King* discloses a simulated attack where Backtracker generates a dependency graph that identifies activity sources requesting to execute underprivileged instructions. *King*, 35, Fig. 4.5. The attack depicted in Figure 4.5 includes an attack that gains unprivileged access via httpd, then downloads and compiles a program (sxp) that overflows an argument buffer and provides a root shell. *King*, 35. The root shell is used to add a privileged user to the password files, log into the machine using this new user, and modify /etc/xinetd.conf (the detection point for this attack). *Id.* A POSITA would have understood that *King*’s graph in Fig. 4.5 identifies at least an activity source that requests to execute underprivileged instructions, as claimed. *Dec.*, ¶¶188-192.

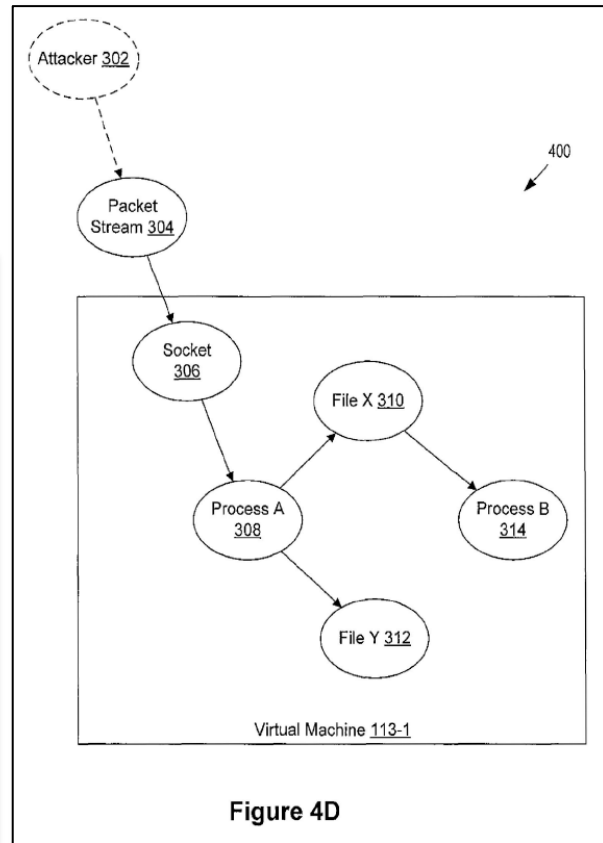
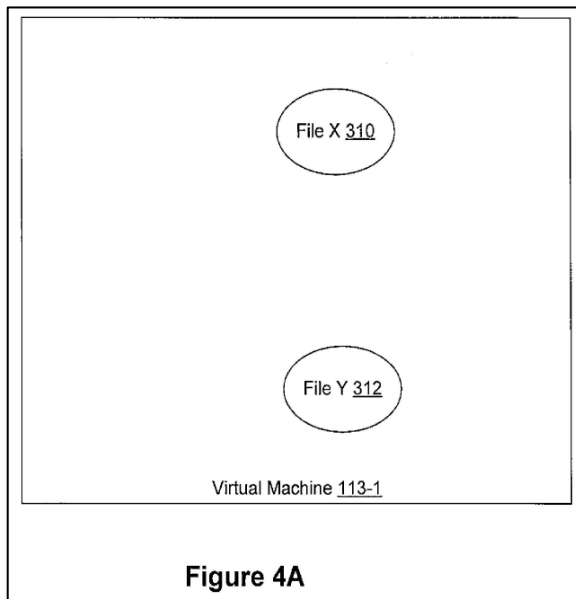
- R. Claim 15: “The method of claim 12, wherein the identifying activity sources that are affected by unauthorized activities includes identifying data stream incoming from outside the first computer system.”**

See Claim 12. *Capalik* discloses identifying activity sources that are affected by unauthorized activities including identifying data streams incoming from outside

the computer system. *Capalik*, [0049] (“When an attacker connects to a network port and begins interacting with a virtualized decoy operating system 512, the introspection module 538 monitors and captures information from the connection, including port numbers, ***data streams***, file uploads, keystrokes, ... and any other data transfers[.]”); Fig. 5 (depicting attacker’s data stream incoming from outside Decoy Computer 502); *Dec.*, ¶193.

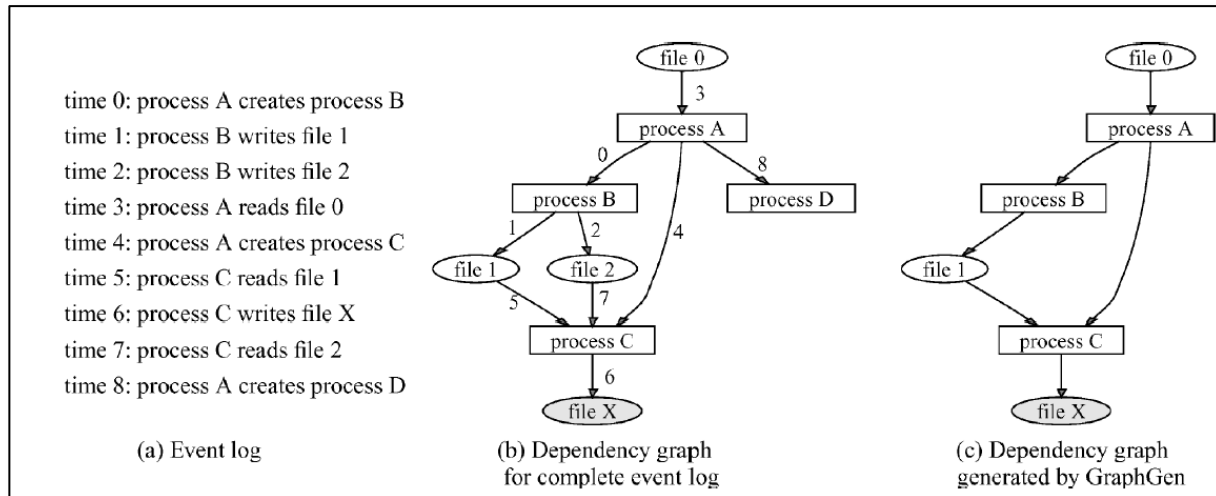
- S. **Claim 16: “The method of claim 1, further comprising: determining an activity status for the stored events, the activity status comprising one of: a complete status, representing that a source of unauthorized activities is identified, and each activity target in the stored activities is associated with the source through one or more associations; and an incomplete status, which represents that the source of unauthorized activities is not identified, or at least one activity target in the stored activities is not associated with the source through one or more associations.”**

The ’872 Patent explains that an example of a complete status is depicted in Fig. 4D, wherein “each activity is associated with the source, packet stream 304 through one or more associations[.]” ’872 *Patent*, 17:23-28. By contrast, Fig. 4A illustrates an incomplete status in which at least one target is not associated with a source. These incomplete (left) and complete (right) statuses are depicted below:



'872, Figs. 4A-D.

As explained above with respect to Claim [1(b)]-[1(d)], *Capalik-King* teaches identifying, storing, and displaying unauthorized activities wherein each of the activities includes an activity source, target, and association. Like Fig. 4D in the '872 Patent, *King's* Fig. 3.2(c) illustrates a complete activity status for an attack. As illustrated below, *King's* system identifies a source of unauthorized activities (e.g., file 0) and that source to every activity target (e.g., Process B, file 1, process C, file X) through one or more identifies associations.



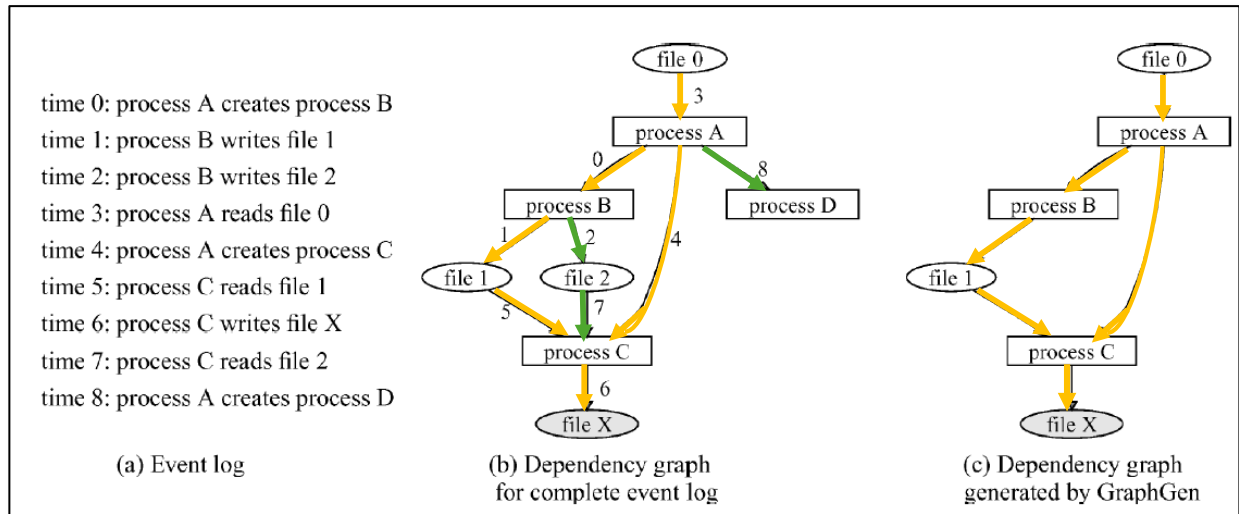
Id., Fig. 3.2; *Dec.*, ¶¶194-196 (explaining that *King*'s process presumes to have identified every association between each source and target before benign associations are filtered by the GraphGen process). *Dec.*, ¶¶194-198.

- T. Claim 18: “The method of claim 1, further comprising: identifying a respective activity as unauthorized based on a determination that the respective activity is associated with an unauthorized activity that is distinct from the respective activity.”**

See Claims 9-10.

- U. Claim 19: “The method of claim 1, wherein the association includes a type that represents whether or not the activity source has modified the activity target to perform the unauthorized activities.”**

As explained above in Claim 7, *Capalik-King* teaches identifying a type of the association between the activity source and the activity target, as depicted in *King*'s dependency graph generated by GraphGen:



Id., Fig. 3.2 (annotated); *Dec.*, ¶200. As discussed above in Claim 7 *King's* GraphGen generates an information-flow graph (i.e., dependency graph) by detecting a suspicious object (e.g., file X having the wrong contents at time 10) and following the attack back in time from this suspicious “detection point” (time 10), tracking the sequence of events and objects that led to the suspicious object. *King*, 17-18 *King's* Figure 3.2(b) depicts the log of events as a complete dependency graph including all activities occurring before the suspicious content was identified in file X. *King*, 17-18. Figure 3.2(c) depicts a filtered graph, ignoring events and objects that do not affect file X as of time 10. *King*, 18-19. In other words, *King's* filtered dependency graph in Fig. 3.2(c) removes any associations that did not contribute (directly or indirectly) to the unauthorized activity identified at time 10 (i.e., suspicious content written into file X). *Dec.*, ¶200. A POSITA would have understood that the associations remaining after this filtering are of a type that

represents a source modifying its target to perform unauthorized activities either directly (e.g., writing incorrect content into file X) or indirectly (e.g., causing a subsequent source to write incorrect content into file X). *Id.*

V. Claim [20(Pre)]: “A first computer system, comprising: one or more processors; and memory storing one or more programs, the one or more programs including instructions for:”

See Claim [1(Pre)], [1(c)]. As discussed above in Claims [1(Pre)]-[1(d)] and [3], *Capalik-King* teaches the following components: (1) hypervisor OS (comprising an introspection module); (2) relational database server; and (3) fingerprint generation engine. A POSITA would have understood that each of these components comprise one or more programs, respectively including instructions for (1) monitoring activity on the computer system and identifying a plurality of activities being performed at the computer system, where each of the activities includes an activity source, target, and an association; (2) storing in the memory the activity sources, targets, and associations; (3) creating, from the stored activities, a fingerprint, and transmitting the fingerprint to prevent future similar attacks. *Dec.*, ¶¶201-202. The programs of these components include instructions for performing the claimed steps below. *See* Claim [20(a)]-[20(d)].

W. Claim [20(a)]: “monitoring activity on the first computer system;”

See Claim [1(a)].

- X. Claim [20(b)]: “identifying a plurality of activities being performed at the first computer system, where each of the activities includes an activity source, an activity target, and an association between the activity source and the activity target;”**

See Claim [1(b)].

- Y. Claim [20(c)]: “storing in the memory a data structure that identifies the activity sources, the activity targets, and the associations for the plurality of activities; and”**

See Claim [1(c)].

- Z. Claim [20(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.”**

See Claim [1(d)].

- AA. Claim [21(Pre)]: “A computer readable storage medium, including one or more programs for execution by one or more processors of a first computer system, the one or more programs including instructions for:”**

See Claims [1(Pre)]-[1(a)], [20(Pre)] (mapping one or more programs including instructions for performing the claimed steps, where the mappings for said programs also apply for Claim 21); *Dec.*, ¶¶207-208. To the extent the preamble is limiting, a POSITA would have understood that *Capalik’s* decoy system discussed in Claim [1(Pre)] includes a computer readable storage medium. *Dec.*, ¶¶207-208.

- BB. Claim [21(a)]: “monitoring activity on the first computer system;”**

See Claim [1(a)].

- CC. Claim [21(b)]: “identifying a plurality of activities being performed at the first computer system, where each of the activities includes an activity source, an activity target, and an association between the activity source and the activity target;”**

See Claim [1(b)].

- DD. Claim [21(c)]: “storing in memory of the first computer system a data structure that identifies the activity sources, the activity targets, and the associations for the plurality of activities; and**

See Claim [1(c)].

- EE. Claim [21(d)]: “transmitting to one or more computer systems other than the first computer system information identifying one or more of the activity sources, activity targets, and 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.”**

See Claim [1(d)].

VI. GROUND 2: *CAPALIK* IN VIEW OF *KING*, AND IN FURTHER VIEW OF *PIKE* RENDERS CLAIMS 13 OBVIOUS

- A. Claim 13: “The method of claim 12, wherein the identifying activity sources that are affected by unauthorized activities includes identifying activity sources that request to access a portion of memory that is set as non-executable.”**

Capalik-King teaches identifying activity sources that are affected by unauthorized activities. *See* Claim 12. As explained below, *Capalik-King-Pike* teaches identifying activity sources affected by unauthorized activities that request to access a portion of memory that is set as non-executable.

The '872 Patent explains that, in some embodiments, identifying activity sources that request to access a portion of memory that is set as non-executable can

be achieved by data execution prevention (DEP) mechanisms, wherein microprocessors allow setting one or more portions of the memory as non-executable (e.g., using a non-executable bit in respective page table entries), and generate an exception when there is an attempt to execute an instruction that is set as non-executable. '872 *Patent*, 16:8-23 (discussing that DEP mechanisms were well-known in the art). As discussed below, a POSITA would have understood that *Pike* teaches one such DEP mechanism. *Dec.*, ¶¶213-217.

As explained above in Section IV.C, *Pike* discloses scanning “input strings” (e.g. data packets), identifying subsequences in the input strings that could be maliciously exploited, and configuring the target computer system to trigger a fault (or other triggered event) when certain execution conditions suggest use of data or code implicated by the identified subsequences. *Pike*, 3:12-19, 5:13-38, 8:65-67. *Pike* discloses several techniques to trigger an action when potentially dangerous data in the input string accesses memory of the target system. *Pike*, 8:65-9:5. *Pike* teaches one technique includes setting a relevant no-execute bit that triggers an exception. *Id.* An exception handler is used to evaluate the likelihood that a particular exception was caused by a dangerous interpretation of a previously received input string. *Pike*, 9:5-12. A POSITA would have understood that potentially dangerous input strings (e.g., data packets) are activity sources affected by unauthorized activities. *Dec.*, ¶¶213-217. Further, a POSITA would have understood that *Pike*'s

technique of setting a relevant no-execute bit to identify potentially dangerous input strings is a DEP mechanism used for identifying activity sources affected by unauthorized activities. *Dec.*, ¶¶213-217. Accordingly, a POSITA would have understood that *Pike* teaches identifying activity sources affected by unauthorized activities that request to access a portion of memory that is set as non-executable. *Dec.*, ¶¶213-217.

1. Motivation to Combine *Capalik-King-Pike*

A POSITA would have been motivated to modify *Capalik-King* per *Pike*'s teachings such that *Capalik-King-Pike* would have identified activity sources affected by unauthorized activities that request to access a portion of memory that is set as non-executable. *Dec.*, ¶¶218-221. As explained above, *Capalik-King* teaches methods and systems that identify a plurality of unauthorized activity sources but fails to expressly describe identifying unauthorized activity sources that request access to a portion of a memory set as non-executable of the target computer system. *See* Claim 12.

A POSITA would have recognized modifying *Capalik-King* per *Pike*'s teachings would allow users to identify a larger variety of unauthorized activity sources, including those that employ infection mechanisms that request access to the memory (including memory set as non-executable) of a target system. *Dec.*, ¶¶218-220. *Pike* discloses using various techniques to identify activity sources that request

access to a system's memory, where one of these techniques includes using no-execute bit (NX bit) technology, which entails marking certain portions of the system's memory as non-executable. *Pike*, 8:65-9:5. Using NX bit technology, also known as Data Execution Prevention (DEP), for malware detection, per *Pike*, was well known in the art. *Dec.*, ¶¶214-220. Thus, such modification would have been use of a known technique to identify unauthorized activity sources that request to access a portion of memory that is set as non-executable. *Dec.*, ¶¶218-220. Although supplying an NX bit could impose additional overhead on the system, a POSITA would have been motivated to accept lower efficiency in exchange for a more robust detection of potential malware attacks. *Dec.*, ¶¶218-220.

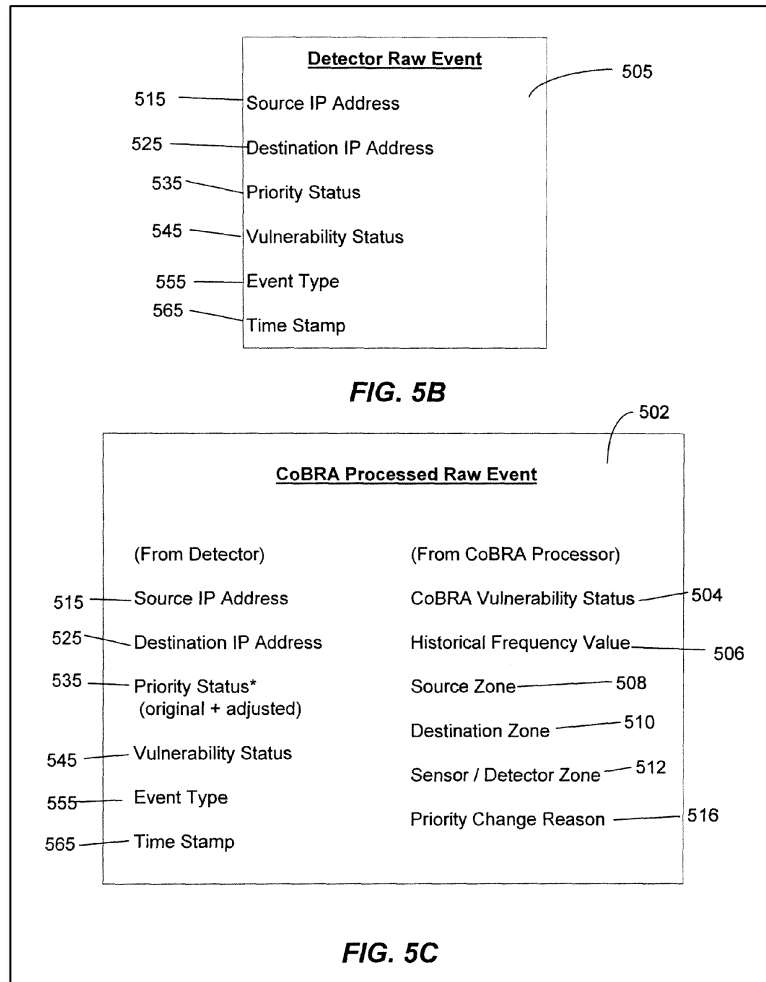
A POSITA would have had a reasonable expectation of success in modifying *Capalik-King* per *Pike's* teachings because *Capalik* already teaches the necessary hardware topology (e.g., one or more processors, memory, VM, computer system) to support *Pike's* technique of identifying activity sources that access portions of the memory that are set as non-executable. *Dec.*, ¶¶218-221. *Pike* teaches implementing its techniques in traditional computers and virtualization systems. *Pike*, 6:19-24. As explained above in Sec. IV.C, *Pike* teaches implementation of its techniques using NX bit technology can be particularly effective against buffer overflow attacks caused by internet worms. *Pike*, 3:12-21. Both *Capalik* and *King* contemplate using their methods for analyzing buffer overflow attacks caused by worms; however,

these references fail to expressly describe the use of NX bit technology or how the activity sources of buffer overflow attacks are identified. *Capalik*, [0057]; *King*, 43, 49. Therefore, a POSITA would have understood that modifying *Capalik-King* to identify activity sources affected by unauthorized activities that request to access a portion of memory that is set as non-executable would require a straightforward application of a well-known technique (NX bit or similar DEP mechanism) that was well within the skillset of a POSITA. *Dec.*, ¶¶218-221.

VII. GROUND 3: *CAPALIK* IN VIEW OF *KING*, AND IN FURTHER VIEW OF *FARLEY* RENDERS CLAIM 17 OBVIOUS

- A. Claim 17: “The method of claim 1, further comprising: determining an activity level representing a frequency of unauthorized activities on the first computer system during a predefined time interval.”**

Farley teaches determining an activity level (i.e., historical frequency value) representing a frequency of unauthorized activities on a computer system during a predefined time interval. Namely, *Farley* discloses a method and system for assembling suspicious computer activity (“raw events”) from multiple data sources (e.g. IDS/IPS, firewalls, audit systems, etc.). *Farley*, 3:47-4:7-11. *Farley*’s system analyzes the assembled information to detect relationships between raw events that may indicate malicious behavior (“mature correlation event”), and provides an organized presentation of the data, as seen in Fig. 5 below:



Farley, Figs 5B-5C, 4:18-34. Figure 5B illustrates an exemplary raw event 505 that is generated by a detector of an intrusion detection system. *Farley*, 11:65-67. The raw event 505 may comprise source Internet protocol address 515, destination Internet protocol address 525, priority status 535, detector assigned vulnerability status 545, event type parameter 555, and time stamp 565. *Farley*, 11:67-12:4.

As explained above in Section IV.D, the priority status of a raw event may be adjusted by a CoBRA processor by comparing the raw event to information stored in the context database such as historical event frequency values. *Farley*, 15:13-26,

Fig. 5C. A historical frequency value “can be a number of events per unit time. Such as events per day, or a mathematically related value, such as an average time between events. . . .[and] typically indicates how frequently raw events of a particular type from a particular source machine to a particular destination machine are seen on the network being monitored[.]” *Farley*, 24:27-36. Accordingly, a POSITA would have understood that *Farley*’s disclosed “historical event frequency value” is an activity level representing a frequency of unauthorized activities on a computer system during a predefined time interval. *Dec.*, ¶¶222-228; *compare with* ’872 Patent, 13:1-13.

1. Motivation to Combine *Capalik-King-Farley*

A POSITA would have been motivated and found it obvious to modify *Capalik-King* per *Farley*’s teachings such that *Capalik-King-Farley* would have determined an activity level (i.e., historical frequency value) representing a frequency of unauthorized activities on the computer system. *Dec.*, ¶¶229-236. Modifying *Capalik* as proposed would have resulted in an improved IDS because, as *Farley* explains, a historical frequency value (and other environmental information) can be used to adjust the priority status of an identified suspicious unauthorized activity, providing a more efficient risk assessment of suspicious activity in a computer system. *Farley*, 15:13-27, 23:53-60; *Dec.*, ¶¶229-236.

Further, the proposed modification adheres to recognized computer security principles—i.e., gathering more detailed threat information provides improved threat analysis and detection where limited data acquisition can hinder performance and result in unperceived threats. *Dec.*, ¶235. Although determining a historical frequency value could have resulted in increased system overhead in *Capalik-King-Farley*, a POSITA would have been motivated to accept increased overhead in exchange for additional security improvements. *Dec.*, ¶235. Thus, the proposed modification would have been use of a known technique of determining a historical frequency value of a suspicious unauthorized activity to improve a forensic analysis method, and would have benefited *Capalik-King* in the same way it benefits *Farley*. *Dec.*, ¶¶229-236.

A POSITA would have had a reasonable expectation of success in modifying *Capalik-King* per *Farley*. *Dec.*, ¶¶229-236. Such a modification would have utilized *Capalik's* processing module that generates an attack report that already includes attack-identifying information. *Id.* Therefore, a POSITA would have understood that modifying *Capalik's* system to additionally capture “a historical frequency value” would require straightforward modifications to programming instructions well-within the skillset of a POSITA. *Id.* A POSITA would have also had a reasonable expectation of success based on the functional similarities between intrusion detection systems taught by *Capalik*, *King*, and *Farley*. *Id.* (explaining these

references teach intrusion detection systems that monitor, store, and graphically display information about suspicious/unauthorized activities). *Dec.*, ¶236.

VIII. DISCRETION UNDER 35 U.S.C. §325(d)

Section 325(d) provides that, in determining whether to institute a post-grant review, “the Director may take into account whether, and reject the petition or request because, the same or substantially the same prior art or arguments previously were presented to the Office.” The Board uses a two-part framework for evaluating arguments under § 325(d):

- (1) whether the same or substantially the same art previously was presented to the Office or whether the same or substantially the same arguments previously were presented to the Office; and
- (2) if either condition of first part of the framework is satisfied, whether the petitioner has demonstrated that the Office erred in a manner material to the patentability of challenged claims.

Advanced Bionics, LLC v. MED-EL Elektromedizinische Geräte GmbH, IPR2019-01469, Paper 6 at 8 (PTAB Feb. 13, 2020) (precedential) (“*Advanced Bionics*”). In applying the two-part framework, the Board considers several nonexclusive factors set forth in *Becton, Dickinson & Co. v. B. Braun Melsungen AG*, IPR2017-01586, Paper 8 at 17–18 (PTAB Dec. 15, 2017).

The Proposed Grounds rely on *Capalik*, which was incorporated by reference in the ’872 Patent but was not discussed by the Examiner during prosecution. *See*

'872 Patent, 1:30-39; *see also generally* '872 Patent File History. The fact that *Capalik* was not applied in a rejection weighs strongly against denying institution under § 325(d). *See CrowdStrike, Inc. v. Webroot Inc.*, IPR2023-00126, Paper 9 at 14 (PTAB May 5, 2023) (“[The Board] cannot determine the extent to which [cited reference] was evaluated during the examination because it was never used in a rejection” and because “[cited reference] was not the basis of a rejection, there is no overlap of arguments.”); *Scientific Design Co., Inc. v. Shell Oil Co.*, IPR2022-00158, Paper 7, 24-26 (PTAB Apr. 4, 2022) (declining to exercise discretion under § 325(d) based on “credit[ing] Petitioner’s argument that although the references ‘were cited in an IDS, together with 111 references, none of them was used in any rejections, or otherwise addressed by the examiner’”); *Bowtech, Inc. v. MCP IP, LLC*, IPR2019-00379, Paper 14 at 18 (PTAB Jul. 3, 2019) (“[The Board has] consistently held that a reference that was neither applied against the claims nor discussed by the Examiner does not weigh in favor of exercising our discretion under § 325(d).”).

As highlighted by the strength of the Proposed Grounds, it is evident that the Examiner did not appreciate *Capalik*’s qualifications as prior art, its disclosures, and how this reference in combination with other references not previously before the Office (*King*, *Farley*, and *Pike*) render the Challenged Claims obvious. When a reference was never “substantively discussed by the Examiner[,]” “a petitioner’s showing that the challenged claims are unpatentable over the asserted prior art may

itself be evidence of material error by the Office during prosecution.” *Quasar Science LLC v. Colt International Clothing, Inc. d/b/a Colt LED*, IPR2023-00611, Paper 10 at 14 (PTAB Oct. 10, 2023) (“Quasar”).

Therefore, denial under 35 U.S.C. § 325(d) is not warranted.

IX. DISCRETIONARY DENIAL IS NOT WARRANTED

A. The *Fintiv* Factors Favor Institution

Under the “*Fintiv* factors,” the Board may consider parallel litigation, including an early trial date, in determining whether to institute under 35 U.S.C. § 314(a). *NHK Spring Co., Ltd., v. Intri-Plex Technologies, Inc.*, IPR2018-00752, Paper 8 at 19–20 (PTAB Sept. 12, 2018) (precedential); *Apple Inc. v. Fintiv, Inc.*, IPR2020-00019, Paper 11 at 2–3, 6 (PTAB Mar. 20, 2020) (precedential).

In the parallel litigation, the parties submitted a proposed schedule that sets a jury trial date of November 16, 2026. Ex. 1026, 8. The Board’s final written decision in this proceeding should precede the scheduled jury trial by more than six months. To avoid overlapping efforts, Petitioner intends to move the district court to stay the litigation pending resolution of this and concurrently filed IPRs. Petitioner is not aware of any precedent supporting a discretionary *Fintiv* denial under similar facts, but reserves its right to seek a preliminary reply in the event Patent Owner does ask the Board to deny institution under *Fintiv*.

X. CONCLUSION

For the forgoing reasons, Petitioner respectfully requests *inter partes* review of the Challenged Claims.

Respectfully submitted,

By: /s/ Paul R. Hart

Paul R. Hart, Reg. No. 59,646

COUNSEL FOR PETITIONER

XI. MANDATORY NOTICES UNDER 37 C.F.R. § 42.8

A. Real Parties-in-Interest Under 37 C.F.R. § 42.8(b)(1)

Petitioner and CrowdStrike Holdings, Inc. are the real parties-in-interest.

B. Related Matters Under 37 C.F.R. § 42.8(b)(2)

The '872 Patent is presently the subject of a patent infringement lawsuit filed against Petitioner in the United States District Court for the Western District of Texas, Austin Division: *GoSecure Inc. v. CrowdStrike, Inc.*, and *CrowdStrike Holdings, Inc.*, Case No. 24-cv-526. As noted in the Claim Construction section, a second petition (IPR2025-00070) is being concurrently filed to account for a distinct interpretation of “association.”

Petitioner is not aware of any other related matters where the '872 Patent has been at issue.

C. Lead and Back-Up Counsel Under 37 C.F.R. § 42.8(b)(3)

Petitioner provides the following designation and service information for lead and back-up counsel. 37 C.F.R. § 42.8(b)(3) and (b)(4).

Lead Counsel	Back-Up Counsel
Paul R. Hart, (Reg. No. 59,646) Paul.Hart@eriseip.com PTAB@eriseip.com	Adam P. Seitz (Reg. No. 52,206) Adam.Seitz@eriseip.com
<u>Postal and Hand-Delivery Address:</u> ERISE IP, P.A. 717 17 th St., Suite 1400 Denver, Colorado 80202 Telephone: (913) 777-5600 Fax: (913) 777-5601	<u>Postal and Hand-Delivery Address:</u> ERISE IP, P.A. 7015 College Blvd., Suite 700 Overland Park, Kansas 66211 Telephone: (913) 777-5600 Fax: (913) 777-5601

APPENDIX OF EXHIBITS

Exhibit 1001	U.S. Patent No. 9,954,872 (the “’872 Patent”)
Exhibit 1002	File History for U.S. Patent 9,954,872 (the “’872 File History”)
Exhibit 1003	Markus Jakobsson Declaration (“ <i>Dec.</i> ”)
Exhibit 1004	U.S. Application Publication No. 2008/0016570 to Capalik (“ <i>Capalik</i> ”)
Exhibit 1005	<i>Intentionally left blank</i>
Exhibit 1006	Samuel T. King, “ <i>Analyzing Intrusions Using Operating System Level Information Flow</i> ,” Dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in The University of Michigan, 2006 (“ <i>King</i> ”)
Exhibit 1007	U.S. Patent No. 8,819,822 to Pike et al. (“ <i>Pike</i> ”)
Exhibit 1008	U.S. Patent No. 7,089,428 to Farley et al. (“ <i>Farley</i> ”)
Exhibit 1009	Declaration of June A. Munford (“ <i>Munford Dec.</i> ”)
Exhibit 1010	IPR2025-0070 of U.S. Patent No. 9,954,872
Exhibit 1011	WO 2004/097584 to Knight (“ <i>Knight</i> ”)
Exhibit 1012	U.S. Application Publication No. 2006/0242701 to Black et al. (“ <i>Black</i> ”)
Exhibit 1013	U.S. Application Publication No. 2007/0208822 to Wang et al. (“ <i>Wang</i> ”)
Exhibit 1014	U.S. Application Publication No. 2008/0098476 to Syversen (“ <i>Syversen</i> ”)
Exhibit 1015	U.S. Patent No., 7,483,972 to Bhattacharya et al. (“ <i>Bhattacharya</i> ”)
Exhibit 1016	U.S. Application Publication No. 2005/0193429 to Demopoulos et al. (“ <i>Demopoulos</i> ”)
Exhibit 1017	WO 2009/032379 to Stolfo et al. (“ <i>Stolfo</i> ”)
Exhibit 1018	U.S. Application Publication No. 2004/0103314 to Liston (“ <i>Liston</i> ”)
Exhibit 1019	U.S. Patent No., 8,407,788 to Kestner et al. (“ <i>Kestner</i> ”)
Exhibit 1020	U.S. Application Publication No. 2009/0037672 to Colbert et al. (“ <i>Colbert</i> ”)
Exhibit 1021	WO 2009/097610 to Moffie et al. (“ <i>Moffie</i> ”)
Exhibit 1022	U.S. Patent No., 8,510,827 to Leake et al. (“ <i>Leake</i> ”)
Exhibit 1023	U.S. Patent No., 9,098,698 to Ghosh et al. (“ <i>Ghosh</i> ”)
Exhibit 1024	U.S. Application Publication No. 2007/0220518 to Verbowski et al. (“ <i>Verbowski</i> ”)
Exhibit 1025	GoSecure Opp’n to Mtn. to Dismiss, Dkt. 26, (Aug. 22, 2024), Case No. 1:24-cv-00526-DII

Exhibit 1026	Jointly Proposed Scheduling Order, Dkt. 48-1, (Oct. 25, 2024), Case No. 1:24-cv-00526-DII
---------------------	---

CERTIFICATION OF WORD COUNT

The undersigned certifies pursuant to 37 C.F.R. §42.24 that the foregoing Petition for *Inter Partes* Review, excluding any table of contents, mandatory notices under 37 C.F.R. §42.8, certificates of service or word count, or appendix of exhibits, contains 13,916 words according to the word-processing program used to prepare this document (Microsoft Word).

Dated: October 31, 2024

Respectfully submitted,

By: /s/ Paul R. Hart
Paul R. Hart, Reg. No. 59,646

COUNSEL FOR PETITIONER

**CERTIFICATE OF SERVICE ON PATENT OWNER
UNDER 37 C.F.R. § 42.105(a)**

Pursuant to 37 C.F.R. §§ 42.6(e) and 42.105(b), the undersigned certifies that on October 31, 2024, a complete and entire copy of this Petition for *Inter Partes* Review and Exhibits were provided via Priority Mail Express to the Patent Owner by serving the correspondence address of record for the '872 patent:

Morgan, Lewis & Bockius LLP (PA)
1400 Page Mill Road
Palo Alto, CA 94304-1124
United States

A complete and entire copy of this Petition for *Inter Partes* Review and Exhibits were also emailed to counsel of record in the parallel litigation:

Giri Pathmanaban - giri.pathmanaban@lw.com
Maximilian A. Grant - max.grant@lw.com
Amit Makker - amit.makker@lw.com
Clement Naples - clement.naples@lw.com
Daniel Todd - daniel.todd@lw.com

Respectfully submitted,

By: /s/ Paul R. Hart
Paul R. Hart, Reg. No. 59,646

COUNSEL FOR PETITIONER