UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

CROWDSTRIKE, INC.,
Petitioner
v.

GOSECURE, INC.,
Patent Owner

Case No. IPR2025-00069
Patent No. 9,106,697

PETITION FOR *INTER PARTES* REVIEW
OF U.S. PATENT NO. 9,106,697

TABLE OF CONTENTS

I. INTRODUCTION 1

II. THE '697 PATENT 1
- A. THE '697 PATENT'S ALLEGED INVENTION 1
- B. THE '697 PATENT'S PROSECUTION 3
- C. CLAIM CONSTRUCTION 3
- D. A PERSON OF ORDINARY SKILL IN THE ART 9

III. REQUIREMENTS UNDER 37 C.F.R. § 42.104 9
- A. STANDING UNDER 37 C.F.R. § 42.104(A) 9
- B. IDENTIFICATION OF CHALLENGE UNDER 37 C.F.R. § 42.104(B) 10

IV. PRIOR ART OVERVIEW 11
- A. CAPALIK 11
- B. GOEL 12
- C. KING 12
- D. PIKE 13
- E. FARLEY 14

V. GROUND 1: *CAPALIK* RENDERS CLAIMS 1-7, 13, 16, 19-20 OBVIOUS 14
- A. CLAIM [1(PRE)]: "A COMPUTER IMPLEMENTED METHOD OF IDENTIFYING UNAUTHORIZED ACTIVITIES ON A DECOY COMPUTER SYSTEM ATTACHED TO A COMPUTER NETWORK, WHEREIN THE DECOY SYSTEM COMPRISES:" 14
- B. CLAIM [1(A)]: "ONE OR MORE PROCESSORS; AND MEMORY STORING:" 17
- C. CLAIM [1(B)]: "A VIRTUAL MACHINE; AND A VIRTUAL MACHINE MONITOR SUPERVISING THE VIRTUAL MACHINE, THE METHOD COMPRISING, AT THE VIRTUAL MACHINE MONITOR: MONITORING ACTIVITY ON THE VIRTUAL MACHINE;" 17
- D. CLAIM [1(C)]: "IDENTIFYING A PLURALITY OF ACTIVITIES BEING PERFORMED AT THE VIRTUAL MACHINE, WHEREIN EACH OF THE ACTIVITIES INCLUDES AN ACTIVITY SOURCE, AN ACTIVITY TARGET, AND AN ASSOCIATION BETWEEN THE ACTIVITY SOURCE AND THE ACTIVITY TARGET;" 23
- E. CLAIM [1(D)]: "STORING IN THE MEMORY THE ACTIVITY SOURCES, ACTIVITY TARGETS, AND ASSOCIATIONS;" 27
- F. CLAIM [1(E)]: "CREATING, FROM THE STORED ACTIVITIES, A FINGERPRINT INDICATIVE OF THE ACTIVITY ON THE VIRTUAL MACHINE; AND" 30

G. CLAIM [1(F)]: "TRANSMITTING THE FINGERPRINT TO PREVENT FUTURE ATTACKS THAT COMPRISE THE SAME OR SIMILAR ACTIVITIES AS INDICATED BY THE FINGERPRINT." 33
H. CLAIM 2: "THE METHOD OF CLAIM 1, WHEREIN THE MONITORING FURTHER COMPRISES MONITORING ALL ACTIVITY ON THE VIRTUAL MACHINE, AND ALL ACTIVITY ON THE VIRTUAL MACHINE IS ASSUMED TO BE UNAUTHORIZED." 34
I. CLAIM 3: "THE METHOD OF CLAIM 1, WHEREIN THE FINGERPRINT INCLUDES FILES THAT ARE AFFECTED BY THE UNAUTHORIZED ACTIVITIES ON THE VIRTUAL MACHINE." 35
J. CLAIM 4: "THE METHOD OF CLAIM 1, WHEREIN THE FINGERPRINT INCLUDES PROCESSES THAT ARE AFFECTED BY THE UNAUTHORIZED ACTIVITIES ON THE VIRTUAL MACHINE." 35
K. CLAIM 5: "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." 37
L. CLAIM 6: "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." 38
M. CLAIM 7: "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." 39
N. CLAIM 13: "THE METHOD OF CLAIM 1, FURTHER COMPRISING IDENTIFYING ACTIVITY SOURCES THAT ARE AFFECTED BY UNAUTHORIZED ACTIVITIES." 39
O. CLAIM 16: "THE METHOD OF CLAIM 13, WHEREIN THE IDENTIFYING ACTIVITY SOURCES THAT ARE AFFECTED BY UNAUTHORIZED ACTIVITIES INCLUDES IDENTIFYING DATA STREAM INCOMING FROM OUTSIDE THE COMPUTER SYSTEM." 39
P. CLAIM [19(PRE)]: 40
Q. CLAIM [19(A)]: 40
R. CLAIM [19(B)]: 40
S. CLAIM [19(C)]: 40
T. CLAIM [19(D)]: 41
U. CLAIM [19(E)]: 41
V. CLAIM [19(F)]: 41
W. CLAIM [19(G)]: 41
X. CLAIM [19(H)]: 41
Y. CLAIM 20(PRE): 41

Z. CLAIM [20(A)]: 42
AA. CLAIM [20(B)]: 42
BB. CLAIM [20(C)]: 42
CC. CLAIM [20(D)]: 42
DD. CLAIM [20(E)]: 42

VI. GROUND 2: *CAPALIK* IN VIEW OF *GOEL* RENDERS CLAIMS 1-7, 13, 15-16, 19-20 OBVIOUS 42
A. *CAPALIK-GOEL* TEACHES IDENTIFYING AND STORING "ACTIVITY SOURCES, ACTIVITY TARGETS, AND ASSOCIATIONS" AS RECITED IN INDEPENDENT CLAIMS 1, 19-20. 43
B. *CAPALIK-GOEL* RENDERS CLAIMS 2-7, 13, 16, 19-20 OBVIOUS. 49
C. *CAPALIK-GOEL* RENDERS CLAIM 15 OBVIOUS. 49

VII. GROUND 3: *CAPALIK* ALONE OR IN VIEW OF *GOEL* AND IN FURTHER VIEW OF *KING* RENDERS CLAIMS 8-12, 15, 17, 21-23 OBVIOUS 50
A. CLAIM 8: "THE METHOD OF CLAIM 1, FURTHER COMPRISING GRAPHICALLY DISPLAYING AT LEAST A SUBSET OF THE STORED ACTIVITIES." 50
B. CLAIM 9: "THE METHOD OF CLAIM 8, WHEREIN THE DISPLAYING INCLUDES VISUALLY DISTINGUISHING AT LEAST A SUBSET OF THE ASSOCIATIONS." .57
C. CLAIM 10: "THE METHOD OF CLAIM 8, WHEREIN 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." 58
D. CLAIM 11: "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." 61
E. CLAIM 12: "THE METHOD OF CLAIM 11, FURTHER COMPRISING TRACKING A TRANSFER OF MALICIOUS ASSOCIATIONS OVER TIME." 62
F. CLAIM 15: "THE METHOD OF CLAIM 13, 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." 63

G. CLAIM 17: "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 TAGET 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." 64
H. CLAIM 21: 66
I. CLAIM 22: "THE METHOD OF CLAIM 1, FURTHER COMPRISING: IDENTIFYING A TYPE OF THE ASSOCIATION BETWEEN THE ACTIVITY SOURCE AND THE ACTIVITY TARGET." 66
J. CLAIM 23: 67

VIII. GROUND 4: *CAPALIK* ALONE OR IN VIEW OF *GOEL,* AND IN FURTHER VIEW OF *PIKE* RENDERS CLAIM 14 OBVIOUS 68
A. CLAIM 14: "THE METHOD OF CLAIM 13, 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." 68

IX. GROUND 5: *CAPALIK* ALONE OR IN VIEW OF *GOEL,* AND IN FURTHER VIEW OF *FARLEY* RENDERS CLAIM 18 OBVIOUS 71
A. CLAIM 18: "THE METHOD OF CLAIM 1, FURTHER COMPRISING DETERMINING AN ACTIVITY LEVEL REPRESENTING A FREQUENCY OF UNAUTHORIZED ACTIVITIES ON THE VIRTUAL MACHINE DURING A PREDEFINED TIME INTERVAL." 71

X. DISCRETION UNDER 35 U.S.C. §325(d) 75

XI. DISCRETIONARY DENIAL IS NOT WARRANTED 77
A. THE *FINTIV* FACTORS FAVOR INSTITUTION 77

XII. CONCLUSION 77

XIII. MANDATORY NOTICES UNDER 37 C.F.R. § 42.8 78
A. REAL PARTIES-IN-INTEREST UNDER 37 C.F.R. § 42.8(B)(1) 78
B. RELATED MATTERS UNDER 37 C.F.R. § 42.8(B)(2) 78
C. LEAD AND BACK-UP COUNSEL UNDER 37 C.F.R. § 42.8(B)(3) 78

I. INTRODUCTION

CrowdStrike, Inc. ("Petitioner") respectfully requests *inter partes* review ("IPR") of all claims ("Challenged Claims") of U.S. Patent No. 9,106,697 ("'697 Patent").

II. THE '697 PATENT

A. The '697 Patent's Alleged Invention

The '697 Patent describes "methods for analyzing malicious activities on a computer system in order to better protect the computer system from future malicious activity." *'697 Patent* (Ex. 1001), 1:33-37. More specifically, the '697 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.*, 2:16-38, Fig. 1. The fingerprint can then be used to protect the remainder of the computer network against similar unauthorized activities. *Id.*, 2:32-38. By generating fingerprints from monitored activities in real-time, the '697 Patent purports to protect against so-called "zero-day attack[s]"—security vulnerabilities previously unknown to software developers or system operators. *Id.*, 1:65-67.

Figs. 3A-3D and 4A-4E illustrate exemplary patterns of attack monitored on a VM. *Id.*, 10:57-63. 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.*, 10:67-11:10. 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.*, 11:18-20, Figs. 3B-3D.

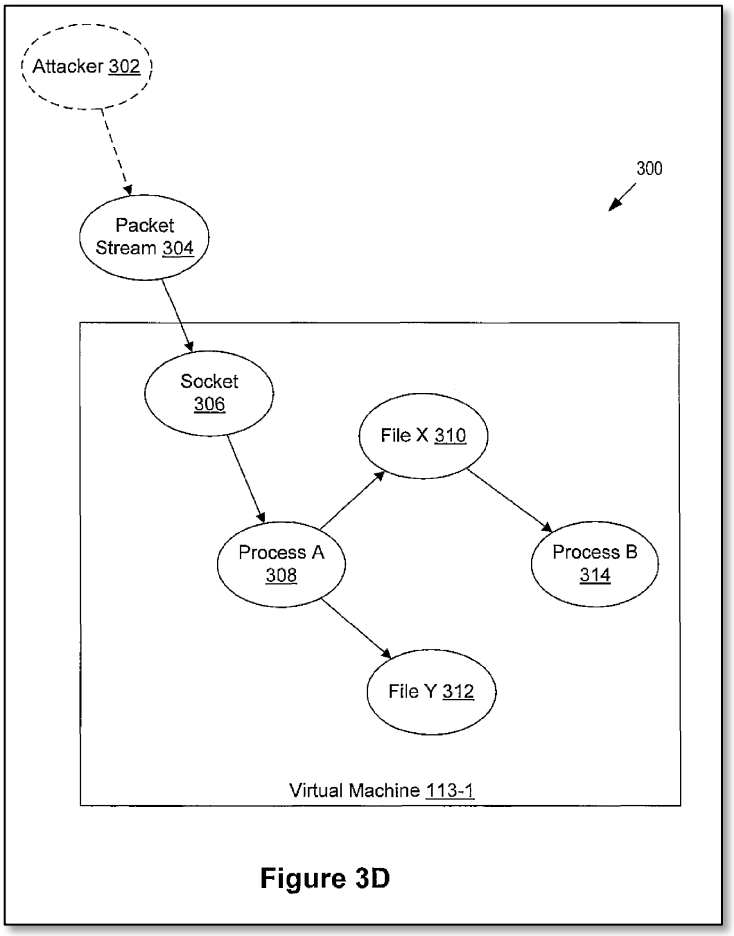


Figure 3D

Id., Fig. 3D, 10:64-11:65.

The '697 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.*, 11:29-30. "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.*, 11:34-38.

B. The '697 Patent's Prosecution

The '697 Patent was filed June 17, 2011, 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 October 25, 2013, rejecting all originally presented claims 1-20. *'697 Patent File History* (Ex. 1002), 216-26. In response, the applicant expressly distinguished the cited prior art on the basis that the claims require "identifying a plurality of activities being performed at the virtual machine, wherein each of the activities includes an **activity source**, an **activity target**, and an **association between the activity source and the activity target.**" *Id.*, 256 (emphasis in original). The applicant further emphasized this distinguishing feature in an applicant-initiated interview. *Id.*, 263. A Notice of Allowance issued thereafter. *Id.*, 265.

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"*

As stressed by the applicant during prosecution, 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 the activity" "from the stored activities[.]" *'697 Patent*, Claims 1, 19-20. 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.*, *'697 Patent*, 13:6-10 (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

[1] Petitioner's challenge is not a waiver of any argument in litigation that any claim term in the '697 Patent is indefinite or invalid. And Petitioner does not waive its right to seek to construe claim terms in litigation, including in the pending district-court litigation.

[2] All emphases added unless otherwise noted.

unauthorized. 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). *'697 Patent*, 13:11-29. A separate petition is being filed concurrently that advances this narrower interpretation of "association." *See* IPR2025-00067 (Ex. 1010).[3] The instant petition applies the broader interpretation, supporting such an interpretation as set forth below.

The Challenged Claims require identifying activities "where each of the activities includes an activity source, an activity target, and an ***association*** between the activity source and the activity target[.]" *'697 Patent*, Claims 1, 19, 20. For a number of reasons, the intrinsic record supports interpreting the claimed "association" as an ***action*** between the source and target. First, each of the independent Challenged Claims defines activities by three key components—an activity source, an activity target, and an association between the source and target. *See '697 Patent*, Claim 1. This same three-component structure is consistently ascribed to activities throughout the specification. But the specification frequently refers to the third mandatory component as an "action" between the source and target, rather than an "association." *Id.*, 12:57-13:37; *see also id.*, 2:59-62 ("***activities***

[3] A "ranking" paper is concurrently filed, explaining and prioritizing two petitions.

(e.g., action ***sources***, action ***targets***, and ***actions***)"). In other words, the specification teaches that an activity is defined by (1) a source, (2) a target, and (3) the action occurring between the source and target. Because the claims refer to component (3) as an "association" between the source and target, consistent with the specification, this term should be construed to mean the "action" between the source and target, where the source acts on the target.

Second, while the specification fails to clearly define "association," it does use the term "associate" interchangeably with the actions occurring between a source and target. For example, when discussing time stamps, the '697 Patent teaches that the time at which an activity source is associated with an activity target coincides with the source performing an action (e.g., calling, accessing, or executing) directed to the target. *Id*., 13:33-37 (describing a time stamp as representing "the time when the activity source 512 **associates** with the activity target 514 (e.g., when the **activity source 512 calls, accesses, or executes the activity target** 514)").

Third, the claims demand that "association" and "association type" have distinct meanings, and interpreting "association" as the action between a source and target aligns with the meaning ascribed to "association type[.]" Initially, limitation [1(d)] recites an "activity source, an activity target, and an ***association*** between the activity source and the activity target[,]" while dependent claim 22 recites, "[t]he method of claim 1, further comprising: identifying ***a type of the association*** between

the activity source and the activity target." Under the doctrine of claim differentiation, there is a presumption that "association" and "a type of the association" have different meanings. *Phillips v. AWH Corp.*, 451 F.3d 1303, 1315 (Fed. Cir. 2005). The specification consistently describes an "association type" as a classification of the action between a source and target as benign, malignant, or remission. *'697 Patent*, 12:2-8, 12:18-28, 13:13-29. In other words, the "association" between a source and target is the action (e.g., event or operation) occurring between the source and target, and the "association type" classifies that action as malignant, benign, etc.

Fourth, other dependent claims confirm that "association" in the independent claims should not be construed to require classifying each individual source and target pair as benign, malignant, or in remission. For example, Claim 2 requires "monitoring all activity on the virtual machine, and all activity on the virtual machine is assumed to be unauthorized." Were "association" construed narrowly to require classifying each individual activity as benign or malignant, Claim 2 would be meaningless because it would be nonsensical to assume that all monitored activity is unauthorized after all the monitored activity was previously classified as benign, malignant or remission.

The specification does appear to confuse the issue, suggesting in the following sentence that an association and association type are the same:

> ***An (optional) association***, type 516, which when identified, ***represents the type of association*** between the activity source 512 and the activity target 514[.]

'697 Patent, 13:11-13. One could reasonably read this sentence to suggest that an association is referred to as "type 516" and that the association "represents the type of association" between a source and target. Examining the full record, however, reveals that the comma between "association" and "type 516" was likely not intended by the applicant. The following excerpt from the '697 File History appears to show that no comma was submitted in the application:

> An (optional) association type 516, which when identified,

Ex. 1002, 20. There is, however, a small dark spot that could easily have been misinterpreted as a comma. Supporting an interpretation that this was an understandable transcription error, U.S. Patent No. 9,954,872, which is a continuation of the '697 Patent, does not include a comma in this location. Removing the apparently errant comma, the sentence no longer suggests that association and association type are one and the same. Instead, the sentence simply states that "[a]n (optional) association type 516 . . . represents the type of association between the activity source 512 and the activity target 514[.]"

In sum, the intrinsic record supports interpreting "association between the activity source and the activity target" as the ***action***[4] occurring between the activity source and activity target.

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 and who would have had a working knowledge of software security analysis and dynamic malware analysis. Dr. Jakobsson Declaration ("*Dec*.") (Ex. 1003), ¶¶41-42. Additional graduate education could substitute for professional experience, or significant experience in the field could substitute for formal education. *Id*.

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

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

Petitioner certifies the '697 Patent is eligible for IPR and Petitioner is not barred or estopped from requesting IPR challenging the claims of the '697 Patent. Petitioner is not the owner of the '697 Patent, has not filed a civil action challenging the validity of any claim of the '697 Patent, and this Petition is filed less than one

[4] An action can be any event or operation, including calling, executing, modifying, or accessing, such as system calls and library calls. *'697 Patent*, 13:6-10.

year after Petitioner was served with a complaint alleging infringement of the '697 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
Ground 1: Claims 1-7, 13, 16, 19-20 of the '697 Patent are obvious under §103(a) over US Application Publication No. 2008/0016570 to Capalik ("*Capalik*")	Ex. 1004
Ground 2: Claims 1-7, 13, 15-16, 19-20 of the '697 Patent are obvious under §103(a) over *Capalik* in view of "*Forensix: A Robust, High-Performance Reconstruction System"* by Goel et al. ("*Goel*").	Ex. 1004 Ex. 1005
Ground 3: Claims 8-12, 15, 17, 21-23 of the '697 Patent are obvious under §103(a) over *Capalik* alone or in view of *Goel* and further in view of "*Analyzing Intrusions Using Operating System Level Information Flow"* by King ("*King*").	Ex. 1004 Ex. 1005 Ex. 1006
Ground 4: Claim 14 of the '697 Patent is obvious under §103(a) over *Capalik* alone or in view of *Goel* and in further view of U.S. Patent No. 8,819,822 to Pike et al. ("*Pike*").	Ex. 1004 Ex. 1005 Ex. 1007
Ground 5: Claim 18 of the '697 Patent is obvious under §103(a) over *Capalik* alone or in view of *Goel* and in further view of US Patent No. 7,089,428 to Farley et al. ("*Farley*").	Ex. 1004 Ex. 1005 Ex. 1008

IV. PRIOR ART OVERVIEW

A. Capalik

Capalik was filed April 20, 2007 and published January 17, 2008. *Capalik* (Ex. 1004). *Capalik* and the '697 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 '697 Patent (June 24, 2010). Therefore, *Capalik* qualifies as prior art to the '697 Patent under 35 U.S.C. § 102(b) (pre-AIA).

Capalik was incorporated by reference in the '697 Patent but was not discussed by the Examiner during prosecution. *See '697 Patent*, 1:21-29; *see also generally '697 Patent File History*. As set forth below in Section X, a discretionary denial under § 325(d) is not warranted on this record.

Capalik, like the '697 Patent, discloses systems and methods for providing computer security by analyzing event data from potential threat events. *Compare Capalik*, Abstract, [0002], [0015] *with '697 Patent*, 1:33-37, 2:22-38, 2:39-62. Thus, *Capalik* is in the same field of endeavor as the '697 Patent. *Dec*., ¶¶82-87.

Capalik is also reasonably pertinent to certain problems addressed by the '697 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* '697 Patent, 2:9-21, 10:7-31. Thus, *Capalik* is analogous art to the '697 Patent. *Dec*., ¶¶82-88.

B. Goel

Goel published in **Error! Reference source not found.**, as held by the **Error! Reference source not found.** library. *See Goel* (Ex. 1005), s*ee also Munford Dec*., (Ex. 1009). The Linda Hall Library first acquired *Goel* as of June 6, 2005, and *Goel* was made publicly available shortly after its initial acquisition date. *See Munford Dec*., (Ex. 1009), ¶¶6-9. Therefore, *Goel* qualifies as prior art under 35 U.S.C. § 102(b).

Goel, like the ’697 Patent, discloses systems and methods for providing computer security by analyzing event data from potential threat events. *Compare Goel*, 1, 3-4, Fig. 2 *with ’697 Patent*, 1:33-37, 2:22-38, 2:39-62. Thus, *Goel* is in the same field of endeavor as the ’697 Patent. *Dec*., ¶¶89-90.

Goel is also reasonably pertinent to certain problems addressed by the ’697 Patent such as providing a forensic analysis of the impact of an attack on a target device. *Compare Goel*, Introduction, 1, *with ’697 Patent*, 2:9-12, 10:7-31. Thus, *Goel* is analogous art to the ’697 Patent. *Dec*., ¶¶89-91.

C. 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 *King* was made publicly

available shortly after its initial acquisition date. *See Munford Dec.,* (Ex. 1009) ¶¶10-13. Therefore, *King* qualifies as prior art under 35 U.S.C. § 102(b) (pre-AIA). *Id*.

King, like the '697 Patent, discloses systems and methods for providing computer security by analyzing event data from potential threat events. *Compare King*, 2, 3 *with '697 Patent*, 1:33-37, 2:22-62. Thus, *King* is in the same field of endeavor as the '697 Patent. *Dec*., ¶¶92-97.

King is also reasonably pertinent to certain problems addressed by the '697 Patent such as providing a forensic analysis of the impact of an attack on a target device. *Compare King*, 2-5, *with '697 Patent*, 2:9-21, 10:7-31. Thus, *King* is analogous art to the '697 Patent. *Dec*., ¶¶92-98.

D. Pike

Pike was filed March 30, 2006 and published August 26, 2014. *Pike* (Ex. 1007). Therefore, *Pike* qualifies as prior art to the '697 Patent under 35 U.S.C. § 102(e).

Pike, like the '697 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 '697 Patent*, 1:33-37, 2:22-38, 2:39-62. Thus, *Pike* is in the same field of endeavor as the '697 Patent. *Dec*., ¶¶99-100.

Pike is also reasonably pertinent to certain problems addressed by the '697 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 '697 Patent*, 2:9-21, 10:7-31. Thus, *Pike* is analogous art to the '697 Patent. *Dec*., ¶¶99-101.

E. Farley

Farley was filed April 27, 2001 and published August 08, 2006. *Farley* (Ex. 1008). Therefore, *Farley* qualifies as prior art to the '697 Patent under 35 U.S.C. § 102(b) (pre-AIA).

Farley, like the '697 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 '697 Patent*, 1:33-37, 2:22-38, 2:39-62. Thus, *Farley* is in the same field of endeavor as the '697 Patent. *Dec*., ¶¶102-104.

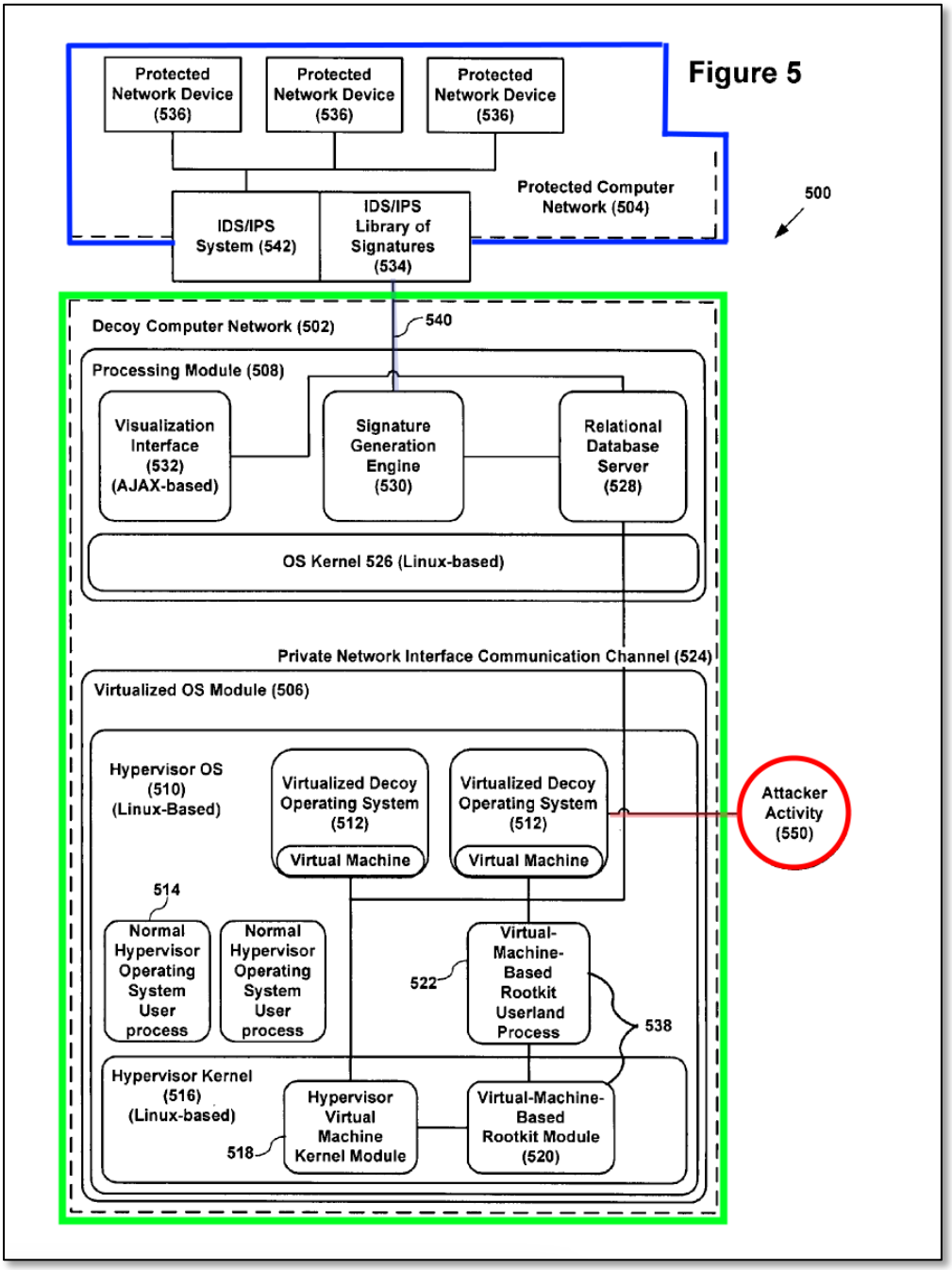
Farley is also reasonably pertinent to certain problems addressed by the '697 Patent such as providing a forensic analysis of the impact of an attack on a target device. *Compare Farley*, 3:25-46, *with '697 Patent*, 2:9-21, 10:7-31. Thus, *Farley* is analogous art to the '697 Patent. *Dec*., ¶¶102-105.

V. GROUND 1: *CAPALIK* RENDERS CLAIMS 1-7, 13, 16, 19-20 OBVIOUS

A. Claim [1(Pre)]: "A computer implemented method of identifying unauthorized activities on a decoy computer system attached to a computer network, wherein the decoy system comprises:"

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]. Decoy computer network 502

(annotated green) comprises multiple modules that identify 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 “[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 where 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 “decoy computer system” (also referred to in the claims as a “decoy 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 Decoy Computer 502. *Dec.*, ¶¶106-107. Accordingly, a POSITA would have understood Decoy Computer 502 is attached to a computer network, as claimed. *Id*.

Additionally, *Capalik* teaches Decoy Computer 502 is attached to "protected computer network 504" via connection 540. *Capalik*, [0054], [0038], Fig. 5; *Dec*., ¶¶106-108.

B. Claim [1(a)]: "one or more processors; and memory storing:"

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.*, ¶109. 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—run on a computing platform that includes at least a processor and memory, as claimed. *Dec*., ¶¶109-10.

C. Claim [1(b)]: "a virtual machine; and a virtual machine monitor supervising the virtual machine, the method comprising, at the virtual machine monitor: monitoring activity on the virtual machine;"

Proposing the same structure and using the same terminology as the '697 Patent, *Capalik* teaches a virtual machine monitor (i.e., Hypervisor module) that monitors activity on a virtual machine.

1. ’697 Patent’s VM and VMM

Figure 2 of the ’697 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, as illustrated below:

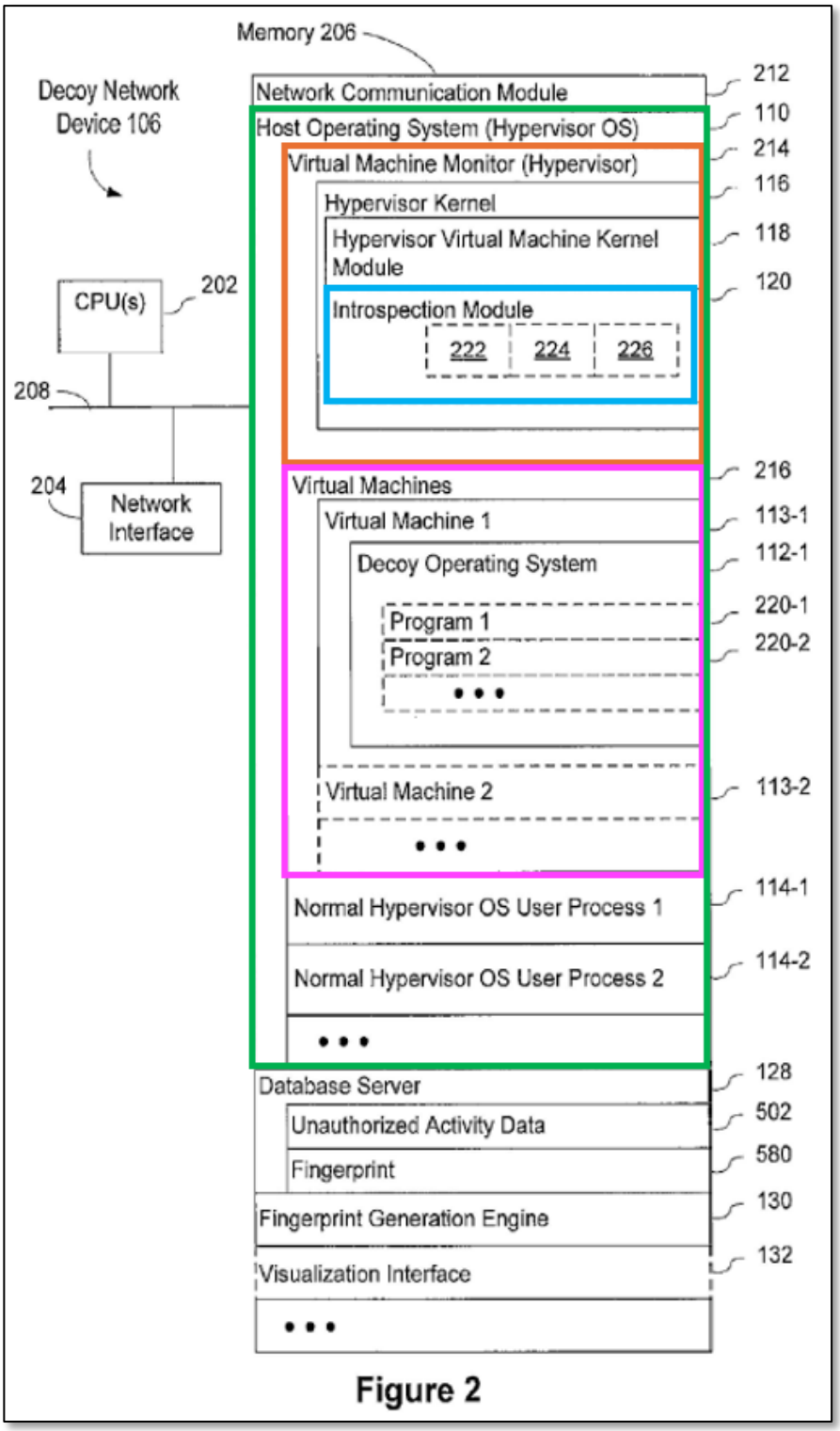


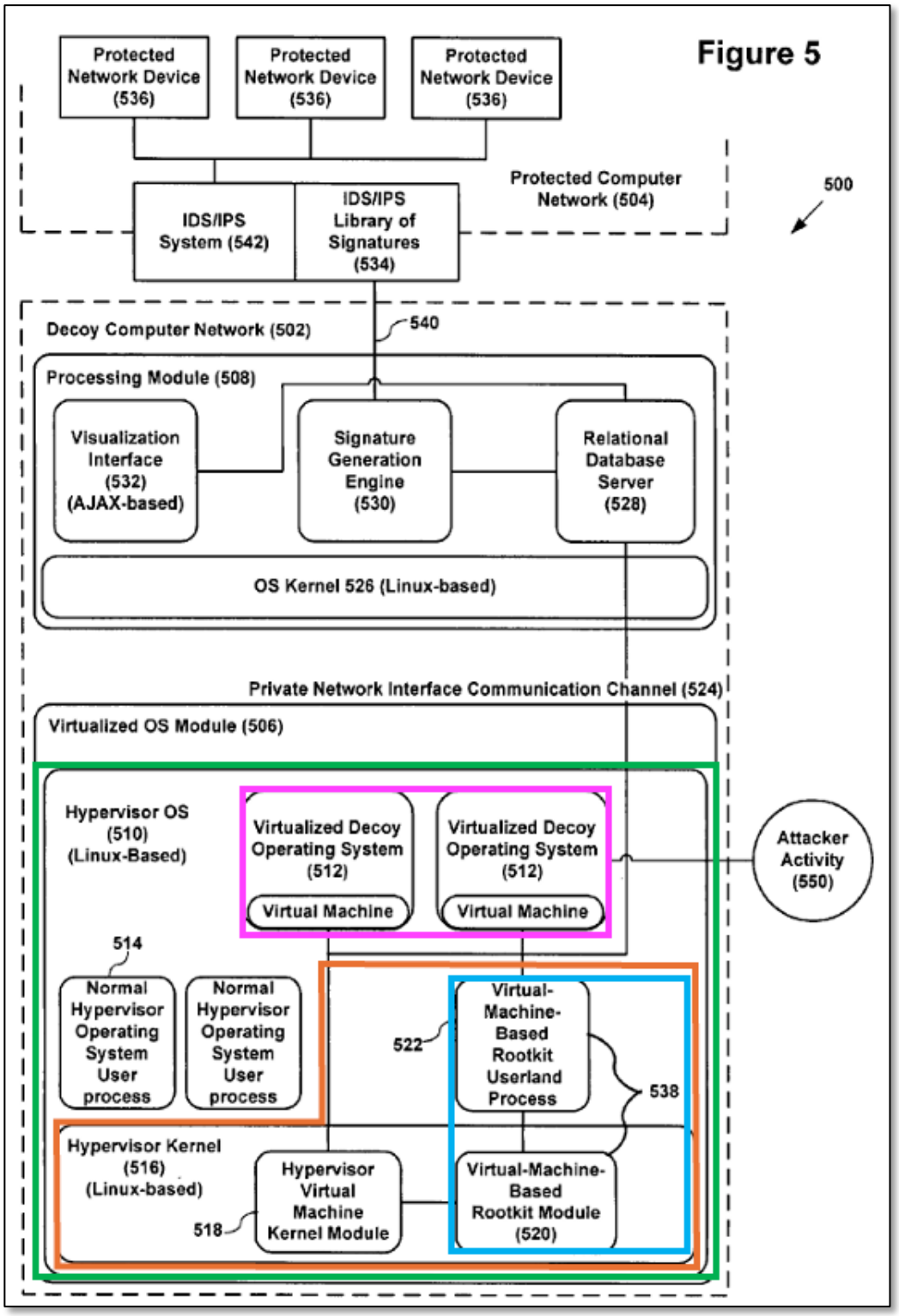
Figure 2

'697 Patent, Fig. 2 (annotated). The '697 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*., 6:21-23. 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*., 6:32-38.

2. *Capalik's* VM and VMM

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



Capalik, Fig. 5 (annotated).

Like the '697 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." *Id.*, [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***." *Id.*, [0047].

In other words, with almost verbatim terminology and functionality, *Capalik* describes the same VM and VMM system as described and claimed in the '697 Patent. Both the '697 Patent and *Capalik* use the term "Hypervisor" interchangeably with virtual machine monitor. Compare *'697 Patent*, 6:12 ("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.*, ¶111-15. 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, as claimed. 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 VM. *Dec*., ¶111-17.

D. Claim [1(c)]: "identifying a plurality of activities being performed at the virtual machine, 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, an ***association*** can be any event or operation, including calling, executing, modifying, or accessing operations, such as system calls and library calls between an activity source and an activity target. *See* Section II.C.

Capalik teaches identifying a plurality of activities being performed at the VM, wherein each of the activities includes an activity source, an activity target, and an association, as discussed below. For example, a POSITA would have understood that *Capalik* teaches identifying a plurality of activities wherein each activity includes an activity source (e.g., data packet stream), an activity target (e.g., a socket), and an association between the activity source and activity target (e.g., receiving the data packet stream at the socket). *Dec*., ¶¶118-28.

1. *Capalik's* plurality of activities performed at the VM

Capalik discloses that when an attacker connects to a network port and begins interacting with virtualized decoy operating system 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* teaches the attack report includes "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]; *Dec.*, ¶120 Thus, a POSITA would have understood that *Capalik* teaches identifying a plurality of activities being performed at the VM. *Dec.*, ¶¶120-21.[5]

[5] GoSecure has argued in the district court litigation that the claims of the'697 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 recite the negative limitation of "not

2. *Capalik's* activity sources and targets

The *'697 Patent* discloses activity sources include access to a particular file, an instruction, a message, data stream, or network port. *'697 Patent*, 12:60-67. *Capalik* teaches identifying at least these activity sources.

As discussed above, *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]. Accordingly, a POSITA would have understood *Capalik's* activity sources included, for example, ports, files, instructions, data streams, and any other data transfers to the decoy computer. *Dec*., ¶¶122-23.

---

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.").

The *'697 Patent* discloses activity targets include a file, an instruction, a data stream, and a network port. *'697 Patent*, 12:60-67. *Capalik* teaches identifying at least these activity targets.

Capalik teaches monitoring processes following receipt of files, malicious payloads, or other data transfers. For example, *Capalik* teaches that attack 550 may be categorized and analyzed by observing operations performed on decoy operating system(s) 512 "to determine, for example, how an attack gained control of the decoy operating system(s) 512." *Capalik*, [0055]-[0057]. In one example, a target may be a socket or port of *Capalik's* decoy computer that receives a data transfer from an attacker. *Capalik*, [0056]-[0057], [0059], [0044]-[0045], [0049]; *Dec*., ¶¶122-25. Further, *Capalik* teaches monitoring and reporting the series of actions that occur after receiving data, for example. *Capalik*, [0056]-[0057]. A POSITA would have understood that such monitoring would have included the "path" or series of steps including and following the receipt of data (or other activity source). *Id*. A POSITA would have understood that *Capalik's* activity targets, then, are processes, files, sockets, ports, or other "recipients" of some action by an activity source, as discussed above. *Id*.; *Capalik*, [0055]-[0057].

Thus, a POSITA would have understood that the "data streams" and "sockets" disclosed in *Capalik* are examples of activity sources and activity targets, respectively. *Dec*., ¶¶122-26; *see '697 Patent*, 12:60-13:5.

3. *Capalik's* associations

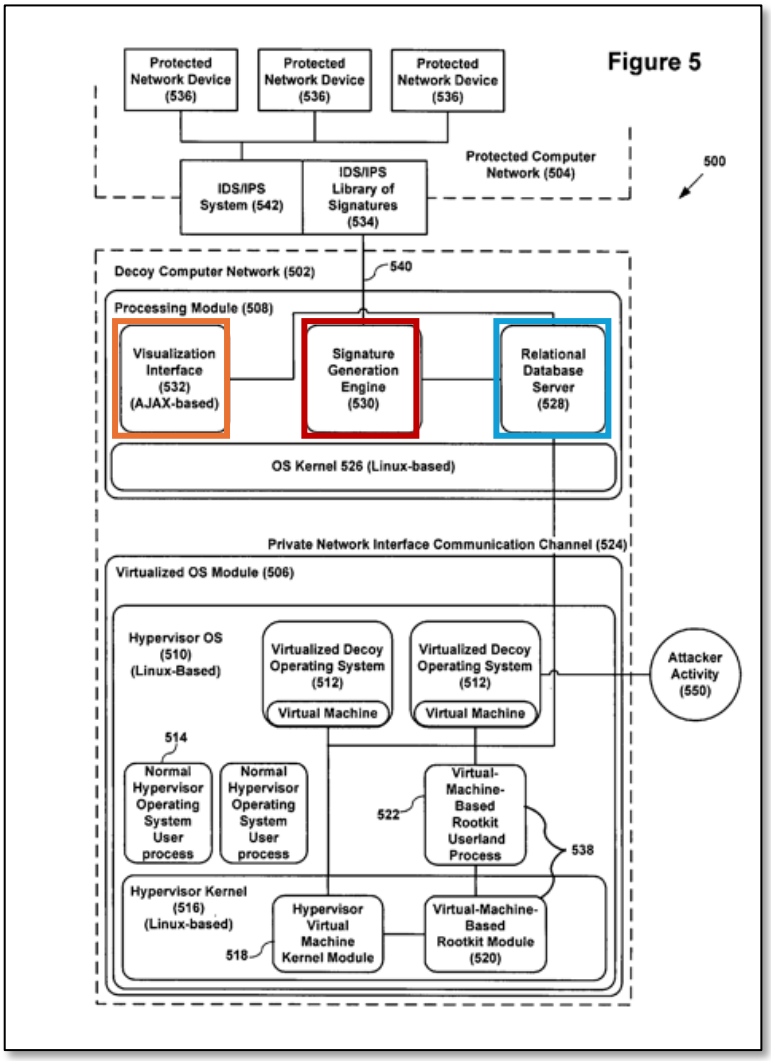
A POSITA would have understood that *Capalik* teaches identifying an "action" (i.e., association) between the activity source and the activity target. *Dec*., ¶127-28; *see* Section II.C; *Capalik,* [0049] ("When an attacker connects to a network port and begins interacting with a virtualized decoy operating system 512, the introspection module 538 monitors and captures information from the connection, including port numbers[and] data streams[.]"), [0056] (teaching that the report of the attack can include information about "what happened at a particular socket …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"); *compare with '697 Patent*, 13:6-8. A POSITA would thus have understood that *Capalik* monitors and reports on how activity sources (e.g., stream of malicious data) interact with an activity target (e.g., socket, file, process). *Dec*., ¶¶127-128. Because *Capalik* monitors and reports the full "path" or series of actions between a plurality of activity sources and activity targets, a POSITA would have understood that *Capalik* teaches identifying associations between each set of the plurality of activity sources and activity targets. *Id*.

E. Claim [1(d)]: "storing in the memory the activity sources, activity targets, and associations;"

As discussed above, *Capalik* teaches a decoy computer comprising a memory. *See* Claims [1(Pre)]-[1(a)]. Also discussed above, *Capalik* teaches identifying

activity sources, activity targets, and associations. *See* Claim [1(c)]. As discussed below, *Capalik* teaches storing ***in the memory*** of the decoy computer the activity sources, activity targets, and corresponding associations.

Capalik discloses storing "attack information" in relational database server 528 (shown in blue) in Decoy Computer 502, as depicted in Figure 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]. "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*., ¶129-130.

As explained in Claim [1(a)], 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*., ¶¶129-132.

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(c)], *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 and used to create the attack report includes activity sources, targets, and associations. *Dec*., ¶129-133.

F. Claim [1(e)]: "creating, from the stored activities, a fingerprint indicative of the activity on the virtual machine; and"

The '697 Patent describes generating a fingerprint[6] indicative of attack activity that "can be used to prevent future attacks[,] by modifying the activity targets[,]" such as "blocking a particular port[.]"[7] *'697 Patent*, 13:38-54; *see also id*., 14:12-19 (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*., 13:40-43. The patent is clear that fingerprints can include both source and target data or even just source or target data. *Id*., 13:55-14:22 ("In some embodiments, the fingerprint [] includes activity targets [] in lieu of or in addition to, the activity sources[.]").

Like the '697 Patent, *Capalik* teaches a decoy system comprising Relational Database Server 528, which stores "attack information" as forensic data. *Capalik*, [0052]; *see* Claim [1(d)]. Also, like the '697 Patent, *Capalik* uses a signature

---

[6] The '697 Patent also refers to a fingerprint as a signature. *'697 Patent*, 10:10-31.

[7] *Capalik* and the '697 Patent expressly teach generating a fingerprint/signature of an attack that identifies the port (or socket) targeted by the attack. The Proposed Grounds are not limited to this teaching.

generation engine to create a "signature of the attack" from the stored forensic data in a relational database. *Compare Capalik*, [0052]*, with '697 Patent*, 5:63-67; *Dec*., ¶¶134-35. *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]; *Dec*., ¶¶134-37. Like the '697 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\xf4\xef\x51\x92\xf9\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*., ¶138. As illustrated, *Capalik's* signature includes many details indicative of the activity on a virtual machine, including identifying the specific port on the virtual machine through which the attack occurred. *Dec*., ¶138 (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*.

G. Claim [1(f)]: "transmitting the fingerprint to prevent future attacks that comprise the same or similar activities as indicated by the fingerprint."

As discussed above, *Capalik* teaches the claimed fingerprint. *See* Claim [1(e)]. Once created, *Capalik* discloses transmitting its signature (i.e., fingerprint) in a similar manner as the '697 Patent. The '697 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." *'697 Patent*, 18:8-12. *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*. ¶¶139-41 (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 to prevent future attacks that comprise the same or similar activities as indicated by the fingerprint. *Dec*., ¶¶139-42.

H. Claim 2: "The method of claim 1, wherein the monitoring further comprises monitoring all activity on the virtual machine, and all activity on the virtual machine is assumed to be unauthorized."

As explained above, *Capalik* teaches monitoring activity on a virtual decoy system. *See* Claim [1(b)]. *Capalik* further teaches monitoring all activity on the VM. *Capalik,* [0059] ("[T]he processing module 508 may generate a signature ***whenever data is found to be transferred through the socket***."), [0049]; *Dec*., ¶143 (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).

Capalik teaches that its decoy system attracts an attacker by leaving ports open on the virtual machines. *Capalik*, [0044]. Further, the decoy system device can be configured to respond to connection attempts made at network addresses that do not exist on protected network 504, and the connections to these non-existent network addresses are assumed to be malicious. *Id*.

Accordingly, a POSITA would have understood that *Capalik* teaches monitoring all activity on the VM wherein all activity on the VM is assumed to be unauthorized. *Dec*., ¶¶143-45.

I. Claim 3: "The method of claim 1, wherein the fingerprint includes files that are affected by the unauthorized activities on the virtual machine."

Capalik expressly teaches that captured forensic data includes information related to files affected by an attack. *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 . . . ***file uploads*** . . ., and any other data transfers or malicious attempts."), [0056] (explaining that *Capalik*'s "signature of the attack" can include "***what files were transferred***"); *see* Claim [1(e)]. Although *Capalik* does not expressly describe signatures that identify files affected by the attack, a POSITA would have understood that such file-related information would have been beneficial to include in *Capalik's* fingerprint. *Dec*., ¶146 (explaining that a signature would beneficially identify any files uploaded, modified, or deleted as part of an attack, allowing the IDS/IPS to block any attempts to similarly upload, modify, or delete files on protected computer network 504).

J. Claim 4: "The method of claim 1, wherein the fingerprint includes processes that are affected by the unauthorized activities on the virtual machine."

Capalik teaches that attack 550 may be categorized and analyzed by observing operations performed on decoy operating system(s) 512 "to determine, for example, ***how an attack gained control*** of decoy operating system(s) 512." *Capalik*, [0055]-

[0057]. *Capalik* teaches an attack report includes "information about which sockets were accessed by the attack 550, ***what happened at a particular socket*** …" and the "***operations performed*** on the decoy OS 512" as well as determining when "data is transferred through the socket." *Capalik*, [0056]-[0057], [0059]. Further, *Capalik* teaches monitoring and reporting the series of actions that occur after receiving data. *Capalik*, [0056], [0057] (describing how the attack-identifying information is analyzed for attack patterns such as "buffer overflow attempts, malicious payloads attempting to execute arbitrary code on the system, memory overwriting attempts, stack attacks, and heap attacks"). Accordingly, a POSITA would have understood that *Capalik* monitors attacks which include processes. *Dec*. ¶147 (explaining that *Capalik's* signature of the attack in Fig. 4 depicts a malicious process, worm W32/RBot-CBQ). Further a POSITA would have understood that *Capalik* monitors attacks that include targets such as ***processes***. *Id*. (explaining that worm W32/RBot-CBQ copies itself into target process "<System>\msprexe.exe"). *Capalik* further teaches monitoring processes following receipt of files, malicious payloads, or other data transfers. *Dec*. ¶¶147-49; *see also* Claim [1(c)].

Moreover, although *Capalik* does not expressly describe signatures that identify processes affected by the attack, a POSITA would have understood that such processes-related information would have been beneficial to include in *Capalik's* fingerprint. *Dec*., ¶147-50 (explaining that a signature would beneficially identify

any processes as part of an attack, allowing the IDS/IPS to block any attempts to upload similar files to protected computer 504 or execute unauthorized processes).

K. Claim 5: "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."

Capalik teaches identifying a plurality of activities performed in a VM, wherein each activity includes an activity source, target, and association. *See* limitation [1(c)]. *Capalik* also teaches a decoy computer attached to a network. *See* limitation [1(Pre)].

When an attacker connects to a network port of decoy computer and begins interacting with virtualized decoy OS 512, *Capalik's* system "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]. Thus, a POSITA would have understood that *Capalik* discloses receiving data through network 502. *Dec*., ¶¶151-52.

Moreover, *Capalik's* figure 3 illustrates a human readable summary of an example attack, wherein worm W32/RBot-CBQ copies itself (reads and then writes) to a file "msprexe.exe" in the "System" directory (step 300), and subsequently first (step 301) and second (step 302) registry entries are created by the attack:

When first run W32/Rbot-CBQ copies itself to <System>\msprexe.exe. ← 300

The following registry entries are created to run msprexe.exe on startup:

HKCU\Software\Microsoft\Windows\CurrentVersion\Run ← 301
Windows Management System
msprexe.exe

HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run ← 302
Windows Management System
msprexe.exe

HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\RunServices
Windows Management System
msprexe.exe

Registry entries are set as follows:

HKLM\SOFTWARE\Microsoft\Ole
EnableDCOM
N

HKLM\SYSTEM\CurrentControlSet\Control\Lsa
restrictanonymous
1

Capalik, Fig. 3, [0061]; *Dec*., ¶153. A POSITA would have understood that W32/Rbot-CBQ initiates execution of both (1) an instruction of reading a file (i.e. W32/Rbot-CBQ reading file "msprexe.exe") and (2) an instruction writing to a file (i.e., W32/Rbot-CBQ writing to file "msprexe.exe"). *Dec*., ¶153.

L. Claim 6: "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."

Capalik teaches activity targets include processes, files, and sockets. *See* Claim [1(c)]. For example, *Capalik* teaches the created attack report includes "which **sockets** were accessed by the attack[.]" *Capalik*, [0056]. A POSITA would have understood that *Capalik's* sockets are "network socket connections." *Dec*., ¶154.

M. Claim 7: "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."

Capalik teaches activity sources include files, malicious payloads, or other data transfers to the decoy computer. *See* Claim [1(c)]; *Capalik*, [0049] (teaching monitoring sources "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**"). A POSITA would have understood these comprise at least a file, an instruction, data stream, and/or network socket connection. *Dec*., ¶155 .

N. Claim 13: "The method of claim 1, further comprising identifying activity sources that are affected by unauthorized activities."

Capalik teaches identifying a plurality of activities being performed on a VM, wherein each of the activities includes an activity source, target, and association. *See* limitation [1(c)]. Because the identified plurality of activities includes unauthorized activities, a POSITA would have understood that *Capalik* also teaches identifying activity sources that are affected by unauthorized activities. *Dec*., ¶156; *see* Claims [1(Pre)], [1(c)].

O. Claim 16: "The method of claim 13, wherein the identifying activity sources that are affected by unauthorized activities includes identifying data stream incoming from outside the computer system."

See Claim 13. *Capalik* discloses identifying activity sources that are affected by unauthorized activities including identifying data stream 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 computer system 500); *Dec*., ¶158.

P. Claim [19(Pre)]:[8]

See Claim [1(Pre)]. To the extent the preamble is limiting, a POSITA would have understood *Capalik's* decoy system discussed in Claim [1(Pre)] is a system. *Dec*., ¶159.

Q. Claim [19(a)]:

See Claim [1(a)].

R. Claim [19(b)]:

See Claim [1(b)].

S. Claim [19(c)]:

See Claim [1(b)]. As discussed above in Claims [1(a)]-[1(e)], *Capalik* 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

[8] Complete claim language is included in the Claim Appendix.

more programs, respectively including instructions for (1) monitoring activity on the VM and identifying a plurality of activities being performed at the VM, 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*., ¶¶162-63. The programs of these components include instructions for performing the claimed steps below. *See* Claim [19(d)]-[19(h)].

T. Claim [19(d)]:

See Claim [1(b)].

U. Claim [19(e)]:

See Claim [1(c)].

V. Claim [19(f)]:

See Claim [1(d)].

W. Claim [19(g)]:

See Claim [1(e)].

X. Claim [19(h)]:

See Claim [1(f)].

Y. Claim 20(Pre):

See Claims [1(Pre)]-[1(a)], [19(c)] (mapping one or more programs including instructions for performing the claimed steps, where the mappings for said programs also apply for Claim 20). To the extent the preamble is limiting, a POSITA would

have understood that *Capalik's* decoy system discussed in Claim [1(Pre)] includes a non-transitory computer readable storage medium (e.g., hard disk). *Dec*., ¶169.

Z. Claim [20(a)]:

See Claim [1(b)].

AA. Claim [20(b)]:

See Claim [1(c)].

BB. Claim [20(c)]:

A POSITA would have understood that "the memory" is the non-transitory computer readable storage medium as claimed in Claim [20(Pre)]. *See* Claim [1(d)], [20(Pre)]; *Dec.* ¶172.

CC. Claim [20(d)]:

See Claim [1(e)].

DD. Claim [20(e)]:

See Claim [1(f)].

VI. GROUND 2: *CAPALIK* IN VIEW OF *GOEL* RENDERS CLAIMS 1-7, 13, 15-16, 19-20 OBVIOUS

To the extent Patent Owner disputes that *Capalik* identifies and stores a source, target, and association for each of the individual activities of an attack, *Goel* demonstrates that it was known in the art to identify and store each of the individual activities of an attack (including the activity source, target, and association) to generate signatures indicative of the activities for forensic analysis.

A. *Capalik-Goel* teaches identifying and storing "activity sources, activity targets, and associations" as recited in independent Claims 1, 19-20.

To avoid repetition, the *Capalik-Goel* combination set forth here highlights *Goel's* role in teaching identifying and storing unauthorized activities wherein each activity includes an activity source, an activity target, and an association, as recited in Claims [1(c)], 19[(e)], and [20(b)]. Petitioner relies on *Capalik's* teachings as set forth in Ground 1 for the rest of the limitations of these independent claims. *See* Ground 1, Claims [1[Pre]-[1(b)], [1(d)]-[1(f)].

Goel teaches analyzing intrusions in a computer system using the Forensix system. *Goel*, 1. Forensix monitors and captures all system activity at a "target system" after an intrusion, including related activities based on a model for attributing system activity. *Goel*, 1-5. Forensix stores captured information in a relational database, supports queries for forensic analysis, and presents the results in human-readable form. *Goel* teaches that Forensix is implemented on Linux OS and comprises (1) an auditing module on the target system, (2) a receiver daemon and a database injector running on the backend storage system, and (3) database queries and scripts that allow replay of system activities for forensic purposes. *Goel*, 4.

1. *Goel* Identifies Activity Source, Activity Target, and Association

Goel teaches a goal of "completeness" for identifying system activity, where "*all* system-call activity is captured and can be attributed to users, processes or

connections" to "accurately reconstruct *all* security incidents, immaterial of the type of attack." *Goel*, 2, 3 ("In its current implementation, all activity across the system-call interface is captured and logged."). *Goel's* target system includes an auditing module comprising a Linux kernel module that traps system calls and logs data in a kernel buffer. *Goel*, 4. The auditing module hijacks the system call table and adds stub code around several system calls to capture the system call, its timing, its parameters, its return value, and the PID of the process making the call. *Id*. The target system's auditing module traces system calls that include networking calls (e.g., *connect*, *accept*, *listen*, and *shutdown*), process management calls (e.g., *fork*, *exec*, *kill*, *exit*, and *setuid*), and file system calls (e.g., *open*, *read*, *write*, *close*). *Goel*, 4-5.

Goel teaches tracing system calls based on a model for "attributing system activity" as depicted in Figure 2, where, in one example, a process is an activity source (annotated blue) and a file (annotated red) is an activity target that the process acts on (i.e., association, annotated yellow):

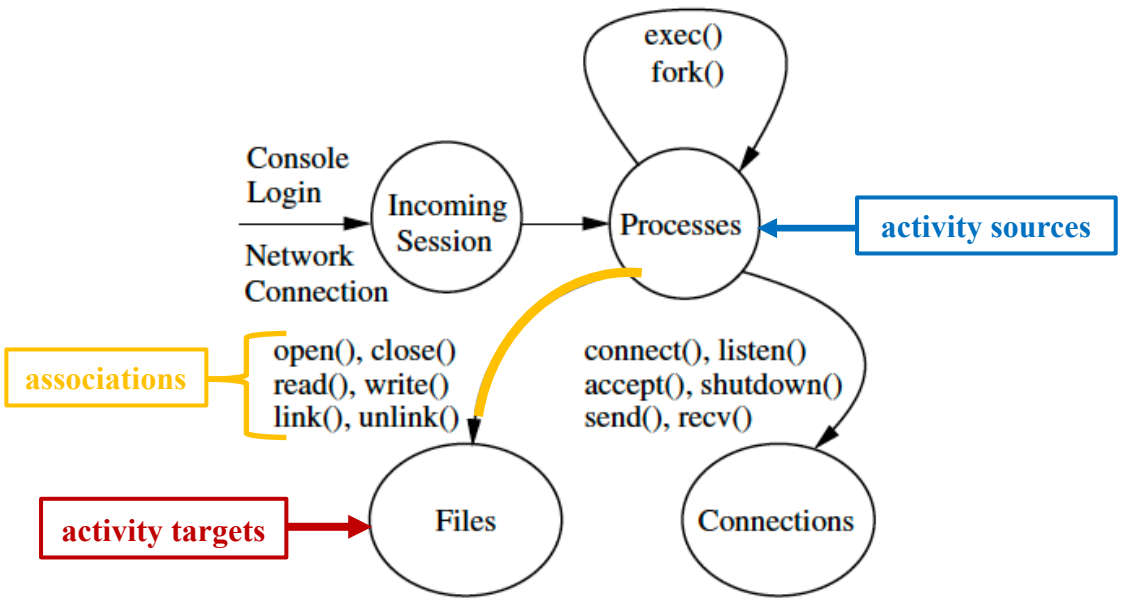


Figure 2. Relationships of system activities.

Goel, 4 ("Figure 2") (annotated), *generally*, 2-4.

Specifically, *Goel* teaches monitoring and capturing intrusion activity starting with a network connection or a console login and following the chain ("cascade") of related system activities including other processes, file accesses, and outgoing connections that are linked to the intrusion activity. *Goel*, 3. *Goel* teaches capturing system activities and their relationships (as depicted in figure 2) by assigning the identifier or the PID of the process that executes the activity as ownership information to each link of the graph. *Id*. At a minimum, "each system-call trace record has an associated PID and a time-stamp that helps to construct the activity relationship" and forensic queries. *Goel*, 3 (*Goel* further captures "each system call

and, among other information, its timing, parameters, return values, the process issuing the call, and the owner of that process, throughout the lifetime of the server").

A POSITA would have understood *Goel* teaches identifying a plurality of activities being performed in a computer system by attributing system activity to, for example, individual connections and sessions (as depicted in Fig. 2) and determining the events that "cascade" from an initial connection or session. *Goel*, 3; *Dec*., ¶¶178-182. Further, a POSITA would have understood that *Goel* teaches identifying activity sources (e.g., processes) and activity targets (e.g., files) by monitoring system calls to identify intrusions and follow as the intrusion "cascade[s] into multiple system activities including other processes, file accesses, and outgoing connections." *Goel*, 3, Fig. 2; *Dec*., ¶178-182. Similarly, a POSITA would have understood *Goel* identifies actions (i.e., associations) such as read() and write() occurring between processes and files, as depicted in Fig. 2. *Dec*., ¶178-182.

2. Motivation to Combine *Capalik-Goel*

A POSITA would have been motivated to modify *Capalik* per *Goel's* teachings to identify and store unauthorized activities including the activity source, activity target, and association of each activity. *Dec*., ¶¶183-90. As discussed above, *Capalik* teaches a decoy system that identifies and stores a plurality of activities being performed at a VM of decoy computer 502. *Goel* teaches a forensic analysis system (Forensix) that similarly identifies and stores a plurality of activities being

performed at a target computer system, wherein each of the activities includes an activity source, an activity target, and an association. *Goel*, 3-4, Fig. 2.

Modifying *Capalik* as proposed would have resulted in an improved IDS. *Dec*., ¶¶187-90. This modification applies the well-known computer security principle that gathering more detailed threat information provides improved threat analysis. *Id*., ¶187. Modifying *Capalik* per *Goel's* teachings would have been use of *Goel's* known technique that would have resulted in *Capalik* identifying and storing individual attack activities (including activity sources, targets, and associations) to improve *Capalik's* similar IDS in the same manner as *Goel's* IDS. *Id*., ¶187.

Goel teaches that although the amount of data collected can be large, the system is feasible given all of the networking, CPU, and storage capacity available at the time. *Goel,* 3. *Goel* teaches that, as a result, "sacrificing some host and networking resources in order to add forensic capability will be a fairly attractive proposition." *Goel*, 3. Thus, a POSITA would have understood that even if there were a modest increase in processing power required to modify *Capalik* per *Goel's* teachings, a POSITA would have been motivated to do so to achieve security improvements at the expense of more processing overhead required. *Dec*., ¶188. Such a modification would have resulted in an improved method and system to analyze unauthorized intrusions in a decoy system. *Id*., ¶188. *Goel* teaches that its methods for identifying and storing an attacker's unauthorized activities (including

activity sources, targets, and associations) in a computer system allow for more efficiently determining how an attacker entered the system and what the attacker did after the intrusion. *Goel*, 1; *Dec*., ¶188.

Further supporting the motivation to combine these references, as set forth in detail above, *Capalik* and *Goel* describe similar systems. *Dec*., ¶183-90. Both describe monitoring unauthorized intrusions being performed on a computer system and identifying related activities after an intrusion. *Id.*, ¶189. Both describe using operating systems such as Linux to interact with the unauthorized intrusion. *Id*. Both describe monitoring and storing information indicative of the unauthorized activities on the computer system to protect computer networks against future attacks. *Id*. Although *Goel* is silent on the use of a decoy system, *Goel* explains that the authors were actively working on implementing the Forensix system on a VM. *Goel*, 2. Given these similarities, a POSITA would have recognized that *Goel's* benefits would have been applicable to *Capalik*. *Dec.* ¶189.

A POSITA would have had a reasonable expectation of success (REOS) when modifying *Capalik* per *Goel's* teachings. *Dec*., ¶183-90. 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*., ¶190. *Goel* teaches that its system implementation can be achieved by adding lines of code. *Goel*, 4-5 (explaining implementation on Linux

OS, and that auditing module code is derived from the SNARE project, database is MySQL). Therefore, a POSITA would have understood that modifying *Capalik's* system (e.g., Linux-based) to additionally monitor, identify, and store each individual activity of an attack including "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.* ¶190. A POSITA would have also had a REOS based on the similarities between *Capalik's* and *Goel's* computer devices—both in structure and in function. *Id*., 183-90 (explaining both references teach forensic analysis systems including monitoring and storing information about unauthorized activities performed on the computer system, and both teach utilizing Linux OS).

B. *Capalik-Goel* Renders Claims 2-7, 13, 16, 19-20 Obvious.

See Ground 1, Claims 2-7, 13, 16, 19-20.

C. *Capalik-Goel* Renders Claim 15 Obvious.

1. Claim 15: "The method of claim 13, 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 Ground 1, Claim 13. *Capalik* teaches one or more processors. *See* Ground 1, Claim 1. As discussed below, *Capalik-Goel* one or more processors execute instructions at various privilege levels (i.e., respective privilege levels). *Goel*

discloses running several local privilege escalation attacks using Forensix, and constructing a local privilege escalation query, which identifies such attacks regardless of the vulnerability they exploit by tracking a common attack pattern. *Goel*, 7-8. A common attack pattern includes using an elevated effective user ID (eu_id) of 0 to change the attacker's real user ID (u_id) permanently to spawn a root shell. *Goel*, 8. As *Goel* explains "[l]ocal privilege escalation attacks allow a user with normal privileges to gain and retain superuser privileges illegally" and "a local root exploit takes a user process with normal privileges and converts it into a shell with superuser privileges[.]" *Goel*, 7. *Goel* teaches writing a local privilege escalation query to identify and extract root exploits that occurred in the system. *Goel*, 8. Accordingly, a POSITA would have understood that *Goel's* queries identify at least an activity source that requests to execute underprivileged instructions, as claimed. *Dec*., ¶¶192-95.

VII. GROUND 3: *CAPALIK* ALONE OR IN VIEW OF *GOEL* AND IN FURTHER VIEW OF *KING* RENDERS CLAIMS 8-12, 15, 17, 21-23 OBVIOUS

A. Claim 8: "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(c)]-[1(d)], *Capalik* (Ground 1) or *Capalik-Goel* (Ground 2) teaches identifying a plurality of activities (including activity sources, activity targets, and associations) performed at the VM, and retaining detailed forensic data that can be used to generate signatures indicative of

the activities. As explained below, *King* teaches graphically displaying the stored activities, including a subset of the stored activities.

1. *'697 Patent's* graphically displaying a subset of the stored activities

The '697 Patent teaches graphically displaying activity diagrams, e.g., FIGS. 3A-3D and 4A-4E. *'697 Patent*, 16:26-30.

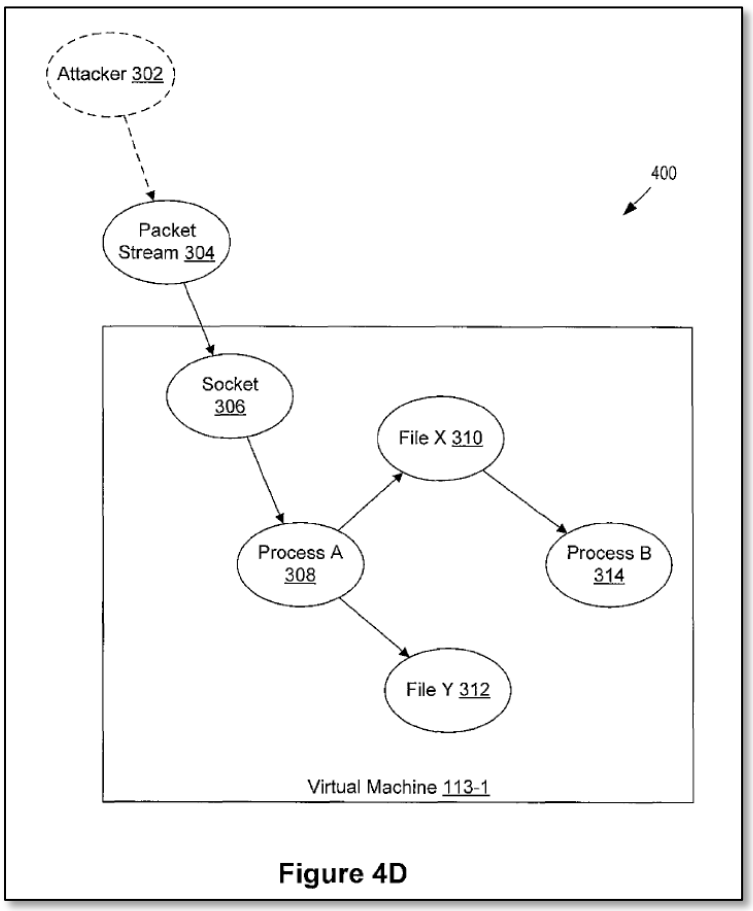
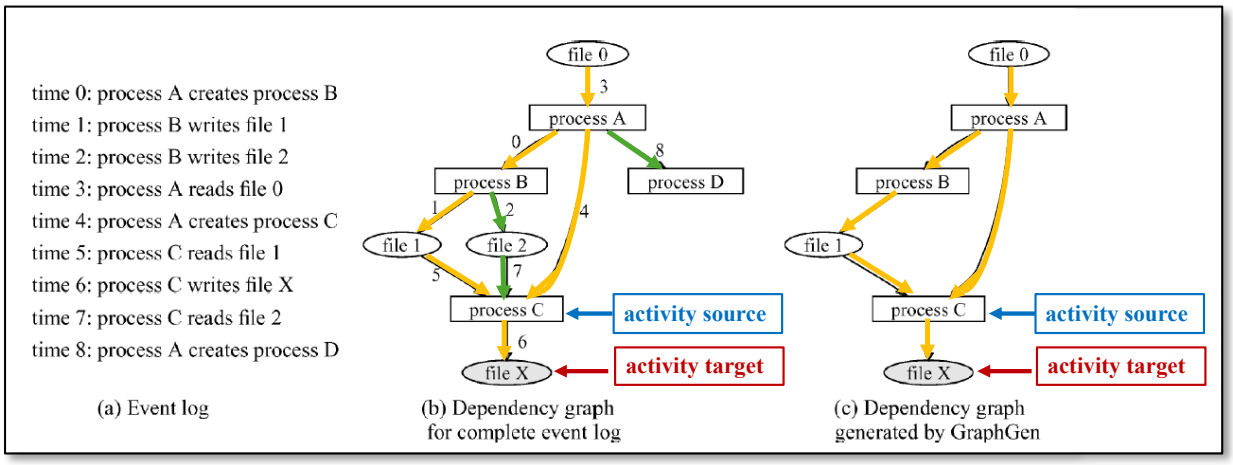


Figure 4D

'697 Patent, Fig. 4D.

2. *King* teaches graphically displaying a subset of stored activities

King teaches analyzing unauthorized activities using operating-system-level (OS-level) information-flow graphs to highlight an attacker's activities inside a VM by following the flow of information from one object to another, and the dependency-causing events between objects. *King*, 10, 13. *King* discloses "EventLogger" identifies and stores information-flow events in real-time, and "GraphGen" processes EventLogger logs to identify and graphically display information-flow graphs of an intrusion. *King*, 12. *King* discloses dependency graphs that "represent the flow of information either backward in time or forward in time[,]" depending on which of *King's* systems (Backtracker or ForwardTracker) was used to generate the graph. *King*, 17. *Dec.*, ¶¶198-200.



King, Fig. 3.2 (annotated).

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, not depicted in Fig. 3.2) 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 (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"). *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.*, ¶204 (noting that each arrow in the dependency graphs represents an activity source and target pair). Figure 3.2(c) depicts a filtered graph, ignoring events and objects that do not affect file X by time 10. *Dec.* ¶201; *King*, 18-19.

A POSITA would have understood that *King's* GraphGen graphically displays a plurality of activities performed at the VM wherein each of the activities includes an activity source, activity target and an association between the activity source and activity target. *Dec.*, ¶¶198-202. A POSITA would have understood each event depicted from time 0 to time 8 in Fig. 3.2(a) describes actions (e.g., reading/writing) between respective activity sources and targets. *Id.*; *'697 Patent*, 13:6-10. 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*., ¶¶203-06. *King's* system prunes away benign associations, resulting in a filtered graph (Fig. 3.2(c)) depicting only activities with malignant associations. *Id*.

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*., ¶206.

A POSITA would have understood *King's* dependency graphs depicted in Fig. 3.2 are similar to Figs. 3A-3D, 4A-4D of the '697 Patent and graphically display at least a subset of the stored activities, including activity sources, targets and associations. *Dec*., ¶¶196-207.

3. Motivation to Combine *Capalik-King/Capalik-Goel-King*

A POSITA would have been motivated to modify *Capalik/Capalik-Goel* per *King* to display a subset of the stored activities, including activity sources, activity targets, and associations for each activity. *Dec*., ¶¶208-16. As discussed above, *Capalik/Capalik-Goel* teaches a decoy computer that identifies, and stores activities performed at the decoy computer's VM. *King* teaches a similar decoy computer that identifies, stores, and graphically displays attack activities including the activity sources, activity targets, associations, and an indication of which individual activities are malignant. *See King*, 10-13, 17-18, Fig. 3.2.; *Dec*., ¶¶208-09.

Modifying *Capalik/Capalik-Goel* as proposed would have resulted in an improved IDS. *Dec*., ¶¶210-13. This modification applies the well-known computer security principle that gathering more detailed threat information provides improved threat analysis. *Dec*., ¶211. Further, additional threat analysis (e.g., displaying a subset of stored activities, including activity sources, targets, and associations) would have enhanced threat detection and would have improved generated signatures. *Dec*., ¶212. Indeed, such displaying would have allowed an administrator to readily identify the attack path. *Id*. *King* teaches its system has "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 an increase in processing power were required to modify *Capalik/Capalik-Goel* 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*., ¶213.

Modifying *Capalik/Capalik-Goel* per *King* would have been use of a known technique of identifying and graphically displaying activity source, target, and association to improve *Capalik's/Capalik-Goel's* similar system in the same way as *King's* system. *Dec*., ¶214. Such a modification would have resulted in an improved analysis of unauthorized intrusions in a decoy system. *Id*. *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; further, *King* teaches preventing future intrusions by updating or disabling any vulnerable applications. *King*, 2; *Dec*., ¶214. The modified *Capalik/Capalik-Goel* would also have appreciated these benefits. *Dec*., ¶214.

Further supporting the motivation to combine these references, as set forth in detail above, *Capalik/Capalik-Goel* and *King* teach similar systems. *Dec*., ¶215 (explaining similarities in detail). Given these similarities, a POSITA would have recognized modifying *Capalik/Capalik-Goel* per *King* would have achieved *King's* security benefits. *Id*.

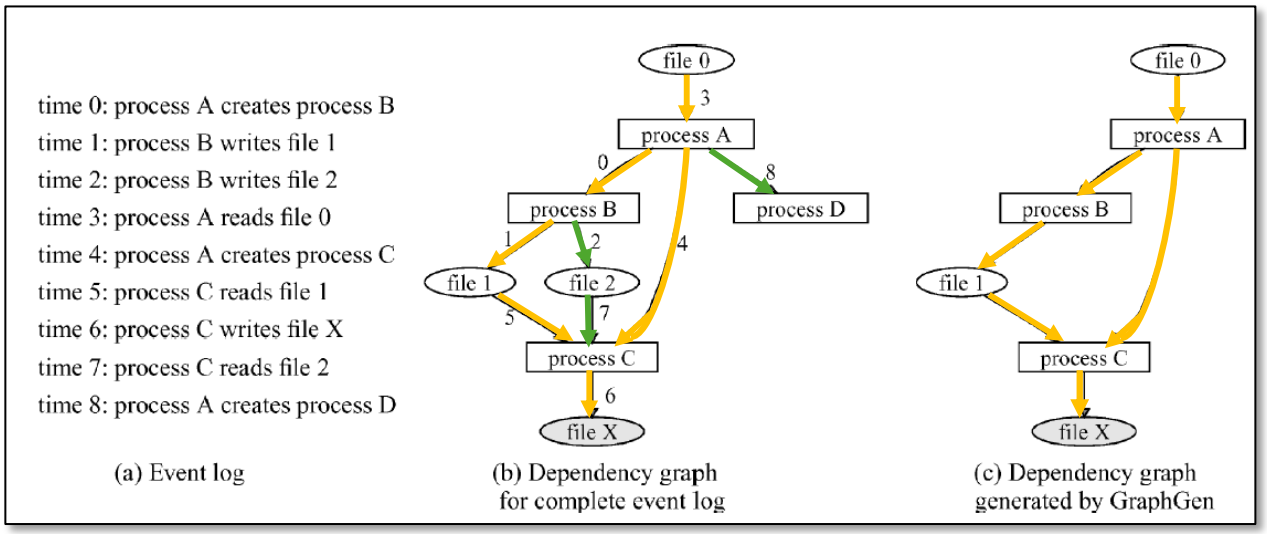
A POSITA would have had a REOS modifying *Capalik/Capalik-Goel* per *King's* teachings. *Dec*., ¶216. Such a modification would have utilized *Capalik's/Capalik-Goel's* VMM, relational database, and visualization interface which already have monitoring, storing, and graphically displaying capabilities for unauthorized activities. *Id*. *King* also teaches GraphGen "runs on any computer used by the administrator[.]" *King*, 25. Therefore, a POSITA would have understood that modifying *Capalik's/Capalik-Goel's* system to additionally graphically display the stored activities (including activity sources, activity targets, and associations) would have required straightforward modifications to program/code instructions, which would have been well-within the skillset of a POSITA. *Dec*., ¶¶208-16.

B. Claim 9: "The method of claim 8, wherein the displaying includes visually distinguishing at least a subset of the associations."

As explained above in Claim 8, *Capalik-King/Capalik-Goel-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 '697 Patent explains that, for example, a bold arrow can represent a malignant association and a regular arrow can represent a benign association. *'697 Patent*, 12:4-12.

As explained in Claim 8, *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, including only malignant associations (annotated yellow):



King, Fig. 3.2 (annotated); *Dec*., ¶219. 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. *Id*. Accordingly, a POSITA would have understood that *Capalik-King/Capalik-Goel-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.*, ¶¶217-19.

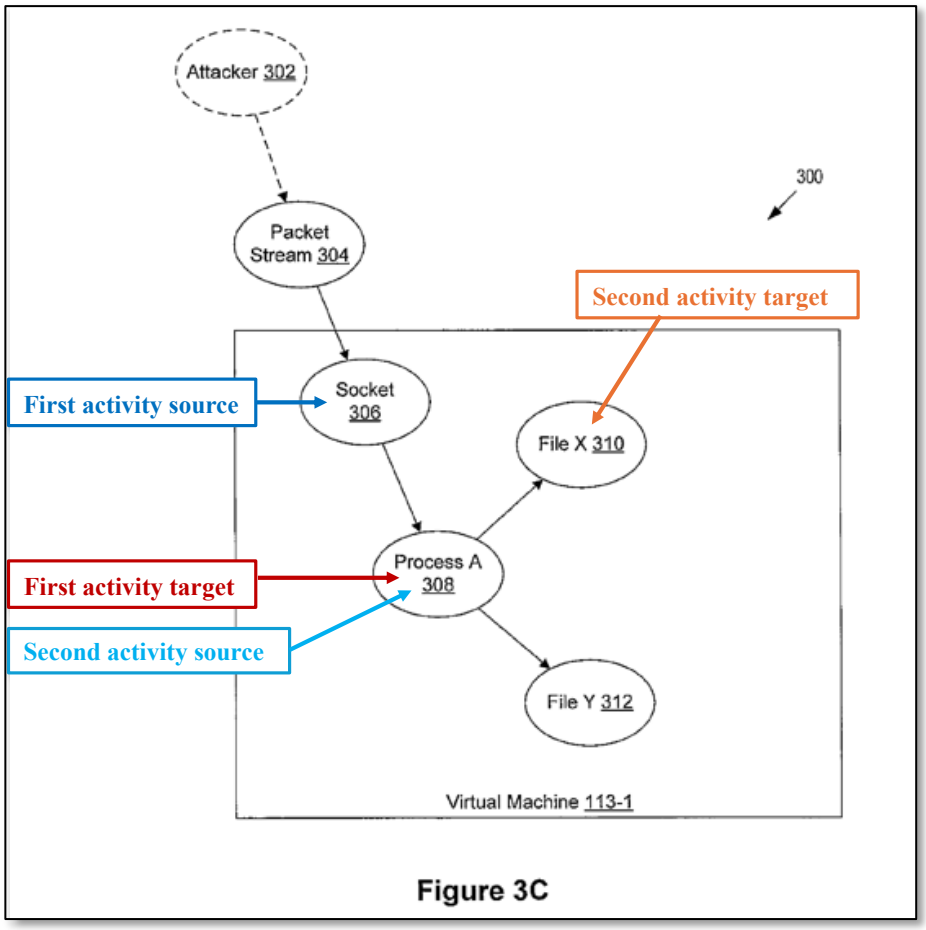
As discussed above, a POSITA would have been motivated to modify *Capalik/Capalik-Goel* per *King's* teachings and would have had a REOS. *See* Claim 8; *Dec.* ¶220.

C. Claim 10: "The method of claim 8, wherein 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."

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 8. As explained below, *Capalik-King/Capalik-Goel-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. '697 Patent's Teachings

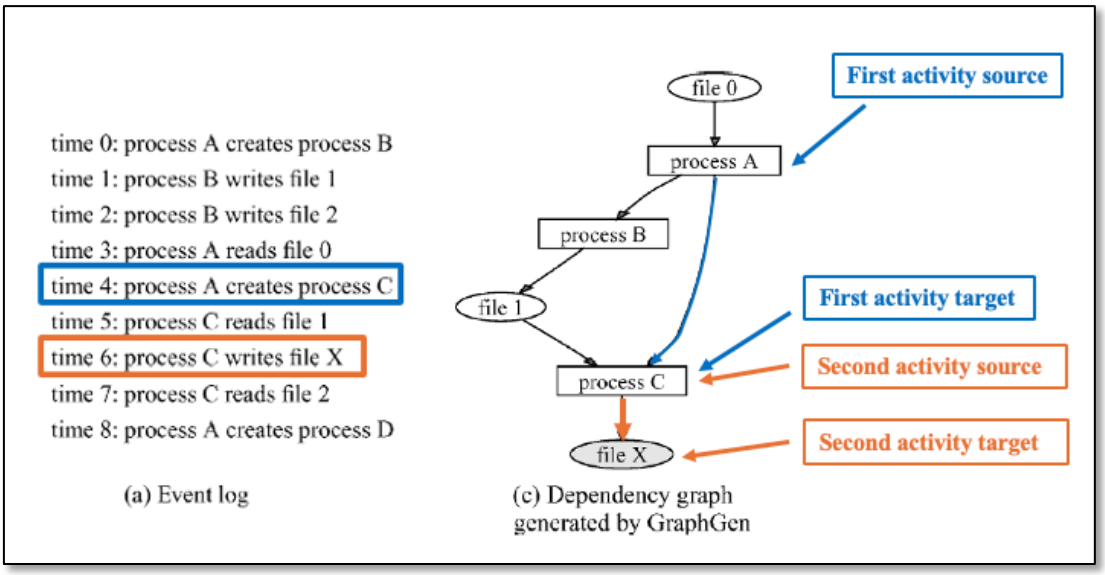
The '697 Patent points to Figure 3C (below) as an exemplary embodiment of this Claim:



'697 Patent, 16:39-49, Fig. 3C (annotated); *Dec*. ¶222. The '697 Patent explains that Figure 3C illustrates a first activity between socket 306 and process A 308 (i.e., data from socket 306 is directed to process A 308) and a second activity between process A 308 and file X 310 (i.e., process A 308 accesses file X). Process A 308 is the activity target of the first activity (annotated red) and the activity source of the second activity (annotated light blue). *Id*., 16:42-46.

2. *King's* Teachings

Similar to the '697 Patent, *King* depicts a similar subset of activities in dependency graph 3.2(c):



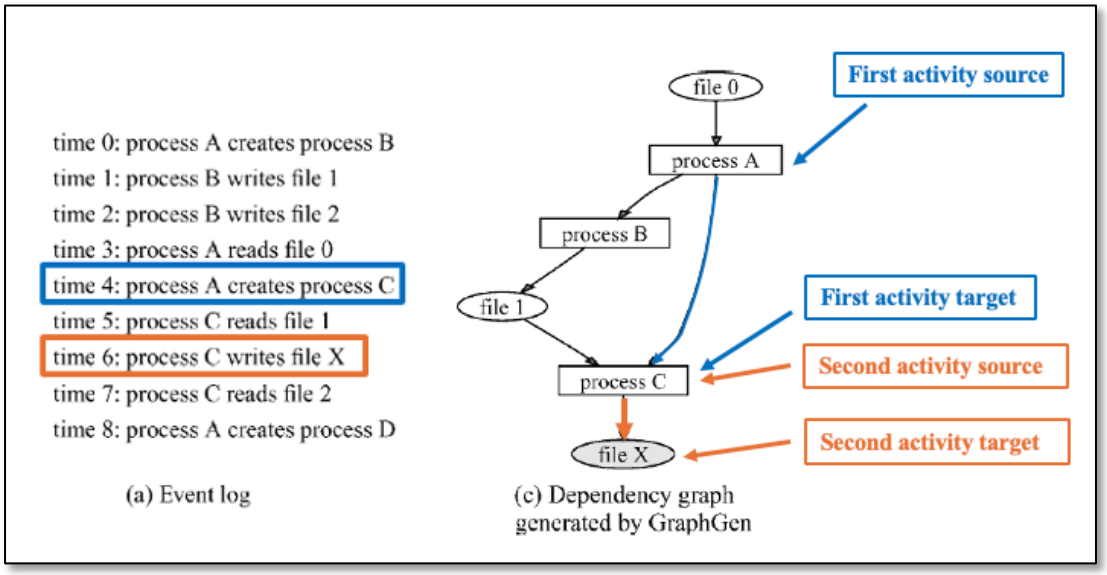
King, Fig. 3.2 (annotated); *Dec.*, ¶¶198-216, 221-24. The annotations above highlight a first unauthorized association (blue) between a first activity source (process A) and a first activity target (process C). Namely, "process A creates process C" at time 4. The annotations above also highlight a second 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.* ¶221-23.

As discussed above, a POSITA would have been motivated to modify *Capalik/Capalik-Goel* per *King's* teachings and would have had a REOS. *See* Claim 8; *Dec.*, ¶¶198-216, 221-24.

D. Claim 11: "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*/*Capalik-Goel* teaches identifying a plurality of activities performed at a VM, wherein each activity includes an activity source, activity target, and association. *See* Claim [1(c)]. As explained below, *Capalik-King/Capalik-Goel-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:



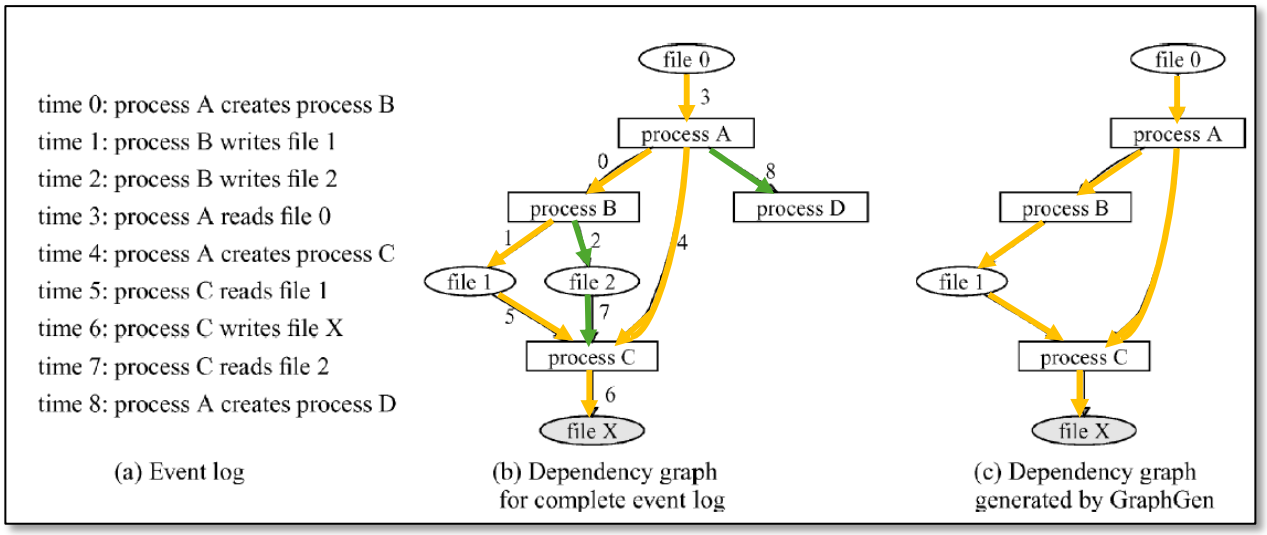
King, Fig. 3.2 (annotated); *Dec.*, ¶¶198-216, 225-229. The annotations above highlight a first unauthorized association (blue) between a first activity source

(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 8, 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.,* ¶¶226.

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.,* ¶¶198-216, 225-229.

E. Claim 12: "The method of claim 11, further comprising tracking a transfer of malicious associations over time."

See Claim 11. As explained above, *Capalik-King/Capalik-Goel-King* teaches identifying and tracking a plurality of activities over time, wherein each activity includes an activity source, activity target, and an association. *See* Claims 8-11. As explained in Claims 8-11, *King's* dependency graphs distinguish between malignant and benign associations, and a filtered graph (Fig. 3.2(c)) depicts only malignant associations:



King, Fig. 3.2 (annotated), *Dec*., ¶230. Accordingly, a POSITA would have understood *Capalik-King/Capalik-Goel-King* teaches tracking a transfer of malicious associations over time. *Dec*., ¶230-31.

F. Claim 15: "The method of claim 13, 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 13. *Capalik/Capalik-Goel* teaches one or more processors. *See* Claim 1. To the extent it is argued *Capalik-Goel* does not teach Claim 15, *Capalik-King/Capalik-Goel-King* teaches Claim 15.

As discussed below, *Capalik-King's/Capalik-Goel-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 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*., ¶¶232-36.

G. Claim 17: "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 '697 Patent describes 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[.]" *'697 Patent*, 16:53-58. In 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:

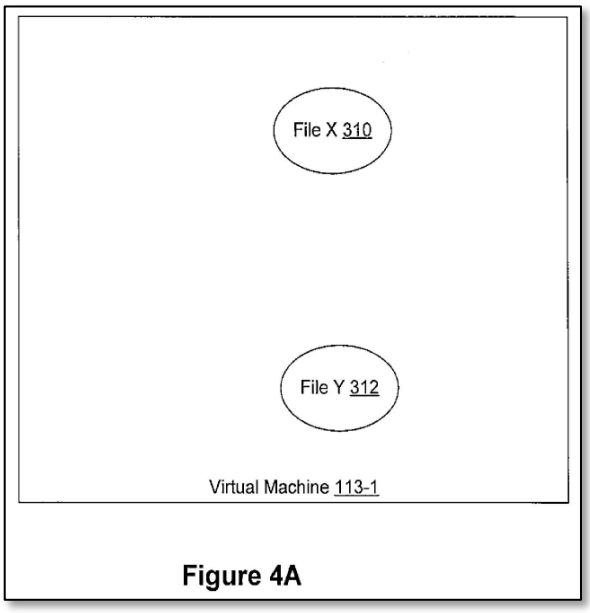


Figure 4A

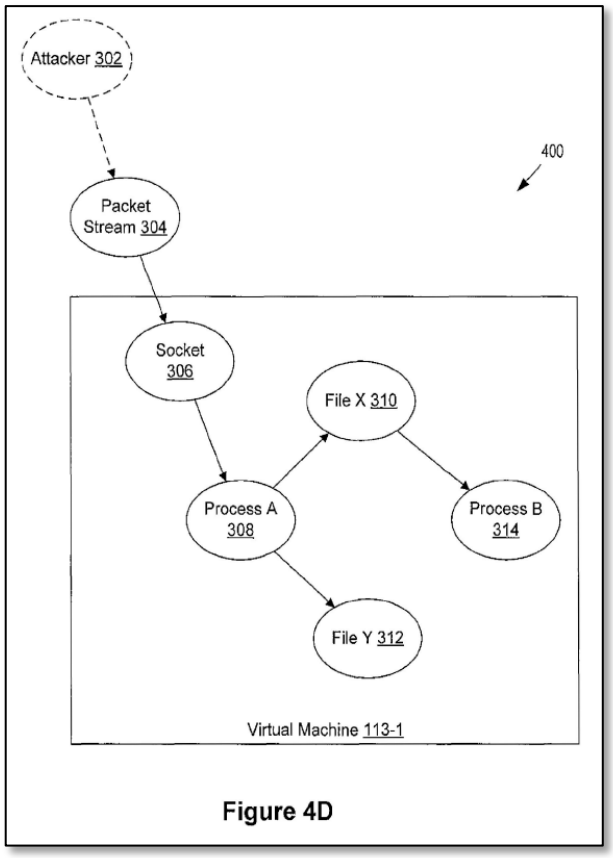
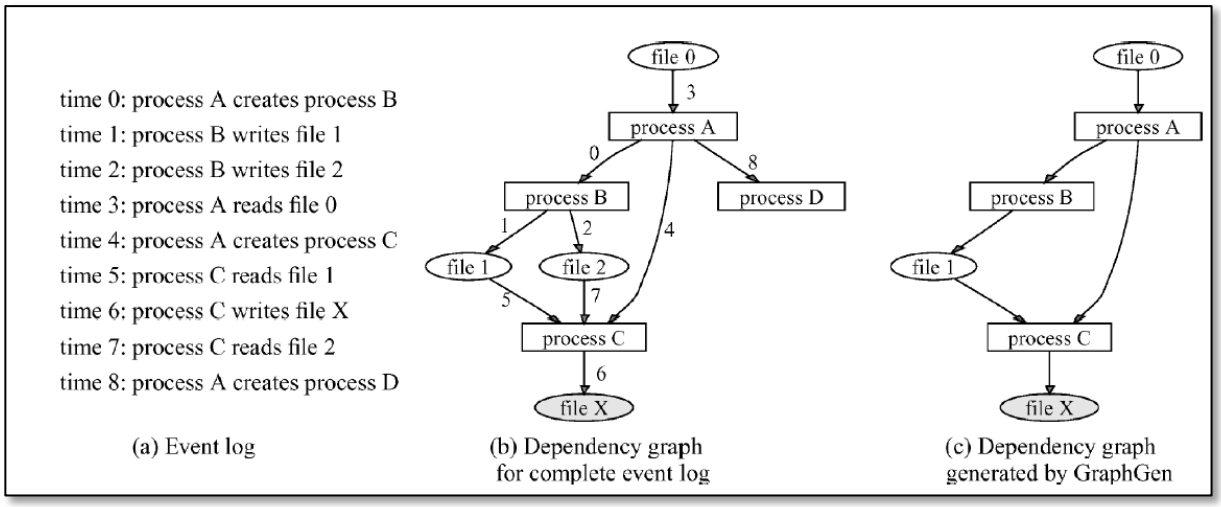


Figure 4D

'697 Patent, Figs. 4A, 4D.

As explained above with respect to Claim [1(c)]-[1(e)], *Capalik*/*Capalik-Goel* teaches identifying, storing, and displaying unauthorized activities wherein each of the activities includes an activity source, target, and association. Like Fig. 4D in the '697 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's relation to every activity target (e.g., process B, file 1, process C, file X) through one or more identified associations.



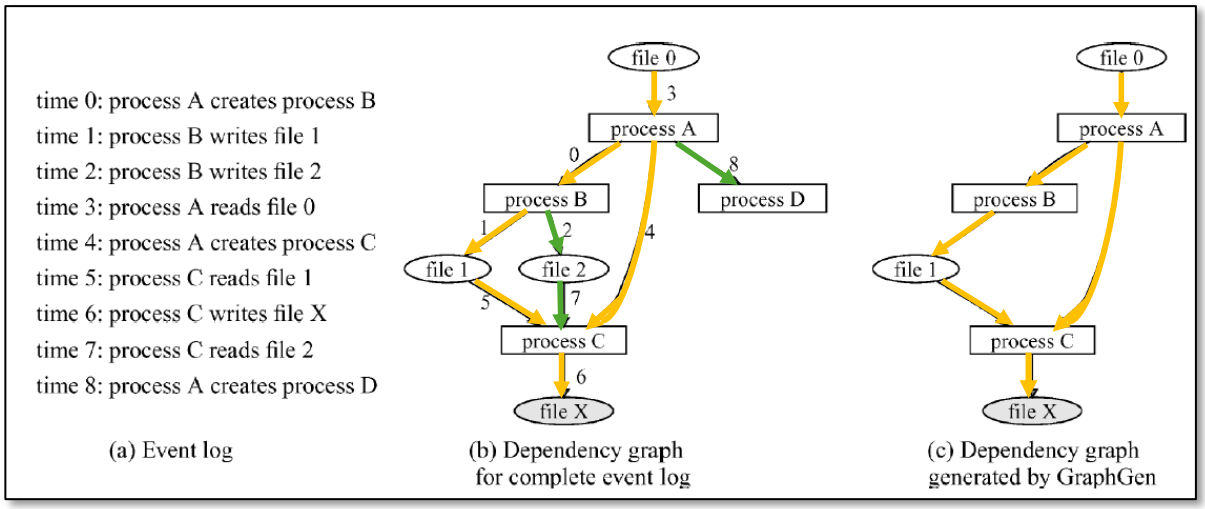
King, Fig. 3.2; *Dec*., ¶¶237-39 (explaining *King* teaches identifying every association between each source and target before benign associations are filtered out by the GraphGen process). Accordingly, *Capalik-King*/*Capalik-Goel-King* teaches determining a complete activity status for the stored events. *Dec*., ¶¶237-41.

H. Claim 21:

See Ground 1 Claim [1(c)], Ground 2 Section A, Ground 3, Claim 10.

I. Claim 22: "The method of claim 1, further comprising: identifying a type of the association between the activity source and the activity target."

As explained above in Claim 8, *Capalik-King/Capalik-Goel-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:



King, Fig. 3.2 (annotated); *Dec*., ¶243. 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*., ¶243; *see* Claim 17. *King* teaches pruning away activities that include benign associations, resulting in a filtered graph (Fig. 3.2(c)) depicting only activities that include malignant associations. *Dec*., ¶243.

J. Claim 23:

See Ground 1 Claim [1(c)], Ground 2 Section A, Ground 3, Claims 10-11.

VIII. GROUND 4: *CAPALIK* ALONE OR IN VIEW OF *GOEL*, AND IN FURTHER VIEW OF *PIKE* RENDERS CLAIM 14 OBVIOUS

A. Claim 14: "The method of claim 13, 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."

As explained above, *Capalik* (Ground 1) or *Capalik-Goel* (Ground 2) teaches Claims 1, 13. *Capalik* expressly teaches monitoring for memory-related errors such as buffer overflow attacks. *Capalik*, [0057]. But *Capalik* does not discuss using NX (i.e., no execute) bit technology or explain how buffer overflow attack sources are identified. As set forth below, a POSITA would have been motivated to supplement *Capalik* (or *Capalik-Goel*) with *Pike's* NX bit technology techniques, which are particularly effective against buffer overflows attacks.

1. *Pike's* teachings

Pike teaches NX bit technology techniques are particularly effective against buffer overflows attacks from internet worms. *Pike*, 3:12-21. As explained below, *Capalik-Pike/Capalik-Goel-Pike* teaches identifying activity sources affected by unauthorized activities that request to access a portion of memory that is set as non-executable.

The '697 Patent describes 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, including using a non-executable bit. *'697 Patent*, 15:40-55 (describing DEP mechanisms were known); *Dec*., ¶247.

As discussed below, *Pike* teaches one such DEP mechanism. *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.

Pike teaches setting a relevant no-execute bit that triggers an exception. *Pike*, 8:65-9:12, 9:24-10:1, 12:58-13:22. An exception handler is used to evaluate the likelihood that a particular exception was caused by a dangerous interpretation of a received input string. *Id*. A POSITA would have understood that potentially dangerous input strings (e.g., data packets) are activity sources affected by unauthorized activities. *Dec*., ¶248-49. 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 that identifies activity sources affected by unauthorized activities. *Id*. 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, as claimed. *Id*., ¶¶245-49.

2. Motivation to Combine *Capalik-Pike/Capalik-Goel-Pike*

A POSITA would have been motivated and found it obvious to modify *Capalik/Capalik-Goel* per *Pike's* teachings such that *Capalik-Pike/Capalik-Goel-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*., ¶250-53. As explained in Claim 13, *Capalik/Capalik-Goel* teaches identifying a plurality of unauthorized activity sources but is silent regarding identifying unauthorized activity sources that request access to a portion of memory of the target computer system. A POSITA would have recognized modifying *Capalik/Capalik-Goel* per *Pike's* teachings would have allowed identifying a larger variety of unauthorized activity sources, including those that employ infection mechanisms that request access to the memory of a target system. *Dec*., ¶¶250-53.

Pike discloses using a variety of techniques to identify the activity sources that request access to a system's memory, 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 (DEP) for malware detection was known in the art per *Pike*. *Dec*., ¶¶247-52. Thus, such modification would have been use of a known technique (DEP) to identify unauthorized activity sources that request to access a portion of memory that is set as non-executable to improve *Capalik's/Capalik-Goel*'s similar system in the same

way as *Pike. Dec*., ¶¶250-53. Although supplying an NX bit could impose overhead on the system, a POSITA would have been motivated to accept additional overhead in exchange for more robust malware attack detection. *Id*.

A POSITA would have had a REOS modifying *Capalik/Capalik-Goel* per *Pike's* teachings because *Capalik/Capalik-Goel* already teaches similar hardware topology (e.g., one or more processors, memory, VM) to support *Pike's* functionality of identifying activity sources that access portions of memory set as non-executable. *Dec*., ¶253; *Pike*, 6:19-24. Modifying *Capalik/Capalik-Goel* per *Pike* to apply a known technique (NX bit/DEP) would have required straightforward modifications to software/programming instructions well within the skillset of a POSITA. *Dec*., ¶¶250-53.

IX. GROUND 5: *CAPALIK* ALONE OR IN VIEW OF *GOEL*, AND IN FURTHER VIEW OF *FARLEY* RENDERS CLAIM 18 OBVIOUS

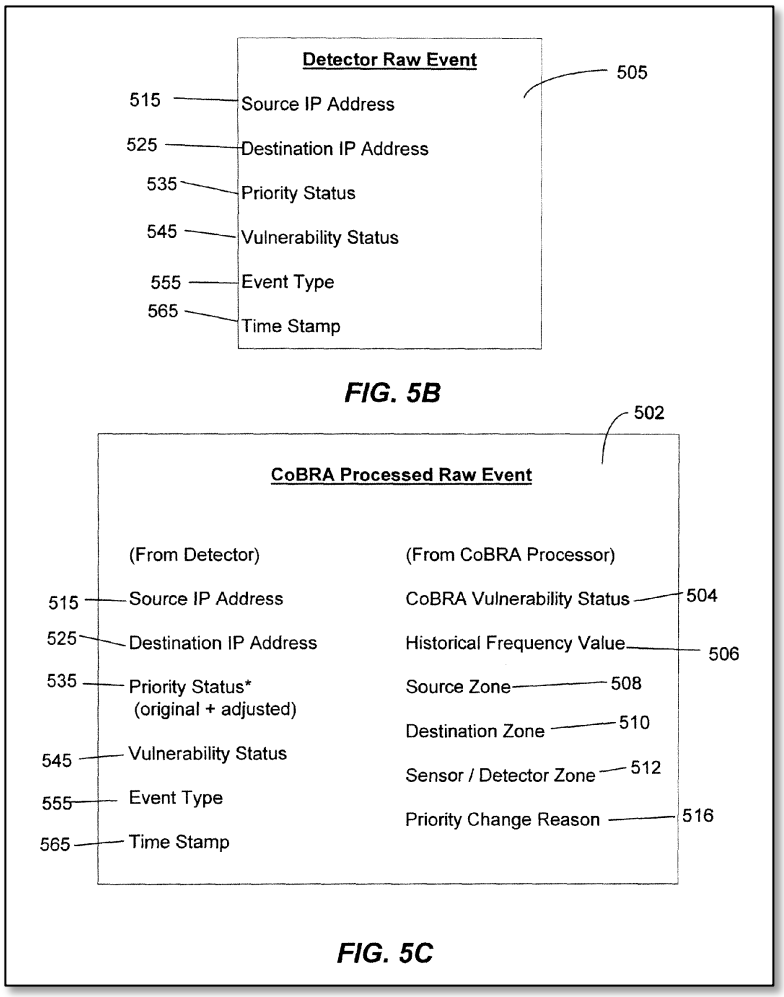
A. Claim 18: "The method of claim 1, further comprising determining an activity level representing a frequency of unauthorized activities on the virtual machine during a predefined time interval."

As explained above, *Capalik* (Ground 1) or *Capalik-Goel* (Ground 2) teaches Claim 1.

1. *Farley's* Teachings

As explained below, *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.

Farley teaches assembling suspicious computer activity ("raw events") from multiple data sources (e.g., IDS/IPS, firewalls, audit systems). *Farley*, 3:47-4:11. *Farley* teaches analyzing assembled activity to detect relationships between raw events that may indicate malicious behavior ("mature correlation event"), and provides an organized data presentation, 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. For example, Fig. 5C includes Historical Frequency Value 506. *Farley*, 12:13-27, 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*, 24:27-39. Accordingly, a POSITA would have understood that *Farley's* "historical event frequency value" is an activity level representing a frequency of unauthorized activities on a computer system during a predefined time interval (e.g., per day). *Dec*., ¶¶254-59; *compare with '697 Patent*, 12:41-52.

2. Motivation to Combine *Capalik-Farley/Capalik-Goel-Farley*

A POSITA would have been motivated and found it obvious to modify *Capalik/Capalik-Goel* per *Farley* such that *Capalik-Farley/Capalik-Goel-Farley* would have determined an activity level (i.e., historical frequency value) representing a frequency of unauthorized activities on the VM. *Dec*., ¶¶260-65.

Modifying *Capalik/Capalik-Goel* as proposed would have utilized a known technique to track historical frequency values of intrusions/attacks and would have improved *Capalik/Capalik-Goel's* IDS in the same way as *Farley's* by tracking unauthorized activity levels on the VM. *Id*. Further, *Farley* teaches a historical frequency value (and/or 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*, 15:11-36, 23:53-60; *Dec*., ¶262-63.

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*., ¶264. Although determining a historical frequency value could result in increased overhead in *Capalik-Farley/Capalik-Goel-Farley*, a POSITA would have been motivated to trade increased overhead for these IDS security improvements. *Id*. Thus, the proposed modification would have used a known technique of determining a historical frequency value of suspicious/unauthorized activities to improve IDS forensic analysis and would have benefited *Capalik/Capalik-Goel's* similar system in the same way as *Farley's*. *Id*.

A POSITA would have had a REOS modifying *Capalik/Capalik-Goel* per *Farley*. *Dec*., ¶265. Such a modification would have utilized *Capalik's/Capalik-*

Goel's processing module that generates an attack report that already includes attack-identifying information from the VM. *Id*. Therefore, a POSITA would have understood that modifying *Capalik's/Capalik-Goel's* system to additionally capture "a historical frequency value" would require straightforward modifications to programming instructions, which was well within the skillset of a POSITA. *Id*. A POSITA would have also had a REOS based on the functional similarities between *Capalik's/Capalik-Goel's* IDS and *Farley's* IDS. *Id*. (explaining that these references teach IDSs that monitor, store and graphically display information about suspicious/unauthorized activities).

X. 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). *Advanced Bionics, LLC v. MED-EL Elektromedizinische Geräte GmbH*, IPR2019-01469, Paper 6 at 8 (PTAB Feb. 13, 2020) (precedential) ("*Advanced Bionics*").

Each of the Proposed Grounds relies on *Capalik* which was incorporated by reference in the '697 Patent but was not discussed by the Examiner during prosecution. *See* '697 Patent, 1:21-29; *see also generally '697 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); *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 (*Goel*, *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.

XI. DISCRETIONARY DENIAL IS NOT WARRANTED

A. The *Fintiv* Factors Favor Institution

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*.

XII. 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

XIII. 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 '697 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 (IPR2025-00067) is being concurrently filed to account for a distinct interpretation of "association."

Petitioner is not aware of any other related matters where the '697 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 Postal and Hand-Delivery Address: ERISE IP, P.A. 717 17th St., Suite 1400 Denver, Colorado 80202 Telephone: (913) 777-5600 Fax: (913) 777-5601	Adam P. Seitz (Reg. No. 52,206) Adam.Seitz@eriseip.com Postal and Hand-Delivery Address: 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. 9,106,697 (the "'697 Patent")
Exhibit 1002	File History for U.S. Patent 9,106,697 (the "'697 File History")
Exhibit 1003	Markus Jakobsson Declaration ("*Dec*.")
Exhibit 1004	U.S. Application Publication No. 2008/0016570 to Capalik ("*Capalik*")
Exhibit 1005	Ashvin Goel et al. *"Forensix: A Robust, High-Performance Reconstruction System,"* Proceedings of the 25th International Conference on Distributed Computing Systems Workshops, 2005 ("*Goel*")
Exhibit 1006	Samuel T. King, "*Analyzing Intrusions Using Operating System Level Information Flow,"* Dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in The University of Michigan, 2006 ("*King*")
Exhibit 1007	U.S. Patent No. 8,819,822 to Pike et al. ("*Pike*")
Exhibit 1008	U.S. Patent No. 7,089,428 to Farley et al. ("*Farley*")
Exhibit 1009	Declaration of June A. Munford ("Munford Dec.")
Exhibit 1010	IPR2025-0067 of U.S. Patent No. 9,106,697
Exhibit 1011	WO 2004/097584 to Knight ("*Knight*")
Exhibit 1012	U.S. Application Publication No. 2006/0242701 to Black et al. ("*Black*")
Exhibit 1013	U.S. Application Publication No. 2007/0208822 to Wang et al. ("*Wang*")
Exhibit 1014	U.S. Application Publication No. 2008/0098476 to Syversen ("*Syversen*")
Exhibit 1015	U.S. Patent No., 7,483,972 to Bhattacharya et al. ("*Bhattacharya*")
Exhibit 1016	U.S. Application Publication No. 2005/0193429 to Demopoulos et al. ("*Demopoulos*")
Exhibit 1017	WO 2009/032379 to Stolfo et al. ("*Stolfo*")
Exhibit 1018	U.S. Application Publication No. 2004/0103314 to Liston ("*Liston*")
Exhibit 1019	U.S. Patent No., 8,407,788 to Kestner et al. ("*Kestner*")
Exhibit 1020	U.S. Application Publication No. 2009/0037672 to Colbert et al. ("*Colbert*")
Exhibit 1021	WO 2009/097610 to Moffie et al. ("*Moffie*")
Exhibit 1022	U.S. Patent No., 8,510,827 to Leake et al. ("*Leake*")
Exhibit 1023	U.S. Patent No., 9,098,698 to Ghosh et al. ("*Ghosh*")

Exhibit 1024	U.S. Application Publication No. 2007/0220518 to Verbowski et al. ("*Verbowski*")
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

CLAIM APPENDIX

Claim 1:

Claim 1[Pre]: A computer implemented method of identifying unauthorized activities on a decoy computer system attached to a computer network, wherein the decoy system comprises:

1(a) one or more processors; and memory storing:

1(b) a virtual machine; and a virtual machine monitor supervising the virtual machine, the method comprising, at the virtual machine monitor: monitoring activity on the virtual machine;

1(c) identifying a plurality of activities being performed at the virtual machine, wherein each of the activities includes an activity source, an activity target, and an association between the activity source and the activity target;

1(d) storing in the memory the activity sources, activity targets, and associations;

1(e) creating, from the stored activities, a fingerprint indicative of the activity on the virtual machine; and

1(f) transmitting the fingerprint to prevent future attacks that comprise the same or similar activities as indicated by the fingerprint.

Claim 2:

The method of claim 1, wherein the monitoring further comprises monitoring all activity on the virtual machine, and all activity on the virtual machine is assumed to be unauthorized.

Claim 3:

The method of claim 1, wherein the fingerprint includes files that are affected by the unauthorized activities on the virtual machine.

Claim 4:

The method of claim 1, wherein the fingerprint includes processes that are affected by the unauthorized activities on the virtual machine.

Claim 5:

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.

Claim 6:

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.

Claim 7:

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.

Claim 8:

The method of claim 1, further comprising graphically displaying at least a subset of the stored activities.

Claim 9:

The method of claim 8, wherein the displaying includes visually distinguishing at least a subset of the associations.

Claim 10:

The method of claim 8, wherein 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.

Claim 11:

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.

Claim 12:

The method of claim 11, further comprising tracking a transfer of malicious associations over time.

Claim 13:

The method of claim 1, further comprising identifying activity sources that are affected by unauthorized activities.

Claim 14:

The method of claim 13, 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.

Claim 15:

The method of claim 13, 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.

Claim 16:

The method of claim 13, wherein the identifying activity sources that are affected by unauthorized activities includes identifying data stream incoming from outside the computer system.

Claim 17:

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.

Claim 18:

The method of claim 1, further comprising determining an activity level representing a frequency of unauthorized activities on the virtual machine during a predefined time interval.

Claim 19:

Claim 19[Pre]: A system, comprising:

19(a) one or more processors; and memory storing:

19(b) a virtual machine; a virtual machine monitor supervising the virtual machine; and

19(c) one or more programs, the one or more programs including instructions for:

19(d) monitoring activity on the virtual machine;

19(e) identifying a plurality of activities being performed at the virtual machine, where each of the activities includes an activity source, an activity target, and an association between the activity source and the activity target;

19(f) storing in the memory the activity sources, activity targets, and associations;

19(g) creating, from the stored activities, a fingerprint indicative of the activity on the virtual machine; and

19(h) transmitting the fingerprint to prevent future attacks that comprise the same or similar activities as indicated by the fingerprint.

Claim 20:

Claim 20[Pre]: A non-transitory computer readable storage medium, including one or more programs for execution by one or more processors of a computer system, the one or more programs including instructions for:

20(a) monitoring activity on the virtual machine;

20(b) identifying a plurality of activities being performed at the virtual machine, where each of the activities includes an activity source, an activity target, and an association between the activity source and the activity target;

20(c) storing in the memory the activity sources, activity targets, and associations;

20(d) creating, from the stored activities, a fingerprint indicative of the activity on the virtual machine; and

20(e) transmitting the fingerprint to prevent future attacks that comprise the same or similar activities as indicated by the fingerprint.

Claim 21:

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.

Claim 22:

The method of claim 1, further comprising: identifying a type of the association between the activity source and the activity target.

Claim 23:

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.

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 13,949 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 ’697 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