

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

AO KASPERSKY LAB
Petitioner

v.

OPEN TEXT INC.
Patent Owner

IPR2024-00292
U.S. Patent No. 8,201,243

**PETITION FOR *INTER PARTES* REVIEW
UNDER 35 U.S.C. § 312 AND 37 C.F.R. § 42.104**

TABLE OF CONTENTS

PETITIONER’S EXHIBIT LIST	iv
I. INTRODUCTION	1
II. MANDATORY NOTICES	2
A. Real Party-in-Interest	2
B. Related Matters.....	2
C. Counsel	3
D. Service Information.....	3
III. CERTIFICATION OF GROUNDS FOR STANDING	4
IV. THE ‘243 PATENT	4
A. Overview of the ‘243 patent.....	4
B. Prosecution History	6
V. LEVEL OF ORDINARY SKILL IN THE ART.....	7
VI. CLAIM CONSTRUCTION	7
VII. STATEMENT OF PRECISE RELIEF REQUESTED FOR EACH CLAIM CHALLENGED.....	9
VIII. PRIOR ART OVERVIEW	10
A. Williamson	10
B. Piccard	12
C. Kidron.....	13
IX. IDENTIFICATION OF HOW THE CLAIMS ARE UNPATENTABLE.....	14
A. Ground 1: Claims 1-7 and 9-14 are obvious under 35 U.S.C. § 103 over Williamson in view of Piccard	14
1. Reasons to Combine Williamson and Piccard	14
2. Claim 1	17
3. Claim 2	34
4. Claim 3	35
5. Claim 4	36
6. Claim 5	37

7. Claim 6	38
8. Claim 7	42
10. Claim 9	43
11. Claim 10	45
12. Claim 11	45
13. Claim 12	46
14. Claim 13	47
14. Claim 14	47
B. Ground 2: Claims 1-14 are obvious under 35 U.S.C. § 103 over Kidron in view of Williamson and Piccard.....	48
1. Reasons to Combine Kidron and Williamson.....	48
2. Reasons to Combine Kidron, Williamson, and Piccard.....	50
3. Claim 1	51
4. Claim 3	55
5. Claim 6	55
6. Claim 8	58
7. Claim 12	59
8. Claims 2, 4-5, 8-11, 13-14.....	60
X. DISCRETIONARY DENIAL WOULD BE INAPPROPRIATE.....	60
A. THE BOARD SHOULD NOT DENY INSTITUTION UNDER SECTION 314(a)	60
1. Factor 1: Whether the district court granted a stay or a stay may be granted if a proceeding is instituted.....	60
2. Factor 2: Proximity of the court’s trial date to the projected statutory deadline for a final written decision	60
3. Factor 3: Investment by the court and the parties	62
4. Factor 4: Overlap between issues raised in the petition and the parallel proceeding.....	62
5. Factor 5: Whether the petitioner and the defendant in the parallel proceeding are the same party	63

6. Factor 6: Other circumstances that impact the Board’s exercise of discretion, including the merits	63
B. THE BOARD SHOULD NOT DENY INSTITUTION UNDER GENERAL PLASTIC	64
1. Factor 1: Whether the Same Petitioner Previously Filed a Petition.....	64
2. Factor 2: Whether at the Time of Filing of the First Petition the Petitioner Knew of the Prior Art Asserted in the Second Petition or Should Have Known of It.....	66
3. Factor 3: Whether at the Time of Filing of the Second Petition the Petitioner Already Received the Patent Owner’s Preliminary Response to the First Petition	66
4. Factors 4 & 5: The Length of Time that Elapsed between the Time the Petitioner Learned of the Prior Art Asserted in the Second Petition and the Filing of the Second Petition, and Whether the Petitioner Provides Adequate Explanation	67
5. Factors 6 & 7: The Finite Resources of the Board, and Requirement to Issue a Final Determination within One Year from Institution	67
C. THE BOARD SHOULD NOT DENY INSTITUTION UNDER 35 U.S.C. § 325(d).....	68
1. The <i>Advanced Bionics</i> Framework Favors Institution.....	68
XI. CONCLUSION	69
CERTIFICATE OF COMPLIANCE UNDER 37 C.F.R. § 42.24(d).....	70
CERTIFICATE OF SERVICE	71

PETITIONER'S EXHIBIT LIST

Exhibit	Description
1001	U.S. Patent No. 8,201,243 (issued June 12, 2012)
1002	Declaration of Michael J. Donahoo, Ph.D. under 37 C.F.R. §1.68
1003	Curriculum Vitae of Michael J. Donahoo, Ph.D.
1004	File History of U.S. Patent No. 8,201,243
1005	U.S. Patent No. 8,719,924 to Williamson et al. ("Williamson")
1006	U.S. Patent Publication No. 2007/0016951 to Piccard et al. ("Piccard")
1007	U.S. Patent No. 7,934,103 to Kidron (filed March 3, 2006) ("Kidron")
1008	WALTER ONEY, PROGRAMMING THE MICROSOFT WINDOWS DRIVER MODEL, (2nd ed. 2003)
1009	Ted Hudek & Don Marshall, <i>KMDF Version History</i> , MICROSOFT (Oct. 31, 2023), https://learn.microsoft.com/en-us/windows-hardware/drivers/wdf/kmdf-version-history
1010	TROY MOTT, LEARNING CARBON, (Apple Computer, Inc. eds., 1st ed. 2001).
1011	JAMES GOSLING, FRANK YELLIN, & THE JAVA TEAM, <i>The Java™ Application Programming Interface, Volume 1: Core Packages in THE JAVA SERIES . . . FROM THE SOURCE.</i>
1012	DICTIONARY OF COMPUTING, (4th ed. 1996).
1013	Sixth Amended Scheduling Order, Open Text Inc. v. AO Kaspersky Lab, 6:22-cv-00243-ADA-DTG (U.S. Dist. Ct. W. Dist. Tx. Waco Div., Oct. 30, 2023)
1014	Claim Construction Order, Webroot, Inc., Open Text Inc. v. AO Kaspersky Lab, 6:22-cv-00243-ADA-DTG (U.S. Dist. Ct. W. Dist. Tx. Waco Div., July 25, 2023).
1015	Fifth Amended Scheduling Order, Open Text Inc. v. AO Kaspersky Lab, 6:22-cv-00243-ADA-DTG (U.S. Dist. Ct. W. Dist. Tx. Waco Div., Mar. 29, 2023)
1016	Plaintiffs' Responsive Claim Construction Brief Regarding Patents Plaintiffs Added by Amendment [263], Open Text Inc. v. AO Kaspersky Lab, 6:22-cv-00243-ADA-DTG (U.S. Dist. Ct. W. Dist. Tx. Waco Div., May 5, 2023)
1017	JAMES BONEY, CISCO IOS IN A NUTSHELL (2001)

Exhibit	Description
1018	DAVID HUCABY, CISCO ASA AND PIX FIREWALL LOGGING, provided by Cisco Press (2005) https://www.ciscopress.com/articles/article.asp?p=424447&seqNum=2
1019	Jennifer M. Anderson, et al., <u>Continuous Profiling: Where Have All the Cycles Gone?</u> , 15 ACM Transactions on Computer Systems 357 (1997)

I. INTRODUCTION

Pursuant to 35 U.S.C. §§ 311, 314(a), and 37 C.F.R. § 42.100, AO Kaspersky Lab (“Petitioner”) respectfully requests that the Board review and cancel as unpatentable claims 1-14 (the “challenged claims”) of U.S. Patent No. 8,201,243 (“‘243 patent,” Ex. 1001). The ‘243 patent describes systems and methods of identifying an origin of activity on a computer that is indicative of pestware by detecting suspected pestware through monitoring and analyzing computer activity to determine the activity is indicative of pestware, identifying a suspected pestware object based on the activity, and identifying the source of the suspected pestware through identifying an externally networked source that originated the suspected pestware object, and reporting the identity of the externally networked source to an externally networked pestware research entity. *See* Ex. 1001, Abstract, Claim 1.

The systems and methods of threat detection claimed in the ‘243 patent were already well known when the inventors filed the application for the ‘243 patent. The prior art describes origin identification of pestware with the claimed elements before the ‘243 patent priority date of April 20, 2006. Notably, Williamson—filed March 3, 2006, and not considered during prosecution—discloses methods and systems for detecting harmful software, Piccard—filed July 13, 2005, and not considered during prosecution—discloses methods and systems for identifying sources of malware,

and Kidron—filed April 15, 2003, and not considered during prosecution—describes systems and methods for detecting and countering known and unknown malicious code.

The evidence in this Petition demonstrates that the challenged claims are unpatentable over the prior art. Accordingly, the Board should institute review.

II. MANDATORY NOTICES

A. Real Party-in-Interest

The sole real party-in-interest for this Petition is AO Kaspersky Lab, Petitioner.

B. Related Matters

The Petitioner is the defendant in a case alleging infringement of the ‘243 patent styled *Webroot, Inc. et al. v. AO Kaspersky Lab* (Case No. 6:22-CV-00243-ADA-DTG), pending in the Western District of Texas, Waco Division.

The Petitioner further understands that the ‘243 patent has been asserted in the following matters, which matters have been consolidated only for pre-trial purposes with the above matter: *Webroot, Inc. et al. v. Sophos Ltd.* (Case No. 6:22-CV-00240-ADA-DTG); *Webroot, Inc. et al. v. CrowdStrike, Inc. et al.* (Case No. 6:22-CV-00241-ADA-DTG); *Webroot, Inc. et al. v. Forcepoint, LLC* (Case No. 6:22-CV-00342-ADA-DTG); and *Webroot, Inc. et al. v. Trend Micro, Inc.* (Case

No. 6:22-CV-00239-ADA-DTG) (collectively the “Parallel Litigation”). Each of the five cases will be tried separately.

Finally, the Petitioner understands that the ‘243 patent is the subject of the following IPR petitions: *Sophos, Ltd. et al., v. Open Text, Inc.* in IPR2023-00731 (“Sophos IPR,” institution denied); *CrowdStrike, Inc. v. Open Text, Inc.* in IPR2023-01052 (“CrowdStrike IPR”); and *Trend Micro, Inc. v. Webroot, Inc.* in IPR 2023-01390 (“Trend IPR”).

C. Counsel

Petitioner designates Joseph T. Miotke (Reg. No. 47,798) as lead counsel and Eric Chadwick (Reg. No. 41,664), and Erin Block (Reg. No. 78,154) as back-up counsel.

D. Service Information

Service of any documents may be made at the postal mailing address of Dewitt LLP, 901 Marquette Ave., 2100 AT&T Tower, Minneapolis, MN 55402, (Tel: (612) 305-1400 and Fax: (612) 305-1414).

Petitioner consents to electronic service by email to the email addresses:

jtm@dewittllp.com;

ehc@dewittllp.com;

eblock@dewittllp.com; and

docketing@dewittllp.com.

III. CERTIFICATION OF GROUNDS FOR STANDING

On December 9, 2022, the Western District of Texas, Waco Division Court directed Patent Owner to file its proposed second amended complaint (“SAC”) that included the ‘243 patent that Petitioner had previously opposed. On the same day, Patent Owner filed SAC and served Petitioner via the court’s electronic filing system. Petitioner certifies that the ‘243 patent is eligible for IPR, and that Petitioner is not barred or estopped from requesting IPR challenging the patent claims. 37 C.F.R. § 42.104(a).¹

IV. THE ‘243 PATENT

A. Overview of the ‘243 patent

The ‘243 patent relates to researching an identity of a source of activity that is indicative of pestware. Ex. 1001, Abstract. The ‘243 patent describes protecting against “trojans, spyware, and adware,” collectively referred to as “malware,” “spyware” or “pestware.” Ex. 1001, 1:43-45. In general, its systems and methods operate as depicted in FIG. 4 below.

¹ See *Amneal Pharmaceuticals, LLC. v. Endo Pharmaceuticals Inc.*, Case IPR2014-00360, (June 27, 2014) (Paper 15)

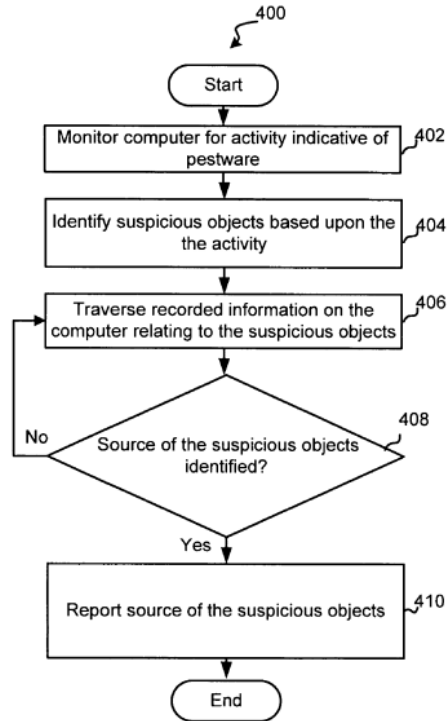


FIGURE 4

Ex. 1001, FIG. 4.

Exemplary claim 1 recites these method steps as monitoring, using a kernel-mode driver, the computer for API call activity that is indicative of pestware, and storing a history of at least a portion of the API calls in an activity log; analyzing, heuristically, computer activity to determine whether one or more weighted factors associated with an activity exceeds a threshold; identifying an object residing on the computer that is a suspected pestware object; accessing recorded history of externally networked sources that the computer received files from to identify a particular externally networked source that the suspected pestware object originated

from; and reporting the identity to an externally networked pestware research entity to research whether the particular externally networked source is a source of pestware. Ex. 1001, Claim 1.

B. Prosecution History

The '243 patent was filed as U.S. Application No. 11/408,146, (“the ’146 application”), on April 20, 2006. During prosecution, the Applicant made several amendments to the claims in response to rejections over the prior art. In particular, in response to a non-final Office Action of February 25, 2010, Applicant amended exemplary independent claim 1 to recite “externally networked” source(s). Ex. 1004, pp. 166-167.

In response to a non-final Office Action of May 9, 2011, Applicant amended claim 1 to add “reporting the identity of the particular externally networked source to an externally networked pestware research entity so as to enable the externally networked pestware research entity to research whether the particular externally networked source is a source of pestware.” Ex. 1004, p. 70.

In a Notice of Allowability of February 9, 2012, the Examiner amended claim 1 with Applicant’s authorization, including “monitoring, using a kernel-mode driver, the computer for activity that is indicative of pestware, wherein the monitoring includes monitoring API calls and storing a history of at least a portion of the API

calls in an activity log; analyzing, heuristically, computer activity to determine whether one or more weighted factors associated with an activity exceeds a threshold so as to arrive at a determination that the activity is indicative of pestware;” and “accessing, in response to the identifying an object.” Ex. 1004, p. 47.

Williamson and Piccard, and Kidron, Williamson, and Piccard expressly disclose the allegedly patentable limitations as cited above.

V. LEVEL OF ORDINARY SKILL IN THE ART

A Person of Ordinary Skill in The Art (“POSITA”) in April 2006 (the earliest alleged priority date) would hold a bachelor’s degree in computer science or a similar field, have about 2 years of experience as a software engineer or similar title, and have about 2 years of experience specifically in the computer security field. A person having more education, such as a master’s degree, would require less experience, and a person not having at least a bachelor’s degree would need more experience. Ex. 1002, ¶¶20-22.

VI. 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); *Phillips v. AWH Corp.*, 415 F.3d 1303 (Fed. Cir. 2005) (en banc).

In the Parallel Litigation, PO argued the contested terms should be given plain and ordinary meaning. The Court construed all contested terms as plain and ordinary meaning except for “pestware,” which was construed to mean “a trojan, spyware, adware, or malware program that collects information about a person or an organization.” Ex. 1014, p. 3.

For purposes of this proceeding only, Petitioner proposes the Board adopt plain and ordinary meaning as proposed by Webroot and adopted by the Western District of Texas court for terms other than “pestware” and adopt the court’s construction of “pestware.” See *Western Digital Corporation v. Spex Technologies, Inc.*, IPR2018 00084, Paper 14, *11 (PTAB Apr. 25, 2018) (finding Petition based on claim constructions urged by Patent Owner satisfies the claim construction requirements and Petitioner is not required to express its subjective agreement regarding correctness of the proffered claim construction or take ownership of the construction); see also *General Electric v. Vestas*, IPR2018- 00928, Paper 9, *16-17 (PTAB Nov. 5, 2018). Should the Board construe “pestware” under its plain and ordinary meaning, the arguments and citations of this Petition would equally apply to that broader interpretation.

VII. STATEMENT OF PRECISE RELIEF REQUESTED FOR EACH CLAIM CHALLENGED

Petitioner requests review under 35 U.S.C. § 311 and cancelation of claims 1-14 of the '243 patent in view of the following references and grounds:

Ground	Claim(s)	Basis
1	1-7, 9-14	§ 103 Williamson and Piccard
2	1-14	§ 103 Kidron, Williamson, and Piccard

U.S. Patent 8,719,924 to Williamson et al. (“Williamson,” Ex. 1005), which application was filed on March 3, 2006, issued May 6, 2014, is thus prior art under at least 35 U.S.C. § 102(a)(2).

U.S. Publication 2007/0016951 to Piccard et al. (“Piccard,” Ex. 1006), which application was filed on July 13, 2005, and published on January 18, 2007, is thus prior art under at least 35 U.S.C. § 102(a)(2).

U.S. Patent 7,934,103 to Kidron (“Kidron,” Ex. 1007), which application was filed on April 15, 2003, and published on April 26, 2011, is thus prior art under at least 35 U.S.C. § 102(a)(2).

Petitioner’s analysis also cites additional prior art to demonstrate the

background knowledge of a POSITA and to provide contemporaneous context to support Petitioner's assertions regarding what a POSITA would have understood from the prior art. *See Yeda Research v. Mylan Pharm. Inc.*, 906 F.3d 1031, 1041-1042 (Fed. Cir. 2018); 37 C.F.R. § 42.104(b); *see also K/S HIMPP v. Hear-Wear Techs., LLC*, 751 F.3d 1362, 1365-66 (Fed. Cir. 2014).

Each of these grounds is discussed in detail below.

VIII. PRIOR ART OVERVIEW

A. Williamson

Williamson was not cited or considered during prosecution of the '243 patent. Like the '243 patent, Williamson recognizes the deficiencies in current technology, particularly in uniquely identifying malicious code. Ex. 1005, 1:16-36, 2:1-8. Williamson relates to detecting harmful software and malicious computer programs by creating and using classifiers to determine whether the program is acting maliciously. Ex. 1005, 1:49-57. An exemplary system that detects various classes of malware is shown in FIG. 2 below:

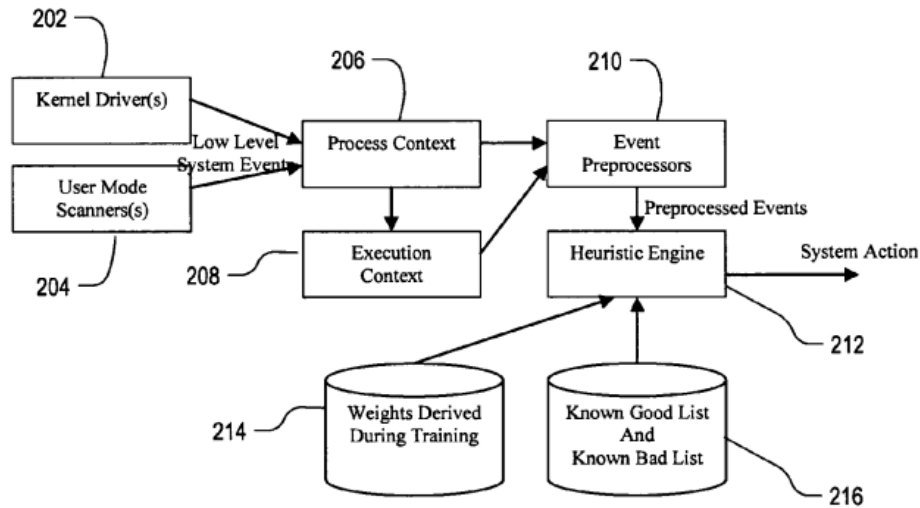


FIG. 2

Ex. 1005, FIG. 2. Williamson describes a “Kernel Drive and User Mode Scanners that hook various API calls.” Ex. 1005, 4:65-67. Low level system events associated with API calls are evaluated by a heuristic engine using a classifier based on weights generated during training, and the process associated with the event is determined to be either a malware or not based on the result. Ex. 1005, 5:5-33.

Williamson, like the ‘243 patent, describes detecting, controlling, and removing security threats to a computer. Ex. 1001, 1:35-39; Ex. 1005, 1:49-53. Williamson is in the same field of endeavor as the ’243 patent. Ex. 1002, ¶¶87-88. Williamson also describes solutions to the same problems as the ‘243 patent (e.g., using definitions to detect malware). Ex. 1001, Abstract, 1:62-2:3 (pestware

definitions are slow and cumbersome to create); Ex. 1005, 1:29-37, 5:34-35, 2:1-3 (“instead of trying to uniquely identify the malicious code, the idea is to detect and constrain the behavior of the malicious program”). Ex. 1002, ¶¶89-91.

B. Piccard

Piccard was not cited or considered during prosecution of the ‘243 patent. Piccard describes systems and methods for managing malware through the collection of information from a history log on the protected computer. Ex. 1006, ¶¶ [0005]-[0007]. “If the malware detection module 118 determines that the protected computer 102 includes malware, the history log module 120 collects information from one or more history logs maintained by the application programs.” Ex. 1006, ¶ [0017]. After identifying malware, the “malware removal module 124 operates to remove the malware on the protected computer 102.” Ex. 1006, ¶ [0019].

Piccard, like the ‘243 patent, describes detecting, controlling, and removing security threats to a computer. Cf. Ex. 1001, 1:35-39, and Ex. 1006, ¶¶[0005]-[0007]. Piccard is in the same field of endeavor as the ’243 patent. Ex. 1002, ¶¶92-94. Piccard also describes solutions to the same problems as the ‘243 patent (e.g., using definitions to detect malware). Cf. Ex. 1001, Abstract, 1:62-2:3 (pestware definitions are slow and cumbersome to create), and Ex. 1006, ¶[0003] (“it is often difficult to initially locate malware in order to generate definitions, particularly since malware

can evolve. It would be desirable to identify sources of malware, such that definitions can be rapidly generated or updated to account for new or evolving malware”). Ex. 1002, ¶¶95-97.

C. Kidron

Kidron was not cited or considered during prosecution of the ‘243 patent. Kidron describes systems and methods for detecting and countering known and unknown malicious code. Ex. 1007, 4:16-18. Like the ‘243 patent, Kidron recognizes the deficiencies in current technology, particularly in uniquely identifying malicious code. Ex. 1007, 2:15-19 (signature evading), 2:34-44 (realignment or reconfiguration for new threats). To overcome these deficiencies, Kidron discloses systems and methods that are not reliant on signatures. Rather, local operations are monitored to detect irregular local behavior patterns and generate an alert after an irregularity in local behavior pattern is detected. If similar alerts are received from a threshold number of local machines, a server selects a countermeasure operation. Ex. 1007, 3:25-40.

Kidron, like the ‘243 patent, describes detecting, controlling, and removing security threats to a computer. Cf. Ex. 1001, 1:35-39, and Ex. 1007, 3:11, 3:25-40. Kidron is therefore in the same field of endeavor as the ‘243 patent. Ex. 1002, ¶¶98-99. Kidron also describes solutions to the same problems as the ‘243 patent (e.g.,

using definitions to detect malware). Cf. Ex. 1001, Abstract, 1:62-2:3 (pestware definitions are slow and cumbersome to create), and Ex. 1007, 2:50-56 (virus signatures files and/or fixed security policy rules must be updated before new malicious codes can be detected). Ex. 1002, ¶¶100-102.

IX. IDENTIFICATION OF HOW THE CLAIMS ARE UNPATENTABLE

A. Ground 1: Claims 1-7 and 9-14 are obvious under 35 U.S.C. § 103 over Williamson in view of Piccard

1. Reasons to Combine Williamson and Piccard

A POSITA would have been motivated to combine the teachings of Williamson and Piccard for multiple reasons.

Williamson's fundamental "idea is to detect and constrain the behavior of the running program. The behavior based mechanism runs the malicious program, and from its running behavior determines whether the program is behaving maliciously" endangers the computer system by allowing execution of malicious operations of the malicious program. Ex. 1005, 2:2-6. Ex. 1002, ¶103. A POSITA would have been motivated to improve Williamson and prevent malicious operations from sources of suspected malware from executing in the first place and thus avoiding harm to the computer system (e.g., Williamson's described damaging self-propagation, allowing unauthorized access, recording information, recording user actions). Ex. 1005, 1:62-

67; Ex. 1002, ¶104. Piccard explicitly solves the problem of the risk to the system of Williamson by detecting malware prior to execution, thereby reducing the risk to the computer system. Ex. 1002, ¶¶105-107.

A POSITA would have readily understood another deficiency described by Williamson. Williamson describes “as the types of programs written evolve, changes would be needed to the characteristics and classifier.” Ex. 1005, 5:38-40. “FIG. 1 shows that a malware database 101, constructed from the ongoing collection of new malware 102, is used to train the system 103.” Ex. 1005, 4:62-64. In particular, the heuristic engine 212 needs to be updated with identified new known malware in order to update its classifier. A POSITA would have understood that more updates to training data would have improved the classification ability of Williamson. Ex. 1005, 6:35-36 (“[a]s a rule of thumb, more training and more diverse training is better.”). Ex. 1002, ¶108.

Piccard explicitly describes a solution to the training and update problem raised by Williamson. Piccard improves Williamson’s malware detection through continuous improvement of malware detection. Ex. 1006, ¶ [0028]. In particular, Piccard improves “the efficiency at which sources of malware can be identified.” Ex. 1006, ¶ [0020]; Ex. 1002, ¶¶109-111 (explaining that “new or updated set of

malware definitions can be provided” to the malware detection systems through a decentralized approach).

In this modified system, Williamson’s heuristic engine would incorporate Piccard’s malware detection module (as coupled to a database as needed), history log module, and reporting module. Williamson discloses that its heuristics engine analysis can evolve (the list of characteristics and the classification mechanisms are extensible, and a mechanism is provided whereby new characteristics and classification types can be added to the system as and when needed), such as those readily incorporated to identify sources suspected of malware, as in Piccard. Ex. 1005, 5:39-43; Ex. 1002, ¶112. Williamson further discloses that one of the characteristics of computer activity that is evaluated is whether the executable accessed the network, “either as a client accessing services on other machines or listening on a certain network port. Malicious programs need to use the network to communicate with their controllers, send out information, receive software updates etc.” Ex. 1005, 8:61-66. Therefore, Williamson already contemplates evaluation of a networked source of the malware. Ex. 1002, ¶113. Williamson’s system modified by Piccard would therefore operate to identify malware prior to execution and implement Williamson’s analysis to identify both identified sources of suspected

malware as in Piccard and unknown code using updated training on new known malware as malware. Ex. 1002, ¶114.

This combination would have improved a known system (Williamson’s system of detecting various classes of malware) with known techniques (Piccard’s updating of new sources and suspected malware), requiring only programming updates for execution on the hardware of Williamson. Ex. 1005, FIG. 3 (computer system 310); Ex. 1006, ¶ [0016] (“the malware detection module 118, the history log module 120, and the reporting module 122 operate to facilitate identification of sources of malware”); Ex. 1002, ¶115. Therefore, a POSITA would have appreciated a reasonable expectation of success making software modifications that require no more than the ordinary skill of a POSITA. Ex. 1002, ¶116.

2. Claim 1

[1.0] A method for identifying an origin of activity on a computer that is indicative of pestware comprising:

To the extent the preamble is limiting, Williamson discloses methods for identifying a source of activity on a computer as malware. Williamson describes “[v]arious embodiments address the problem of malicious code running on computers,” and in one example, a “behavior based mechanism” that “determines

whether the program is behaving maliciously. This allow[s] **unknown code to be identified as malware.**”² Ex. 1005, 1:61-62, 2:4-7; Ex. 1002, ¶117.

The ‘243 patent explains that “trojans, spyware, and adware” are “collectively referred to as ‘malware,’ ‘spyware’ or ‘pestware.’” Ex. 1001, 1:43-54. Williamson’s methods are for addressing “worms and viruses which are programs that self-propagate; **trojans** or backdoors which are programs that allow unauthorized access to machines, as well as record information (passwords etc); and **spyware** which is software to record user actions,” “some malicious programs (particularly **adware**) install themselves in [the Program Files] directory,” and “[t]rojans are used as an example of a particular **malware** in this discussion,” which are all categorized under the ‘243 patent definition of “pestware” and further under the narrower adopted construction of “pestware” as “a trojan, spyware, adware, or malware program that collects information about a person or an organization.” Ex. 1005, 1:61-67, 9:56-58, 2:28-30. In particular, Williamson’s disclosed solutions to address “record[ing] information (passwords etc)” and “software to record user actions,” correspond to the collection of information about a person or an organization. Ex. 1005, 1:61-67; Ex. 1002, ¶¶118-120.

² **Emphasis** in quoted material has been added. Claim terms are presented in *italics*.

As explained above in Section IX.A.1. and with respect to limitation [1.4] below, Williamson in view of Piccard also discloses the claim. It would have been obvious to modify Williamson in view of Piccard for the reasons discussed above, and in particular to identify the source of the malicious code, which is explicitly disclosed by Piccard (“directing the protected computer to convey the information to a host computer, such that the information can be used to ***identify a source of the malware.***”) Ex. 1006, ¶ [0007]; *supra*, Section IX.A.1.; *infra*, [1.4]; Ex. 1002, ¶121.

[1.1] monitoring, using a kernel-mode driver, the computer for activity that is indicative of pestware, wherein the monitoring includes monitoring API calls and storing a history of at least a portion of the API calls in an activity log;

First, Williamson utilizes “a ***Kernel Drive*** [sic] 202 and User Mode Scanners 204 that **hook various components of the operating system.** The ***kernel driver*** 202 looks at the requests made by programs running on the protected system, such as system calls, graphics, and windows calls (*monitoring the computer using a kernel-mode driver for activity that is indicative of pestware*).” Ex. 1005, 4:63-5:2; FIG. 2; Ex. 1002, ¶¶50-53, 65-73, 123-125.

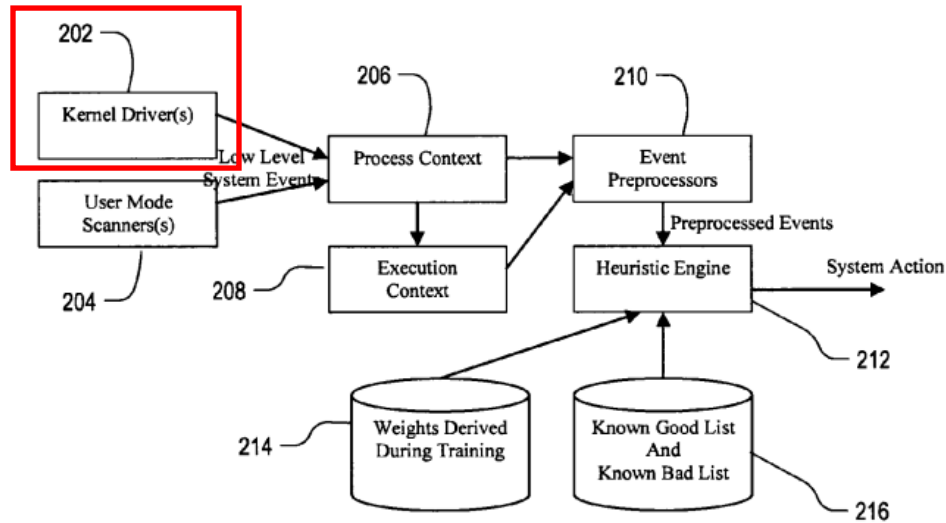


FIG. 2

Ex. 1005, FIG. 2 (annotated). The system calls, graphics, and windows calls monitored using hooks by kernel driver 202 are examples of API calls. Ex. 1002, ¶¶54-64, 126-127.

Second, Williamson discloses storing a history of at least a portion of the API calls in an activity log. When the mode scanner “detect[s] an interesting operation that generates a Low Level System event, **the events are passed to a process content 206.**” Ex. 1005, 5:5-7. “[E]xamples of low level system events include creation and setting of registry values, installation of global windows hooks, taking a snapshot of a screen and so forth.” Ex. 1005, 5:11-15; Ex. 1002, ¶¶84-85, 128-130 (explaining that API calls are examples of system events).

Williamson explains that **“events that are kept at the process context”** (*storing a history of at least a portion of the API calls in an activity log*). Ex. 1005, 5:16; Ex. 1002, ¶¶84-85, 131. Williamson further discloses storage of data constructs such as the process context. **“Storage Subsystem 324 stores the basic programming and data constructs that provide the functionality of certain embodiments of the present invention.”** Ex. 1005, 14:59-61.

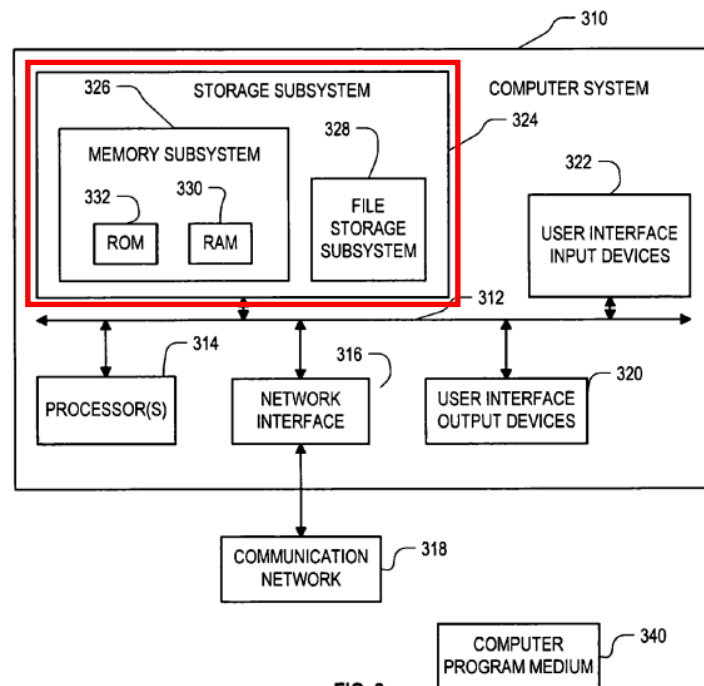


FIG. 3

Ex. 1005, FIG. 3 (annotated). A POSITA would understand that the events kept at the process context, for example, by storage subsystem 324 could be stored in volatile RAM 330 and/or file storage subsystem 328 (e.g., as described by the ‘243

patent, “file storage device includes an activity log 120.”) Ex. 1001, 3:10-11; Ex. 1002, ¶132 (explaining that events kept at the process context requires storage).

[1.2] analyzing, heuristically, computer activity to determine whether one or more weighted factors associated with an activity exceeds a threshold so as to arrive at a determination that the activity is indicative of pestware;

As discussed above, Williamson’s kernel driver 202 monitors computer activity including API calls and a corresponding Low Level System event. *Supra*, [1.1]. On every new event passed to an “event preprocessor 210 from the process context 206 or the execution context 208, the **characteristics are calculated** [*analyzing computer activity*] by the **heuristic engine 212 based on weights** generated during training 214, based on [Evaluation Equation] (see Bayes Classifier) and the process is determined to be either a malware [*so as to arrive at a determination that the activity is indicative of pestware*] or not based on the result.” Ex. 1005, 5:28-33; Ex. 1002, ¶134-135.

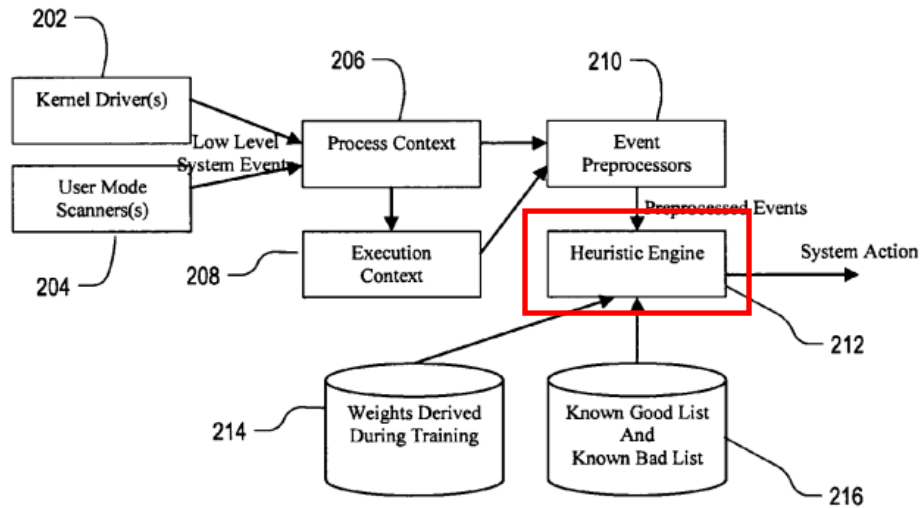


FIG. 2

Ex. 1005, FIG. 2 (annotated). Williamson’s heuristics engine 212 well-known classifier “trained based on the data and samples as shown in FIG. 1 . . . calculates the probability that a class of programs (e.g. trojans) will exhibit a particular characteristic (e.g. have a visible window).” Ex. 1005, 5:45-46, 55-58; Ex. 1002, ¶¶78-83, 136.

Williamson identifies “the key premise . . . that most of malware of the same type exhibit similar behaviors. These **behaviors are categorized and detectors for them are created and weights are automatically assigned** to through a learning process.” Ex. 1005, 2:12-16. In one specific example, Williamson describes “two extra tuning parameters that we have introduced. Some embodiments have one, both,

or neither. One is **a weight on each characteristic**, that allows the importance of each characteristic to be varied.” Ex. 1005, 6:40-43. “For example, **some of the characteristics are based on which directory in the file system the malicious code runs from** [*weighted factors associated with an activity*]. . . . A particular directory may be commonly used by different forms of malware, thus resulting in the classifier learning that executing from that directory is a good indication of maliciousness.” Ex. 1005, 6:50-52; Ex. 1002, ¶137.

Williamson describes evaluation of multiple weighted factors against a threshold. “[I]ndividual values of $P(x_i|c_j)$ can be estimated from the data, and for a new data point this equation can be evaluated. **If the function is >0 , then it is predicted to belong to class c_1 , else c_2** ” (*determining whether one or more weighted factors associated with an activity exceeds a threshold*). Ex. 1005, 5:47-6:22; Ex. 1002, ¶¶138-140. Williamson also teaches evaluation of a single weighted factor against a threshold. For example, “**determining an updated probability** that the software application is malicious **based on [a] behavior**, and in response to the **updated probability exceeding a threshold value** (*determining whether one or more weighted factors associated with an activity exceeds a threshold*) . . . identifying the software application as harmful.” Ex. 1005, Claim 1; Ex. 1002, ¶¶141-142. A POSITA would have understood that Williamson’s updated

probability based on a behavior in claim 1 refers to evaluating different aspects of program behavior (computer activity) using a weighted characteristic (equating “aspects of the program behavior” being measured to “characteristics”). Ex. 1005, 2:9-10, Ex. 1005, 3:27-45 (Examples of these characteristics include: 1) Surviving a reboot, 2) Stealth, 3) Disguised executable, 4) Network, 5) Keylogger, 6) Installation actions, 7) Injecting code into other processes), Ex. 1005, 2:30-32; (“FIG. 1 describes an exemplary overall process for updating and heuristics and probabilities of the behavioral approach.”); Ex. 1002, ¶143.

To the extent PO argues that Williamson’s described updated probability exceeding a threshold value of claim 1 is in a separate embodiment from the heuristic engine 212 embodiment, it would have been obvious to modify the heuristic engine 212 to include evaluation against thresholds for individual activities. Ex. 1002, ¶144.

Williamson explains that “[o]ne of the key premises behind this technology is that most of malware of the same type exhibit similar behaviors.” Ex. 1005, 3:12-14. Williamson further explains particular emphasis on individual characteristics of a behavior, such as “ α_i allows the ‘strength’ of each characteristic to be independently varied.” Ex. 1005, 6:49-50. A POSITA would have recognized that evaluating a fundamental characteristic of malware against a threshold for that behavior characteristic, as in claim 1 (e.g., “each of the plurality of behaviors defines

a specific action that the software application performs”), would advantageously speed up malware detection. Ex. 1002, ¶¶145.

This would address a concern described in Williamson that “malicious code spreads quickly.” Ex. 1005, 1:30-31. A POSITA would have recognized by the effective date of the ‘243 patent that timely and efficient detection of malware is critical to any malware detection system. In particular, by evaluating a fundamental action of malware behavior against a threshold for malware, detection speed and efficiency is improved. Ex. 1002, ¶¶145-146.

A POSITA would have had a reasonable expectation of success in making this modification. Williamson explains both combined characteristics evaluation against a probability threshold and individual characteristic evaluation against a probability threshold. Ex. 1005, 3:51-59 (“combination mechanism for the characteristics . . . to calculate an overall score for each class”), 4:49-52 (“extra set of weights that vary the importance of the individual probabilistic calculations, so for example increasing the importance of the keylogger characteristic”), Claim 1 (“updated probability that the software application is malicious based on” “the specific action that the software application performs”). Thus, this is just combining already-present prior art elements according to known methods already described expressly in Williamson to

predictably yield a system for evaluating one or more factors of computer activity.
Ex. 1002, ¶147.

Thus, it would have been obvious to modify Williamson to evaluate computer activity to determine whether one or more weighted factors associated with an activity exceeds a threshold so as to arrive at a determination that the activity is indicative of pestware, as claimed. Ex. 1002, ¶148.

[1.3] identifying, based upon the activity, an object residing on the computer that is a suspected pestware object;

Williamson describes systems and methods to identify a suspected pestware object (running program or executable or process instance of an executable) associated with the monitored computer activity. Ex. 1002, ¶¶74-75, 150. As explained above, Williamson teaches monitoring requests made by programs running on the protected system. Low Level System events (any type of resource, action, or status request that a running program makes from the operating system or any of its extensions) (e.g., *based upon the activity*) applied to the process context 206 “only apply to **that particular instance of a running program or executable**” (*an object residing on the computer that is a suspected pestware object*). Ex. 1005, 5:5-20. Ex. 1002, ¶151. Heuristic engine 212 evaluates corresponding characteristics of an event using a classifier and the executable or “**the process is determined** [*identifying*] to be either a malware or not based on the result.” Ex. 1005, 5:30-33.

The program/executable or process instance(s) of an executable are necessarily *identified* as a particular instance, and that the determined “executable” or “process” is an “object” as described by the ‘243 patent and consistent with the prosecution history. Ex. 1002, ¶¶151-152; Ex. 1001, 8:14-15, 8:27-28 (“suspicious objects (e.g., files and/or processes)”); Ex. 1004, p. 86, p. 139 (“unwanted software application” discussed as the claimed “object”).

Williamson provides numerous examples of evaluation of suspicious objects *residing on the computer*. “Various embodiments address the problem of **malicious code running on computers**.” Ex. 1005, 1:61-62. Explaining characteristics to evaluate, “[t]he above four characteristics [EXEC_FROM_CACHE, EXEC_FROM_WINDIR, EXEC_FROM_PGM_FILES, OTHER_PATH] are a report of **where in the file system the executable resides**. . . . it provide[s] some hints about the type of executable that is running.” Ex. 1005, 12:42-54. In another example, Williamson explains “running a **software application on a computing device**, wherein the software application is **associated with a probability that the software application is harmful**.” Ex. 1005, Claim 1. “The executable has the same name as a legitimate executable. This is evidence of a common mechanism that trojans and other malicious code use to **hide themselves on a computer**.” Ex. 1005, 10:19-22; Ex. 1002, ¶154.

[1.4] accessing, in response to the identifying an object, at least a portion of a recorded history of externally networked sources that the computer received files from so as to identify a reference to an identity of a particular externally networked source that the suspected pestware object originated from; and

As discussed above, Williamson discloses identifying a suspected pestware object (specific instance of a running program or executable or process instance of an executable) residing on the computer, based upon monitored API call activity. *Supra*, [1.0]-[1.3]. Williamson does not explicitly disclose the claimed accessing. Piccard discloses this element. Ex. 1002, ¶¶156-157.

Piccard discloses similar methods for identifying sources of malware. Ex. 1006, Abstract. Similar to Williamson’s methods that operate based upon monitored computer activity, Piccard likewise discloses that its malware detection module can “monitor activities of the protected computer to detect the presence of the malware.” Ex. 1006, ¶ [0024] (including computer activities “related to third-party cookies” or “entries or modifications of registry files of an operating system.”); Ex. 1002, ¶158.

Piccard discloses several history logs for access by a history log module. Ex. 1006, ¶ [0014]. In one example, Piccard explains that “Web browser 114 maintains a **history log 116** [*at least a portion of a recorded history*], which serves to provide a record of Web browsing events. In particular, when a user visits a Web site, the Web browser 114 records an entry that is indicative of that Web site in the history log 116. For example, when a file is accessed and downloaded from a Web site, the

Web browser 114 can record a Web address of the file in the history log 116. A Web address typically specifies a location of a file within a Web site.” Ex. 1006, ¶ [0014]. “It is also contemplated that the application programs 112 can maintain **a common history log to provide a record of events** for all of the application programs 112.” Ex. 1006, ¶ [0014]; Ex. 1002, ¶¶159-160.

Piccard further discloses “if **malware detection module 118 determines that the protected computer 102 includes malware**, the **history log module 120 collects information from one or more history logs** maintained by the application programs 112” (*in response to the identifying an object, accessing at least a portion of a recorded history*). Ex. 1006, ¶ [0017]; Ex. 1002, ¶159. “In particular, the history log module accesses the Web browser's history log to identify the *n* most recently **recorded Web addresses in the Web browser's history log** [*recorded history of externally networked sources that the computer received files from*] Thus, by collecting the set of Web addresses from the Web browser's history log, the history log module can facilitate **identification of the set of Web sites from which the malware may have been downloaded, such as in terms of domain names of the set of Web sites**” (*identify a reference to an identity of a particular externally networked source that the suspected pestware object originated from*). Ex. 1006, ¶ [0026]; Ex. 1002, ¶¶76-77, 161-162.

With reference to the claimed *externally networked source*, during prosecution, the applicant explained an “externally networked” entity means “any entity external to the computer.” Ex. 1004, p. 71. Consistent with PO’s definition, Piccard explains and depicts, for example, in annotated FIG. 1 below, “the Web browser 114 is operated by a user who visits various Web sites that are **included in the computer network 104**. For example, the user can access and download various files from those Web sites.” Ex. 1006, ¶ [0013]. The web sites recorded in the Web browser’s history log are thus external to the protected computer 102. Ex. 1006, ¶ [0013]; Ex. 1002, ¶163.

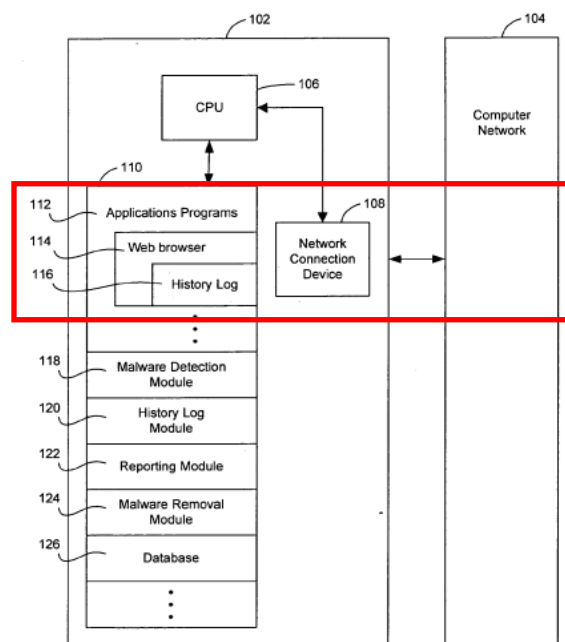


FIG. 1

Ex. 1006, FIG. 1 (annotated).

In another example, with reference to “additional history logs can be maintained to provide a record of other types of events, such as events related to operation of an e-mail program,” (*at least a portion of a recorded history*) “other sources of malware that can be identified include **sources that are external to the protected computer 102**, [*externally networked sources*] such as a **sender of an e-mail that includes the malware or an external database from which the malware was accessed or downloaded**” (*a reference to an identity of a particular externally networked source that the suspected pestware object originated from*). Ex. 1006, ¶¶ [0014], [0031]; Ex. 1002, ¶164.

As discussed above, a POSITA would have been motivated to incorporate Piccard’s identification of a suspected source of malware functionality into Williamson’s heuristic engine to identify the source of the files associated with suspected malware. Ex. 1002, ¶165; *supra*, Section IX.A.1.

[1.5] reporting the identity of the particular externally networked source to an externally networked pestware research entity so as to enable the externally networked pestware research entity to research whether the particular externally networked source is a source of pestware.

Piccard discloses identifying an identity of a particular externally networked source of a suspected pestware object using its history log module. *Supra*, [1.4]. Piccard further discloses “[o]nce the history log module 120 collects the information from one or more history logs, the **reporting module 122 then reports this**

information [*reporting the identity of the particular externally networked source*] to a **remotely located host computer** that is included in the computer network 104” (*externally networked pestware research entity*). Ex. 1006, ¶ [0018]; Ex. 1002, ¶167. With reference to the claimed *externally networked pestware research entity*, Piccard explicitly describes the host computer is “remotely located” and “included in the computer network,” thus “external to the computer.” Ex. 1006, ¶¶ [0018], [0028]; *supra*, [1.4]; Ex. 1002, ¶168.

Piccard explains, “[f]or example, the reporting module 122 can report the n most recently visited Web sites [*identity of a particular externally networked source of a suspected pestware object*] to the host computer, and **the host computer** or a user at the host computer **can evaluate those Web sites to determine whether any of those Web sites is, in fact, a malware distribution site**” (*enable the externally networked pestware research entity to research whether the particular externally networked source is a source of pestware*). Ex. 1006, ¶ [0018]. Piccard teaches research including “targeted evaluation of Web sites that are being visited by users and that may be distributing malware.” Ex. 1006, ¶ [0020]; Ex. 1002, ¶169.

As discussed above, a POSITA would have been motivated to incorporate Piccard’s reporting module into Williamson’s heuristic engine to enable a pestware

research entity to research if an externally networked source is a source of pestware.

Ex. 1002, ¶170; *supra*, Section IX.A.1.

3. Claim 2

[2.0] The method of claim 1,

See [1.0]-[1.5]. Ex. 1002, ¶171.

[2.1] wherein the particular externally networked source is identified by an identifier selected from the group consisting of an I.P address and a URL.

As explained above, Williamson in view of Piccard discloses identification of *a reference to an identity of a particular externally networked source*. For example, “by collecting the set of Web addresses from the Web browser's history log, the history log module can facilitate **identification of the set of Web sites** from which the malware may have been downloaded.” Ex, 1006, ¶ [0026]; *Supra*, [1.4]; Ex. 1002, ¶172. Piccard further explains that a “[w]eb browser 114 records an entry that is indicative of that Web site in the history log 116. For example, when a file is accessed and downloaded from a Web site [*particular externally networked source*], the Web browser 114 can record a Web address of the file in the history log 116. A Web address typically specifies a location of a file within a Web site. For example, a Web address can be a **Uniform Resource Identifier (‘URI’) of a file**, such as a **Uniform Resource Locator (‘URL’) of the file**. It is also contemplated that a Web address can be defined in various other ways, such as using an **Internet Protocol**

(‘IP’) address or any other identifier of a source of a file.” Ex. 1006, ¶ [0014]. “[B]y collecting the set of Web addresses from the Web browser's history log, the history log module can facilitate identification of the set of Web sites from which the malware may have been downloaded.” Ex. 1006, ¶[0026]; Ex. 1002, ¶173.

In the Parallel Litigation, PO asserts that this Markush group is unbounded, such that “unrecited group members” may fall within the scope of this claim. Ex. 1016, pp. 25-28. Petitioner applies PO’s construction with respect to this group in light of the group recited in Claim 1 of associated U.S. Patent No. 8,719,932, which uses substantially similar language. To the extent the Board agrees that unrecited group members fall within the scope of this claim, Piccard explicitly describes that “any other identifier of a source of file” can be used, which applies to such construction. Ex. 1006, ¶ [0014]; Ex. 1002, ¶174.

4. Claim 3

[3.0] The method of claim 2

See [1.0]-[1.5] and [2.0]-[2.1]. Ex. 1002, ¶175.

[3.1] wherein the identifying includes accessing an activity log that includes information that relates the activity to the suspected pestware object.

Williamson discloses identifying a process as a suspected pestware object based on evaluation by heuristic engine 212. *Supra*, [1.2]. Ex. 1002, ¶176.

Specifically, Williamson explains that “most of malware of the same type exhibit similar behaviors. These **behaviors are categorized and detectors for them are created and weights are automatically assigned to through a learning process.**” Ex. 1005, 2:14-16. Williamson’s monitoring of after-effects of system operations cause low-level system events to be generated. In turn, the events are passed to a process context. Ex. 1005, 5:2-14. **Events that are kept at the process context** (*activity log*) only apply to that particular instance of a running program or executable. Ex. 1005, 5:16-18. “[O]n every new event passed [*accessing*] to the event preprocessor 210 from the process context 206 or the execution context 208, **the characteristics are calculated by the heuristic engine 212 based on weights generated during training . . . and the process is determined to be either a malware or not based on the result.**” Ex. 1005, 5:28-33. Heuristic engine 212 therefore evaluates events from the process context for characteristics associated with particular behaviors of particular instance of a running program or executable as malware/pestware (*includes information that relates the activity to the suspected pestware object*). Ex. 1002, ¶¶176-178.

5. Claim 4

[4.0] The method of claim 1,

See [1.0]-[1.5]. Ex. 1002, ¶179.

[4.1] wherein the recorded history resides in a browser cache, which includes information about files downloaded to the computer and a source of each of the files.

Williamson in view of Piccard teaches a recorded history. For example, Piccard explains “Web browser 114 maintains a **history log 116** (*recorded history*), which serves to provide a record of Web browsing events. In particular, when a user visits a Web site, the Web browser 114 records **an entry that is indicative of that Web site in the history log 116.**” Ex. 1006, ¶ [0014]. “[A] Web address can be a URL of a file that is downloaded from a Web site. For example, a Web address can have the following format: `http://www.DomainName.com/Subdirectory/FileName.html`, where ‘`http://`’ specifies a communication protocol used to download a file, ‘**www.DomainName**’ specifies a domain name of a Web site from which the file was downloaded, [*a source of each of the files*] ‘`/Subdirectory/`’ specifies a subdirectory within the Web site from which the file was downloaded, and ‘**FileName.html**’ specifies a name of the file” (*information about files downloaded to the computer*). Ex. 1006, ¶ [0026]; Ex. 1002, ¶¶181-182.

A POSITA would have understood that Williamson’s system as modified by Piccard utilizes a history log in browser cache as claimed. Ex. 1002, ¶183.

6. Claim 5

[5.0] The method of claim 1

See [1.0]-[1.5]. Ex. 1002, ¶184.

[5.1] wherein the recorded history resides in at least one log selected from the group consisting of an activity log, a browser history, browser cache, browser settings, operating system settings, an event log, a debugging log, a firewall log, file information and monitoring software logs.

Piccard explains “each of the application programs 112 maintains a separate **history log**, which serves to provide a **record of events** [*event log*] related to operation of that application program.” Ex. 1006, ¶ [0014]; Ex. 1002, ¶185. In a specific example, Piccard explains, “Web browser 114 maintains a **history log 116**, which serves to provide a **record of Web browsing events**” (*browser history*). Ex. 1006, ¶ [0014]. A POSITA would have understood that the example records of events are stored in a log selected from at least an event log, a browser history, and a browser cache. Ex. 1002, ¶86.

7. Claim 6

[6.0] A system for identifying a source of activity on a computer that is indicative of pestware including:

To the extent the preamble is limiting, Williamson discloses **systems** and methods **for identifying sources of malware**. A system includes a malware detection module configured to determine that a **protected computer includes malware**. Ex. 1005, Abstract, 14:18-21 (“block diagram of a computer system 310 suitable for use with embodiments of the present invention. [c]omputer system 310 typically includes at least one processor 314”); Ex. 1002, ¶¶187-188.; *supra*, [1.0].

[6.1] an activity monitor configured to monitor API calls and to store a history of at least a portion of the API calls in an activity log, wherein the activity monitor includes a kernel-mode driver adapted to intercept the API calls;

As discussed above with respect to claim 1, Williamson describes an activity monitor as a “*Kernel Drive* [sic] 202 and User Mode Scanners 204 that **hook** [intercept] various components of the operating system. The *kernel driver* 202 looks at the requests made by programs running on the protected system, such as system calls, graphics, and windows calls” (*monitor API calls*). Ex. 1005, 4:63-5:2, FIG. 2; Ex. 1002, ¶190; *supra*, [1.1].

Williamson explains that low level system events of the API calls are “kept at the process context” (*store a history of at least a portion of the API calls in an activity log*), and further that data constructs (such as the process context) are stored by storage subsystem 324. Ex. 1005, 5:16, 14:59-61, FIG. 3; Ex. 1002, ¶191.; *supra*, [1.1].

[6.2] a heuristics module configured to identify an activity on the computer that is indicative of pestware residing on the computer and to analyze the activity to determine whether one or more weighted factors associated with the activity exceeds a threshold; and

Williamson describes that for every new event for computer activity initiated by an API call “**characteristics are calculated** [*analyze the activity*] by the *heuristic engine* 212 [*heuristics module*] **based on weights** [*one or more weighted factors*] generated during training 214, based on [Evaluation Equation] (see Bayes Classifier)

and the process is determined to be either a malware [*configured to identify an activity on the computer that is indicative of pestware residing on the computer*] or not based on the result.” Ex. 1005, 5:28-33; Ex. 1002, ¶193.

Williamson further describes “**determining an updated probability** that the software application is malicious **based on [a] behavior**, and in response to the **updated probability exceeding a *threshold* value** [*determine whether one or more weighted factors associated with the activity exceeds a threshold*] . . . identifying the software application as harmful.” Ex. 1005, Claim 1; Ex. 1002, ¶194; *supra*, [1.2].

[6.3.a] a research portion configured to access, in response to a prompt from the heuristics module, a first set of recorded information on the computer that relates the activity to at least one file residing on the computer,

Piccard discloses, “the illustrated embodiment allows **automated collection** and reporting of relevant information **once malware is detected on the protected computer 102**” (*in response to a prompt from the modified heuristic engine*). Ex. 1006, ¶ [0020]. A POSITA would have understood that Williamson’s heuristic engine 212 as modified by Piccard to include the functions of the malware detection module, history log module, and reporting module, as described above, would include a research portion. Ex. 1002, ¶195 (describing a prompt consistent with the definition of prompt in the ‘243 patent “the heuristics module 110 prompts the

research module 108 to initiate research into the origin of the pestware initiating the activity”); Ex. 1001, 4:53-55.

As described above, *accessed* information includes “the history log module 120 can identify which one of the application programs 112 was used to access or download the malware” (*a first set of recorded information on the computer that relates the activity to at least one file residing on the computer*). Ex. 1006, ¶ [0017]; Ex. 1002, ¶196-197; *supra*, [1.3]-[1.4].

[6.3.b] and wherein the research portion is configured to access a second set of recorded information on the computer that relates the at least one file to an externally networked source from which the file was received; and

Piccard discloses that “the history log module 120 can collect information from one or more predetermined history logs, such as the history log 116.” Ex. 1006, ¶ [0017]. “For example, if the malware is downloaded using the Web browser 114, the history log module 120 can access the history log 116 to identify the **n most recently visited Web sites**” (*a second set of recorded information on the computer that relates the at least one file to an externally networked source from which the file was received*). Ex. 1006, ¶ [0017]. “In such manner, the history log module 120 can identify those Web sites from which the malware may have been downloaded.” Ex. 1006, ¶ [0017]; Ex. 1002, ¶198; *supra*, [1.3]-[1.4].

[6.4] a reporting portion configured to generate a report that identifies the externally networked source of the file and to report an identity of the externally networked source to an externally networked pestware research entity so as to enable the externally networked pestware research entity to research whether the externally networked source is a source of pestware.

Piccard discloses a “**reporting module 122**” (*reporting portion*) that “**reports** this information to a remotely-located host computer [*externally networked pestware research entity*] that is included in the computer network 104.” Ex. 1006, ¶ [0018]. “For example, the reporting module 122 can **report the n most recently visited Web sites** [*identity of the externally networked source*] to the host computer.” Ex. 1006, ¶ [0018]. “This information as well as any additional relevant information can be analyzed at the host computer to identify whether the externally networked source is a source of pestware” (*research whether the externally networked source is a source of pestware*). Ex. 1002, ¶¶200-201.; Ex. 1006, ¶ [0018]; *supra*, [1.5].

8. Claim 7

[7.0] The system of claim 6,

See [6.0]-[6.4]. Ex. 1002, ¶202.

[7.1] wherein activity monitor is configured to store an identity of each process that made each of the API calls in the stored history, and wherein the activity monitor is configured to store an identity of each file that corresponds to each the processes that made the API calls in the stored history so as to create a relation between each API call and at least one file.

As discussed above, Williamson describes an activity monitor as a kernel-mode driver and user mode scanners. The kernel-mode driver “looks at the requests

made by programs running on the protected system, such as system calls, graphics, and windows calls.” Ex. 1005, 4:65-5:2. Williamson further explains “[w]hen these components [e.g. kernel driver] detect an interesting **operation that generates a Low Level System event**, the events are passed to a process content 206.” Ex. 1005, 5:5-7; Ex. 1002, ¶203.

“The events that are kept at the process context [*stored history*] **only apply to that particular instance** [*identity of each process that corresponds to each [of] the processes*] **of a running program or executable**” (*identity of each file*). Ex. 1005, 5:16-18; Ex. 1002, ¶204 (explaining that Williamson monitors and stores events with respect to the processes and to executables corresponding to the processes, and to do so, stores identities of each process and each corresponding file, *a relation between each API call and at least one file*).

10. Claim 9

[9.0] The system of claim 6,

See [6.0]-[6.4]. Ex. 1002, ¶205.

[9.1] wherein the first set of recorded information and the second set of recorded information reside in the same log file on a file storage device of the computer.

As discussed above, Williamson’s modified system discloses or renders obvious first and second sets of recorded information. *Supra*, [6.3.a]-[6.3.b]. Piccard

further teaches that the first and second sets are recorded in the same history log file 116 on a file storage device of the computer. Ex. 1002, ¶206.

In particular, Piccard teaches “[i]dentification of which one of the application programs 112 was used to access or download the malware (*first set*) can be based on . . . **information from one or more predetermined history logs, such as the history log 116.**” Ex. 1006, ¶ [0017]. Further, “the history log module 120 can access **the history log 116 to identify the n most recently visited Web sites**” (*second set*). Ex. 1006, ¶ [0017]. As annotated below, FIG. 1 of Piccard illustrates history log 116 being stored on “memory 110” of the “protected computer 102.”

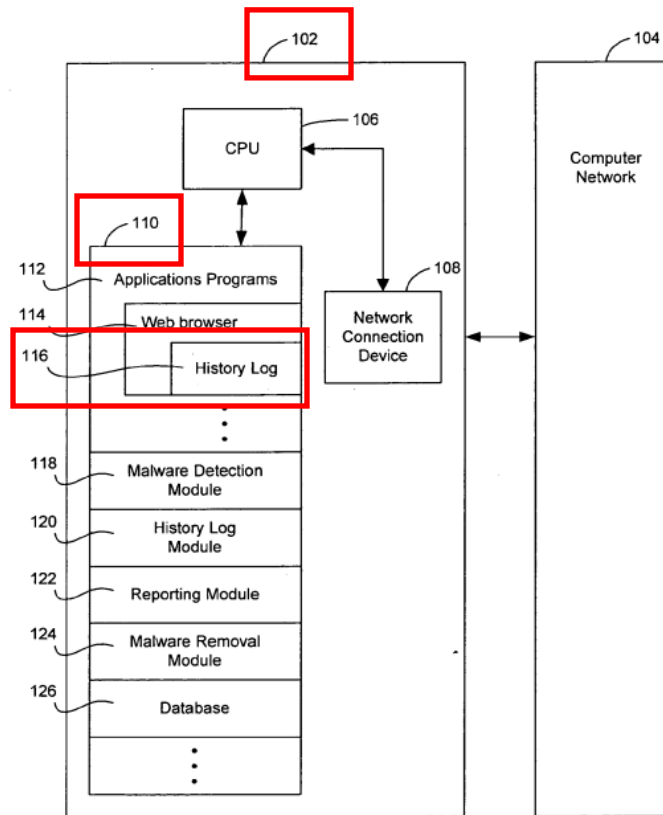


FIG. 1

Ex. 1006, FIG. 1; ¶¶ [0017], [0012]; Ex. 1002, ¶207.

11. Claim 10

[10.0] The system of claim 6,

See [6.0]-[6.4]. Ex. 1002, ¶208.

[10.1] wherein the externally networked source is identified by an identifier selected from the group consisting of an I.P. address and a URL.

See [2.1]. Ex. 1002, ¶209.

12. Claim 11

[11.0] The system of claim 6

See [6.0]-[6.4]. Ex. 1002, ¶210.

[11.1] wherein the recorded history resides in at least one log selected from the group consisting of an activity log, a browser history, browser cache, browser settings, operating system settings, an event log, a debugging log, a firewall log, file information and monitoring software logs.

See [5.1]. Ex. 1002, ¶211.

13. Claim 12

[12.0] A non-transitory computer-readable medium including processor-executable instructions for identifying an origin of activity on a computer that is indicative of pestware, the instructions including instructions for:

Williamson in view of Piccard discloses a method of identifying an origin of activity on a computer that is indicative of pestware. *Supra*, [1.0]. To the extent the preamble is limiting, Williamson teaches instructions for the method of identifying stored on a non-transitory computer-readable medium. Ex. 1005, 14:66-15:3, 15:15-17; Ex. 1002, ¶¶212-213.

[12.1] monitoring, with a kernel-mode driver, the computer for activity that is indicative of pestware, wherein the instructions for monitoring include instructions for monitoring API calls and storing a history of at least a portion of the API calls in an activity log;

See [1.1]. Ex. 1002, ¶214.

[12.2] analyzing, heuristically, computer activity to determine whether one or more weighted factors associated with an activity exceeds a threshold so as to arrive at a determination that the activity is indicative of pestware;

See [1.2]. Ex. 1002, ¶215.

[12.3] identifying, based upon the activity, an object residing on the computer that is a suspected pestware object;

See [1.3]. Ex. 1002, ¶216.

[12.4] accessing, in response to the identifying an object, at least a portion of a recorded history of externally networked sources that the computer received files from so as to identify a reference to an identity of a particular externally networked source that the suspected pestware object originated from; and

See [1.4]. Ex. 1002, ¶217.

[12.5] reporting the identity of the particular externally networked source to an externally networked pestware research entity so as to enable the externally networked pestware research entity to research whether the particular externally networked source is a source of pestware.

See [1.5]. Ex. 1002, ¶218.

14. Claim 13

[13.0] The non-transitory computer-readable medium of claim 12,

See [12.0]-[12.5]. Ex. 1002, ¶219.

[13.1] wherein the particular externally networked source is identified by an identifier selected from the group consisting of an I.P. address and a URL.

See [2.1]. Ex. 1002, ¶220.

14. Claim 14

[14.0] The non-transitory computer-readable medium of claim 12,

See [12.0]-[12.5]. Ex. 1002, ¶221.

[14.1] wherein the recorded history resides in at least one log selected from the group consisting of an activity log, a browser history, browser cache, browser settings, operating system settings, an event log, a debugging log, a firewall log, file information and monitoring software logs.

See [5.1]. Ex. 1002, ¶222.

B. Ground 2: Claims 1-14 are obvious under 35 U.S.C. § 103 over Kidron in view of Williamson and Piccard

1. Reasons to Combine Kidron and Williamson

A POSITA would have been motivated to combine Kidron and Williamson for several reasons.

Kidron's system of detecting and countering malicious code operates in an enterprise network system having a server and a plurality of local machines. First, local machine operations are monitored to detect irregular local behavior patterns to generate an alert from the local machine to the server. Then, if similar alerts are received from a threshold number of local machines, as determined by the server, countermeasure operations are implemented. Ex. 1007, 3:25-40. A POSITA would have understood that Kidron's system of relying on data from other local machines to implement countermeasures exposes local machines to malicious code, due to delays in those machines experiencing the same malicious behavior. Also, Kidron relies on functioning client-server communication, which puts at risk local machines that are not properly connected. Ex. 1002, ¶¶223-224. Therefore, to increase identification speed by local operation and provide greater resilience against server failure and network partition, a POSITA would have been motivated to incorporate Williamson's heuristic weighting analysis for API call activity into Kidron's similar

pestware security methods which monitors API call activity to identify and mitigate pestware. Ex. 1002, ¶225-226.

Williamson explicitly discloses improving performance based on adjustment of the “strength” of each characteristic. Ex. 1005, 6:40-7:14. A POSITA would therefore be motivated to improve Kidron’s detection performance based on weighted factors. Ex. 1002, ¶227. Kidron’s pattern recognition processor 12a would incorporate functions from Williamson’s heuristic engine 212 to perform Williamson’s weighted threshold analysis. In fact, Kidron already discloses that its pattern recognition processor may be trained based on known classifications, including heuristics. Ex. 1007, 26-47; Ex. 1002, ¶228.

The modified Kidron-Williamson system would therefore implement a weighted analysis using Kidron’s pattern recognition processor trained based on Williamson’s weighted factors. This combination would have improved a known system (Kidron’s system of detecting malicious code) with known techniques (Williamson’s heuristic engine implementing weights as tuning parameters), requiring only programming updates for execution on the hardware of Kidron. Ex. 1007, FIGS. 1A-1B (local machine); Ex. 1002, ¶229. Therefore, a POSITA would have appreciated a reasonable expectation of success making software modifications that require no more than the ordinary skill of a POSITA. Ex. 1002, ¶230.

2. Reasons to Combine Kidron, Williamson, and Piccard

A POSITA would have found it obvious to include Piccard's accessing of recorded history to identify an identity of a particular externally networked source, and subsequent reporting of the entity to a pestware research entity to identify a source of the malware with the combination of Kidron and Williamson. Ex. 1002, ¶¶231-232.

As discussed above, Piccard describes similar systems and methods for identifying sources of malware. Ex. 1006, ¶ [0001]. If Piccard's malware detection module determines the protected computer has malware, information is collected from a history log, and this information is reported to a host computer (e.g., server) to identify a source of the malware. Ex. 1006, ¶¶ [0017]-[0018]; Ex. 1002, ¶233.

One of the goals of the modified Kidron-Williamson system is to take appropriate countermeasures to protect a local machine, including "pinpoint the exact flow of the malicious code through the network" or "shutting down a certain TCP/IP port" in the enterprise architecture of local machines coupled to a server machine. Ex. 1007, 7:50-55; Ex. 1002, ¶234. A POSITA would have recognized the advantages of identifying sources of malware of Piccard in a similar enterprise environment. Ex. 1002, ¶235.

Modifying Kidron-Williamson’s method with Piccard in such a manner would have had a reasonable expectation of success. A POSITA would have understood that Kidron’s remote control core 12c would have readily integrated the functionality of Piccard’s history log module 120. Such a modification would have required only programming updates for execution on the hardware of Kidron-Williamson. Ex. 1002, ¶236.

3. Claim 1

[1.0] A method for identifying an origin of activity on a computer that is indicative of pestware comprising:

To the extent the preamble is limiting, Kidron teaches “a **method of detecting** and countering malicious code in an enterprise network system” as one of its “flexible methodologies for *identifying* and countering new computer viruses and malicious code.” Ex. 1002, ¶237; Ex. 1007, 3:41-42, 3:19-21, 1:41-43 (“attack by malicious software, such as viruses, Trojans, worms and other malicious computer codes, is a continuous and increasing threat to **computers**”); *Supra*, Section IX.A.2. [1.0] (Trojans, spyware, and adware categorized under the ‘243 patent definition of “pestware”). Kidron further discloses identification of the source (*origin of activity*) of suspicious operating system calls. Ex. 1007, 6:48-61; Ex. 1002, ¶238.

[1.1] monitoring, using a kernel-mode driver, the computer for activity that is indicative of pestware, wherein the monitoring includes monitoring API calls and storing a history of at least a portion of the API calls in an activity log;

Kidron discloses a “signal monitor module *monitors* and *logs* operating system call *activity* [*storing a history of at least a portion of the API calls in an activity log*], innocent or malicious” (*indicative of pestware*). Ex. 1007, 6:22-23, 6:18-19 (operating system calls including file input-output, network input-output, memory management); Ex. 1002, ¶50-73, 84-85, 239-240 (explaining such operating system calls are examples of API calls).

Kidron further teaches “signal monitor 12b may be provided **for hooking into the operating system on the local machine**” and “may be closely fused into the operating system, for example, as an **extension to the operating system kernel**” (*using a kernel-mode driver*). Ex. 1007, 6:15-21; Ex. 1002, ¶241.

[1.2] analyzing, heuristically, computer activity to determine whether one or more weighted factors associated with an activity exceeds a threshold so as to arrive at a determination that the activity is indicative of pestware;

Kidron discloses heuristic analysis of computer activity so as to arrive at a determination that the activity is indicative of pestware. As explained above, Kidron’s signal monitor module monitors and logs operating system call activity. Then, “pattern recognition processor 12a periodically *analyzes* the log of **operating system calls** [*computer activity*] to **detect any irregular patterns**” (*determination that the activity is indicative of pestware*). Ex. 1007, 6:24-25; Ex. 1002, ¶242.

(explaining irregular patterns compared to expected patterns as activity indicative of pestware).

For example, *heuristic analyzing*, “[m]alicious code deployment of unknown nature (that is, of which there is no known signature or behavior pattern) in an enterprise environment may be detected through pattern recognition technology (such as **neural nets, clustering techniques, decision tree techniques**, etc.).” Ex. 1007, 4:58-62. In further examples, the methodologies may include a combination of **neural net, rule-based and state analysis techniques**. Ex. 1007, 5:54-56. Also, pattern recognition processor may use “integrity checking (e.g., checksum or detecting a change to a program's file size), polymorphic code detection, interrupt monitoring (monitoring of access to restricted system resources), statistic analysis, signature and/or **heuristic scanning**.” Ex. 1007, 6:38-42; Ex. 1002, ¶¶78-83, 243.

Kidron does not explicitly disclose whether one or more weighted factors associated with an activity exceeds a threshold. Williamson discloses this element. *Supra*, Section IX.A.2. [1.2]; *supra*, Section IX.B.1.; Ex. 1002, ¶244.

[1.3] identifying, based upon the activity, an object residing on the computer that is a suspected pestware object;

Kidron’s modified method teaches pattern recognition processor analysis of computer activity. *Supra*, Section IX.B.3. [1.2]. Protective measures can then be taken in response to irregular patterns of a suspected pestware object. Ex. 1007, 6:49-

51 (transmitting **locally assessed information** (*based upon the activity*) to the server and receiving instructions from the server for remote management of the local workstation). In particular, Kidron teaches **detection of irregular behavior at a local machine** (*residing on the computer*) may trigger issuance of an alert to the network, **additional confirmation of irregular behavior may trigger interventional measures**. Ex. 1007, 5:9-12. Local protective (or interventional) security measures include “blocking any access to selected files, setting a quarantine on a **suspect file** to prevent copying of the **suspect file**” (*identifying an object that is a suspected pestware object*). Ex. 1007, 6:53-58; Ex. 1002, ¶245. (explaining that a file is an example object); Ex. 1007, 2:3-4 (“**file viruses** may copy themselves into essential system files”).

[1.4] accessing, in response to the identifying an object, at least a portion of a recorded history of externally networked sources that the computer received files from so as to identify a reference to an identity of a particular externally networked source that the suspected pestware object originated from; and

As explained in Ground 1, Piccard teaches this element. *Supra*, Section IX.A.2. [1.4]; *supra*, Section IX.B.2.; Ex. 1002, ¶246.

[1.5] reporting the identity of the particular externally networked source to an externally networked pestware research entity so as to enable the externally networked pestware research entity to research whether the particular externally networked source is a source of pestware.

As explained in Ground 1, Piccard teaches this element. *Supra*, Section IX.A.2. [1.5]; *supra*, Section IX.B.2.; Ex. 1002, ¶247.

4. Claim 3

[3.0] The method of claim 2

See Ground 2 [1.0]-[1.5], [2.0]-[2.1] and Ground 1, [2.0]-[2.1]. Ex. 1002, ¶248.

[3.1] wherein the identifying includes accessing an activity log that includes information that relates the activity to the suspected pestware object.

Kidron's modified method discloses a signal monitor that monitors and logs operating system call activity (*activity log*). *Supra*, Section IX.B.3. [1.1]. Kidron's pattern recognition processor **analyzes the log of operating system calls** (*accessing*) to detect any irregular patterns, which may **correspond to, for example, known or unknown viruses, Trojans, worms, or other malicious code** (for example, **with characteristics of memory resident, file, and/or macro viruses** (*information that relates the activity to the suspected pestware object*)). Ex. 1007, 6:23-31; Ex. 1002, ¶249.

5. Claim 6

[6.0] A system for identifying a source of activity on a computer that is indicative of pestware including:

To the extent the preamble is limiting, Kidron discloses systems for identifying sources of malware. For example, Kidron provides a **system** for detecting and countering known and unknown malicious code in an enterprise network. Ex. 1007, 3:19-21, 3:25-27 (new, more flexible methodologies for *identifying* and

countering new computer viruses and malicious code); 7:31-35 (**local machine** may be any of the known **computing platforms**); Ex. 1002, ¶250; *supra*, Section IX.B.3. [1.0].

[6.1] an activity monitor configured to monitor API calls and to store a history of at least a portion of the API calls in an activity log, wherein the activity monitor includes a kernel-mode driver adapted to intercept the API calls;

As discussed above with respect to claim 1, Kidron describes a signal monitor (*activity monitor*) that **monitors** and **logs operating system call activity** (*store a history of at least a portion of the API calls in an activity log*). Ex. 1007, 6:22-23; 6:18-19 (operating system calls including file input-output, network input-output, memory management); Ex. 1002, ¶¶54-64, 251 (explaining such operating system calls are examples of API calls).

Kidron explains “signal monitor 12b may be provided **for hooking** [*intercept*] **into the operating system on the local machine**” and “may be **closely fused into the operating system**, for example, as an **extension to the operating system kernel**” (*a kernel-mode driver*). Ex. 1007, 6:15-21; Ex. 1002, ¶252; *supra*, Section IX.B.3. [1.1].

[6.2] a heuristics module configured to identify an activity on the computer that is indicative of pestware residing on the computer and to analyze the activity to determine whether one or more weighted factors associated with the activity exceeds a threshold; and

As explained above, Kidron's signal monitor module monitors and logs operating system call activity. *Supra*, Section IX.B.5. [6.1]. Then, "**pattern recognition processor 12a** [*heuristics module*] periodically **analyzes the log of operating system calls** [*analyze the activity*] to detect any irregular patterns." Ex. 1007, 6:24-25; Ex. 1002, ¶253. **Irregular behavior patterns** (*identify an activity on the computer*) may correspond to, for example, known or unknown viruses, Trojans, worms, or other malicious code (for example, with characteristics of memory resident, file, and/or macro viruses) (*indicative of pestware*). Ex. 1007, 6:28-31. And that such detection of irregular behavior is **at a local machine** (*residing on the computer*). Ex. 1007, 5:9-12; Ex. 1002, ¶254.

Kidron further discloses many examples of heuristics analysis by the signal monitor module. *Supra*, Section IX.B.3. [1.2]. Kidron does not explicitly disclose whether one or more weighted factors associated with an activity exceeds a threshold. Williamson discloses this element. *Supra*, Section IX.A.2. [1.2]; *supra*, Section IX.B.1.; Ex. 1002, ¶255.

[6.3.a] a research portion configured to access, in response to a prompt from the heuristics module, a first set of recorded information on the computer that relates the activity to at least one file residing on the computer,

As explained in Ground 1, Williamson in view of Piccard teaches this element.

A POSITA would have understood that Kidron's pattern recognition processor as modified to include the weighted evaluation of Williamson's heuristic engine 212 as modified by Piccard to include the functions of the malware detection module, history log module, and reporting module would include a research portion. Ex. 1002, ¶256.; *supra*, Section IX.A.7. [6.3.a]; *supra*, Section IX.B.2.

[6.3.b] and wherein the research portion is configured to access a second set of recorded information on the computer that relates the at least one file to an externally networked source from which the file was received; and

As explained in Ground 1, Piccard teaches this element. *Supra*, Section IX.A.7. [6.3.b]; *supra*, Section IX.B.2.; Ex. 1002, ¶257.

[6.4] a reporting portion configured to generate a report that identifies the externally networked source of the file and to report an identity of the externally networked source to an externally networked pestware research entity so as to enable the externally networked pestware research entity to research whether the externally networked source is a source of pestware.

As explained in Ground 1, Piccard teaches this element. *Supra*, Section IX.A.7. [6.4]; *supra*, Section IX.B.2.; Ex. 1002, ¶258.

6. Claim 8

[8.0] The system of claim 6,

See Ground 2, [6.0]-[6.4]. Ex. 1002, ¶259.

[8.1] wherein the heuristics module is configured to receive information about the API calls from the activity monitor and determine whether the API calls are indicative of pestware.

As discussed above, Kidron’s modified system discloses a pattern recognition processor (*heuristics module*) configured to receive information about operating system calls (*API calls*) from signal monitor (*activity monitor*) as a log of operating system calls. *Supra*, Section IX.B.5. [6.1]-[6.2]. Kidron further teaches “pattern recognition processor 12 a periodically analyzes the log of operating system calls to detect any irregular patterns.” Ex. 1007, 6:24-25; Ex. 1002, ¶260. Irregular behavior patterns may correspond to, for example, known or unknown viruses (*whether the API calls are indicative of pestware*). Ex. 1007, 6:28-29; Ex. 1002, ¶261.

7. Claim 12

[12.0] A non-transitory computer-readable medium including processor-executable instructions for identifying an origin of activity on a computer that is indicative of pestware, the instructions including instructions for:

Kidron-Williamson-Piccard discloses a method of identifying an origin of activity on a computer that is indicative of pestware. *Supra*, Section IX.B.3. [1.0]. To the extent the preamble is limiting, Kidron teaches instructions for the method of identifying stored on a non-transitory computer-readable medium. Ex. 1007, 4: 18-24, 5: 43-45, 7:31-35, Claim 24; Ex. 1002, ¶262.

[12.1]-[12.5]

See Ground 2 [1.1]-[1.5]. Ex. 1002, ¶263.

8. Claims 2, 4-5, 7, 9-11, 13-14

Petitioner incorporates by reference its arguments from Ground 1 regarding these claims, and in particular a method, system, and non-transitory computer-readable medium of Kidron modified by Williamson and Piccard. *Supra*, Section IX.A.; Ex. 1002, ¶264.

X. DISCRETIONARY DENIAL WOULD BE INAPPROPRIATE

A. THE BOARD SHOULD NOT DENY INSTITUTION UNDER SECTION 314(a)

The factors considered for §314 denial favor institution. *See Apple Inc. v. Fintiv, Inc.*, IPR2020-00019, Paper 11 (Mar. 20, 2020) (precedential).

1. Factor 1: Whether the district court granted a stay or a stay may be granted if a proceeding is instituted

No motions to stay have been entered or are pending in the co-pending parallel litigation. At worst, this factor is neutral because the Board should not presume how the district court would rule should a motion to stay be filed. *Sand Revolution II, LLC v. Continental Intermodal Group-Trucking LLC*, IPR2019-01393, Paper 24, 7 (PTAB Jun. 16, 2020); *see also Dish Network L.L.C. v. Broadband iTV, Inc.*, IPR2020-01359, Paper 15 at 11 (Feb. 12, 2021).

2. Factor 2: Proximity of the court's trial date to the projected statutory deadline for a final written decision

The District Court's fifth amended scheduling order set the trial date for

August 19, 2024, in a consolidated matter involving five separate lawsuits. Ex. 1015, p. 8. Consolidation is limited to pretrial issues; thus, separate trials will be required. Given that there remain multiple cases, trial of them all on the scheduled date is unlikely. Rather, the Court will likely try one case on that date and try the other cases as its docket allows in the months thereafter.

Moreover, in the litigation involving Kaspersky, the plaintiffs recently amended their complaint to add two new parties. As a result, the defendants moved the Court to extend the dates in the scheduling order by approximately 75 days only for the Kaspersky litigation. Plaintiffs did not oppose the motion. The Court granted the unopposed motion and re-set the case schedule including the trial date. The current trial date for the Kaspersky litigation is November 4, 2024. Ex. 1013, p. 3. Plaintiffs in the case have also recently amended their infringement contentions to add more than two dozen additional accused products, which likely will necessitate an additional modification of the scheduling order and trial date well into 2025.

Although the Board's FWD deadline in this case falls after the earliest possible trial date might occur, the gap between completion dates is small, especially considering that the Kaspersky trial date was recently postponed by 11 weeks. This gap will shrink further or perhaps disappear altogether if an additional modification is granted. Further, in view of any post-trial motion practice that is certain to occur,

any gap between completion will be negligible. Given the postponement of the Kaspersky litigation trial date, this factor is neutral.

3. Factor 3: Investment by the court and the parties

As indicated above, litigation is in its early stages. Beyond claim construction, the parties have invested little in the parallel proceeding. No depositions have yet been taken. Fact discovery does not close until April 4, 2024. Opening expert reports are not due until April 16, 2024, and expert discovery does not close until May 29, 2024. Ex. 1013, p. 2 Given the minimal investment in the parallel proceeding, this factor weighs slightly in favor of institution.

4. Factor 4: Overlap between issues raised in the petition and the parallel proceeding

Petitioner did not raise Williamson, Piccard, or Kidron, the prior art relied upon in the Petition, in preliminary invalidity contentions filed in the parallel litigation. However, final infringement and invalidity contentions are not due until January 5, 2024. Ex. 1013, p. 1. Therefore, the extent of overlap is currently unknown. Further, if IPR is instituted, Petitioner stipulates to not raise in the parallel District Court proceedings any ground raised or that could have been reasonably raised in the IPR. *Sotera Wireless, Inc., v. Masimo Corp.*, No. IPR2020-01019, Paper 12 at 19 (P.T.A.B. Dec. 1, 2020) (precedential, designated Dec. 17, 2020). This factor weighs heavily in favor of institution.

5. Factor 5: Whether the petitioner and the defendant in the parallel proceeding are the same party

Petitioner and PO are parties in the parallel litigation. PO is the petitioner in the parallel proceeding and the Petitioner is one of several defendants in the consolidated co-pending litigation. This factor weighs slightly in favor of institution. *See HP Inc. v. Slingshot Printing LLC*, IPR2020-01084, Paper 13 at 9 (Jan. 14, 2021); *see also Cisco Sys., Inc. v. Ramot at Tel Aviv Univ. Ltd.*, IPR2020-00122, Paper 15, at 11 (J. Crumbley dissenting) (PTAB May 15, 2020) (noting that disfavoring a defendant in parallel litigation is “contrary to the goal of providing district court litigants an alternative venue to resolve questions of validity.”).

6. Factor 6: Other circumstances that impact the Board’s exercise of discretion, including the merits

This factor includes consideration of the merits of the proceeding, which weighs in favor of institution. None of Williamson, Piccard, or Kidron were considered in prosecution. A different publication listing Piccard as an inventor was listed in an IDS, but never used by the examiner during prosecution. The combinations of Williamson and Piccard and Kidron, Williamson, and Piccard were not considered during prosecution. Because the art and arguments being asserted in this Petition are materially different from those presented in prosecution, the considerations as to art previously before the USPTO, are favorable to institution.

Hewlett Packard Enterprise Co., et al. v. Biljco LLC, IPR2022-00420, Paper 17, 6 (PTAB July 12, 2022).

Further, PO has asserted the ‘243 patent against multiple defendants. Therefore, there is strong public interest in the Board instituting the IPR and adjudicating the validity of the patent. This factor weighs in favor of institution.

B. THE BOARD SHOULD NOT DENY INSTITUTION UNDER GENERAL PLASTIC

The Board considers the *General Plastic* factors when multiple petitions are filed against the same patent. *See Gen. Plastic Indus. Co. v. Canon Kabushiki Kaisha*, IPR2016-01357, Paper 19, 16 (Sept. 6, 2016) (Section II.B.4.i. precedential). The ‘243 patent has been challenged in the Sophos IPR, CrowdStrike IPR, and Trend IPR. Discretionary denial of this Petition would unfairly deprive Kaspersky of access to *inter partes* review with respect to the ’243 patent.

1. Factor 1: Whether the Same Petitioner Previously Filed a Petition

This factor weighs against denial because Petitioner is a *different* petitioner than those in the Sophos IPR, CrowdStrike IPR, and Trend IPR. Moreover, there is no significant relationship between Petitioner and Sophos, CrowdStrike or Trend Micro. Although the Board held in *Valve Corp. v. Elec. Scripting Prods., Inc.*, IPR2019-00062, Paper 11 (Apr. 2, 2019) (precedential) that application of the first *Generic Plastic* factor is not limited to instances where multiple petitions are filed

by the same petitioner, the facts here are distinguishable from *Valve*. First, in *Valve*, both the petitioner (Valve) and HTC (who filed the earlier IPR) were co-defendants in the same district court case and were accused of infringing the patent-at-issue based on the same product. *See Valve*, IPR2019-00062, at 9. Conversely, here Petitioner was sued separately, and will receive a separate trial relating to its own, different products that provided the basis of alleged infringement.

Counsel for Petitioner have not coordinated with Sophos, CrowdStrike, or Trend Micro in preparing the Petition. *See Twitter, Inc., v. Palo Alto Research Center Inc.*, IPR2021-01458, Paper 11 at 33 (April 6, 2022) (declining to discretionarily deny the petition under *General Plastic* and finding that there was no evidence that Petitioner had coordinated with previous filers). Moreover, Petitioner has no corporate affiliation with or contractual obligations to the other petitioners.

The Board routinely finds that “*General Plastic* and *Valve* do not apply” when, as here, petitioners are sued independently based on different products, and have no significant relationship. *See, e.g., NetNut Ltd. V. Bright Data Ltd.*, IPR2021-00465, Paper 11, at 8-11 (Aug. 12, 2021) (“*NetNut*”); *Sony Mobile Commc’ns AB. V. Ancora Techs., Inc.*, IPR2021-00663, Paper 17, 7-11 (Jun. 10, 2021) (“*Sony Mobile*”).

Accordingly, this factor heavily weighs against discretionary denial.

2. Factor 2: Whether at the Time of Filing of the First Petition the Petitioner Knew of the Prior Art Asserted in the Second Petition or Should Have Known of It

Because Petitioner is not a petitioner in the Sophos IPR, CrowdStrike IPR, or Trend IPR, factor 2 “is of little probative value.” *NetNut*, IPR2021-00465 at 9; *Western Digital Corp. v. Spex Technologies, Inc.*, IPR2018-00084, Paper 14 at 17 (April 25, 2018) (“Because Petitioner has not previously filed a petition against the same patent, factors 2–5 bear little relevance in this case.”).

Regardless, Petitioner learned of its references during its own prior art search, not from other petitioners. The references and combinations asserted in the Petition differ from those asserted by Sophos, CrowdStrike, and Trend. This factor weighs against discretionary denial.

3. Factor 3: Whether at the Time of Filing of the Second Petition the Petitioner Already Received the Patent Owner’s Preliminary Response to the First Petition

Factors 1-3 only weigh in favor of discretionary denial when there are “road mapping” concerns under factor 3 or other concerns under factor 2. *See Code200, UAB, et al. v. Bright Data Ltd.*, IPR2022-00861 and IPR2022-00862, Paper 18 (PTAB Aug. 23, 2022) (Vidal Decision), at 5 (quoting *Cuozzo Speed Techs., LLC v. Lee*, 579 U.S. 261, 272 (2016) (quoting H.R. Rep. No. 112-98, pt. 1, at 45, 48)). Such road mapping concerns arise where “follow-on petitions would allow petitioners the opportunity to strategically stage their prior art and arguments in multiple petitions,

using [the Board’s] decisions as a roadmap, until a ground is found that results in the grant of review.” *Id.* (citing *General Plastic*, at 17).

While PO has filed a Patent Owner’s Preliminary Response (POPR) in the Sophos IPR, the prior art is different than the prior art in the Petition. Kaspersky has thus gained no tactical advantage or road map.

This factor weighs against discretionary denial.

4. Factors 4 & 5: The Length of Time that Elapsed between the Time the Petitioner Learned of the Prior Art Asserted in the Second Petition and the Filing of the Second Petition, and Whether the Petitioner Provides Adequate Explanation

Because this is Petitioner’s *first* petition challenging the ‘243 patent, the fourth and fifth factors are inapplicable. *See, e.g., Sony Mobile*, IPR2021-00663, 13-15. Moreover, Petitioner learned of its references during its own prior art search and then worked diligently to prepare the Petition, filed within the 1-year statutory deadline. 35 U.S.C. § 315(b). Accordingly, these factors weigh “strongly” against discretionary denial. *Sony Mobile* at 14, 15.

5. Factors 6 & 7: The Finite Resources of the Board, and Requirement to Issue a Final Determination within One Year from Institution

There is no relation between the one-year deadlines for final written decisions on the grounds advanced in the Petition and the grounds advanced in the Sophos

IPR, CrowdStrike IPR, and Trend IPR. The Petition presents no overlapping prior art with those three IPRs. These factors weigh against discretionary denial.

C. THE BOARD SHOULD NOT DENY INSTITUTION UNDER 35 U.S.C. § 325(d)

The Board should institute review and should *not* exercise its discretion to deny institution under § 325(d). *Advanced Bionics, LLC v. Med-El Elektromedizinische Geräte GMBH*, IPR2019-01469, Paper 6, at 6-11 (PTAB Feb. 13, 2020) (precedential).

1. The *Advanced Bionics* Framework Favors Institution

The first condition of the *Advanced Bionics* framework is not satisfied and discretionary denial under §325(d) is inappropriate.

The Petition presents new grounds of unpatentability relying on new prior art references: Williamson, Piccard, and Kidron. No reference was cited by the Examiner in any office action. Ex. 1001, Cover. And although, in an IDS submitted during prosecution of the ‘243 patent, applicant cited a *different* patent application, PCT/US2006/025378, claiming priority to U.S. Application No. 11/171,924 (“Piccard ‘924”), the examiner never cited Piccard ‘924 in any office action.

Since part one of the *Advanced Bionics* framework is not met, Petitioner need not address part two. *Id. Becton, Dickinson* factors (a), (b), and (d) relate to part one of the *Advanced Bionics* framework—whether the art or arguments presented in the

Petition are the same or substantially the same as those previously presented to the Office. *Advanced Bionics*, IPR2019-01469 at 10.

XI. CONCLUSION

Petitioner asks that the Board institute review and cancel each challenged claim.

Dated: December 8, 2023

Respectfully submitted,

By: /Joseph T. Miotke/
Joseph T. Miotke, Reg. No. 47,798
Dewitt LLP
901 Marquette Avenue
2100 AT&T Tower
Minneapolis, MN 55402
Tel: (612) 305-1400

CERTIFICATE OF COMPLIANCE UNDER 37 C.F.R. § 42.24(d)

Pursuant to 37 C.F.R. § 42.24(d), Petitioner hereby certifies, in accordance with and reliance on the word count provided by the word-processing system used to prepare this Petition, that the number of words in this paper is 13,303. Pursuant to 37 C.F.R. § 42.24(d), this word count excludes the table of contents, table of authorities, mandatory notices under § 42.8, certificate of service, certificate of word count, appendix of exhibits, and any claim listing.

Dated: December 8, 2023

Respectfully submitted,

By: /Joseph T. Miotke/
Joseph T. Miotke, Reg. No. 47,798
Dewitt LLP
901 Marquette Avenue
2100 AT&T Tower
Minneapolis, MN 55402
Tel: (612) 305-1400

CERTIFICATE OF SERVICE

The undersigned certifies that, in accordance with 37 C.F.R. § 42.6(e) and 37 C.F.R. § 42.105, service was made on Patent Owner as detailed below.

Date of service December 8, 2023

Manner of service FEDERAL EXPRESS

Documents served Petition for Inter Partes Review Under 35 U.S.C. § 312

and 37 C.F.R. § 42.104 of U.S. 8,201,243; Petitioner's Exhibit List; Exhibits 1001-1019; and Petitioner's power of attorney.

Persons served Merchant & Gould P.C.

P.O. Box 2903

Minneapolis, MN 55402-0903

Dated: December 8, 2023

Respectfully submitted,

By: /Joseph T. Miotke/

Joseph T. Miotke, Reg. No. 47,798

Dewitt LLP

901 Marquette Avenue

2100 AT&T Tower

Minneapolis, MN 55402

Tel: (612) 305-1400