

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

TREND MICRO, INC.
Petitioner,

v.

WEBROOT INC.,
Patent Owner.

Case No. IPR2024-00106
U.S. Patent No. 8,418,250

Petition for *Inter Partes* Review of U.S. Patent No. 8,418,250

Filed on behalf of Trend Micro
By: Robert M. Hansen
Matthew J. Anderson
Scott B. Amankwatia
The Marbury Law Group, PLLC
11800 Sunrise Valley Drive
15th Floor
Reston, VA 20191
Tel: (703) 391-2900
Fax: (703) 391-2901

TABLE OF CONTENTS

I. INTRODUCTION	1
II. MANDATORY NOTICES	3
A. Real Parties-in-Interest 37 C.F.R. § 42.8(b)(1).....	3
B. Related Matters Under 37 C.F.R. § 42.8(b)(2)	4
1. Current Litigation.....	4
2. Administrative Proceedings	4
D. Service of Information Under 37 C.F.R. § 42.8(b)(4)	5
III. PAYMENT OF FEES.....	6
IV. REQUIREMENTS OF <i>INTER PARTES</i> REVIEW	6
A. Standing	6
B. Identification of Challenge and Relief Requested	6
C. How the Challenged Claims Are to Be Construed Under 37 C.F.R. § 42.104(b)(3)	8
D. How the Challenged Claims Are Unpatentable Under 37 C.F.R. § 42.104(b)(4)	10
E. Supporting Evidence Under 37 C.F.R. § 42.104(b)(5).....	10
V. THRESHOLD REQUIREMENT FOR <i>INTER PARTES</i> REVIEW	10
VI. PERSON OF ORDINARY SKILL IN THE ART.....	10
VII. STATE OF THE ART.....	11
A. Malware	11
B. Malware Detection Software	11
VIII. OVERVIEW OF THE '250 PATENT	12
IX. DETAILED CHALLENGE	17

A. Ground 1: Claims 1, 2, 12–14, and 24 Are Rendered Obvious Over Halperin in View of Mankins	17
1. Prior Art Overview	17
2. Motivation to Combine and Reasonable Expectation of Success	20
3. Analysis	23
B. Ground 2: Claims 1–30 Are Rendered Obvious Over Halperin in view of Mankins and Kirby	44
1. Prior Art Overview	44
2. Motivation to Combine and Reasonable Expectation of Success	45
3. Analysis	47
X. PTAB DISCRETION SHOULD NOT PRECLUDE INSTITUTION	62
A. PTAB Should Not Exercise Its Discretion to Deny Institution Under <i>Fintiv</i> ..	62
1. Factor 1: Likelihood of a Stay	63
2. Factor 2: District Court Schedule	63
3. Factor 3: Petitioners’ Investment in the IPR Outweighs Forced Investment in Litigation to Date	66
4. Factor 4: The Petition Raises Unique Issues	67
5. Factor 5: Institution Will Provide the Board an Opportunity to Invalidate Claims That Later Could Be Reasserted Against Others	68
6. Factor 6: Other Circumstances Support Institution	68
B. PTAB Should Not Exercise its Discretion to Deny Institution Under <i>Becton</i> and <i>Advanced Bionics</i>	69
XI. CONCLUSION	69
APPENDIX: CHALLENGED CLAIM LISTING	73

TABLE OF AUTHORITIES

Cases

<i>10X Genomics, Inc. v. Bio-Rad Labs., Inc.</i> , IPR2020-00086, Paper 8 at 17–22 (PTAB Apr. 27, 2020)	9
<i>Advanced Bionics LLC v. MED-EL Elektromedizinische Gerate GmbH</i> , IPR2019-01469, Paper 6 (PTAB Feb. 13, 2020)	69
<i>Apple v. Seven Networks</i> , IPR2020-00255, Paper 13 at 12–15 (July 28, 2020)	67
<i>Becton, Dickinson & Co. v. B. Braun Melsungen AG</i> , IPR2017-01586, Paper 8 (Dec. 15, 2017)	69
<i>Cizion, LLC v. Kerr Machine Co.</i> , PGR2020-00065, Paper 10 at 26 (Dec. 3, 2020)	64, 65
<i>HTC Corp. v. Cellular Communications Equip., LLC</i> , 877 F.3d 1361, 1367–68 (Fed. Cir. 2017)	8
<i>KSR Int’l Corp. v. Teleflex, Inc.</i> , 550 U.S. 398, 417 (2007)	20
<i>Micron Tech., Inc. v. Godo Kaisha IP Bridge 1</i> , IPR2020-01008, Paper 10 at 14 (Dec. 7, 2020)	64
<i>Nidec Motor Corp. v. Zhongshan Broad Ocean Motor Co.</i> , 868 F.3d 1013, 1017 (Fed. Cir. 2017.	9
<i>Oticon Medical AB v. Cochlear Ltd.</i> , IPR2019-00975 Paper 15, 20 (PTAB Oct. 16, 2019)	69
<i>Samsung Elec. ’s Co., Ltd. et all v. Staton Techniya, LLC</i> , IPR202200302, Paper 13, 10 (PTAB July 11, 2022)	65
<i>Sand Revolution II, LLC v. Continental Intermodal Grp.</i> , IPR2019-01393, Paper 24 (June 16, 2020)	63, 67

<i>Sotera Wireless, Inc., v. Masimo Corp.</i> , No. IPR2020-01019, Paper 12 at 19 (P.T.A.B. Dec. 1, 2020)	67
<i>Volkswagen Grp. of Am., Inc. v. Arigna Tech. Ltd.</i> , IPR2021-01321, Paper 10 at 7 (Feb. 15, 2022)	63
<i>Western Digital Corp. v. Kuster</i> , IPR2020-01391, Paper 10 at 9 (Feb. 16, 2021)	63
<i>Western Digital Corp. v. SPEX Techs., Inc.</i> , IPR2018-00082, Paper 11 at 12 (PTAB Apr. 25, 2018)	9

Statutes

35 U.S.C. § 102 (2011)	8
35 U.S.C. § 103	1, 6, 8
35 U.S.C. § 311	1
35 U.S.C. § 314(a)	10
35 U.S.C. § 325(d)	69

Rules

37 C.F.R. § 42.8(b)(1)	3
37 C.F.R. § 42.8(b)(2)	4
37 C.F.R. § 42.8(b)(4)	5
37 C.F.R. § 42.15(a)	6
37 C.F.R. § 42.63(e)	10
37 C.F.R. § 42.103	6
37 C.F.R. § 42.104	6
37 C.F.R. § 42.104(a)	6
37 C.F.R. § 42.104(b)	6
37 C.F.R. § 42.104(b)(3)	8
37 C.F.R. § 42.104(b)(4)	10
37 C.F.R. § 42.104(b)(5)	10
37 C.F.R. § 42.122	6

TABLE OF EXHIBITS

Exhibit No.	Document
1001	U.S. Patent No. 8,418,250 (“the ’250 Patent”)
1002	File History for U.S. Patent No. 8,418,250
1003	Complaint, <i>Webroot Inc, et al. v. Sophos Ltd.</i> , No. 6:22-cv-240 (W.D. Tex.)
1004	Declaration of Dr. Richard Newman
1005	Curriculum Vitae of Dr. Richard Newman
1006	U.S. Patent Appl. Pub. No. 2004/0111632 to Halperin (“Halperin”)
1007	U.S. Patent No. 7,694,150 to Kirby (“Kirby”)
1008	U.S. Patent No. 7,900,194 to Mankins (“Mankins”)
1009	Stipulation Waiving IPR Claims in District Court Litigation
1010	S. Forrest <i>et al.</i> , <i>Computer Immunology</i> , Comms. of the ACM, Vol. 40, No. 10, 88–96 (1997)
1011	A. Snoeren <i>et al.</i> , <i>Single-Packet IP Traceback</i> , IEEE/ACM Trans. on Networking (TON), Vol. 10, No. 6, 721–34 (2002)
1012	S. Hofmeyr, <i>Intrusion Detection Using Sequences of System Calls</i> , J. Computer Security, Vol. 6, Issue 3, 151–180 (Aug. 1998)
1013	S. Forrest <i>et al.</i> , <i>A Sense of Self for Unix Processes</i> , Proc. 1996 IEEE Symp. on Security and Privacy, 120–28 (1996)
1014	Order Granting Fifth Amended Scheduling Order, Dkt. 253, <i>Webroot, Inc., et al. v. AO Kaspersky Lab</i> , No. 6:22-cv-00243-ADA-DTG (W.D. Tex.) (Mar. 29, 2023)
1015	Excerpt from Transcript of Motions Hearing in <i>Webroot, Inc., et al. v. AO Kaspersky Lab</i> , No. 6:22-cv-00243-ADA-DTG (W.D. Tex.) (Dec. 9, 2022)
1016	<i>Threat Graphs examples: Malware detection</i> , Sophos Ltd., KB-000036359, https://support.sophos.com/support/s/article/KB000036359?language=en_US (Feb. 23, 2023)

I. INTRODUCTION

Pursuant to 35 U.S.C. § 311, Trend Micro, Inc. (“Petitioner”) submits this Petition for *inter partes* review (“IPR”), seeking cancellation of claims 1–30 of U.S. Patent No. 8,418,250 (Ex. 1001). This Petition demonstrates there is a reasonable likelihood Petitioner will prevail on at least one of the challenged claims based upon prior art references that the USPTO did not have before it during prosecution. Claims 1–30 should be canceled as unpatentable under 35 U.S.C. § 103 over the prior art references identified in this Petition.

The ’250 Patent describes and claims a purportedly novel malware detection method and apparatus that automatically generates a mask of “acceptable” behavior for a computer object (*i.e.*, an executable file) at a “base computer” based on “normal” behavioral data about the computer object provided by one or more “remote computers.” The computer object is run on one or more of the “remote computers,” and its operation is automatically monitored. The computer object is allowed to continue to run if its behavior is permitted by the mask of “acceptable” behavior, and is disallowed to run when the monitored behavior extends beyond that permitted by the mask. And, if the monitored behavior extends beyond the “acceptable” behavior permitted by the mask, the computer object is classified as malware. *See, e.g.*, Ex. 1001, cl. 1.

However, behavior-based malware detection generally, and monitoring for anomalous behavior specifically, as well as the generic host and remote computer architecture claimed in the '250 Patent, already were well known at the time of the '250 Patent's earliest priority date. For example, and as elaborated further below, Halperin describes a behavior-based malware detection system wherein multiple individual computers (*i.e.*, "remote computers") "detect associated target behavior" and provide "target behavior information" to a server (*i.e.*, a "host computer"). The Halperin server detects suspicious behavior patterns by determining whether "correlated target behavior corresponds to a predefined suspicious behavior pattern of target behavior." In other words, among other things, the Halperin server correlates and evaluates the behavior of computer objects on remote computers. Similarly, Mankins discloses a behavior-based malware detection system that automatically generates a "Bloom filter data object" that represents the "normal" behavior of an executable, and subsequently "monitor[s] the behavior of the computer program" and "takes an appropriate predefined action" when "the Bloom filter data object does not contain a match for the monitor[ed] behavior." In other words, among other things, Mankins uses a Bloom filter "mask" of acceptable behavior to determine when an object's monitored behavior extends beyond that permitted by the Bloom filter mask.

In combination, Halperin and Mankins render each limitation of the independent method claims of the '250 Patent obvious (Ground 1). And all of the claims of the '250 Patent are obvious over Halperin and Mankins in view of Kirby, which is yet another behavior-based malware detection system (Ground 2). Among other things, Kirby describes a two-tier system that classifies an object as either benign or “malware” based on data about that object as well as the actual behavior of the object, and explicitly accounts for relationships between objects. Halperin, Mankins, and Kirby are all in the same field, directed toward solving the same problems. They are compatible, and it would have been obvious to combine them and to modify one based on teachings of another. Petitioner respectfully requests the cancellation of claims 1–30 of the '250 Patent.

II. MANDATORY NOTICES

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

The real parties-in-interest are Trend Micro, Inc., Trend Micro America, Inc. and Trend Micro, Incorporated. Trend Micro, Inc. is a wholly owned subsidiary of Trend Micro America, Inc., which is a wholly owned subsidiary of Trend Micro Incorporated, a Japanese Corporation.

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

1. Current Litigation

The '250 Patent is presently the subject of litigations in five suits brought by Patent Owner against Petitioner Trend Micro, Inc. and four other defendants in: *Webroot, Inc. et al. v. Trend Micro, Inc.*, No. 6:22-cv-00239-ADA-DTG (W.D. Tex.), (the “Trend Micro Action”); *Webroot Inc, et al. v. Sophos Ltd.*, No. 6:22-cv-00240-ADA-DTG (W.D. Tex.) (Ex. 1003); *Webroot, Inc. et al. v. CrowdStrike Holdings, Inc. et al.*, No. 6:22-cv-00241-ADA-DTG (W.D. Tex.), (the “CrowdStrike Action”)¹; and *Webroot, Inc. et al. v. AO Kaspersky Lab*, No. 6:22-cv-243-ADA-DTG (W.D. Tex.), (the “Kaspersky Action”).

2. Administrative Proceedings

On December 29, 2022, CrowdStrike, Inc. filed a petition for *inter partes* review challenging all claims of the '250 Patent. *See Crowdstrike, Inc. v. Webroot Inc.*, IPR2023-00289. On March 2, 2023, Trend Micro, Inc. filed its first petition for *inter partes* review challenging all claims of the '250 Patent. *See Trend Micro, Inc. v. Webroot Inc.*, IPR2023-00655. On March 8, 2023, Sophos Ltd. and Sophos Inc. filed a petition for *inter partes* review challenging all claims of the '250 Patent. *See Sophos Inc. v. Webroot Inc.*, IPR 2023-00699. IPR2023-00699 was instituted

¹ The *CrowdStrike* Action was recently dismissed by stipulation.

on October 11, 2023 (“the Sophos IPR”). Institution of IPR2023-00289 and IPR2023-00655 have been denied.

Petitioner provides the following designation of counsel:

Lead Counsel	Backup Counsel
Robert M. Hansen (Reg. No. 43,656) The Marbury Law Group PLLC 11800 Sunrise Valley Drive 15 th Floor Reston, VA 20191 Tel:(703)-391-2900 bhansen@marburylaw.com	Matthew J. Anderson (Reg. No. 78,257) The Marbury Law Group PLLC 11800 Sunrise Valley Drive 15 th Floor Reston, VA 20191 Tel:(703)-416-9721 manderson@marburylaw.com
	Scott B. Amankwatia (Reg. No. 78,226) The Marbury Law Group PLLC 11800 Sunrise Valley Drive 15 th Floor Reston, VA 20191 Tel:(703)-391-2900 samankwatia@marburylaw.com

D. Service of Information Under 37 C.F.R. § 42.8(b)(4)

Service of any documents via hand delivery may be made at the mailing address of lead and backup counsel designated above. Petitioner also hereby consents to electronic service, and service via electronic mail may be made at the email addresses provided above for the lead and backup counsel.

III. PAYMENT OF FEES

Pursuant to 37 C.F.R. §§ 42.103 and 42.15(a), the required fee was submitted, but the Office is authorized to charge any fee deficiency, or credit overpayment, to Deposit Account 18-1579, including additional fees due in connection with this Petition.

IV. REQUIREMENTS OF *INTER PARTES* REVIEW

Pursuant to 37 C.F.R. § 42.104, each requirement for *inter partes* review of the '250 Patent is satisfied.

A. Standing

Pursuant to 37 C.F.R. § 42.104(a), Petitioner certifies that the '250 Patent is available for IPR and that Petitioner is not barred or estopped from requesting IPR of the claims on the ground identified herein. Specifically, Petitioner certifies that: Petitioner has not filed a civil action challenging the validity of the '250 Patent; and this petition seeks to join the Sophos IPR (IPR2023-00699). The provisions of 37 C.F.R. § 42.122 do not prohibit this *inter partes* review because this Petition was filed less than one month after institution of the Sophos IPR (October 11, 2023).

B. Identification of Challenge and Relief Requested

Pursuant to 37 C.F.R. § 42.104(b), Petitioner requests that the Board institute *inter partes* review of claims 1–30 (the “Challenged Claims”) of the '250 Patent on one or more grounds under pre-AIA 35 U.S.C. § 103. The precise relief requested

by Petitioner is that claims 1–30 be canceled in view of the following prior art references:

Patent/ Publication	Filing/ Priority Date	Date of Publication	Exhibit No.
Halperin	May 5, 2003	June 10, 2004	Ex. 1006
Kirby	June 22, 2004	April 6, 2010	Ex. 1007
Mankins	March 25, 2004	March 1, 2011	Ex. 1008

The '250 Patent issued from application No. 13/543,866 filed June 8, 2012, which is a continuation of application No. 11/477,807, filed June 30, 2006, and which claims priority to foreign patent GB 0513375.6, filed June 30, 2005. Claims 1–30 of the '250 Patent are, therefore, not entitled to claim priority any earlier than June 30, 2005, and, in the related district court litigation, Patent Owner has asserted that the claims are entitled to a priority date of June 30, 2005. For purposes of this Petition, Petitioner assumes that the claims of the '250 Patent are entitled to a priority date of June 30, 2005.²

² Although Petitioner assumes for purposes of this Petition a priority date of June 2005, Petitioner does not concede that any claim of the '250 Patent is entitled to that or any priority date earlier than June 8, 2012. The Board need not resolve any priority issues for the purposes of this Petition.

Each of the references cited above is prior art. Halperin was filed on May 5, 2003 and published on June 10, 2004. Halperin therefore qualifies as prior art to the '250 Patent under at least 35 U.S.C. § 102(b).³ Mankins issued on March 1, 2011, from an application filed on March 23, 2005. Mankins therefore qualifies as prior art under at least § 102(e). Kirby issued on April 6, 2010, from an application filed on June 22, 2004. Kirby therefore qualifies as prior art under at least § 102(e). None of Halperin, Mankins, or Kirby were cited or considered during prosecution of the '250 Patent.

In the instant *inter partes* review, Petitioner applies the above references and asserts the following grounds of rejection under 35 U.S.C. § 103:

Ground	Claims	Proposed Statutory Rejections for the '250 Patent
1	1, 2, 12–14, 24	Obvious over Halperin in view of Mankins
2	1–30	Obvious over Halperin in view of Mankins and Kirby

C. How the Challenged Claims Are to Be Construed Under 37 C.F.R. § 42.104(b)(3)

Petitioner does not believe any constructions have an impact on the invalidity analyses set forth herein. *See HTC Corp. v. Cellular Communications Equip., LLC*,

³ Unless specifically noted, all references to Title 35 of the United States Code refer to code pre-America Invents Act. *See, e.g.*, 35 U.S.C. § 102 (2011).

877 F.3d 1361, 1367–68 (Fed. Cir. 2017) (affirming the Board’s application, absent an express construction of a term). Although Petitioner and Patent Owner disagree over the construction of certain terms in the parallel district court litigation, for the purpose of this Petition, Petitioner relies on Patent Owner’s proposed constructions (plain and ordinary meaning). *See 10X Genomics, Inc. v. Bio-Rad Labs., Inc.*, IPR2020-00086, Paper 8 at 17–22 (PTAB Apr. 27, 2020) (a Petitioner need only identify “[h]ow the challenged claim is to be construed” for purposes of the petition, and “judicial efficiency will be enhanced by allowing Petitioner to rely upon a claim construction that Patent Owner is relying upon in the related district court litigation to assert infringement of the challenged patent”); *see also Western Digital Corp. v. SPEX Techs., Inc.*, IPR2018-00082, Paper 11 at 12 (PTAB Apr. 25, 2018).

Moreover, the Board “need only construe terms ‘that are in controversy, and only to the extent necessary to resolve the controversy.’” *Nidec Motor Corp. v. Zhongshan Broad Ocean Motor Co.*, 868 F.3d 1013, 1017 (Fed. Cir. 2017) (citation omitted). Because Petitioner bases this Petition on constructions *Patent Owner* has advocated, the Petition does not present any controversy over claim meaning and the Board need not construe any terms. *Nidec*, 868 F.3d at 1017. Petitioner maintains the Challenged Claims would have been obvious under any reasonable construction.

D. How the Challenged Claims Are Unpatentable Under 37 C.F.R. § 42.104(b)(4)

An explanation of how the Challenged Claims are unpatentable, including the identification of where each element of the claim is found in the prior art patents or printed publications, is provided below.

E. Supporting Evidence Under 37 C.F.R. § 42.104(b)(5)

The exhibit numbers of the supporting evidence relied upon and the relevance of the evidence to the Challenged Claims, including an identification of specific portions of the evidence that support the challenge, are provided below. The technical information and ground for rejection explained in the petition are further supported by the Expert Declaration of Dr. Richard Newman, attached as Exhibit 1004.

A List of Exhibits is included in this petition pursuant to 37 C.F.R. § 42.63(e).

V. THRESHOLD REQUIREMENT FOR *INTER PARTES* REVIEW

Under 35 U.S.C. § 314(a), institution of *inter partes* review requires “a reasonable likelihood that the petitioner would prevail with respect to at least one of the claims challenged in the petition.” This Petition meets this threshold for the asserted ground of unpatentability.

VI. PERSON OF ORDINARY SKILL IN THE ART

A person of ordinary skill in the art (“POSITA”) in June 2005 would have had at least a Bachelor’s degree in Computer Science, Computer Engineering,

Electrical Engineering, or an equivalent discipline, and at least two years' worth of experience developing dynamic malware detection and computer networking or network architecture, or a Master's degree in a related field. Alternatively, a POSITA could have four or more years of work experience in computer networking and in the development of client-server systems and dynamic malware detection. Ex. 1004, ¶17.

VII. STATE OF THE ART

All concepts in this section were well known and widely used by POSITAs at least as of June 30, 2005.

A. Malware

Malware, or malicious software, refers to software developed to modify, steal, or destroy data on a computer or a computer network. Ex. 1004, ¶32. Malware is often introduced into a computer or computer network through the Internet or an external storage device. *Id.*, ¶¶33–34.

B. Malware Detection Software

Traditional malware detection software relies on signature matching to identify malicious code. *Id.*, ¶35. A signature is a sequence of bytes that uniquely identify a computer object based on characteristics such as the content or the metadata of the object. When a new computer object is introduced into a user's computer, the malware detection software generates one or more signatures for that

object and searches a malware detection software provider's database for a match. If the generated signature is found in the database, the object can be classified as malware and isolated. *Id.*

Although the use of signatures is effective, it has well-known, significant disadvantages. *Id.*, ¶36. Signatures can be written only after a malware sample has been seen by the malware detection provider, so new malicious code that is sufficiently different may spread before the provider can add a signature for that new code into the signature database. *Id.*, ¶37.

Another method of malware detection that was known in the art is behavior analysis. *Id.*, ¶38. This method of malware detection evaluates the actions or intended actions of a computer object. Behavior-based analysis does not require the computer object to be previously known because it examines the actual or expected behavior of a computer object. *Id.* Often, behavior-based antivirus software includes policies that define the actions an object can or cannot do. *Id.*, ¶¶39–40. If an object performs or tries to perform an action that is considered malicious or is not known to be safe, the object may be classified as malware. *Id.*, ¶40.

VIII. OVERVIEW OF THE '250 PATENT

The '250 Patent purports to describe “methods and apparatus for dealing with malware” that employ a network including a central server (or “base computer”)

and various remote computers. Ex. 1001, 1:4–5; cl. 1. Computer objects are classified, or reclassified, as malware based on comparing their observed behavior with a profile, or “mask,” that defines acceptable behavior for computer objects. *Id.*, cl. 1.

According to the patent, a “typical anti-malware product” “scans objects or the results of an algorithm applied to the object or part thereof to look for signatures in the object that are known to be indicative of the presence of a virus.” *Id.*, 1:19–22.

When new viruses are discovered by anti-malware service providers, the “service providers in the field [would] generate signatures that attempt to deal with these and these signatures are then released as updates to their anti-malware programs.” *Id.*, 1:25–28. In addition, “[h]euristic methods have also been employed” for dealing with malware. *Id.*, 1:28–29. Although these systems are effective for detecting already-known threats, “since they rely on signature files being generated and/or updated, there is inevitably a delay between a new piece of malware coming into existence or being released and the signatures for combating that malware being generated or updated and supplied to users.” *Id.*, 1:30–35. In addition, “in order to try to defeat anti-virus products, the malware writers use obfuscation techniques in order to attempt to hide the signature or signature base data of the virus code from detection.” *Id.*, 1:37–42.

The '250 Patent purports to address this known problem by describing a distributed process that exchanges information between remote computers on which potential malware runs and a central server (“base computer”). Figure 1 (reproduced below) depicts a conventional network on which this process runs. Figure 2, also below, depicts the process itself.

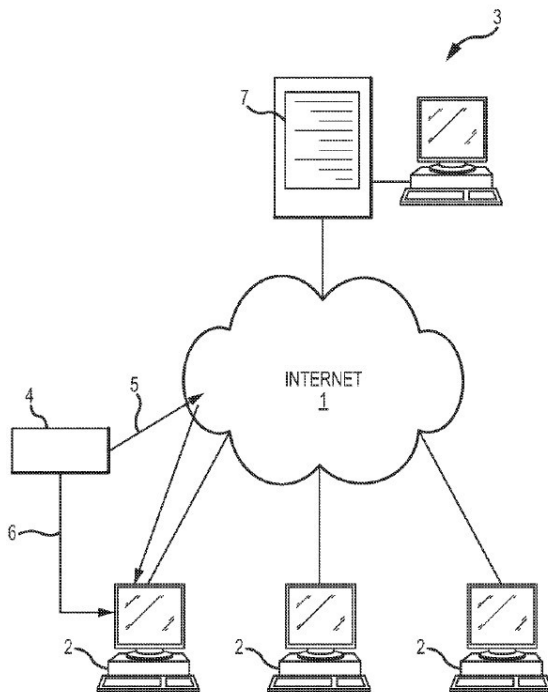


FIG.1

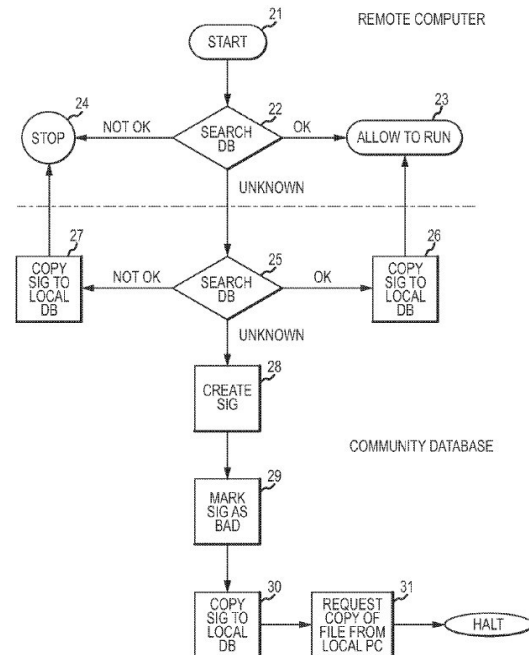


FIG.2

When a remote computer 2 discovers an unknown object, “details of the object are passed over the Internet 1 or other network to the base computer 3 for storing in the community database 7 and preferable for further analysis at the base computer 3.” *Id.*, 8:44–47. Next, “the community database 7 is searched for signature of the object that was not previously known at the remote computer 2.”

Id., 8:60–62. If a match is found in the community database that indicates the object is safe, “a copy of the signature or at least a message that the object is safe is sent to the local database of the remote computer 2 concerned at step 26 to populate the local database.” *Id.*, 8:62–66. If a match is found in the community database 7 that indicates the object is not safe, “the signature is copied back to the local database and marked ‘unsafe’ at step 27.” *Id.*, 9:4–7.

Figure 3 (reproduced below) is a flowchart that shows further aspects of the '250 Patent that operate when the safety of a process (“computer object”) is unknown.

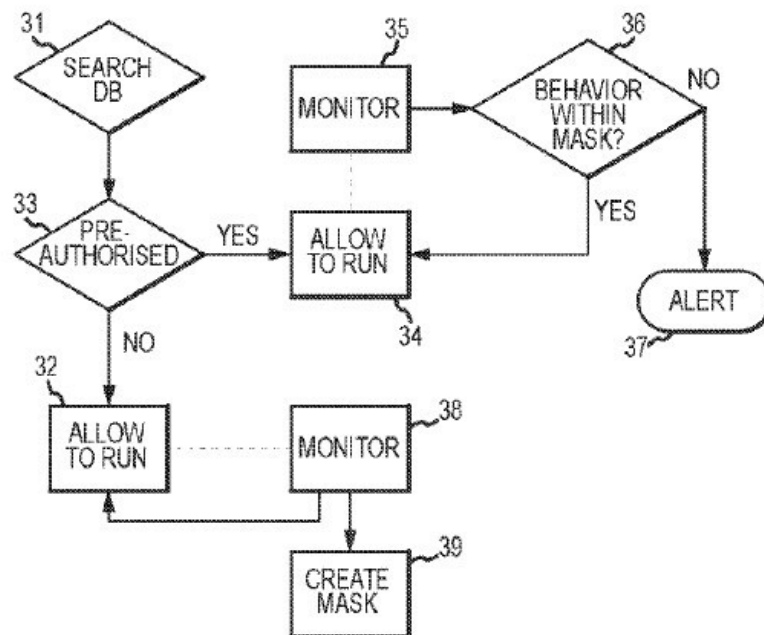


FIG.3

When the safety of a computer object is *unknown* at the base computer, the process is allowed to run and it “is monitored at step 38 in order to generate a mask 39 representative of the normal behaviour of that process.” *Id.*, 15:59–62. The process is allowed to run on the remote computers and is “continually monitor[ed]” for “any behaviour that differs from previous behaviour of the process” and “the mask can be modified, or this behaviour might be used to determine that a process that was once considered safe should now be considered to be unsafe.” *Id.*, 15:64–16:3.

The Challenged Claims recite elements of a method and apparatus of “classifying a computer object as malware.” *Id.*, cl. 1. The method describes a “base computer” that receives “data about a computer object” from “plural remote computers.” The data includes “information about the behaviour of the object.” The base computer then uses the received data to determine and classify the object (that is the subject of the received data) as malware or not malware. Next, the ’250 Patent discloses a “mask” for the object that is “automatically generat[ed]” and “defines acceptable behaviour of the object.” The object is allowed to run on the plural remote computers while being monitored. While the object performs actions that are “acceptable” and included in the mask, the object is allowed to continue to run. But if the object performs an action that is not included in the mask, the system stops the object from running and reclassifies it as malware. *Id.*

As demonstrated below, the Challenged Claims would have been obvious over the prior art. This petition cites prior art references (Halperin, Mankins, Kirby) that describe the threat-protection methods and apparatuses that the '250 Patent claims. These prior art references were not before the Examiner during the prosecution of the '250 Patent.

IX. DETAILED CHALLENGE

A. Ground 1: Claims 1, 2, 12–14, and 24 Are Rendered Obvious Over Halperin in View of Mankins

As described below, the combination of Halperin and Mankins, in view of the knowledge of a POSITA, render obvious claims 1, 2, 12–14, and 24.

1. Prior Art Overview

a. Halperin

Like the '250 Patent, Halperin discloses “detecting malicious activity in a computer network,” Ex. 1006, Abstract, “by correlating behavior and/or data from different computers.” *Id.*, [0037]. Halperin proposes a solution “that overcomes disadvantages of the prior art,” which Halperin describes as “virus scanners [that] operate by searching a computer for tell-tale byte sequences known as ‘signatures’ that exist in known viruses,” and “antivirus software that employs heuristic techniques to identify typical virus behavior . . . and then identifying any deviation from such behavior.” *Id.*, [0005]–[0007]. Figure 5 of Halperin (shown below) is “a simplified conceptual illustration”:

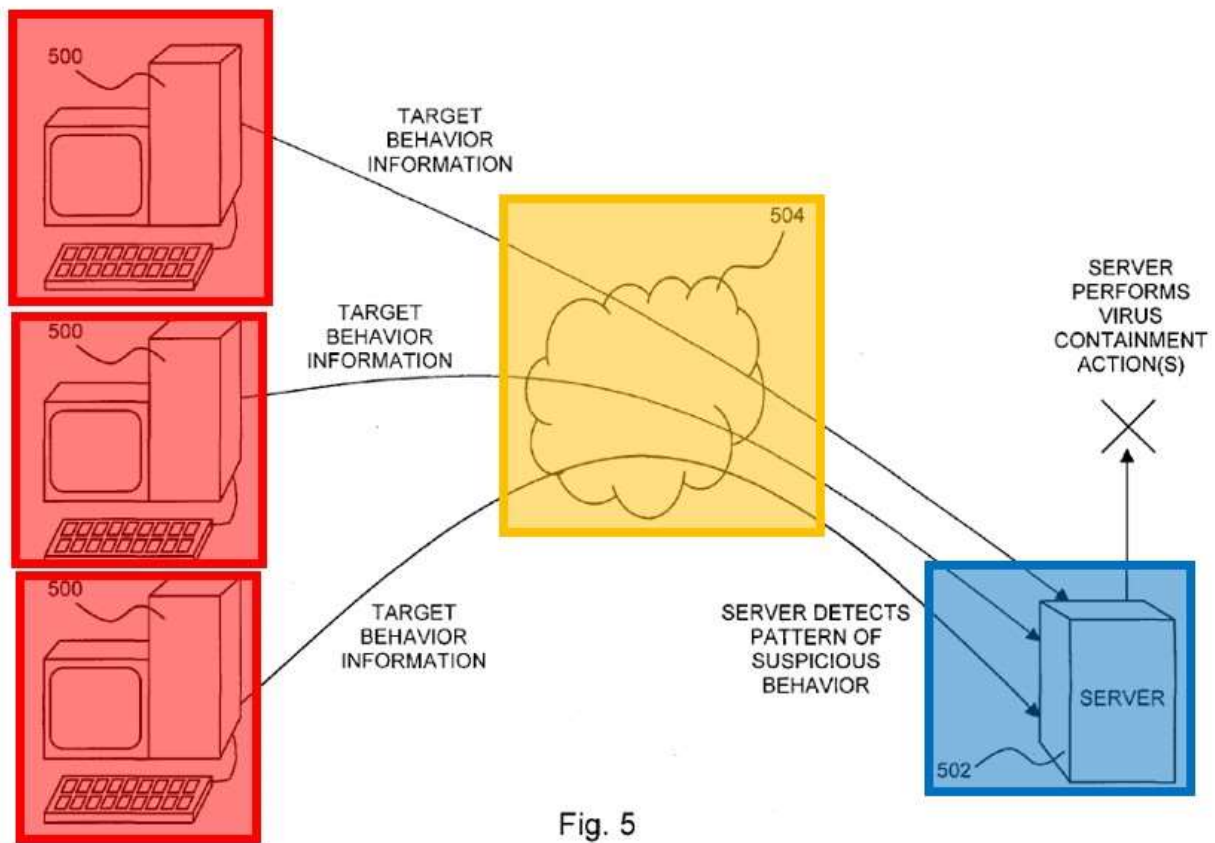


Fig. 5

“In the system of Figure 5 one or more **computers 500** are shown, being configured to communicate with a **server 502** via a **network 504**, such as the Internet.” *Id.*, [0036]. **Server 502** “may be configured with such target behavior profiles and may detect associated target behavior at **computers 500** using conventional techniques.” *Id.*, [0047]. “After collecting information regarding target behavior detected at two or more of **computers 500**, **server 502** may then correlate the presence of target behavior detected at two or more of **computers 500** in order to determine whether the correlated target behavior corresponds to a predefined suspicious behavior pattern.” *Id.*, [0047]. Halperin then compares the

correlated data to another object or entity, specifically a “predefined suspicious behavior pattern of target behavior.” *Id.*, [0038]. On the basis of the comparison between the set of correlated data and the “predefined suspicious behavior pattern of target behavior,” Halperin determines “that a computer virus may have infected those computers.” *Id.*, [0047].

b. Mankins

Mankins discloses a “system and method for kernel-based intrusion detection using Bloom filters.” Ex. 1008, 3:18–19. Mankins explains that “normal (i.e., expected) system-call behavior is defined by monitoring system-call behavior during operation of the computer program under training conditions (e.g., when the computer program is not subject to intrusion attacks).” *Id.*, 3:37–40. Mankins’ “intrusion detection system then uses a Bloom filter to generate a Bloom filter data object containing data representative of the normal system-call behavior.” *Id.*, 3:40–43. This “predefined Bloom filter data is representative of ‘normal’ computer program behavior, which refers to behavior that is expected during operation of the computer programs when not subject to intrusion attempts.” *Id.*, 3:26–30. Accordingly, “[a]nomalous behavior is identified when the actual system-call behavior does not have a match in the Bloom filter data object.” *Id.*, 3:53–55.

2. Motivation to Combine and Reasonable Expectation of Success

It would have been obvious to a POSITA to incorporate Mankins's Bloom filter data object into Halperin's malicious activity detection, in order to enable the identification of malware-based intrusion based on expected system call behavior. Ex. 1004, ¶62. Halperin employs one or more suspicious event sensors to detect computer program behavior that corresponds to predefined suspicious patterns, and Mankins teaches a sensor that uses a behavioral approach to malware detection that monitors the actions of a computer program for behavior (system calls) outside known "normal" or acceptable behavior. *Id.*

Intrusion detection using sequences of system calls was known in the art as early as the mid-1990s. *Id.*, ¶63. A POSITA would have recognized that the examples of "target behavior" described in Halperin at [0039]–[0046] would be detected by monitoring system calls. *See id.* Accordingly, incorporating Mankins's Bloom filter data object into Halpern's malicious activity detection approach would have been nothing more than a predictable use of a prior art element according to its established function. *KSR Int'l Corp. v. Teleflex, Inc.*, 550 U.S. 398, 417 (2007).

It would have been obvious to a POSITA to add Mankins's Bloom filter approach to Halperin, and a POSITA would have recognized that Mankins's Bloom filter data object could be incorporated at either a remote computer or a base

computer using the system architecture described in Halperin (including a “server 502” and “computers 500” that communicate over a “network 504”). *See* Ex. 1006, [0036], [0047], Fig. 5; Ex. 1004, ¶¶64–66. For example, because each remote computer in Halperin’s system monitors the “target behavior” of objects and sends it to a “central database” which “logs” the monitored behavior for comparison against predefined patterns, it would be an obvious modification to collect system call sequences as part of the data and compare them against a Bloom filter data object stored there. Ex. 1004, ¶¶66–68. Indeed, Mankins teaches that its “intrusion detection system 100 may be implemented *in a distributed fashion over a network of computing devices.*” Ex. 1008, 4:29–30 (emphasis added); *see also id.*, 4:50–52 (“In many embodiments, the computer 110 is in the form of one or more *network servers.*” (emphasis added)). Likewise, Mankins describes “network embodiments of the intrusion detection system” wherein “multiple network devices are configured to work together to identify intrusion attacks,” and the advantages of such a configuration:

[W]hen several network devices generate separate Bloom filter data objects 180, are monitored together, and are subject only to attacks that affect each of the network devices within a small amount of time, it is likely that at least one of the network devices would detect a false positive that is produced by another device. Thus, the sizes of the Bloom filter data objects 180 may be reduced because the different network devices can work together to identify false positives.

Id., 10:53–62. Mankins indicates that such an arrangement would have been a

“[c]onventional system-call-based intrusions detection technique[]” where “tracked system-call sequences are inserted into a *database* to form a profile of system-call sequences that are considered to be normal operations of the particular program,” and where “the monitored system-call sequences” that are “tracked in the OS kernel” are compared “with data of a *profile database stored outside of the OS*.” *Id.*, 1:60– 22 (emphases added).

Alternatively, it would have been obvious to a POSITA that the combined system could log the sequence of system calls made by the object in the kernel and maintain the Bloom filter at each remote computer, as in the preferred embodiment in Mankins. Ex. 1004, ¶67. Mankins explains that because “the amount of memory required by a particular Bloom filter data object 180 (*e.g.*, 3 KB) will be a small memory outlay,” it “may easily fit in the process structure of the OS kernel 120.” Ex. 1008, 10:48–51.

In either implementation, if the system does not identify malware based on the behavioral data received from remote “computers 500,” Mankins’s process would begin. *Id.*, ¶68. Whether at each remote computer or at the server, the system would follow Mankins’ teaching to: (i) use the monitored behavior containing the “normal” system call patterns to generate a (or update an existing) Bloom filter data object for that program and (ii) monitor the behavior of the program (*e.g.*, by monitoring system call sequences) and compare the behavior to the Bloom filter

data object directly at the remote computer, or, alternatively, transmit the information for comparison against the Bloom filter data object stored at the server.

Id.

Adding these additional steps from Mankins would optimize the system's ability to identify unknown malware, and, as explained by Dr. Newman, the benefits of this addition would not be offset by any cost. *Id.*, ¶69. Ultimately, a POSITA would have understood that the chosen configuration is an obvious matter of system design. *Id.*, ¶70.

3. Analysis

a. Independent Claim 1

[1P]: A method of classifying a computer object as malware, the method comprising:

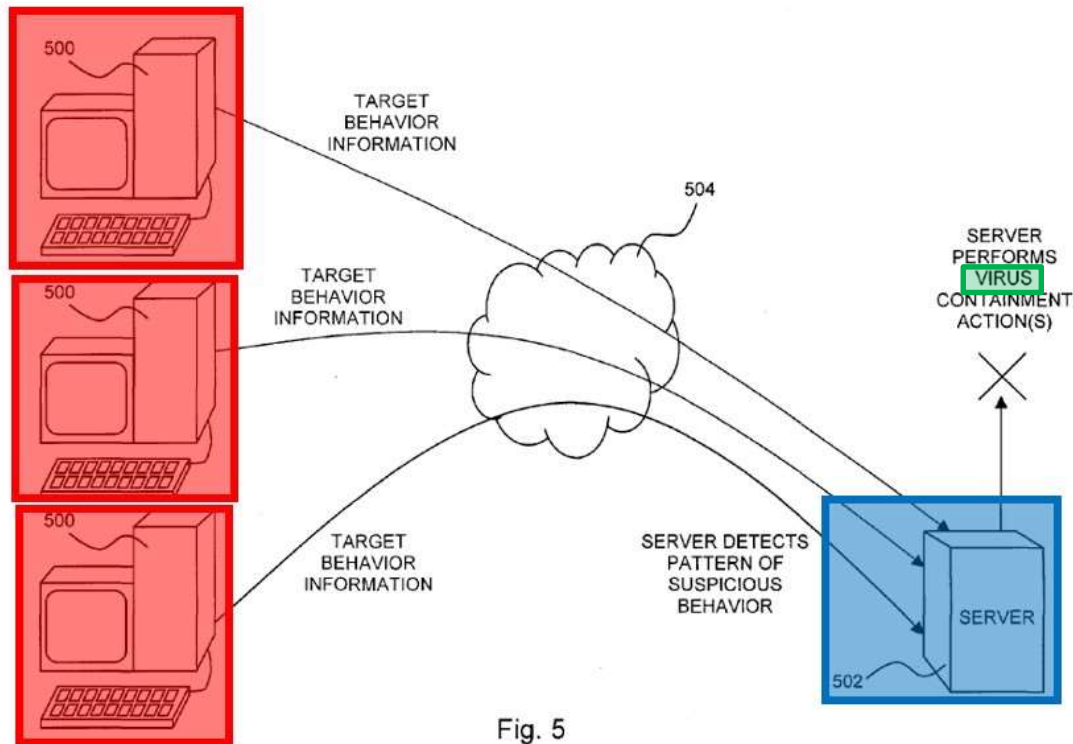
To the extent the preamble of claim 1 is limiting, Halperin discloses or renders obvious [1P]. Ex. 1004 ¶¶72–73. The '250 Patent is directed to a method of classifying a computer object as malware. Halperin discloses a system and method of “detection and prevention of malicious computer programs.” Ex. 1006, [0002]. Halperin discloses “malicious software detection algorithms,” *id.* [0074], *i.e.*, methods for classifying software (“a computer object”) as malware, such that “when malicious software has been identified . . . it is possible to isolate the mechanism that has been spreading the malicious software,” *id.* [0073]. Thus, a POSITA would understand Halperin's disclosure of methods for detecting and

identifying software as “malicious” as disclosing a “method of classifying a computer object as malware.”

[1A]: at a base computer, receiving data about a computer object from each of plural remote computers on which the object or similar objects are stored, the data including information about the behaviour of the object running on one or more remote computers;

This limitation includes multiple parts, each of which is disclosed or rendered obvious by Halperin. Ex. 1004 ¶¶74–77.

Base computer. Halperin discloses a base computer, for example, “**server 502**” depicted in Figure 5. Ex. 1006, [0047], Fig. 5. Halperin further discloses that a base computer (*i.e.*, “**server 502**”) receives data about a computer object—for example, “target behavior information” associated with a potential “**virus**”—from each of plural remote computers (*i.e.*, **computers 500**). *Id.*, [0036]–[0037], [0047].



Id., Fig. 5.

Receiving data about a computer object from each of plural remote computers on which the object or similar objects are stored. Halperin explains that “target behavior profiles are defined for **computers 500**,” and that “[e]ach target behavior profile describes behavior that should be the subject of correlation analysis.” *Id.*, [0038]. Thus, **computers 500** can “detect associated target behavior and notify **server 502** accordingly” and **server 502** “collect[s] information regarding target behavior detected at two or more of **computers 500**.” *Id.*, [0047]; Ex. 1004, ¶76. Further, with reference to “the embodiments described hereinabove” (including the Figure 5 embodiment), Halperin discloses “each computer” logging

“any predefined suspicious event that takes place” and discloses “[t]hat information is logged in a central database,” *i.e.*, at the **server 502**. Ex. 1006, [0086]. A POSITA would have understood the “target behavior” and the “predefined suspicious event[s]” detected by the computers and received by the **server 502** to be data about a computer object (*i.e.*, a potential **virus**) from each of plural remote **computers 500** on which the object or similar computer objects stored, as the purpose of Halperin’s method is to determine whether “a computer virus may have infected those computers.” *Id.*, [0047]; Ex. 1004, ¶77.

The data including information about the behaviour of the object running on one or more remote computers. A POSITA would have understood Halperin’s “information regarding target behavior detected” at the **computers 500** to “includ[e] information about the behaviour of the object,” (*i.e.*, a potential **virus**) running on one or more remote computers. Thus, a POSITA would have understood Halperin to disclose or render obvious all of the parts of [1A]. *Id.*

[1B]: determining in the base computer whether the data about the computer object received from the plural computers indicates that the computer object is malware;

Halperin discloses or renders obvious [1B]. Ex. 1004 ¶¶78–79. Once Halperin’s base computer (e.g., server 502) correlates the “target behavior,” which is the claimed “data about the computer object” received from remote computers 500, the base computer “determine[s] whether the correlated target behavior

corresponds to a predefined suspicious behavior pattern of target behavior as an indication that a computer virus may have infected those computers.” Ex. 1006, [0047]. An “indication that a computer virus may have infected those computers” is an example of the claimed indication “that the computer object is malware.” Thus, a POSITA would have understood Halperin to disclose or render obvious [1B]. Ex. 1004, ¶79.

[1C]: classifying the computer object as malware when the data indicates that the computer object is malware;

Halperin discloses or renders obvious [1C]. Ex. 1004, ¶¶80–82. Halperin explains that if the base computer determines “the correlated target behavior corresponds to a predefined suspicious behavior pattern of target behavior,” it classifies the computer object as malware (*e.g.*, a “computer virus”). *See* Ex. 1006, [0047], [0057], [0073]–[0074]; *see also id.*, [0087] (“The proposed system works in two phases to identify malicious code (from here on referred to as ‘viruses.’)”; *id.* [0061] (“[T]he identification of virus activity may include automatic identification of suspicious activity by a server or a combination of automatic identification and a notification of a system operator and approval by that operator that the suspicious activity is truly a virus, before notifying other servers.”). Ex. 1004, ¶81. Thus, Halperin discloses classifying the computer object as malware when the data indicates that the computer object is malware. A POSITA would have

further understood from Halperin that when Halperin takes actions responsive to the detection of a virus, such as to “activate a virus alarm” or “initiate virus containment actions,” Ex. 1006 [0090], it is because the system has classified the computer object as malware. Ex. 1004, ¶82.

[1D]: when the determining does not indicate that the computer object is malware, initially classifying the computer object as not malware;

Both Halperin and Mankins disclose or render obvious [1D]. Ex. 1004, ¶¶83–86. As explained with respect to [1B] and [1C], if Halperin’s base computer determines that “the correlated target behavior corresponds to a predefined suspicious behavior pattern of target behavior,” it classifies the computer object as malware (*e.g.*, a “computer virus”) and the system takes responsive action, such as “virus containment actions.” *See* Ex. 1006, [0047], [0057], [0061], [0073]–[0074], [0090]; Ex. 1004, ¶84. Conversely, when the correlated data *does not* correspond to “a suspicious behavior pattern,” no virus containment actions are initiated. Ex. 1004, ¶85; *cf.* Ex. 1006, [0047]. A POSITA would have understood that this element is simply the converse of the prior element, and so it would have been obvious from Halperin’s disclosure of classifying an object as malware, that the same object would be classified as not malware if the determining step generated the opposite result. Ex. 1004, ¶85.

Mankins likewise discloses that “normal behavior” of a particular computer program “generally refers to observable and acceptable behavior characteristics expected of networks when not under attack,” Ex. 1008, 1:40–43, for example, by running the program “under training conditions” and “monitoring system-call behavior at the OS kernel.” Ex. 1008, 6:63–7:7. Once the “normal behavior” is defined, a “Bloom filter data object” is built, and the program is allowed to run so long as “a match exists” for the program’s behavior in the Bloom filter data object. *Id.*, 11:30–37, 11:56–12:14, Fig. 3. A POSITA would have understood Mankins’ disclosure of running a computer program to obtain data on *normal* behavior, and then allowing the program to run so long as its behavior matches the predefined normal behavior to be the same as “initially classifying the computer object as not malware” as required by [1D]. Ex. 1004, ¶86.

[1E]: automatically generating a mask for the computer object that defines acceptable behaviour for the computer object, wherein the mask is generated in accordance with normal behaviour of the object determined from said received data;

Mankins discloses or renders obvious [1E]. Ex. 1004, ¶¶87–97.

In Halperin, “target behavior profiles” of acceptable and/or unacceptable behavior are “defined,” *see, e.g.*, Ex. 1006, [0047] (“[O]ne or more target behavior profiles are defined for computers 500.”), but Halperin does not state explicitly how the “target behavior profiles” are generated. Ex. 1004, ¶88.

Mankins “uses a Bloom filter to generate a Bloom filter data object containing data representative of normal system-call behavior” of a computer program. Ex. 1008, 3:40–43. Mankins explains that the “predefined Bloom filter data is representative of ‘normal’ computer program behavior that is expected during operation of the computer programs when not subject to intrusion attempts.” *Id.*, 3:26–30. Like the mask in the ’250 Patent, the “Bloom filter data object 180 is built from the monitored behavior” of the computer object. It would have been obvious to a POSITA that Mankins’s “Bloom filter data object,” which is “representative of ‘normal’ computer program behavior” (*id.*, 3:40–43), is an example of the ’250 Patent’s “mask for said object that defines acceptable behaviour for the object” (Ex. 1001 at 4:22–23). Ex. 1004, ¶89.

Moreover, Mankins discloses that its “intrusion detection system 100 may be configured to *automatically generate* Bloom filter data objects for computer programs 130 and then automatically use the Bloom filter data objects 180 to monitor system-call behavior.” Ex. 1008, 8:8–13 (emphasis added); *see also id.*, 11:24–45 (describing “an example of a process for defining normal system-call behavior under training conditions” and noting that the process “may be performed automatically to generate the Bloom filter data object 180 in an automated fashion, without requiring user intervention”).

Mankins discloses an exemplary configuration of its intrusion detection system implemented on a single computer 110 at the operating system kernel level. Thus, it would have been obvious to a POSITA to automatically generate Bloom filter data objects directly at the remote “computers 500” in the combined Halperin/Mankins system. *Id.*, ¶¶90–91. Mankins also discloses that its “intrusion detection system 100 may be implemented *in a distributed fashion over a network of computing devices.*” Ex. 1008, 4:29–31 (emphasis added); *see also id.*, 4:50–52 (“In many embodiments, the computer 110 is in the form of one or more *network servers.*” (emphasis added)). Accordingly, it would have also been obvious to a POSITA to implement Mankins’s intrusion detection system in a distributed fashion, such as by using the system architecture described in Halperin and automatically generating Bloom filter data objects at the “server 502.” *See* Ex. 1006, [0047], Fig. 5; Ex. 1004, ¶92.

Further, Mankins generates the Bloom filter data object (e.g., the mask) in accordance with the normal behavior of the object. Mankins’s Figure 2, reproduced below, is a flowchart that illustrates an example process for defining “normal system-call behavior” (*i.e.*, “normal” or “acceptable” behavior) for a computer program (*i.e.*, a “computer object”) under training conditions:

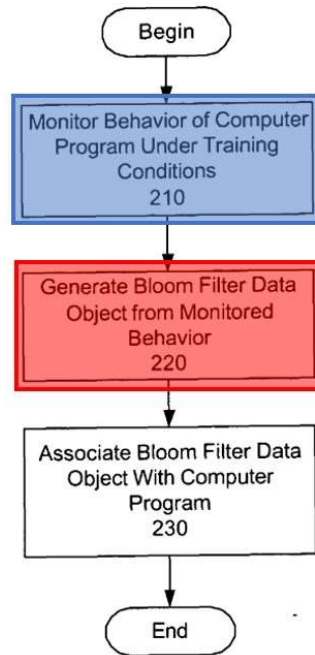


FIG. 2

Ex. 1008, 11:19–23; Fig. 2. For example, Mankins describes “monitoring behavior of a computer program under training conditions at **step 210**,” which may be performed by “monitoring system-call behavior at the OS kernel 120 when the computer program 130 is not subject to intrusion attacks.” *Id.*, 11:24–29. At **step 220**, “a profile Bloom filter data object 180 is built from the monitored behavior,” which can “contain data representative of predefined normal behavior” that “is defined in the form of . . . sequences of system calls.” *Id.*, 11:30–36. The Bloom filter data object is then “associated with the computer program.” *Id.*, 11:37–38; *see* Ex. 1004, ¶93.

As discussed above, a POSITA would have been motivated to incorporate the

Bloom filter mask of Mankins into the detection system of Halperin. Halperin's behavioral approach to virus detection determines that a computer object is malware if the "correlated target behavior corresponds to a predefined suspicious behavior pattern of target behavior," Ex. 1006, [0047], and Halperin discloses that another existing technique was to "characteriz[e] legitimate software behavior and then identify[] any deviation from such behavior." *Id.*, [0006]. It would have been obvious to a POSITA to use Mankins's Bloom filter mask with Halperin to identify predefined suspicious behavior patterns and determine an object is malware if that computer object performs actions beyond its "normal (i.e. expected) system-call behavior" (Ex. 1008, 3:37). This would have been a simple substitution of one known behavior-based approach to malware behavior detection for another. Ex. 1004, ¶94.

Indeed, a POSITA would have recognized that using Mankins's mask approach in a distributed system such as Halperin's would improve the system by reducing false positives. Ex. 1004, ¶95. Halperin discloses that because "computer user behavior is quite dynamic and tends to vary over time and between different users," the "application of heuristic techniques thus often results in a false alarm." Ex. 1006, [0006]. That is why Halperin discloses a distributed approach using information collected from multiple remote computers. Mankins likewise acknowledges that a distributed approach can correct for false positives: "In

particular, when several network devices generate separate Bloom filter data objects 180, are monitored together, and are subject only to attacks that affect each of the network devices within a small amount of time, it is likely that at least one of the network devices would detect a false positive that is produced by another device.” Ex. 1008, 10:53–62. Thus, a POSITA would have understood that implementation of Mankins’s Bloom filter mask in a *distributed system* like that of Halperin would have *reduced* the occurrence of false positives because the Bloom filter data objects would be generated from behavioral data collected from different computers having different users, and at different times. Ex. 1004, ¶96.

Thus, because Mankins’s Bloom filter mask was a known technique for improving virus detection software, adding it to Halperin’s system would have been no more than using a known technique to improve a similar device in a similar way. Ex. 1004, ¶97. For the same reason, a POSITA would have had at least a reasonable expectation of successfully modifying Halperin in view of Mankins in the manner described above.

**[1F]: running said object on at least one of the remote computers;
automatically monitoring operation of the object on the at least one of the
remote computers;**

Mankins discloses or renders obvious [1F]. Ex. 1004, ¶¶98–101. Mankins’s Figure 3, reproduced below, is a flowchart that illustrates “an example process for detecting intrusion attempts by monitoring system-call behavior”:

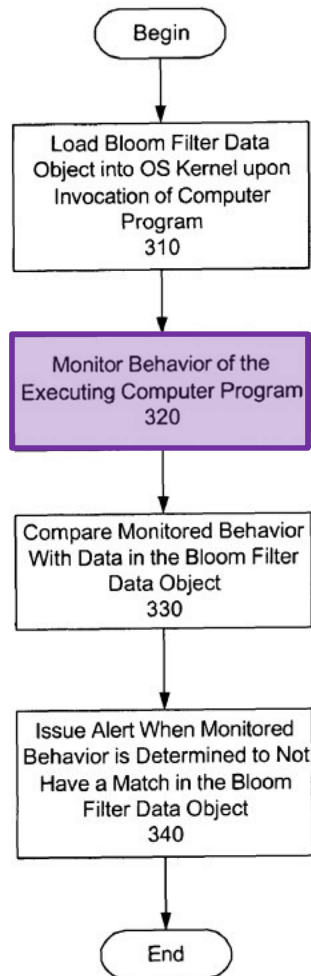


FIG. 3

Ex. 1008, 11:50–52, Fig. 3. At **step 320** of Figure 3, “behavior of the computer program is monitored as it is executed by the computer 110.” *Id.*, 11:63–65. Mankins teaches “*automatically* us[ing] the Bloom filter data objects 180 to *monitor* system-call behavior in the OS kernel 120, thereby allowing intrusions to be detected in real time.” *Id.*, 8:8–13 (emphases added). As explained above, the automatically monitored behavior may be “system-call behavior at the OS kernel.” *Id.*, 11:65–66; *see also id.*, 6:3–56 (describing “the monitoring subsystem 140” that

may “use [a] rolling log of system calls to monitor system-call sequences occurring in the OS kernel”). It would have been obvious to a POSITA that an object that is “execut[ing]” or producing “system-call sequences” is running. Ex. 1004, ¶99.

Further, Mankins describes one embodiment in which “several network devices generate separate Bloom filter data objects 180, [and] are monitored together.” Ex. 1008, 10:57–58. It would have been obvious to a POSITA that an object whose behavior is being monitored is necessarily “running.” Ex. 1004, ¶100. In addition, Mankins explains that “the intrusion detection system 100 is prepared to detect intrusion attempts by monitoring the behavior of the computer program 130 when it is executed.” Ex. 1008, 11:46–50.

Moreover, in this distributed embodiment, each of “several network devices [*i.e.*, plural remote computers] generate separate Bloom filter data objects 180.” *Id.*, 10:57–58. Mankins explains that the purpose of this arrangement is to “reduce[]” “the size of the Bloom filter data object 180.” *Id.*, 10:53–56. This reduction occurs because each remote computer has its own Bloom filter data object. Further, this arrangement requires the program to be run on the remote computers, where its behavior will be compared to their computer’s Bloom filter data object for that program. Ex. 1004, ¶100.

As explained above, it would have been obvious to a POSITA to incorporate Mankins’s Bloom filter data object at either a remote computer or a base computer

as part of the system architecture described in Halperin (including a “server 502” and “computers 500” that communicate over a “network 504”). See Ex. 1004, ¶101. A POSITA would have understood that a combined Halperin/Mankins system could log the sequence of system calls made by the object in the kernel and maintain the Bloom filter at each remote computer, as in the preferred embodiment in Mankins. Alternatively, it would have been an obvious modification to have collected system call sequences at the remote “computers 500,” and transmitted the sequences as part of the data that is received at the “server 502,” and to have compared them against a Bloom filter data object stored at the server. Moreover, this choice of configuration would have been nothing more than an obvious matter of system design to a POSITA. *Id.*

[1G]: allowing the computer object to continue to run when behaviour of the computer object is permitted by the mask;

Mankins discloses or renders obvious [1G]. Ex. 1004 ¶¶102–05. At [step 330](#) of Mankins’s Figure 3, reproduced below, “the monitored behavior is compared with data in the Bloom filter data object,” and “the monitoring subsystem” may “determine whether a match exists for any particular set of monitored behavior (e.g., a system-call sequence).”

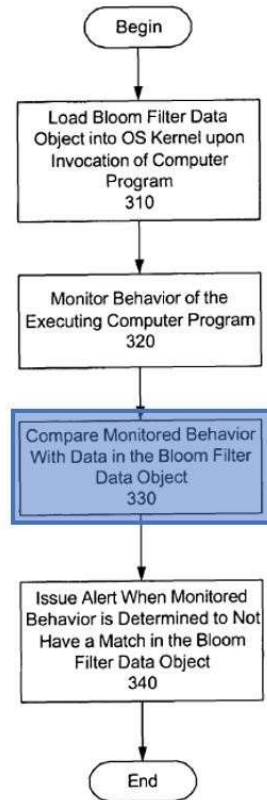


FIG. 3

Ex. 1008, 12:1–6, Fig. 3; *see also id.*, 6:37–39 (“The monitoring sub-system 150 may then compare this system-call sequence with the predefined normal system-call sequences.”). As with conventional system-call based intrusion detection techniques, “[a]s long as the monitored system-call sequences have a match” in the Bloom filter data object, the “operation is considered to be normal” and the object or program will be permitted to continue to run. *Id.*, 1:60–63, 2:8–10; *cf. id.*, 6:39–41 (“If a match is not found, the observed system-call sequence is determined to be anomalous and may indicate an intrusion attempt.”); Ex. 1004, ¶103.

As previously discussed, a POSITA would have been motivated to combine Mankins's Bloom filter intrusion detection system with Halperin's distributed/networked malware detection system. Both relate to behavior-based approaches for detecting unknown malware, and a person of ordinary skill would be motivated to employ various known methods of behavior-based malware detection software. Ex. 1004, ¶104. A POSITA would have had a reasonable expectation of successfully modifying Halperin in view of Mankins in the manner described above, as Halperin already performs a similar behavioral analysis function. And Halperin recognizes that "characterizing legitimate software behavior and then identifying any deviation from such behavior," which is the technique described in Mankins, was an existing malware detection approach. Ex. 1006, [0006].

It would thus have been obvious to a POSITA to modify Halperin's behavior step with the teachings of Mankins's Bloom filter data object. For the reasons stated above, Halperin, while utilizing Mankins's Bloom filter technique, would have allowed a computer object to run when the actual monitored behavior is permitted by the mask. Ex. 1004, ¶105.

[1H]: disallowing the computer object to run when the actual monitored behaviour of the computer object extends beyond that permitted by the mask; and,

Halperin and Mankins disclose or render obvious [1H]. Ex. 1004, ¶¶106–08.

Mankins explains that “[i]f a match [between the Bloom filter data object 180 and the actions of the computer object] is not found, the observed system-call sequence is determined to be anomalous and may indicate an intrusion attempt.” Ex. 1008, 6:39–41. As illustrated by step 340 of Mankins’s Figure 3, “an alert is issued in response to a determination that the Bloom filter data object 180 does not contain a match for the monitor behavior,” and “[a]ny appropriate predefined action may be performed in response to the alert.” *Id.*, 12:7–14; *see id.*, 11:8–11 (“[T]he report subsystem 170 may be configured to automatically invoke *predetermined actions designed to protect the computer.*” (emphasis added)).

As explained by Petitioners’ expert, it would have been obvious to a POSITA to disallow the program that performed the system-call sequence flagged as an intrusion attempt from running. Ex. 1004, ¶108. Indeed, Halperin discloses that if suspicious behavior is detected, “the system can . . . initiate virus containment actions.” Ex. 1006, [0090]. Thus, it would have been obvious to a POSITA to augment Halperin’s disclosure of “virus containment actions” with Mankins’s disclosure of “predetermined actions designed to protect the computer,” such as disallowing a program from running when its behavior does not match that in the Bloom filter data object. Ex. 1004, ¶108.

[1I]: reclassifying the computer object as malware when the actual monitored behaviour extends beyond that permitted by the mask.

Halperin and Mankins disclose or render obvious [1I]. Ex. 1004, ¶¶109–11. As explained above with respect to element [1G], Mankins teaches that after the comparison of the monitored system-call behavior with data in the Bloom filter data object, “[i]f a match is not found, the observed system-call sequence is determined to be anomalous and may indicate an intrusion attempt.” Ex. 1008, 6:39–41. And the system may issue an alert, and take “predetermined actions designed to protect the computer.” *Id.*, 11:8–11. As explained by Petitioners’ expert, a POSITA would understand that taking actions to protect the computer indicates the computer object has been reclassified as malware. Ex. 1004, ¶¶110–11. At a minimum, it would have been obvious to reclassify the program that performed the system-call sequence flagged as an intrusion attempt as malware. *Id.*, ¶111.

Again, Halperin discloses that if suspicious behavior is detected, “the system can . . . initiate virus containment actions.” Ex. 1006, [0090]. A POSITA would have understood that Halperin’s disclosure of “virus containment actions” and Mankins’s disclosure of “predetermined actions designed to protect the computer” to include reclassifying the computer object as malware. Ex. 1004, ¶112.

b. Independent Claim 13

Claim 13 is materially identical to claim 1, except that claim 13 is an apparatus version. Halperin in view of Mankins, thus, renders claim 13 obvious for the same reasons as claim 1. Ex. 1004, ¶113.

c. Dependent Claims 2 & 14

Claims 2 and 14 depend from claims 1 and 13, respectively. The combination of Halperin and Mankins renders claims 1 and 13 obvious for the reasons discussed above. Claims 2 and 14 further require “wherein the data about the computer object that is sent from the plural remote computers to the base computer includes one or more of: . . . the current name of the computer object.”

Halperin discloses or renders obvious this element. Ex. 1004, ¶114. In one embodiment, Halperin identifies malware by detecting “computers in the network sending the same file attachment via a message in the last 15 minutes.” Ex. 1006, [0054]. A common way to determine if two files attachments are the same is to compare file names. Ex. 1004, ¶115. A POSITA would have understood that the current name of the computer objects would be sent to the base computer as part of determining whether they are the “same file attachment.” At a minimum, it would have been obvious to a POSITA to include the current name of the computer object to aid in that determination. *Id.* Halperin in view of Mankins thus renders claims 2 and 14 obvious. *Id.*, ¶116.

d. Dependent Claims 12 & 24

Claims 12 and 24 depend from claims 1 and 13, respectively. The combination of Halperin and Mankins renders claims 1 and 13 obvious for the reasons discussed above. Claims 12 and 24 further require “comprising if the monitored behaviour of the object running on the remote computer exhibits nonpermitted behaviour that extends beyond that permitted by the mask, comparing at the base computer the non-permitted behaviour with the behaviour of other objects and if said behaviour is known to be not malicious for said other objects, allowing said behaviour for the object and modifying the mask to allow that behaviour for that object in future.”

Mankins discloses or renders obvious this element. Ex. 1004, ¶117. Mankins discloses that upon “detection of anomalous program behavior” (*i.e.*, “non-permitted behaviour that extends beyond that permitted by the mask”), its report subsystem may issue “an alert and *allow execution of detected anomalous behavior to proceed.*” Ex. 1006, 11:4–7 (emphasis added). As explained by Petitioners’ expert, it would have been obvious to limit the allowance of execution of detected anomalous behavior only if that behavior was known to not be malicious for another object, and to modify the Bloom filter data object corresponding to the object exhibiting anomalous behavior to include that behavior. Ex. 1004, ¶118–19. Moreover, it would have been obvious for this to be done at the base computer

because, as disclosed by Halperin (*see* [1A] above), the base computer receives data about the object from each of the remote computers. *Id.* The combination of Halperin and Mankins therefore renders claims 12 and 24 obvious. Ex. 1004, ¶120.

B. Ground 2: Claims 1–30 Are Rendered Obvious Over Halperin in view of Mankins and Kirby

As described below, the combination of Halperin, Mankins, and Kirby, in view of the knowledge of POSITA, render claims 1–30 obvious.

1. Prior Art Overview

a. Kirby

Kirby—like Halperin and Mankins—discloses a method and apparatus for virus detection. Ex. 1007, Abstract. Kirby is directed to “overcome[] the above described shortcomings of the signature approach alone by providing an integrated approach which applies features of both the behavioral and signature approach in a parallel and complementary manner.” *Id.*, 2:56–62. Kirby’s behavior-based approach monitors actions performed by computer objects “to determine if the detected actions correspond to the suspicious behavioral actions of the behavior heuristic 112.” *Id.*, 10:54–60. If they do, the object is classified as “undesirable activity” (malware). *Id.*, 10:63–11:6; *see also id.*, 15:7–20. In addition, Kirby discloses that when the behavioral actions indicate an object is malware, the computer on which the object is found creates a “hash signature” of the object,

“therefore enabling a signature based (immediate) prevention mechanism against subsequent occurrences of the malware on this or other computers.” *Id.*, 15:15–20.

2. Motivation to Combine and Reasonable Expectation of Success

Both Halperin and Kirby are directed to malware detection systems and processes that use behavioral monitoring to address the “day zero” problem of traditional signature approaches. *See* Ex. 1004, ¶122; Ex. 1006, [0005], [0007] (“[W]hile virus scanners can effectively detect known computer viruses, they generally cannot reliably detect unknown computer viruses. . . .The present invention seeks to provide for the detection and containment of malicious computer programs that overcomes disadvantages of the prior art.”); Ex. 1007, 1:62–2:7, 2:12–25 (“Embodiments of the invention provide a behavioral approach to virus detection and processing of other malicious processing activity, that in contrast to conventional techniques, exhibits a substantial departure from the conventional day Zero problem.”). Kirby focuses on combining behavioral and signature matching approaches. Ex. 1007, 2:56–62. And, as described above, Mankins is directed to a specific technique for behavioral monitoring, *i.e.*, comparing tracked system call sequences with expected system call behavior data stored in a Bloom filter data object. *See* Ex. 1008, 3:18–57.

A POSITA would have been motivated to combine the teachings of Halperin, Mankins, and Kirby for multiple reasons, including to produce the obvious, beneficial, and predictable result of incorporating Kirby’s signature matching techniques to aid in the classification of objects as either malware or non-malware. Ex. 1004, ¶123. Kirby would have been a natural place for a POSITA to look because Kirby’s system of classifying an object as malware using integrated behavioral analysis and signature matching was known as an effective way to detect malware that would speed up the detection process. *Id.*

Further, a POSITA would have been motivated to use Kirby’s teachings regarding identifying as malware objects that “touch” malware. *Id.*, ¶124. As Halperin discloses, viruses are “designed to infiltrate computer files and other sensitive areas on the computer.” Ex. 1006, [0003]. Thus, viruses modify existing programs or files to include viral code. Ex. 1004, ¶124. A POSITA would understand that, as a result, an object that is modified by a virus is likely to be malware itself. *Id.* Thus, a POSITA would have been motivated to modify Halperin and Mankins by incorporating Kirby’s teachings on classifying as malware objects that “touch,” or have interacted with known malware. *Id.*, ¶¶124–25; *see also* Ex. 1007, 5:48–5.

3. Analysis

As demonstrated in Ground 1, Halperin and Mankins render obvious all the limitations of claims 1, 2, 12–14, and 24. To the extent that Patent Owner argues that the combination of Halpern and Mankins does not sufficiently teach or suggest any element of claims 1, 2, 12–14, and/or 24, Kirby discloses or renders obvious these features. As explained above, a POSITA would have been motivated to combine the teachings of Kirby with Halperin and Mankins and would have reasonably expected this combination to be successful. *See* Section IX.B.2, *supra*. Thus, the combination of Halperin, Mankins, and Kirby renders obvious these claims, and further renders obvious independent claim 25 and the remaining dependent claims.

Except for the specific limitations and claims discussed below, Petitioner incorporates by reference the analysis of the combination of Halperin and Mankins in Ground 1, above.

a. Independent Claim 1

[1D]: when the determining does not indicate that the computer object is malware, initially classifying the computer object as not malware;

As explained above, both Halperin and Mankins disclose or render obvious [1D]. Kirby also discloses or renders obvious this limitation. Ex. 1004, ¶¶126–28. Kirby explains that “[i]f the pattern matcher concludes that a content based check against the baseline of previously stored realtime signatures in the signature

database indicates a *benign* transmission, then the behavior monitor evaluates the actions performed by, on behalf of or as a result of the incoming transmission, based on the operations exhibited.” Ex. 1007, 7:65–8:4 (emphasis added). A POSITA would have understood that by determining that the object is benign, Kirby is classifying that object as “not malware.” Ex. 1004, ¶128.

[1F]: running said object on at least one of the remote computers; automatically monitoring operation of the object on the at least one of the remote computers;

As explained above, Mankins discloses or renders obvious [1F]. Kirby also discloses or renders obvious this limitation. Ex. 1004, ¶¶129–30. Kirby allows objects that have not yet been identified as malware to operate on remote computer while monitoring the actions of that object from a base computer. In one configuration of Kirby, “the security agent 110 is configured to intercept and *monitor* a variety of subject (process or thread) *actions*, such as file accesses, I/O port accesses, and other security pertinent operations. Therefore, the security agent 110 is empowered to scrutinize relevant operations on the host node 120.” Ex. 1007, 11:51–56 (emphasis added).

[1I]: reclassifying the computer object as malware when the actual monitored behaviour extends beyond that permitted by the mask.

As explained above, Halperin and Mankins disclose or render obvious [1I]. Kirby additionally discloses or renders obvious this limitation. Ex. 1004, ¶¶131–

32. As explained with respect to element [1D], Kirby classifies a computer object as not malware (specifically, “benign”) if the computer object passes the signature check. Kirby explains that the “signature check . . . generally occurs immediately upon introduction of a new entity (e.g., file or object).” Ex. 1006, 14:30–33. Kirby continues “the behavioral monitoring and prevention, which occurs as an ongoing monitoring effort.” *Id.* The behavioral monitoring step is therefore performed *after* the computer object is initially classified as not malware (benign). Thus, a POSITA would have understood that when the behavioral monitoring step determines the computer object is malware, Kirby reclassifies that computer object as malware. Ex. 1004, ¶132.

b. Independent Claim 13

Claim 13 is materially identical to claim 1, except that claim 13 is an apparatus version. Halperin in view of Mankins and Kirby, thus, renders claim 13 obvious for the same reasons as claim 1. Ex. 1004, ¶133.

c. Dependent Claims 2 & 14

As explained above, Halperin in view of Mankins renders claims 2 and 14 obvious. For the same reasons, claims 2 and 14 are obvious over the combination of Halperin, Mankins, and Kirby. *Id.*, ¶134.

d. Dependent Claims 3 & 15

Claims 3 and 15 depend from claims 1 and 13, respectively. The combination of Halperin, Kirby, and Mankins renders claims 1 and 13 obvious for the reasons discussed above. Claims 3 and 15 further require “wherein the determining identifies relationships between the object and other objects.”

Kirby discloses or renders obvious this additional element. Ex. 1004, ¶135. Kirby teaches “an incoming email attachment file may be deemed as ‘untrusted.’ A process reading the file and writing another file creates another untrusted file.” Ex. 1007, 5:49–50. Kirby continues: “any process or file which ‘touches’ an untrusted file may ‘inherit’ an untrustworthy trait, which may then be passed on until the pattern indicates a correspondence with undesirable, and hence, preventable, behavior.” *Id.*, 5:52–55. Here, Kirby explains that one way to identify potentially malicious code is by determining that code is *related to* malicious code. Ex. 1004, ¶136. Thus, a POSITA would understand Kirby to teach determining that a computer object is malware by identifying relationships between the object and other objects. *Id.*, ¶136–37.

e. Dependent Claims 4 & 16

Claims 4 and 16 depend from claims 3 and 15, respectively. The combination of Halperin, Kirby, and Mankins renders claims 3 and 15 obvious for the reasons discussed above. Claims 4 and 16 further require “wherein if at least one other

object to which said object is related is classed as malware, then classifying said object as malware.”

Kirby discloses or renders obvious this element. Ex. 1004, ¶138. As explained above with respect to claims 3 and 15, Kirby discloses that “any process or file which ‘touches’ an untrusted file may ‘inherit’ an untrustworthy trait, which may then be passed on until the pattern indicates a correspondence with undesirable, and hence, preventable, behavior.” Ex. 1007, 5:52–55. Kirby further discloses that the signature is computed from a malware object that “is deemed operable to indicate subsequent undesirable activity,” for example, by other objects that perform the same suspicious behavioral actions. Ex. 1007, 10:61–11:6. Kirby continues “the same *or similar* object appearing elsewhere is likely indicative of the same virus or undesirable transmission.” *Id.*, 12:27–30 (emphasis added). “In other words, any process or file which ‘*touches*’ an untrusted file may ‘inherit’ an untrustworthy trait.” *Id.*, 5:52–53 (emphasis added). A POSITA reading Kirby would understand *similar* objects to include objects that “touch” or are related objects, such as the child or parent of an “untrusted” file. The computed signature of “the same or similar object appearing elsewhere” would likewise be classified as malware based on the previous classification. *See* Ex. 1004, ¶¶139–40. Thus, Halperin in view of Mankins and Kirby renders claims 4 and 16 obvious. Ex. 1004, ¶140–41.

f. Dependent Claims 5 & 17

Claims 5 and 17 depend from claims 3 and 15, respectively. The combination of Halperin, Kirby, and Mankins renders claims 3 and 15 obvious for the reasons discussed above. Claims 5 and 17 further require “wherein said other objects include the object or similar objects stored on at least some of the remote computers.”

Halperin and Kirby disclose or render obvious this element. Ex. 1004, ¶142. Halperin discloses, for example, identifying “[a] certain percentage of the computers in the network sending file attachments via one or more messages in the last hour where each of the files includes the same string of bits.” Ex. 1006, [0047], [0055]. A POSITA would understand that “the same string of bits” refers to a string of bits in the virus (“the object”) itself and would further understand that the files are stored on at least some of the remote computers. Ex. 1004, ¶143. Moreover, as explained with respect to claims 3 and 15, Kirby teaches that one way to identify potentially malicious code is by determining that code is related to malicious code found on at least some of the remote computers. Ex. 1007, 5:49–50; Ex. 1004, ¶144. Kirby further teaches identifying an object as malware if it is determined that the same or similar object is malware. *Id.*, 12:27–30 (“[T]he same or similar object appearing elsewhere is likely indicative of the same virus or undesirable transmission.” (emphasis added)). A POSITA would therefore understand that the

related objects include the object or similar objects stored on at least some of the remote computers. Ex. 1004, ¶144.

Halperin in view and Mankins and Kirby therefore render claims 5 and 17 obvious. Ex. 1004, ¶¶144–45.

g. Dependent Claims 6 & 18

Claims 6 and 18 depend from claims 3 and 15, respectively. The combination of Halperin, Kirby, and Mankins renders claims 3 and 15 obvious for the reasons discussed above. Claims 6 and 18 further require “wherein said other objects include other objects that are parent objects or child objects or otherwise process-related objects to said object.”

Kirby discloses or renders obvious this element. Ex. 1004, ¶146. As explained above, Kirby teaches that a “process reading the [malicious] file and writing another file creates another untrusted file.” Ex. 1007, 5:49–50. It would have been obvious to a POSITA that “another file” that is created by a “process reading the [malicious] file” is a child object of a malicious object. Ex. 1004, ¶¶147–48. In addition, Kirby discloses that “any process or file which ‘touches’ an untrusted file may ‘inherit’ an untrustworthy trait, which may then be passed on until the pattern indicates a correspondence with undesirable, and hence, preventable, behavior.” Ex. 1007, 5:52–55. Again, it would have been obvious to a POSITA that a file that is touched could be a parent or child object. Ex. 1004,

¶149. Halperin in view of Kirby and Mankins consequently renders claims 6 and 18 obvious. *Id.*, ¶¶149–50.

h. Dependent Claims 7 & 19

Claims 7 and 19 depend from claims 1 and 13, respectively. The combination of Halperin, Kirby, and Mankins renders claims 1 and 13 obvious for the reasons discussed above. Claims 7 and 19 further require “wherein the data is sent in the form of key [sic] that is obtained by a hashing process carried out in respect of the objects on the respective remote computers.”

Kirby discloses or renders obvious this element. Ex. 1004, ¶151. Kirby teaches a two-tiered virus detection system that scans for viruses and monitors the behavior of objects. Kirby teaches that a “security agent” on the host computer (*i.e.*, remote computer) includes “a scanner/extractor 124 for parsing objects such as files and delivering byte sequences to the signature generator 114 for computing a signature or hash.” Ex. 1007, 11:11–16. Kirby further discloses that the signature generated by the remote computer “is an identifier such as a hash, *i.e.* MD4, MD5, or SHA to name several known in the art, which are rapidly computable,” and that “other hashes or computations which are smaller than the entity they represent yet highly unlikely to be duplicated by a dissimilar entity” could be used. *Id.*, 12:37–43. Further, Kirby teaches the remote computer calculating the hash on the objects stored on the remote computers and transmitting it to the base computer, as a way

of providing the base computer with data about the computer objects. *Id.*, 9:66–10:4, 15:21–26; Ex. 1004, ¶¶152–53. Modifying Halperin’s disclosures of sending data about a computer object to a base computer by sending the data in the form of a key that is obtained by a hashing process carried out in respect of the objects on the respective remote computers, as Kirby teaches, would have been an obvious combination of prior art elements according to known methods to yield predictable results. Ex. 1004, ¶153. Halperin in view of Kirby and Mankins consequently renders claims 7 and 19 obvious. *Id.*, ¶153–54.

i. Dependent Claims 8 & 20

Claims 8 and 20 depend from claims 7 and 19, respectively. The combination of Halperin, Kirby, and Mankins renders claims 7 and 19 obvious for the reasons discussed above. Claims 8 and 20 further require “wherein the key has at least one component that represents executable instructions contained within or constituted by the object.”

Kirby discloses or renders obvious this element. Ex. 1004, ¶155. As discussed with respect to claims 7 and 19 above, Kirby discloses sending data about computer objects to a base computer (or AV server) in the form of a hash. Kirby further teaches “the signature is *computed from* the associated object (e.g. infected email attachment) of the behavioral actions and operable to indicate subsequent undesirable activity.” Ex. 1007, 12:23–26 (emphasis added). A POSITA would

have understood that one way Kirby’s signature would be “operable to indicate subsequent undesirable activity” would be to include a component that represents executable instructions contained within or constituted by the malware (i.e., the computer object), and it would have been obvious to a POSITA to include that component to improve detection. Ex. 1004, ¶156. As explained with respect to claims 7 and 19 above, it would have been obvious to combine Halperin with Kirby’s key. *See id.*, ¶153. Moreover, for purposes of its infringement allegations, Patent Owner asserts that sending a SHA256 hash meets this limitation, without specifying what the hash contains. *See* Ex. 1016, 5. Thus, although Petitioner disputes Patent Owners infringement allegations, Halperin meets this limitation under Patent Owner’s interpretation because it sends data about the computer object in the form of a key that is obtained by a hashing process. Ex. 1004, ¶157. Thus, the combination of Halperin, Kirby, and Mankins also renders claims 8 and 20 obvious. *Id.*, ¶¶157–58.

j. Dependent Claims 9 & 21

Claims 9 and 21 depend from claims 7 and 19, respectively. The combination of Halperin, Kirby, and Mankins renders claims 7 and 19 obvious for the reasons discussed above. Claims 9 and 21 further require “wherein the key has at least one component that represents data about said object.”

Kirby discloses or renders obvious this element. Ex. 1004, ¶159. Kirby explains that “the set of signatures may include signatures generated from predetermined mathematical computations *on at least a portion of the object.*” Ex. 1007, 6:6–10 (emphasis added). It would have been obvious to a POSITA that a signature generated from computations on at least a portion of the object would include data about the object as at least one component of the signature. Ex. 1004, ¶160. As explained with respect to claims 7 and 19 above, it would have been obvious to a POSITA to modify Halperin with the key disclosed by Kirby. Thus, the combination of Halperin, Kirby, and Mankins consequently renders claims 9 and 21 obvious. Ex. 1004, ¶¶161–62.

k. Dependent Claims 10 & 22

Claims 10 and 22 depend from claims 9 and 21, respectively. The combination of Halperin, Kirby, and Mankins renders claims 9 and 21 obvious for the reasons discussed above. Claims 10 and 22 further require

“wherein said data about said object includes at least one of:
the current name of the object; the physical and folder location of the object on disk; the original name of the object; the creation and modification dates of the object; vendor, product and version and any other information stored within the object; the object header or header held by the remote computer; and, events initiated by or involving the object when the object is created, configured or runs on the respective remote computers.”

As discussed with respect to claims 9 and 21 above, Kirby discloses generating a key that is “computed from” the computer object. *See* Ex. 1007, 9:66–10:4. A POSITA would have known that a key computed from the computer object could have included any of the information recited in claims 10 and 22, and it would have been obvious to a POSITA to represent at least one of the items recited in claim 12 to improve detection. Ex. 1004, ¶¶163–64. Thus, for the reasons explained above with respect to claims 9 and 21, the combination of Halperin, Mankins, and Kirby also renders claims 10 and 22 obvious. *Id.*, ¶¶164–65.

I. Dependent Claims 11 & 23

Claims 11 and 23 depend from claims 7 and 19, respectively. The combination of Halperin, Kirby, and Mankins renders claims 7 and 19 obvious for the reasons discussed above. Claims 11 and 23 further require “wherein the key has at least one component that represents the physical size of the object.”

Kirby discloses or renders obvious this element. Ex. 1004, ¶166. As explained with respect to dependent claims 7 and 19, Kirby discloses that “the signature is an identifier such as a hash, i.e. MD4, MD5, or SHA to name several known in the art, which are rapidly computable as an automated response.” Ex. 1007, 12:37–40. Kirby likewise discloses generating a key that is “computed from” the computer object. *Id.*, 12:23–26. A POSITA would have known that metadata about the computer object, including its physical size, could be used to help

determine whether it was legitimate or likely malware. Ex. 1004, ¶167. For example, if an object purports to be a known, legitimate object, comparing its characteristics like physical size to the known characteristics of the object it purports to be can help identify whether the object is malware. *Id.*, ¶¶168–69. Thus, it would have been obvious to a POSITA to include in a key representing data about the computer object at least one component that represents the physical size of the computer object. *See also* Claims 9 and 14, above.

Thus, the combination of Halperin, Mankins, and Kirby renders claims 11 and 23 obvious. Ex. 1004, ¶¶169–70.

m. Dependent Claims 12 & 24

As explained above, Halperin in view of Mankins renders claims 12 and 24 obvious. For the same reasons, claims 12 and 24 are obvious over the combination of Halperin, Mankins, and Kirby. Ex. 1004, ¶171.

n. Independent Claim 25

[25P]: A method of providing data about a computer object from a remote computer to a base computer so that a comparison can be made at the base computer with similar data received from other remote computers, the method comprising:

See Element [1P], above (Ground 1).

[25A]: providing from a remote computer to a base computer data about a computer object that is stored on the remote computer;

See Element [1A], above (Ground 1).

[25B]: the data including information about the behaviour of the object running on one or more remote computers and including one or more of: executable instructions contained within or constituted by the object; the size of the object; the current name of the object; the physical and folder location of the object on disk; the original name of the object; the creation and modification dates of the object; vendor, product and version and any other information stored within the object; the object header or header held by the remote computer; and, events initiated by or involving the object when the object is created, configured or runs on the respective remote computers;

See Element [1A] and claims 2 and 14, above (Ground 1).

[25C]: the data being sent in the form of key that is obtained by a hashing process carried out in respect of the object on the remote computer,

See Claims 7 & 19, above.

[25D]: automatically generating a mask for the computer object, the mask defining acceptable behaviour for the object built in accordance with normal behaviour of the object determined from said received data;

See Element [1E], above (Ground 1).

[25E]: executing the computer object at the remote computer; automatically monitoring operation of the computer object at the remote computer compared to the behaviour permitted by the mask;

See Element [1F], above.

[25F]: allowing the object to continue to run when behaviour of the computer object is permitted by the mask;

See Element [1G], above (Ground 1).

[25G]: disallowing the object to run when the actual monitored behaviour of the object extends beyond that permitted by the mask; and,

See Element [1H], above (Ground 1).

[25H]: reclassifying the object as malware when the actual monitored behaviour extends beyond [t]hat permitted by the mask,

See Element [1I], above.

[25I]: wherein the base computer is arranged to receive information about the behaviour of the object running on one or more remote computers,

See Element [1A], above (Ground 1).

[25J]: wherein the step of generating a mask for that object comprises building the mask with the base computer in accordance with normal behaviour of the object determined from said received information.

See Element [1E], above (Ground 1).

o. Dependent Claim 26

See Claims 8 and 20, above.

p. Dependent Claim 27

See Claims 9 and 21, above.

q. Dependent Claim 28

See Claims 10 and 22, above.

r. Dependent Claim 29

See Claims 11 and 23, above.

s. Dependent Claim 30

[30]: A non-transitory storage medium storing a computer program comprising program instructions for causing a computer to perform the method of claim 25.

Claim 30 depends from claim 25. The combination of Halperin, Kirby, and Mankins renders claim 25 obvious for the reasons discussed above. Claim 30 further requires “[a] non-transitory storage medium storing a computer program comprising program instructions for causing a computer to perform the method of claim 25.”

Kirby discloses this additional element. Ex. 1004, ¶187. Kirby teaches that “[o]ne such embodiment comprises a computer program product that has a computer-readable medium including computer program logic encoded thereon that, when performed in a multiprocessing computerized device having a coupling of a memory and a processor, programs the processor to perform the operations disclosed herein as embodiments of the invention to carry out data access requests.” Ex. 1007, 6:48–55. Thus, the combination of Halperin, Kirby, and Mankins renders claim 30 obvious. Ex. 1004, ¶188.

X. PTAB DISCRETION SHOULD NOT PRECLUDE INSTITUTION

A. PTAB Should Not Exercise Its Discretion to Deny Institution Under *Fintiv*

Consistent with congressional intent and the goals of *NHK/Fintiv*, Petitioner takes steps to promote the Board’s efficiency and fairness objectives. Petitioner

files this petition to join the Sophos IPR in an understudy role. Sophos served a stipulation (Ex. 1009) broader than previously found mitigated concerns of duplicative efforts between the district court and the Board and potentially conflicting decisions. *Sand Revolution II, LLC v. Continental Intermodal Grp.*, IPR2019-01393, Paper 24 (June 16, 2020). Petitioner also stipulates below to be bound by this proceeding. These facts, along with the merits of Grounds 1–3, strongly favor institution.

1. Factor 1: Likelihood of a Stay

Institution will enable the Board to resolve the issue of validity, and a finding of invalidity will relieve the District Court of the need to thereafter continue with the litigation. The District Court has made clear in prior cases that a motion to stay prior to institution of an IPR would be premature, and the Board recently “decline[d] to speculate on the likelihood of Judge Albright entering a stay after institution” and thus found Factor 1 neutral. *Western Digital Corp. v. Kuster*, IPR2020-01391, Paper 10 at 9 (Feb. 16, 2021); *see also Volkswagen Grp. of Am., Inc. v. Arigna Tech. Ltd.*, IPR2021-01321, Paper 10 at 7 (Feb. 15, 2022) (declining to speculate about the likelihood of a stay). Thus, this factor is at worst neutral.

2. Factor 2: District Court Schedule

The District Court schedule weighs in favor of institution. Pursuant to the District Court’s fifth amended scheduling order on March 29, 2023 (Dkt. 252), the

trial in this case will not begin until August 19, 2024. Ex. 1014, 6. Based on an 18 month IPR schedule, even if the trial began exactly as currently projected, a Final Written Decision (“FWD”) arising from this petition would still be likely to issue at approximately the same time as the trial.

Even if that were not the case, uncertainty about the trial date and the high likelihood of further delay weigh against discretionary denial. In general, District Court patent-validity litigation schedules are uncertain. Indeed, the Board has held that this factor was “neutral” even in cases where the projected trial date was before the Board’s statutory final-decision deadline because it has recognized that projected trial dates are uncertain, especially when projected in the early stages of litigation. In *Cizion, LLC v. Kerr Machine Co.*, for instance, trial was set “three months prior to [the Board’s] (anticipated) deadline for a final written decision.” PGR2020-00065, Paper 10 at 26 (Dec. 3, 2020). The Board explained that “[a]lthough the current schedule [in *Cizion*] indicates that trial will commence a few months prior to the statutory due date to issue a final written decision in this proceeding, evidence in the record cited by the Petitioner does indicate that the actual trial date is uncertain.” *Id.*; see also *Micron Tech., Inc. v. Godo Kaisha IP Bridge 1*, IPR2020-01008, Paper 10 at 14 (Dec. 7, 2020) (“Accordingly, even if we consider the Texas court to have set a formal trial date for a few months before the

final written decision, due to the uncertainty as to this trial date, this factor is, at most, neutral to whether we should exercise our discretion to deny the Petition.”).

Of particular relevance here, the *Cizion* Board credited the Petitioner’s evidence that “in the Western District of Texas, 70% of trial dates have slipped following PTAB discretionary denials.” PGR2020-00065, Paper 10 at 26 (Dec. 3, 2020). The Board further noted that the evidence included “multiple proceedings with trial dates close to the” same trial date. *Id.* The same is true here, where the District Court has set a single trial date for five separate litigations with dozens of asserted patents. Ex. 1014.

Thus, general uncertainty of District Court patent-validity actions, and the high likelihood that this litigation will extend beyond its projections, weighs in favor of institution. Indeed, recent Interim Guidance⁴ dictates that because scheduled trial dates “are unreliable and often change,” a scheduled trial date “is not by itself a good indicator of whether the district court trial will occur before the statutory deadline for a [FWD].” *Samsung Elec.’s Co., Ltd. et al v. Staton Techniya, LLC*, IPR202200302, Paper 13, 10 (PTAB July 11, 2022).

⁴ https://www.uspto.gov/sites/default/files/documents/interim_proc_discretionary_denials_aia_parallel_district_court_litigation_memo_20220621_.pdf.

Finally, trial is more likely to get pushed further back in this case because the litigation is especially complex. Patent Owner originally asserted nine patents against five defendants, including the '250 Patent, and filed an amended complaint that asserts additional patents against Trend Micro, Inc., and two amended complaints asserting additional patents against other defendants. In granting Patent Owner's motion to amend, the Court acknowledged that this case is "about as complicated as you're going to get with patent cases" and said it would be "amenable to adjusting the schedule" by "granting more time to address things." Ex. 1015, 2. Because this multi-defendant consolidated case is exceptionally complicated, it is unlikely that the District Court will dispose of the dispute over the '250 Patent before a FWD would issue.

Thus, this factor weighs strongly against discretionary denial.

3. Factor 3: Petitioners' Investment in the IPR Outweighs Forced Investment in Litigation to Date

Petitioners' substantial investment in these IPR proceedings should counterbalance—if not outweigh—the resources invested in the co-pending litigation. It would be unjust to consider resources expended in the District Court (equally by both parties), without considering Petitioner's resources expended to prepare this IPR petition that would be irretrievably lost without consideration on the merits, in addition to the extensive expenses that may be foregone through

institution of this petition and that will otherwise certainly follow in co-pending litigation for which significant milestones exist. *See, e.g., Apple v. Seven Networks*, IPR2020-00255, Paper 13 at 12–15 (July 28, 2020) (declining to exercise discretionary denial on petitions filed nine months after the commencement of copending litigation where a large number of patents were included in the litigation, and a large number of claims from each patent were challenged in the petitions).

In *Sand Revolution*, even after invalidity contentions and the *Markman* hearing, the Board found that this factor weighed “only marginally, if at all, in favor of exercising discretion.” *Sand Revolution*, IPR2019-01393, Paper 24 at 11. Given Petitioners’ diligence and significant investment in filing this petition, this factor should weigh against a discretionary denial.

4. Factor 4: The Petition Raises Unique Issues

Factor 4 strongly supports institution. Consistent with *Fintiv*, Petitioner asks the Board to consider the unique challenges raised in the petition.

Petitioner hereby stipulates that, if institution of this IPR is granted, Petitioner will 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 “mitigates any concerns of duplicative efforts

between the district court and the Board, as well as concerns of potentially conflicting decisions”. *Id.*

5. Factor 5: Institution Will Provide the Board an Opportunity to Invalidate Claims That Later Could Be Reasserted Against Others

In the District Court litigation, as noted above, Patent Owner has asserted 15 patents against five defendants. Of those 15 patents, five are asserted against all five defendants; five are asserted against four defendants; and the average patent is asserted against 3.6 defendants. In fact, Patent Owners asserted the ’250 Patent against three other co-defendants. *See CrowdStrike Action, Sophos Action, and Kaspersky Action.* In other words, Patent Owner has a demonstrated history of asserting claims against multiple defendants, and the Board has an opportunity to invalidate some of those claims here. Factor 5 weighs in favor of institution.

6. Factor 6: Other Circumstances Support Institution

As explained above, the merits of the unpatentability grounds are particularly strong, weighing in favor of institution. *See Fintiv*, 14–15. And the effect on “the economy [and] the integrity of the patent system” further supports institution. PTAB Consolidated Trial Practice Guide, p. 56. The claims of the ’250 Patent are very broad and purport to cover receiving, determining, and storing information about potential malware, which was already disclosed several years prior to even the

earliest priority date of the patent. The Board should consider and invalidate the claims such that they may not be asserted against other entities in the future.

B. PTAB Should Not Exercise its Discretion to Deny Institution Under *Becton* and *Advanced Bionics*.

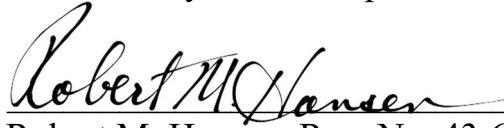
Institution should not be denied under the framework set forth in *Becton, Dickinson & Co. v. B. Braun Melsungen AG*, IPR2017-01586, Paper 8 (Dec. 15, 2017) (precedential as to § III.C.5, first paragraph) and *Advanced Bionics LLC v. MED-EL Elektromedizinische Gerate GmbH*, IPR2019-01469, Paper 6 (PTAB Feb. 13, 2020) (precedential). Neither Halperin, Kirby, nor Mankins were before the Examiner during prosecution. Therefore, the arguments presented in this Petition are not the same as, or substantially similar to, arguments previously presented to the Office. Thus, the first prong of the *Advanced Bionics* framework is not met. And because the first prong is not met, there is no need to reach the second prong to resolve against discretionary denial under Section 325(d)—though in any event, there are no pending proceedings before the PTAB concerning the '250 Patent, so the second prong is not met either. *Oticon Medical AB v. Cochlear Ltd.*, IPR2019-00975 Paper 15, 20 (PTAB Oct. 16, 2019) (precedential).

XI. CONCLUSION

Petitioner has established a reasonable likelihood that the Challenged Claims are unpatentable. Petitioner therefore respectfully requests that IPR be instituted.

Respectfully submitted,

The Marbury Law Group, PLLC

A handwritten signature in cursive script, reading "Robert M. Hansen", is written over a horizontal line.

Robert M. Hansen, Reg. No. 43,656

Matthew J. Anderson, Reg. No. 78,257

Scott B. Amankwatia, Reg. No. 78,226

Date: November 8, 2023

The Marbury Law Group, PLLC

11800 Sunrise Valley Drive

15th Floor

Reston, VA 20191

Tel: (703) 391-2900

Attorneys for Petitioner

CERTIFICATION UNDER 37 C.F.R. § 42.24(d)

Under the provisions of 37 C.F.R. § 42.24(d), the undersigned attorney hereby certifies that the word count for Sections I and III–XI of the foregoing Petition for *Inter Partes* Review is 13,959, according to the word count tool in Microsoft Word, excluding the table of contents, table of authorities, exhibit list, mandatory disclosures, signature block, certificates and attached exhibits of the Petition.

DATED: November 8, 2023

By: 
Robert M. Hansen
Reg. No. 43,656
The Marbury Law Group, PLLC
11800 Sunrise Valley Drive
15th Floor
Reston, VA 20191
Tel: 703-391-2900
rhansen@marburylaw.com

CERTIFICATE OF SERVICE

The undersigned certifies service pursuant to 37 C.F.R. §§ 42.6(e) and 42.105(a), (b) on the Patent Owner via FedEx overnight mail of a copy of this Petition for Inter Partes Review and supporting materials at the Patent Owner at the correspondence address of record for the '250 Patent:

Assignee:

Webroot Inc.
385 Interlocken Crescent
Suite 800
Broomfield, CO 80021

Counsel of Record:

Merchant & Gould P.C.
150 South Fifth Street
Suite 2200
Minneapolis, MN 55402-0903
United States

DATED: November 8, 2023

By: 
Robert M. Hansen
Reg. No. 43,656
The Marbury Law Group, PLLC
11800 Sunrise Valley Drive
15th Floor
Reston, VA 20191
Tel: 703-391-2900
rhansen@marburylaw.com

APPENDIX: CHALLENGED CLAIM LISTING

No.	Limitation
[1P]	A method of classifying a computer object as malware, the method comprising:
[1A]	at a base computer, receiving data about a computer object from each of plural remote computers on which the object or similar objects are stored, the data including information about the behaviour of the object running on one or more remote computers;
[1B]	determining in the base computer whether the data about the computer object received from the plural computers indicates that the computer object is malware;
[1C]	classifying the computer object as malware when the data indicates that the computer object is malware;
[1D]	when the determining does not indicate that the computer object is malware, initially classifying the computer object as not malware;
[1E]	automatically generating a mask for the computer object that defines acceptable behaviour for the computer object, wherein the mask is generated in accordance with normal behaviour of the object determined from said received data;
[1F]	running said object on at least one of the remote computers; automatically monitoring operation of the object on the at least one of the remote computers;
[1G]	allowing the computer object to continue to run when behaviour of the computer object is permitted by the mask;
[1H]	disallowing the computer object to run when the actual monitored behaviour of the computer object extends beyond that permitted by the mask; and,
[1I]	reclassifying the computer object as malware when the actual monitored behaviour extends beyond that permitted by the mask.

No.	Limitation
[2]	A method according to claim 1, wherein the data about the computer object that is sent from the plural remote computers to the base computer includes one or more of: executable instructions contained within or constituted by the object; the size of the object; the current name of the object; the physical and folder location of the object on disk; the original name of the object; the creation and modification dates of the object; vendor, product and version and any other information stored within the object; the object header or header held by the remote computer; and, events initiated by or involving the object when the object is created, configured or runs on the respective remote computers.
[3]	A method according to claim 1, wherein determining identifies relationships between the object and other objects.
[4]	A method according to claim 3, wherein if at least one other object to which said object is related is classed as malware, then classifying said object as malware.
[5]	A method according to claim 3, wherein said other objects include the object or similar objects stored on at least some of the remote computers.
[6]	A method according to claim 3, wherein said other objects include other objects that are parent objects or child objects or otherwise process-related objects to said object.
[7]	A method according to claim 1, wherein the data is sent in the form of key that is obtained by a hashing process carried out in respect of the objects on the respective remote computers.
[8]	A method according to claim 7, wherein the key has at least one component that represents executable instructions contained within or constituted by the object.
[9]	A method according to claim 7, wherein the key has at least one component that represents data about said object.

No.	Limitation
[10]	A method according to claim 9, wherein said data about said object includes at least one of: the current name of the object; the physical and folder location of the object on disk; the original name of the object; the creation and modification dates of the object; vendor, product and version and any other information stored within the object; the object header or header held by the remote computer; and, events initiated by or involving the object when the object is created, configured or runs on the respective remote computers.
[11]	A method according to claim 7, wherein the key has at least one component that represents the physical size of the object.
[12]	A method according to claim 1, comprising if the monitored behaviour of the object running on the remote computer exhibits non-permitted behaviour that extends beyond that permitted by the mask, comparing at the base computer the non-permitted behaviour with the behaviour of other objects and if said behaviour is known to be not malicious for said other objects, allowing said behaviour for the object and modifying the mask to allow that behaviour for that object in future.
[13P]	Apparatus for classifying a computer object as malware, the apparatus comprising:
[13A]	a base computer constructed and arranged to receive data about a computer object from each of plural remote computers on which the object or similar objects are stored, the data including information about the behaviour of the object running on one or more remote computers;
[13B]	the base computer being constructed and arranged to determine whether the data about the computer object received from said plural computers indicates that the computer object is malware; and,
[13C]	the base computer being constructed and arranged to classify the computer object as malware when the base computer determines that the data indicates that the computer object is malware,

[13D]	wherein the base computer is constructed and arranged to automatically generate a mask for an object that is initially classed not as malware, said mask defining acceptable behaviour for the object built in accordance with normal behaviour of the object determined from said received data,
--------------	---

No.	Limitation
[13E]	wherein operation of the object is automatically monitored when the object is running on at least one of the remote computers,
[13F]	the object being allowed to continue to run when behaviour of the object is permitted by the mask,
[13G]	the object not being permitted to run when the actual monitored behaviour of the object extends beyond that permitted by the mask,
[13H]	and the object is reclassified as malware when the actual monitored behaviour extends beyond that permitted by the mask.
[14]	Apparatus according to claim 13, the data includes one or more of: executable instructions contained within or constituted by the object; the size of the object; the current name of the object; the physical and folder location of the object on disk; the original name of the object; the creation and modification dates of the object; vendor, product and version and any other information stored within the object; the object header or header held by the remote computer; and, events initiated by or involving the object when the object is created, configured or runs on the respective remote computers.
[15]	Apparatus according to claim 13, wherein the base computer is constructed and arranged so that the determining identifies relationships between the object and other objects.
[16]	Apparatus according to claim 15, the base computer is constructed and arranged so that if at least one other object to which said object is related is classed as malware, then said object is classified as malware.
[17]	Apparatus according to claim 15, wherein the base computer is constructed and arranged so that said other objects include the object or similar objects stored on at least one of said remote computers.

[18]	Apparatus according to claim 15, wherein the base computer is constructed and arranged so that said other objects include other objects that are parent objects or child objects or otherwise processrelated objects to said object.
[19]	Apparatus according to claim 13, wherein the base computer is constructed and arranged so as to be able to process data that is sent in the form of key that is obtained by a hashing process carried out in respect of the objects on said respective remote computers.

No.	Limitation
[20]	Apparatus according to claim 19, wherein the key has at least one component that represents executable instructions contained within or constituted by the object.
[21]	Apparatus according to claim 19, wherein the key has at least one component that represents data about said object.
[22]	Apparatus according to claim 21, wherein said data about said object includes at least one of: the current name of the object; the physical and folder location of the object on disk; the original name of the object; the creation and modification dates of the object; vendor, product and version and any other information stored within the object; the object header or header held by the remote computer; and, events initiated by or involving the object when the object is created, configured or runs on the respective remote computers.
[23]	Apparatus according to claim 19, wherein the key has at least one component that represents the physical size of the object.
[24]	A method according to claim 13, wherein, if the monitored behaviour of the object running on the remote computer exhibits non-permitted behaviour that extends beyond that permitted by the mask, the base computer is arranged to compare the non-permitted behaviour with the behaviour of other objects and if said behaviour is known to be not malicious for said other objects the base computer is arranged to allow said behaviour for the object and to modify the mask to allow that behaviour for that object in future.

[25P]	A method of providing data about a computer object from a remote computer to a base computer so that a comparison can be made at the base computer with similar data received from other remote computers, the method comprising:
[25A]	providing from a remote computer to a base computer data about a computer object that is stored on the remote computer;

No.	Limitation
[25B]	the data including information about the behaviour of the object running on one or more remote computers and including one or more of: executable instructions contained within or constituted by the object; the size of the object; the current name of the object; the physical and folder location of the object on disk; the original name of the object; the creation and modification dates of the object; vendor, product and version and any other information stored within the object; the object header or header held by the remote computer; and, events initiated by or involving the object when the object is created, configured or runs on the respective remote computers;
[25C]	the data being sent in the form of key that is obtained by a hashing process carried out in respect of the object on the remote computer,
[25D]	automatically generating a mask for the computer object, the mask defining acceptable behaviour for the object built in accordance with normal behaviour of the object determined from said received data;
[25E]	executing the computer object at the remote computer; automatically monitoring operation of the computer object at the remote computer compared to the behaviour permitted by the mask;
[25F]	allowing the object to continue to run when behaviour of the computer object is permitted by the mask;
[25G]	disallowing the object to run when the actual monitored behaviour of the object extends beyond that permitted by the mask; and,
[25H]	reclassifying the object as malware when the actual monitored behaviour extends beyond [t]hat permitted by the mask,

[25I]	wherein the base computer is arranged to receive information about the behaviour of the object running on one or more remote computers,
[25J]	wherein the step of generating a mask for that object comprises building the mask with the base computer in accordance with normal behaviour of the object determined from said received information.
[26]	A method according to claim 25, wherein the key has at least one component that represents executable instructions contained within or constituted by the object.
[27]	A method according to claim 25, wherein the key has at least one component that represents data about said object.
No.	Limitation
[28]	A method according to claim 27, wherein said data about said object includes at least one of: the current name of the object; the physical and folder location of the object on disk; the original name of the object; the creation and modification dates of the object; vendor, product and version and any other information stored within the object; the object header or header held by the remote computer; and, events initiated by or involving the object when the object is created, configured or runs on the respective remote computers.
[29]	A method according to claim 25, wherein the key has at least one component that represents the physical size of the object.
[30]	A non-transitory storage medium storing a computer program comprising program instructions for causing a computer to perform the method of claim 25.