

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

TREND MICRO, INC.
Petitioner

v.

WEBROOT, INC.

Case No. IPR2023-01389
Patent No. 8,856,505

**PETITION FOR INTER PARTES REVIEW
OF U.S. PATENT NO. 8,856,505**

Filed on behalf of Trend Micro, Inc.

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.	Mandatory Notices (37 C.F.R. § 42.8(a)(1)).....	1
II.	Grounds for Standing (37 C.F.R. § 42.104(a)).....	3
III.	Identification of Challenge (37 C.F.R. § 42.104(b)).....	3
	A. Citation of Prior Art	3
	B. Statutory Grounds for Challenge.....	4
IV.	The '505 Patent	5
	A. Claim of Priority of the '505 patent	5
	B. Specification of the '505 Patent	5
	C. Summary of the Prosecution History	6
	D. Level of Ordinary Skill in the Art	8
	E. Claim Construction.....	8
V.	Grounds of Rejection.....	10
	A. Ground 1: Claims 1-20 are unpatentable under 35 U.S.C. § 103(a) as obvious in view of Garfinkel (EX1006) and Hoglund (EX1007).....	10
	1. Claim 1	13
	a. A method for managing pestware on a computer comprising:.....	13
	b. monitoring events during a boot sequence of the computer;.....	13
	c. managing pestware-related events during a first period in a boot sequence of the computer, the first period in the boot sequence occurring before the computer becomes configured to run native applications, before a subsystem of an operating system is loaded, and after a kernel is loaded;	15
	d. managing pestware-related events in accordance with a set of behavior rules during a second period in the boot sequence occurring when the computer is configured to run native applications;	16

e.	generating, in response to the monitoring, a record of events, the record of events including the pestware-related events;	18
f.	analyzing the record of events so as to identify the pestware-related events	19
g.	modifying the set of behavior rules so as to prevent the pestware related events; and	19
h.	scanning data in a registry of the computer for pestware during the second period in the boot sequence.	21
2.	Claim 2	22
	The method of claim 1, further comprising launching, after an operating system is initiated, a pestware management engine; and wherein the set of behavior rules includes behavior rules compiled by the pestware management engine.	22
3.	Claim 3	24
	The method of claim 1, wherein the record of events includes information selected from a group consisting of: process identification information, file identification information, and hook generation information.	24
4.	Claim 4	26
	The method of claim 1, further comprising: managing pestware-related events after the computer becomes configured to run native applications.	26
5.	Claim 5	26
	The method of claim 1, wherein the monitoring includes monitoring the boot sequence while boot drivers are initiated.	26
6.	Claim 6	28
a.	A system for managing pestware on a computer comprising:	28
b.	a processor;	28
c.	a memory comprising:	28

d. a first instruction set configured to monitor events during a boot sequence of the computer;	29
e. a second instruction set configured to manage pestware-related events during first period in a boot sequence of the computer, the first period in the boot sequence occurring before the computer becomes configured to run native applications, before a WIN32 subsystem is loaded, and after a kernel is loaded;	30
f. a third instruction set configured to manage pestware-related events in accordance with a set of behavior rules during a second period in the boot sequence occurring when the computer is configured to run native applications;	31
g. a fourth instruction set configured to generate, in response to the monitoring, a record of events, the record of events including the pestware-related events;	31
h. a fifth instruction set configured to analyze the record of events so as to identify the pestware-related events; and	32
i. a sixth instruction set configured to modify the set of behavior rules so as to prevent the pestware related events; and	32
j. a seventh instruction set configured to scan data in a registry of the computer for pestware during the second period in the boot sequence. ..	33
7. Claim 7	33
The system of claim 6, wherein the memory further comprises an eighth instruction set configured to launch, after an operating system is initiated, a pestware management engine; and wherein the set of behavior rules includes behavior rules compiled by the pestware management engine. 33	
8. Claim 8	34
The system of claim 1, wherein the record of events includes information selected from a group consisting of: process identification information, file identification information, and hook generation information.	
	34

9. Claim 9	34
The system of claim 6, wherein the memory further comprises: a ninth instruction set configured to manage pestware-related events after the computer becomes configured to run native applications.	
10. Claim 10	35
The system of claim 6, wherein the first instruction set is further configured to monitor the boot sequence while boot drivers are initiated.	
11. Claim 11	36
a. A non-transitory tangible computer-readable storage medium comprising program instructions for:	
b. monitoring events during a boot sequence of a computer;	
c. managing pestware-related events during a first period in a boot sequence of the computer, the first period in the boot sequence occurring before the computer becomes configured to run native applications, before a WIN32 subsystem is loaded, and after a kernel is loaded;	
d. managing pestware-related events in accordance with a set of behavior rules during a second period in the boot sequence occurring when the computer is configured to run native applications;	
e. generating, in response to the monitoring, a record of events, the record of events including the pestware-related events;	
f. analyzing the record of events so as to identify the pestware-related events;	
g. modifying the set of behavior rules so as to prevent the pestware related events; and	
h. scanning data in a registry of the computer for pestware during the second period in the boot sequence.	
12. Claim 12	39

The non-transitory tangible computer-readable storage medium of claim 11 further comprising program instructions for launching, after an operating system is initiated, a pestware management engine; and wherein the set of behavior rules includes behavior rules compiled by the pestware management engine.....	39
13. Claim 13	40
The non-transitory tangible computer-readable storage medium of claim 11, wherein the record of events includes information selected from a group consisting of: process identification information, file identification information, and hook generation information.	40
14. Claim 14	40
The non-transitory tangible computer-readable storage medium of claim 11 further comprising program instructions for managing pestware-related events after the computer becomes configured to run native applications.....	40
15. Claim 15	41
The non-transitory tangible computer-readable storage medium of claim 11, wherein the monitoring includes monitoring the boot sequence while boot drivers are initiated.....	41
16. Claim 16	42
The method of claim 2, further comprising modifying, by the pestware management engine, the behavior rules utilized by a kernel-level monitor, wherein the kernel-level monitor manages pestware-related events in the first and second periods in the boot sequence.....	42
17. Claim 17	44
The method of claim 1, further comprising modifying the set of behavior rules based on the scanned registry so as to prevent the pestware-related events.	44
18. Claim 18	45

The method of claim 1, wherein the pestware-related invents include events associated with at least one of loading pestware, executing pestware, and one or more behaviors of pestware.	45
19. Claim 19	46
The system of claim 6, wherein the pestware-related invents include events associated with at least one of loading pestware, executing pestware, and one or more behaviors of pestware.	46
20. Claim 20	47
The non-transitory tangible computer-readable storage medium of claim 11, wherein the pestware-related invents include events associated with at least one of loading pestware, executing pestware, and one or more behaviors of pestware.	47
VI. NO BASIS FOR DISCRETIONARY DENIAL.....	47
VI. CONCLUSION	51

TABLE OF AUTHORITIES

Cases

<i>Advanced Bionics, LLC v. MED-EL Elektromedizinische Gerate GmbH</i> , IPR2019-01469, Paper 6, 8-9 (Feb. 13, 2020)	49
<i>Cellco Partnership v. Huawei Device Co.</i> , IPR2020-01117, Paper 10, 13 (Feb. 3, 2021)	50
<i>General Electric v. Vestas</i> , IPR2018-00928, Paper 9, *16-17 (PTAB Nov. 5, 2018)	10
<i>Graham v. John Deere Co.</i> , 383 U.S. 1 (1966)	10
<i>KSR Int’l Co. v. Teleflex, Inc.</i> , 550 U.S. 398 (2007)	11
<i>Sand Revolution II, LLC v. Cont’l Intermodal Group – Trucking LLC</i> , IPR2019-01393, Paper 24 at 11-12 (PTAB June 16, 2020)	48
<i>Western Digital Corporation v. Spex Technologies, Inc.</i> , IPR2018-00084, Paper 14, *11 (PTAB Apr. 25, 2018)	9

Statutes

35 U.S.C. § 103	passim
35 U.S.C. § 325(d)	46

Rules

37 C.F.R. § 42.10(a)	2
37 C.F.R. § 42.10(b)	2
37 C.F.R. § 42.104(a)	3
37 C.F.R. § 42.8(b)(2)	1
37 C.F.R. § 42.8(b)(3)	2
37 C.F.R. § 42.100(b)	8

EXHIBIT LIST

Exhibit No.	Description
1001	U.S. Patent No. 8,856,505 to Schneider.
1002	Prosecution History of '505.
1003	Expert Declaration of Dr. Seth Nielson.
1004	Curriculum Vitae of Dr. Seth Nielson.
1005	Webroot, Inc. and Open Text, Inc. v. Trend Micro Inc., Case No. 6:22-cv-00243-ADA-DTG, Claim Construction Order, Dkt. 391 (W.D. Tex., July 25, 2023).
1006	Tal Garfinkel, <i>A Virtual Machine Introspection Based Architecture for Intrusion Detection</i>, Computer Science Department, Stanford University (2003).
1007	Greg Hoglund, <i>Subverting the Windows Kernel: Rootkits</i>, Addison-Wesley Professional (Jul. 22, 2005).
1008	Andreas Buntten, UNIX and Linux based Rootkits Techniques and Countermeasures, DFN-CERT Services GmbH (Apr. 30, 2004).
1009	Catherine Dodge, A Study of Initialization in Linux and OpenBSD, ACM SIGOPS Operating Systems Review 39(2), pp.79-93, (April, 2005).
1010	Phrack Staff, Linux on-the-fly kernel patching without LKM (58), Linux on-the-fly kernel patching without LKM, http://phrack.org/issues/58/7.html (Dec. 28, 2001).
1011	Red Hat Linux 6.2, <i>The Official Red Hat Linux Reference Guide</i> (2000).

Exhibit No.	Description
1012	Kathy Ivens, <i>Walkin' Through the Boot Process</i>, (Aug. 30, 2004).
1013	Henrique de Moraes Holschuh, <i>System Init Scripts and the Debian O.S.</i>, 3rd Debian Conference (Jun. 2002).
1014	Frank Apap, <i>Detecting Malicious Software by Monitoring Anomalous Windows Registry Accesses</i>, LNCS (2516), (Oct. 2002).
1015	Declaration of Ingrid Hsieh-Yee, Ph.D.

Trend Micro, Inc. (“Petitioner”) petitions for inter partes review of claims 1-20 of U.S. Patent No. 8,856,505 to Boney, entitled “Malware management through kernel detection during a boot sequence” (“the ’505 patent,” EX1001), which according to PTO records, is assigned to Webroot, Inc.

I. Mandatory Notices (37 C.F.R. § 42.8(a)(1))

REAL PARTY IN INTEREST:

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.

RELATED MATTERS:

Pursuant to 37 C.F.R. § 42.8(b)(2), Petitioner identifies the following related matters.

A petition for *inter partes* review was filed previously by CrowdStrike with respect to U.S. Patent No. 8,856,505, as IPR2023-01159.

The ’505 patent is also at issue in the following district court litigations:

- *Webroot, Inc. and Open Text, Inc. v. Trend Micro Inc.*, Case No. 6:22-cv-00239 (W.D. Tex.), which was originally filed on March 4, 2022, and an amended complaint listing the ’505 patent filed on September 7, 2022.

- *Webroot, Inc. and Open Text, Inc. v. Sophos LTD.*, Case No. 6:22-cv-00240 (W.D. Tex.), which was originally filed on March 4, 2022, and an amended complaint listing the '505 patent filed on September 7, 2022.
- *Webroot, Inc. and Open Text, Inc. v. CrowdStrike, Inc. and CrowdStrike Holdings, Inc.*, Case No. 6:22-cv-00241 (W.D. Tex.), which was originally filed on March 4, 2022, and an amended complaint listing the '505 patent filed on September 7, 2022.
- *Webroot, Inc. and Open Text, Inc. v. AO Kaspersky Lab*, Case No. 6:22-cv-00243 (W.D. Tex.), which was originally filed on March 4, 2022, and an amended complaint listing the '505 patent filed on September 7, 2022.

LEAD AND BACKUP COUNSEL:

Pursuant to 37 C.F.R. § 42.8(b)(3) and 37 C.F.R. § 42.10(a), Petitioner appoints:

Robert M. Hansen (Reg. No. 43,656) as lead counsel,

Scott B. Amankwatia (Reg. No. 78,226) as back-up counsel, and

Matthew J. Anderson (Reg. No. 78,257) as back-up counsel.

Pursuant to 37 C.F.R. § 42.10(b), a power of attorney is being filed with this designation of counsel.

SERVICE INFORMATION:

Petitioner provides the following service information for designated counsel:

The Marbury Law Group, PLLC
11800 Sunrise Valley Drive
15th Floor
Reston, VA 20191
Tel: (703)-391-2900
Fax: (703)-391-2900

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

rhansen@marburylaw.com;
samankwatia@marburylaw.com;
manderson@marburylaw.com; and
pat-docketing@marburylaw.com.

PAYMENT OF FEES

The PTO is authorized to charge any fees due during this proceeding to
Deposit Account No. 18-1579.

II. Grounds for Standing (37 C.F.R. § 42.104(a))

Petitioner certifies that the '505 patent is available for *inter partes* review
and that Petitioner is not barred or estopped from requesting an *inter partes* review
challenging the patent claims on the grounds identified in this Petition.

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

A. Citation of Prior Art

Petitioner cites the following prior art references:

**Garfinkel, et al., *A Virtual Machine Introspection Based Architecture for
Intrusion Detection*, Computer Science Dept., Stanford Univ. (2003).**

(“Garfinkel,” EX1006) is prior art under pre-AIA § 102(a)/(b) because it was published and publicly available at least by October 2003, more than one (1) year before the earliest possible priority date of the ’505 patent. EX1015, p.18. Garfinkel was not submitted to or considered by the Examiner who allowed the ’505 patent.

Hoglund, et al., *Subverting the Windows Kernel: Rootkits*, Addison-Wesley Professional (Jul. 22, 2005) (“Hoglund,” EX1007) is prior art under pre-AIA § 102(a) because it was published and publicly available at the latest on May 27, 2006, before the earliest possible priority date of the ’505 patent. EX1015, p.25. Hoglund was not disclosed to or considered by the Examiner who allowed the ’505 patent.

B. Statutory Grounds for Challenge

Petitioner requests review of claims 1-20 on the following ground:

GROUND 1: Claims 1-20 are unpatentable under pre-AIA 35 U.S.C. § 103(a) as obvious in view of Garfinkel (EX1006) and Hoglund (EX1007).

IV. The '505 Patent

A. Claim of Priority of the '505 patent

The '505 patent, entitled “Malware management through kernel detection during a boot sequence,” issued on 10/7/14, from U.S. Patent Application No. 13/460,655, filed 4/30/12, which is a continuation of U.S. Patent Application No. 11/462,827, filed 8/7/06. EX1001, Front Cover.

B. Specification of the '505 Patent

The '505 patent describes a method, system, and apparatus for managing pestware on a computer with a non-transitory tangible computer-readable storage medium. EX1001, 6:34-8:54.

Specifically, claims 1-5 are directed to a method for managing pestware on a computer that includes monitoring events, managing pestware-related events, generating a record of events, analyzing the record of events, modifying a set of behavior rules, and scanning data in a registry. EX1001, 6:34-55.

Dependent claims 2-5 recite further limitations regarding launching a pestware management engine, the record of events, managing pestware-related events, and monitoring the boot sequence. EX1001, 6:56-7:2.

Claims 6-10 are directed to the system for managing pestware on the computer including a processor and a memory with at least seven instruction sets.

Ex. 1001, 7:3-47. Dependent claims 7-10 recite further limitations regarding the memory and record of events. *Id.* at 7:31-47.

Claims 11-20 are directed to a non-transitory tangible computer-readable storage medium. *Id.* at 7:48-8:54. Dependent claims 12-20 recite further limitations regarding program instructions, record of events, monitoring the boot sequence, set of behavior rules, and pestware-related invents. *Id.* at 8:14-54.

C. Summary of the Prosecution History

The application that issued as the '505 patent, U.S. Patent Application No. 13/460,655, was filed on 4/30/12. EX1001, cover page.

Prosecution began with the non-final Office Action that issued on 10/18/13, claims 1-18 were rejected. Claims 1-18 were rejected on the ground non-statutory double patenting over claims 1-5, 9, 10, 12, and 14 of U.S. Patent No. 8,190,868. EX1002, p.160. Claims 1, 2, 5-8, 11-14, 17, and 18 were rejected under 35 U.S.C. § 103(a) as unpatentable over U.S. Patent Publication No. 2005/0114687 to Zimmer *et al.* ("Zimmer") in view of WIPO Patent Publication No. 2006/077443, Rothwell, and further in view of U.S. Patent Publication No. 2005/0216759 to Rothman *et al.* ("Rothman"). EX1002, pp.160-174. Claims 3, 9, and 15 were rejected under 35 U.S.C. § 103(a) as unpatentable over Zimmer, Rothwell, Rothman and further in view of U.S. Patent Publication 2005/0125687 to

Townsend *et al.* (“Townsend”). Claims 4, 10, and 16 were rejected under 35 U.S.C. § 103(a) as unpatentable over Zimmer, Rothwell, Rothman, Townsend, and further in view of U.S. Patent Publication 2003/0135791 to Natvig. EX1002, pp.174-180.

An Examiner Interview occurred on 1/20/14 discussing the Office Action and amendments to obviate the claim rejections. In the Amendment and Response, dated 1/22/14, Applicant filed a terminal disclaimer and Applicant asserted the cited references do not disclose the feature of: managing pestware-related events in accordance with a set of behavior rules during a second period in the boot sequence occurring when the computer is configured to run native applications and scanning data in a registry of the computer for pestware during the second period in the boot sequence. EX1002, p.150.

In the Notice of Allowability, dated 4/29/14, the Examiner entered an examiner’s amendment that amended claims 1, 4, 7, 8, 10, 13, and 16, canceled claims 3, 9, 15, and added new claims 21-23 (that issued as claims 18-20 of the ’505 patent). EX1002, p.90. In these amendments, independent claims 1 and 13 were amended to recite “generating, in response to the monitoring, a record of events, the record of events including the pestware-related events; analyzing the record of events so as to identify the pestware-related events; modifying the set of behavior rules so as to prevent the pestware related events”; and independent claim

7 was amended to recite, “ a fourth instruction set configured to generate, in response to the monitoring, a record of events, the record of events including the pestware-related events; a fifth instruction set configured to analyze the record of events so as to identify the pestware-related events; and a sixth instruction set configured to modify the set of behavior rules so as to prevent the pestware related events.” EX1002, pp.91-94. The Examiner indicated that claims 1, 2, 4-8, 10-14, and 16-23 were allowed over the prior art of record. EX1002, p.94.

D. Level of Ordinary Skill in the Art

As established by the Declaration of Dr. Seth Nielson dated 9/6/2023 (*See* EX1003, ¶117), a person having ordinary skill in the art (PHOSITA) as of the earliest priority date for the '505 patent would have had a bachelor's degree in computer science or a related field, or an equivalent technical degree or equivalent work experience, and an additional one to two years of education or experience in the field of computer security topics, such as malware/exploit detection. More education can supplement practical experience and vice versa. EX1003, ¶117.

E. Claim Construction

In this proceeding, claims are interpreted under the same standard applied by Article III courts (i.e., the *Phillips* standard). 37 C.F.R. §42.100(b); *Phillips v. AWH Corp.*, 415 F.3d 1303 (Fed. Cir. 2005) (en banc).

In parallel litigation, Webroot argues claim terms in asserted patents should be given their plain and ordinary meanings. Likewise, the claim construction order in this parallel litigation in the Western District of Texas construed contested terms in the '505 patent under their plain and ordinary meaning with the exception of “pestware.” EX1005, pp.3. Specifically, the court construed “pestware” to mean “any program that collects and/or reports information about a person or an organization and any ‘watcher processes’ related to the pestware”. *Id.* For purposes of this proceeding only, Petitioner proposes the Board adopt plain and ordinary meaning as proposed by Webroot and adopted by the Western District of Texas court for terms other than “pestware” and adopt the court’s construction of “pestware.” *See Western Digital Corporation v. Spex Technologies, Inc.*, IPR2018-00084, Paper 14, *11 (PTAB Apr. 25, 2018) (finding Petition based on claim constructions urged by Patent Owner satisfies the claim construction requirements and Petitioner is not required to express its subjective agreement regarding correctness of the proffered claim construction or take ownership of the construction); *see also General Electric v. Vestas*, IPR2018-00928, Paper 9, *16-17 (PTAB Nov. 5, 2018). Should the Board construe “pestware” under its plain and ordinary meaning, the arguments and citations of this petition would equally apply to that broader interpretation.

V. Grounds of Rejection

The following discussion and the accompanying evidence demonstrate that there is a reasonable likelihood that Petitioner will prevail with respect to the challenged claims.

Petitioner evaluates the scope and content of the prior art, any differences between the art and the claim, and the knowledge of person of ordinary skill in the art in accordance with *Graham v. John Deere Co.*, 383 U.S. 1 (1966) and *KSR Int'l Co. v. Teleflex, Inc.*, 550 U.S. 398 (2007). Petitioner is unaware of any secondary considerations relevant to the '505 patent and a more than adequate written rationale is provided to support the combination of Garfinkel and Hoglund under *KSR*. 550 U.S. at 418.

A. Ground 1: Claims 1-20 are unpatentable under 35 U.S.C. § 103(a) as obvious in view of Garfinkel (EX1006) and Hoglund (EX1007).

A PHOSITA would have been motivated to combine the teachings of Garfinkel and Hoglund. Garfinkel and Hoglund are both directed to the same problem of identifying malicious objects (e.g., malware and pestware) including rootkits and rootkit operations. EX1003, ¶¶138-140,146; EX1006, pp.2,6; EX1007, p.17. Garfinkel discloses an intrusion detection system (IDS) that detects malicious activity during the full boot sequence by detecting and blocking rootkits that attack the Linux operating system (OS), including tracking memory access

events and identifying those that indicate malicious behavior. EX1003, ¶147; EX1006, pp.6-7. Hoglund describes how Windows-based rootkits infiltrate and compromise a Windows system, describing different approaches used to survive reboot, including the re-loading of the rootkit at various time periods within the boot sequence. EX1003, ¶148; EX1007, pp.93-128 (citing example code). The host OS that runs the IDS of Garfinkel also boots and runs the guest OS as another application on the host OS. EX1006, p.3. A PHOSITA would have understood that a Windows OS could also be the guest OS and monitored by the VM IDS of Garfinkel. EX1003, ¶151. A PHOSITA would have been motivated to combine Hoglund's teachings about Windows rootkits with Garfinkel's rootkit detection and blocking IDS system, including applying Garfinkel's teachings to Windows rootkits as Windows was a common operating system and the Windows rootkits were a known threat. *Id.* at ¶149.

Because the VM (host OS) of Garfinkel can host Linux or Windows OSes and monitor each with the same IDS tools or modules from the beginning of boot, a PHOSITA would have understood that the teachings of Garfinkel regarding visibility into a boot stage in Linux or Windows are interchangeable for equivalent stages in either boot process. EX1003, ¶150; EX1007, p.13. Specifically, Garfinkel discloses that the placement of the IDS outside of the guest OS allows this total visibility into the boot process. EX1006, pp.3-5. Additionally, a PHOSITA would

have looked to Hoglund for details regarding timing of rootkit intrusion on Windows and Linux since Hoglund is a textbook devoted to the subject with numerous examples. EX1003, ¶148; EX1007, pp.2,17.

A PHOSITA would have had an expectation of success in combining the teachings of Garfinkel and Hoglund because the PHOSITA would have understood that the two references were compatible. Garfinkel’s architecture for detecting Linux rootkits is not Linux-specific. *Id.* at ¶149. Garfinkel explains “[t]he operating system running inside of a VM is traditionally referred to as the guest OS, and applications running on the guest OS are similarly referred to as guest applications.” EX1006, p. 3. A PHOSITA would have known that virtual machines, and VMware specifically, support the Windows operating system. EX1003, ¶150. Thus, a PHOSITA would have understood that the VMM core component of Garfinkel could work with Linux or Windows kernels, and that the Garfinkel policy modules could be updated with behavior rules that detect the Windows rootkits disclosed by Hoglund. *Id.* at ¶151. Because of these similarities, commodity components, and modular design, a PHOSITA would have had a high expectation of success of combining Garfinkel and Hoglund with minimal experimentation. *Id.*

1. Claim 1

a. A method for managing pestware on a computer comprising:

Garfinkel teaches a virtual machine monitor that gives visibility into the state of the host machine, which is a computer, including inspection for malware. EX1003, ¶¶155-162; EX1006, pp.1,14. Therefore, Garfinkel teaches the preamble.

b. monitoring events during a boot sequence of the computer;

Garfinkel discloses a virtual machine (VM) that monitors an operating system on a host computer as well as hardware level events (corresponding to “monitoring events ...of the computer”). EX1003, ¶166; EX1006, p.1. Garfinkel is particularly concerned with identifying rootkits and kernel backdoors which a PHOSITA would have understood are difficult to detect. EX1003, ¶¶169-170; EX1006, pp.9-10, Table-1. For example, Garfinkel discloses identifying and blocking the “SucKIT” rootkit, which inserts itself during the boot sequence, by blocking an attempt to modify an interrupt dispatch table (corresponding to “monitoring events during a boot sequence...”). EX1003, ¶¶99-103, 171-172 (citing EX1010 and EX1014); EX1006, pp.9-10. The VM of Garfinkel, however, is able to detect and block these rootkits and backdoors during installation and activation because “there is no state in the monitored system that the IDS [of Garfinkel] cannot see”. EX1003, ¶172; EX1006, p.4. A PHOSITA would have understood that this disclosure coupled with the capability to block the rootkits and kernel backdoors disclosed as being blocked in Garfinkel indicates that the VM

IDS of Garfinkel monitors and checks events during the boot sequence as recited in this limitation. EX1003, ¶¶173-174; EX1006, p.10.

Further, Hoglund describes various rootkits in detail and the procedures necessary to identify and block them. For example, Hoglund states:

Any .sys file or executable that is to be loaded at boot time can be replaced, or the loader code can be inserted similarly to the way a virus can infect a file. Ironically, one of the best things to infect is a virus scanning or security product. A security product will typically start when the system is booted. EX1007, p.38.

Accordingly, a PHOSITA would have understood that the boot sequence is a key time when these intrusions activate. EX1003, ¶174; EX1007, p.38. In addition, the Windows-based rootkits of Hoglund operate like the Linux-based rootkits of Garfinkel, which means that a PHOSITA would understand that the detection capabilities of the VM IDS of Garfinkel would also be able to detect boot-time intrusions on Windows and Linux operating systems. EX1003, ¶174.

Therefore, Garfinkel alone or in combination with Hoglund discloses this limitation. EX1003, ¶176.

c. managing pestware-related events during a first period in a boot sequence of the computer, the first period in the boot sequence occurring before the computer becomes configured to run native applications, before a subsystem of an operating system is loaded, and after a kernel is loaded;

Garfinkel discloses a virtual machine that monitors an operating system on a host computer as well as hardware level events. EX1003, ¶¶179-181; EX1006, p.1. As explained above, Garfinkel discloses monitoring and blocking events during a boot period that would lead to intrusions (corresponding to “managing pestware-related events during a first period in a boot sequence of the computer”). EX1003, ¶182; EX1006, pp.9-10. Further, Garfinkel discloses that the VM has visibility into hardware and software events of the operating system that it hosts and sees any state of the monitored system including those during boot. EX1003, ¶¶166-167; EX1006, pp.1,4.

This limitation relates the pestware management to a specific time period of boot (i.e., the first period) based on certain characteristics of that period. EX1003, ¶179. A PHOSITA would have understood that rootkits can infect any portion of the boot sequence and, further, that infections of the initialization functions of /sbin/init would occur after the kernel had been loaded. EX1003, ¶¶67,80-84,97-103 (citing EX1008). For Linux operating systems, the kernel launches and then launches the /sbin/init functions that initialize the user space environment. EX1003, ¶¶180-181; EX1011; EX1013. Thus, the SucKIT malware activates after

the kernel loads (in order to infect the init script) but before the computer is configured to execute native applications. EX1003, ¶¶85,99-103,181-182 (citing EX1010). This time period after the kernel attempts to launch “/sbin/init” but before “/sbin/init” has finished running is a “first period... occurring before the computer becomes configured to run native applications, before a subsystem of an operating system is loaded, and after a kernel is loaded.” EX1003, ¶181.

Further, by the nature of the attempt to infect the init script that is initializing subsystems, a PHOSITA would understand that SucKIT operates before a subsystem of an operating system is loaded (e.g., networking or graphics). EX1003, ¶¶86,183. Since the VM IDS of Garfinkel is capable of detecting and blocking this SucKIT malware by its disclosed event monitoring (e.g., memory registers, instructions, system calls), Garfinkel discloses “managing pestware-related events during a first period in a boot sequence” including the “first period” as defined in the claim. EX1003, ¶¶182-185 (citing EX1011); EX1006, pp.9-10.

Therefore, Garfinkel teaches this limitation. EX1003, ¶187.

d. managing pestware-related events in accordance with a set of behavior rules during a second period in the boot sequence occurring when the computer is configured to run native applications;

Garfinkel discloses various detector modules including a lie detector module, a signature detector, a memory protection enforcer, and a user program integrity checker. EX1003, ¶¶134,190; EX1006, pp.9-10. While all of these

modules may launch on the VM IDS before the host machine OS launches, different modules operate to monitor different aspects of the hosted OS operation and boot. EX1003, ¶190; EX1006, pp.4-6. For example, the lie detector module operates by querying the operating system state via various OS utilities such as “ps” and “netstat” which are user-level programs. EX1003, ¶¶190-191; EX1006, pp.9-10. Accordingly, a PHOSITA would understand that the lie detector (during the boot sequence) operates to identify and halt malware that injects itself during a runtime when user space programs are operating (corresponding to “a second period in the boot sequence occurring when the computer is configured to run native applications”). EX1003, ¶¶68-81,191 (citing EX1009). This time after “/sbin/init” has started and the necessary subsystems are loaded is “a second period in the boot sequence occurring when the computer is configured to run native applications.” *Id.*

Further, Garfinkel teaches performing memory checks and sys-call monitoring in order to detect rootkits. EX1003, ¶¶190; EX1006, pp.9-10. A PHOSITA would have known that rootkit attacks attempt to modify memory and perform malicious system calls in the period of the boot sequence after /sbin/init/ is completed and after OS subsystems are initiated (corresponding to “managing pestware-related events ...during a second period”). EX1003, ¶¶191-195 (citing EX1007, p.38 noting any sysfile or executable can be modified to launch a virus).

Garfinkel also discloses “managing pestware-related events in accordance with a set of behavior rules.” Specifically, Garfinkel teaches a policy engine that runs policy modules that include access rules (e.g., policies and alarms) and event/action monitoring. EX1003, ¶¶197; EX1006, p.7. Further, Garfinkel teaches halting the hosted OS if certain rules or policies are violated (e.g., modifying protected memory). EX1003, ¶¶198-199; EX1006, p.10.

Therefore, Garfinkel teaches this limitation. EX1003, ¶207.

e. generating, in response to the monitoring, a record of events, the record of events including the pestware-related events;

Garfinkel discloses event-driven policies and polling policies that are enforced as part of the virtual machine monitor (VMM). EX1003, ¶¶203-204; EX1006, p.8-9. The event driven policies utilize event driven checkers that register the possibly-malicious events (corresponding to “pestware-related events”) that the VMM should flag / hook for inspection. EX1003, ¶¶203-204; EX1006, pp.7,9. Garfinkel also discloses that this event-driven inspection may be based on a log of events (corresponding to “generating, in response to the monitoring, a record of events”) to reduce computing load. EX1003, ¶¶205; EX1006, p.13.

Therefore, Garfinkel teaches this limitation. EX1003, ¶206.

f. analyzing the record of events so as to identify the pestware-related events

Garfinkel teaches monitoring events in memory and networks including via a log of those events as described above for limitation 1[e]. EX1003, ¶¶209-210; EX1006, p.13. Garfinkel also discloses that the VMM actively prevents actions that violate policies or restrictions (e.g., memory restrictions) such that the VMM assesses whether the event or action violates the policy or configuration (e.g., modify read-only memory or reconfigure network). EX1003, ¶¶212-214; EX1006, p.10. A PHOSITA would understand this assessment of events for malware to correspond to the elements of this limitation. EX1003, ¶¶211-215. Further, Garfinkel discloses processing the logs collected by the system (corresponding to “analyzing the record of events”) in the policy engine where the VMM determines whether malicious actions are being taken and responding accordingly. EX1003, ¶216; EX1006, p.7,13.

Therefore, Garfinkel teaches this limitation. EX1003, ¶217.

g. modifying the set of behavior rules so as to prevent the pestware related events; and

Garfinkel teaches a couple of ways to modify or update the policies that dictate which actions are prevented on the hosted machine. EX1003, ¶220. First, Garfinkel discloses that policy modules may be used alone or together with others and that the policy modules can be turned on and off or managed such that they

work together. EX1003, ¶¶221-223; EX1006, pp.8-10,Table-1. A PHOSITA would have understood that running a policy module as a standalone module would impose a different set of rules than operating them “in concert with other policy modules” (corresponding to “modifying the set of behavior rules”).

EX1003, ¶222. Further, Garfinkel discloses that the different modules enable different detections and different levels of intrusion containment throughout the hosted machine. EX1003, ¶¶223-225; EX1006, pp.8-10,Table-1. For example, the rootkit “cdoor” would only be detected by raw socket detector, whereas the user level rootkit “lrk5” can be detected by four different modules. *Id.* Table 1 of Garfinkel below also discloses that prevention (P) and detection (D) are separate, and that an event may be detected by one module (see columns) and prevented by another.

Name	Description	nic	raw	sig	int	lie	mem
cdoor	Stealth user level remote backdoor		D				
t0rn	Precompiled user level rootkit			D		D	
Ramen	Linux Worm			D			
lrk5	Source based user level rootkit	P		D	D	D	
knark-0.59	LKM based kernel backdoor/rootkit			D		D	P
adore-0.42	LKM based kernel backdoor/rootkit			D		D	P
dsniff 2.4	All-purpose packet sniffer for switched networks	P					
SUCKIT	/dev/kmem patching based kernel backdoor			D		D	P

Table 1. Results of Livewire policy modules against common attacks. Within the grid, “P” designates a prevented attack and “D” a detected attack.

Therefore, a PHOSITA would have understood that changing the different modes of the policy engine to incorporate different prevention and detection capabilities corresponds to this limitation. EX1003, ¶¶226.

Second, Garfinkel describes the need for the VM IDS to adapt and protect other mutable but sensitive kernel elements as they become targets of future threats, which a PHOSITA would understand to be modifying behavior rules of the VM IDS. EX1003, ¶¶226-227; EX1006, p.12.

Therefore, Garfinkel teaches this limitation. EX1003, ¶228.

h. scanning data in a registry of the computer for pestware during the second period in the boot sequence.

Garfinkel discloses a signature scanner that scans files, buffer cache, and memory with access to memory registers. EX1003, ¶231; EX1006, pp.1,9-10. Additionally, one of the trojans (“Back Orifice”) identified by the scanner described in Garfinkel would have been understood by a PHOSITA to modify a Windows registry. EX1003, ¶232; EX1006, p.9. Therefore, Garfinkel discloses a system with the necessary access and inspection capabilities at any time during boot to scan a registry and describes malware that would motivate scanning that registry (corresponding to “scanning data ...of the computer for pestware during the second period in the boot sequence”). EX1003, ¶¶231-232.

Further, Hoglund teaches that rootkits often need to insert registry keys or modify the windows registry so that the malware can get further access (e.g., memory write capability). EX1003, ¶233; EX1007, pp.38,78-79,96-97. Hoglund discloses that to defend against such intrusions, the registry should be hooked and monitored and discloses a scanning tool (“RootkitRevealer”) that parses files and

the Windows registry for malicious files (corresponding to “scanning data in a registry of the computer”). EX1003, ¶¶234-235; EX1007, pp.347,359. A PHOSITA would have been motivated to combine the scanning functionality during boot of Garfinkel with the registry scanning of Hoglund for hosted Windows machines. EX1003, ¶236.

Therefore, Garfinkel in view of Hoglund teaches this limitation. EX1003, ¶237.

For the foregoing reasons, a PHOSITA would have found claim 1 to be obvious based on the Garfinkel-Hoglund combination. Accordingly, claim 1 should be found unpatentable over Garfinkel-Hoglund under 35 U.S.C. § 103.

2. Claim 2

The method of claim 1, further comprising launching, after an operating system is initiated, a pestware management engine; and wherein the set of behavior rules includes behavior rules compiled by the pestware management engine.

Garfinkel discloses launching an intrusion detection system (IDS) as a part of the virtual machine (VM) that hosts the guest operating system (OS) of the monitored host. EX1003, ¶¶240-241; EX1006, p.7. Figure 1 of Garfinkel is annotated below:

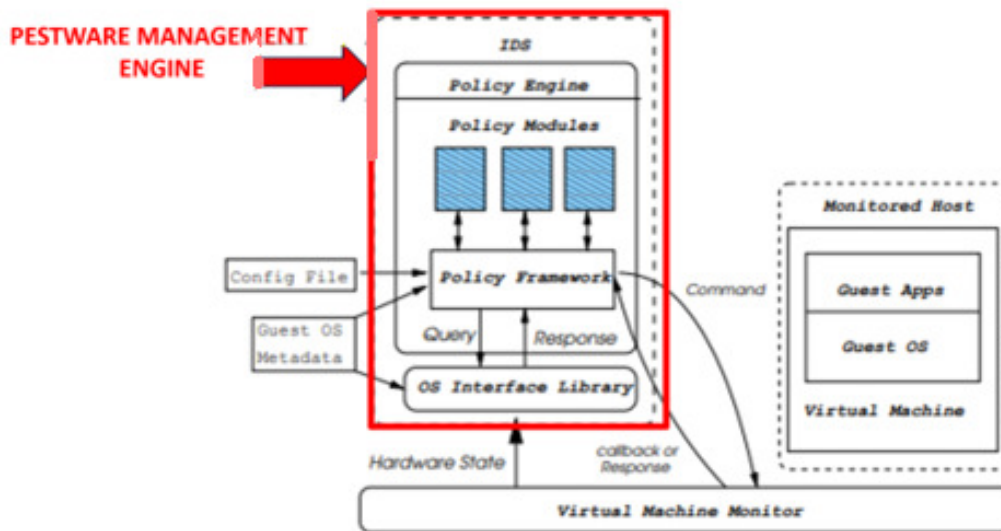


Figure 1 of Garfinkel – Annotated

A PHOSITA would understand that while the IDS and VMM are loaded before the guest OS, the IDS running on the VMM would be loaded after the host virtual machine’s OS (corresponding to “launching, after an operating system is initiated”) on which the VMM runs. EX1003, ¶¶242; EX1006, pp.3,12. A PHOSITA would understand this IDS to be a pestware management engine and understand its functionality as an application on the host OS to disclose “launching, after an operating system is initiated, a pestware management engine.” EX1003, ¶¶240-242. Further, the policy engine of the IDS illustrated above in Figure 1 stores the policies and writes them to the policy engine via a common API (corresponding to “the set of behavior rules includes behavior rules compiled by the pestware management engine”). EX1003, ¶¶243; EX1006, p.8.

Therefore, Garfinkel discloses the limitations of claim 2. EX1003, ¶244.

As shown, the Garfinkel-Hoglund combination discloses all elements recited in claim 1 from which claim 2 depends. Accordingly, claim 2 should be found invalid under 35 U.S.C. § 103 based on Garfinkel and Hoglund.

3. Claim 3

The method of claim 1, wherein the record of events includes information selected from a group consisting of: process identification information, file identification information, and hook generation information.

As described regarding claim 1[e], Garfinkel discloses creating a log of events which may then be processed by the policy engine to detect malware. EX1003, ¶¶205; EX1006, p.13. Part of this log of events includes unauthorized network access including user processes using raw sockets (corresponding to “process identification information”) and unauthorized network changes. EX1003, ¶¶247; EX1006, pp.8-9. Further, this log is built by marking or registering specific events (e.g., system calls, memory states) with the VMM so that it notifies a policy module with the information generated by that event. EX1003, ¶¶248-249; EX1006, pp.9-10. Therefore, when filesystem integrity checking is performed in the IDS of the VMM or filesystem buffer cache is scanned (corresponding to “file identification information”), these events or findings would be recorded. EX1003, ¶¶248; EX1006, pp.9,14. Indeed as Table 1 of Garfinkel shows, events from multiple modules are collocated for enhanced monitoring.

Name	Description	nic	raw	sig	int	lie	mem
cdoor	Stealth user level remote backdoor		D				
t0rn	Precompiled user level rootkit			D		D	
Ramen	Linux Worm			D			
lrk5	Source based user level rootkit	P		D	D	D	
knark-0.59	LKM based kernel backdoor/rootkit			D		D	P
adore-0.42	LKM based kernel backdoor/rootkit			D		D	P
dsniff 2.4	All-purpose packet sniffer for switched networks	P					
SUCKIT	/dev/kmem patching based kernel backdoor			D		D	P

Table 1. Results of Livewire policy modules against common attacks. Within the grid, "P" designates a prevented attack and "D" a detected attack.

Further, Garfinkel discloses that the memory checker can inspect memory via inspection commands that allow the memory checker to interpose itself into memory events such as interrupts and updates to device and memory state, which a PHOSITA would understand to include hooks. EX1003, ¶¶251; EX1006, pp.6-7. Hoglund supports this interpretation by noting that rootkits routinely use inline function hooking or detours to overwrite protected memory and redirect execution flow, whereas the memory checker of Garfinkel explicitly looks for writes to protected memory. EX1003, ¶¶252-254; EX1006, pp.7,10; EX1007, pp.95,139.

Therefore, Garfinkel or Garfinkel in view of Hoglund teaches the limitations of this claim. EX1003, ¶257.

As shown, the Garfinkel-Hoglund combination discloses all elements recited in claim 1 from which claim 3 depends. Accordingly, claim 3 should be found invalid under 35 U.S.C. § 103 based on Garfinkel and Hoglund.

4. Claim 4

The method of claim 1, further comprising: managing pestware-related events after the computer becomes configured to run native applications.

As explained with regard to claim 1[d], Garfinkel teaches managing malware events in a boot sequence after a point where native applications can run. EX1003, ¶¶260. For example, the lie detector uses various native programs (e.g., netstat) to evaluate the runtime environment in this period. EX1003, ¶¶190-191; EX1006, pp.9-10.

Therefore, Garfinkel discloses the limitations of claim 4. EX1003, ¶262.

As shown, the Garfinkel-Hoglund combination discloses all elements recited in claim 1 from which claim 4 depends. Accordingly, claim 4 should be found invalid under 35 U.S.C. § 103 based on Garfinkel and Hoglund.

5. Claim 5

The method of claim 1, wherein the monitoring includes monitoring the boot sequence while boot drivers are initiated.

Garfinkel discloses that its services (e.g., the memory checker) can monitor all states in the monitored system (i.e., the hosted machine) (corresponding to “wherein the monitoring includes monitoring the boot sequence while boot drivers are initiated”). EX1003, ¶¶263-265; EX1006, pp.1,4. This high visibility into the monitored system is accomplished by “the ability to interpose at the architecture interface of the monitored host, yielding even better visibility than normal OS-

level mechanisms by enabling monitoring of both hardware and software level events.” EX1003, ¶265; EX1006, pp.1. A PHOSITA would have understood that after the kernel is loaded in the boot process, then the kernel drivers are loaded sequentially over a period of time, and then the system moves on to setup of user space which in Linux-based machines begins with the execution of init scripts (i.e., /sbin/init). EX1003, ¶¶75-85,89-93,265-266 (citing EX1012). As described above regarding claim 1[c], at least the SucKIT malware would have modified protected memory in this time period. *Id.*

To the extent that Garfinkel’s detection of memory violations by the SucKIT malware in this period is not explicit, Hoglund teaches that rootkits register themselves as drivers that are loaded during boot, which Garfinkel would have detected as writes to protected memory or the protected file system. EX1003, ¶¶267-268; EX1006, p.7; EX1007, p.38.

Therefore, Garfinkel or Garfinkel in view of Hoglund discloses the limitations of claim 5. EX1003, ¶270.

As shown, the Garfinkel-Hoglund combination discloses all elements recited in claim 1 from which claim 5 depends. EX1003, ¶269. Accordingly, claim 5 should be found invalid under 35 U.S.C. § 103 based on Garfinkel and Hoglund. *Id.* at ¶270.

6. Claim 6

a. A system for managing pestware on a computer comprising:

The preamble of claim 6 is similar to the preamble of claim 1, replacing “method of” with “system for.” As discussed regarding 1[a], Garfinkel discloses a virtual machine monitor that gives visibility into the state of the host machine, which is a computer, including inspection for malware. EX1003, ¶¶271-275; EX1006, pp.1,14. Therefore, Garfinkel teaches the preamble.

b. a processor;

As discussed regarding 1[a], Garfinkel discloses a virtual machine monitor that runs on a host machine, which is a computer, and a PHOSITA would have understood that a host OS requires both a processor and a memory. EX1003, ¶¶271,276-277. Garfinkel explicitly discloses “[a] common way to virtualize the CPU is to run the VMM in the most privileged mode of the processor.” EX1006, p.1.

c. a memory comprising:

Garfinkel discloses both a processor and a memory for use by the VMM: “Among these techniques is configuring the real machine so that virtual machines can safely and directly execute using the machine’s CPU and memory.” EX1006, p.3; EX1003, ¶¶271,276-277. Further, Garfinkel disclosed that the prototype system used VMware plus customized software components. “Each of these

components runs in its own process in Linux, our host OS.” EX1006, p.7. A PHOSITA would have known that components running a Linux process and integrated with VMware are program instructions (corresponding to “instruction set”) loaded from a non-transitory tangible computer readable storage medium, such as a hard drive. EX1003, ¶275.

d. a first instruction set configured to monitor events during a boot sequence of the computer;

As addressed above, a PHOSITA would have understood that Garfinkel's disclosure of processes running in Linux discloses “instruction sets” that cause components to perform the disclosed processes. EX1003, ¶275. The rest of this limitation is equivalent to claim element 1[b]. *Id.* at ¶271. As described re 1[b] Garfinkel discloses a virtual machine (VM) that monitors an operating system on a host computer as well as hardware level events (corresponding to “monitoring events ...of the computer”). EX1003, ¶271; EX1006, p.1. Further, Hoglund describes various in detail and the procedures necessary to identify and block rootkit executables. EX1007, p.38. For the reasons discussed re 1[b], Garfinkel alone or in combination with Hoglund discloses this limitation. EX1003, ¶¶176,271-276.

- e. a second instruction set configured to manage pestware-related events during first period in a boot sequence of the computer, the first period in the boot sequence occurring before the computer becomes configured to run native applications, before a WIN32 subsystem is loaded, and after a kernel is loaded;**

As addressed above, a PHOSITA would have understood that Garfinkel's disclosure of processes running in Linux discloses "instruction sets" that cause components to perform the disclosed processes. EX1003, ¶¶271, 275. The rest of this limitation is similar to claim element 1[c] except that this limitation recites "before a WIN32 subsystem is loaded." *Id.* at ¶¶271,279. As described re 1[c] Garfinkel discloses monitoring and blocking events in Linux based rootkits during a boot period that would lead to intrusions (corresponding to "managing pestware-related events during a first period in a boot sequence of the computer"). EX1003, ¶¶271,279; EX1006, pp.9-10. A PHOSITA would have looked to Hoglund because it discloses that Windows-based rootkits behaved like Linux-based rootkits in that they typically activate during the boot sequence. EX1003, ¶¶184-185. In addition, Hoglund discloses numerous code samples for rootkit detection and management each with instructions for the Windows operating system. EX1007, pp.345-356. Windows kernel drivers are loaded before the Win32 Subsystem. Thus, a PHOSITA would have understood that any rootkit installation in a kernel driver would come after the kernel was loaded but before the Win32 Subsystem was loaded and native applications are configured to run. EX1003,

¶185. For these reasons Garfinkel in view of Hoglund discloses this limitation.

EX1003, ¶¶185,187,271-276.

- f. a third instruction set configured to manage pestware-related events in accordance with a set of behavior rules during a second period in the boot sequence occurring when the computer is configured to run native applications;**

As addressed above, a PHOSITA would have understood that Garfinkel's disclosure of processes running in Linux discloses “instruction sets” that cause components to perform the disclosed processes. EX1003, ¶275. In addition, Hoglund discloses numerous code samples for rootkit detection and management each with instructions for the Windows operating system. EX1007, pp.345-356. The rest of this limitation is equivalent to claim element 1[d]. *Id.* at ¶271. For the reasons discussed re 1[d], Garfinkel discloses this limitation. EX1003, ¶¶200,271-276.

- g. a fourth instruction set configured to generate, in response to the monitoring, a record of events, the record of events including the pestware-related events;**

As addressed above, a PHOSITA would have understood that Garfinkel's disclosure of processes running in Linux discloses “instruction sets” that cause components to perform the disclosed processes. EX1003, ¶275. The rest of this limitation is equivalent to claim element 1[e]. *Id.* at ¶271. As described re 1[e] Garfinkel discloses an event-driven inspection that may be based on a log of events

to reduce computing load. EX1003, ¶¶271; EX1006, p.13. For the reasons discussed re 1[e], Garfinkel discloses this limitation. EX1003, ¶¶206, 271-276.

h. a fifth instruction set configured to analyze the record of events so as to identify the pestware-related events; and

As addressed above, a PHOSITA would have understood that Garfinkel's disclosure of processes running in Linux discloses “instruction sets” that cause components to perform the disclosed processes. EX1003, ¶275. The rest of this limitation is equivalent to claim element 1[f]. *Id.* at ¶271. As described re 1[f] Garfinkel discloses processing the logs of memory and network events collected by the system (corresponding to “analyzing the record of events”) in the policy engine where the VMM determines whether malicious actions are being taken and responding accordingly. EX1003, ¶¶271; EX1006, p.7,13. For the reasons discussed re 1[f], Garfinkel discloses this limitation. EX1003, ¶¶217, 271-276.

i. a sixth instruction set configured to modify the set of behavior rules so as to prevent the pestware related events; and

As addressed above, a PHOSITA would have understood that Garfinkel's disclosure of processes running in Linux discloses “instruction sets” that cause components to perform the disclosed processes. EX1003, ¶275. The rest of this limitation is equivalent to claim element 1[g]. *Id.* at ¶271. For the reasons discussed re 1[b], Garfinkel discloses this limitation. EX1003, ¶¶228, 271-276.

- j. a seventh instruction set configured to scan data in a registry of the computer for pestware during the second period in the boot sequence.**

As addressed above, a PHOSITA would have understood that Garfinkel's disclosure of processes running in Linux discloses “instruction sets” that cause components to perform the disclosed processes. EX1003, ¶275. The rest of this limitation is equivalent to claim element 1[h]. *Id.* at ¶271. For the reasons discussed re 1[h], Garfinkel in view of Hoglund discloses this limitation. EX1003, ¶¶237, 271-276.

For the foregoing reasons, a PHOSITA would have found claim 6 to be obvious based on the Garfinkel-Hoglund combination. Accordingly, claim 6 should be found unpatentable over Garfinkel-Hoglund under 35 U.S.C. § 103.

7. Claim 7

The system of claim 6, wherein the memory further comprises an eighth instruction set configured to launch, after an operating system is initiated, a pestware management engine; and wherein the set of behavior rules includes behavior rules compiled by the pestware management engine.

As addressed above, a PHOSITA would have understood that Garfinkel's disclosure of processes running in Linux discloses “instruction sets” that cause components to perform the disclosed processes. EX1003, ¶275. The rest of claim 7 is equivalent to claim 2. *Id.* at ¶271. For the reasons discussed re claim 2,

Garfinkel in view of Hoglund discloses claim 7. EX1003, ¶¶244, 271-276. As shown, the Garfinkel-Hoglund combination discloses all elements recited in claim 6 from which claim 7 depends. Accordingly, claim 7 should be found invalid under 35 U.S.C. § 103 based on Garfinkel and Hoglund.

8. Claim 8

The system of claim 1, wherein the record of events includes information selected from a group consisting of: process identification information, file identification information, and hook generation information.

While system claim 8 depends from method claim 1, rendering the claim ambiguous, the claim language after “system” is identical to claim 3. Therefore, for the reasons discussed re claim 3, Garfinkel or Garfinkel in view of Hoglund discloses claim 8. EX1003, ¶¶257, 271-276. As shown, the Garfinkel-Hoglund combination discloses all elements recited in claim 1 (as well as claim 6) from which claim 8 depends. Accordingly, claim 8 should be found invalid under 35 U.S.C. § 103 based on Garfinkel and Hoglund.

9. Claim 9

The system of claim 6, wherein the memory further comprises: a ninth instruction set configured to manage pestware-related events after the computer becomes configured to run native applications.

As addressed above, a PHOSITA would have understood that Garfinkel's disclosure of processes running in Linux discloses “instruction sets” that cause

components to perform the disclosed processes. EX1003, ¶275. The rest of claim 9 is equivalent to claim 4. *Id.* at ¶271. For the reasons discussed re claim 4, Garfinkel discloses claim 9. EX1003, ¶¶262, 271-276. As shown, the Garfinkel-Hoglund combination discloses all elements recited in claim 6 from which claim 9 depends. Accordingly, claim 9 should be found invalid under 35 U.S.C. § 103 based on Garfinkel and Hoglund.

10. Claim 10

The system of claim 6, wherein the first instruction set is further configured to monitor the boot sequence while boot drivers are initiated.

As addressed re claim element 6[b], a PHOSITA would have understood that Garfinkel's disclosure of processes running in Linux discloses “instruction sets” that cause components to perform the disclosed processes. EX1003, ¶275. The rest of claim 10 is equivalent to claim 5. *Id.* at ¶271. For the reasons discussed re claim 5, Garfinkel discloses claim 10. EX1003, ¶¶262, 271-276. As shown, the Garfinkel-Hoglund combination discloses all elements recited in claim 6 from which claim 10 depends. Accordingly, claim 10 should be found invalid under 35 U.S.C. § 103 based on Garfinkel and Hoglund.

11. Claim 11

a. A non-transitory tangible computer-readable storage medium comprising program instructions for:

The preamble of claim 11 is similar to the preamble of claim 1, replacing “method of” with “non-transitory tangible computer-readable storage medium comprising program instructions.” As discussed regarding 1[a], Garfinkel discloses a virtual machine monitor, which is a computer, that used VMware plus customized software components. EX1003, ¶275; EX1006, p.7. A PHOSITA would have known that the Garfinkel disclosed program instructions (e.g., VMware) executing on a computer are loaded from a non-transitory tangible computer readable storage medium, such as a hard drive. EX1003, ¶275.

Therefore, Garfinkel teaches the preamble.

b. monitoring events during a boot sequence of a computer;

This limitation is identical to claim element 1[b]. *Id.* at ¶271. Therefore, for the reasons discussed re 1[b], Garfinkel alone or in combination with Hoglund discloses this limitation. EX1003, ¶¶176, 271-276

c. managing pestware-related events during a first period in a boot sequence of the computer, the first period in the boot sequence occurring before the computer becomes configured to run native applications, before a WIN32 subsystem is loaded, and after a kernel is loaded;

This limitation is similar to claim element 1[c] except that this limitation recites “before a WIN32 subsystem is loaded,” and is almost identical to claim element 6[e]. *Id.* at ¶¶271, 279. As described re 1[c] and 6[e], Garfinkel discloses monitoring and blocking events in Linux-based rootkits during a boot period that would lead to intrusions (corresponding to “managing pestware-related events during a first period in a boot sequence of the computer”). EX1003, ¶271; EX1006, pp.9-10. A PHOSITA would have looked to Hoglund because it discloses that Windows-based rootkits behaved like Linux-based rootkits in that they typically activate during the boot sequence. EX1003, ¶¶184-185. Windows kernel drivers are loaded before the Win32 Subsystem. Thus, a PHOSITA would have understood that any rootkit installation in a kernel driver would come after the kernel was loaded but before the Win32 Subsystem was loaded and native applications are configured to run. EX1003, ¶185. For these reasons, Garfinkel in view of Hoglund discloses this limitation. EX1003, ¶¶185, 187, 271-276.

d. managing pestware-related events in accordance with a set of behavior rules during a second period in the boot sequence occurring when the computer is configured to run native applications;

This limitation is identical to claim element 1[d]. *Id.* at ¶271. For the reasons discussed re 1[d], Garfinkel discloses this limitation. EX1003, ¶¶200, 271-276.

**e. generating, in response to the monitoring, a record of events,
the record of events including the pestware-related events;**

This limitation is identical to claim element 1[e]. *Id.* at ¶271. As described re 1[e] Garfinkel discloses an event-driven inspection that may be based on a log of events to reduce computing load. EX1003, ¶271; EX1006, p.13. For the reasons discussed re 1[e], Garfinkel discloses this limitation. EX1003, ¶¶206, 271-276.

**f. analyzing the record of events so as to identify the pestware-
related events;**

This limitation is identical to claim element 1[f]. *Id.* at ¶271. As described re 1[f] Garfinkel discloses processing the logs of memory and network events collected by the system (corresponding to “analyzing the record of events”) in the policy engine where the VMM determines whether malicious actions are being taken and responding accordingly. EX1003, ¶271; EX1006, p.7,13. For the reasons discussed re 1[f], Garfinkel discloses this limitation. EX1003, ¶¶217, 271-276.

**g. modifying the set of behavior rules so as to prevent the
pestware related events; and**

This limitation is identical to claim element 1[g]. *Id.* at ¶271. For the reasons discussed re 1[b], Garfinkel discloses this limitation. EX1003, ¶¶228, 271-276.

h. scanning data in a registry of the computer for pestware during the second period in the boot sequence.

This limitation is identical to claim element 1[h]. *Id.* at ¶271. For the reasons discussed re 1[h], Garfinkel in view of Hoglund discloses this limitation. EX1003, ¶¶237, 271-276.

For the foregoing reasons, a PHOSITA would have found claim 11 to be obvious based on the Garfinkel-Hoglund combination. Accordingly, claim 11 should be found unpatentable over Garfinkel-Hoglund under 35 U.S.C. § 103.

12. Claim 12

The non-transitory tangible computer-readable storage medium of claim 11 further comprising program instructions for launching, after an operating system is initiated, a pestware management engine; and wherein the set of behavior rules includes behavior rules compiled by the pestware management engine.

A PHOSITA would have known that the Garfinkel disclosed program instructions (e.g., VMware) executing on a computer are loaded from a non-transitory tangible computer readable storage medium, such as a hard drive. EX1003, ¶275. The rest of claim 12 is equivalent to claim 2. *Id.* at ¶271. For the reasons discussed re claim 2, Garfinkel in view of Hoglund discloses claim 12. EX1003, ¶¶244, 271-276. As shown, the Garfinkel-Hoglund combination discloses all elements recited in claim 11 from which claim 12 depends. Accordingly, claim 12 should be found invalid under 35 U.S.C. § 103 based on Garfinkel and Hoglund.

13. Claim 13

The non-transitory tangible computer-readable storage medium of claim 11, wherein the record of events includes information selected from a group consisting of: process identification information, file identification information, and hook generation information.

A PHOSITA would have known that the Garfinkel disclosed program instructions (e.g., VMware) executing on a computer are loaded from a non-transitory tangible computer readable storage medium, such as a hard drive. EX1003, ¶275. The rest of the claim language is equivalent to claim 3. Therefore, for the reasons discussed re claim 3, Garfinkel or Garfinkel in view of Hoglund discloses claim 13. EX1003, ¶¶257, 271-276. As shown, the Garfinkel-Hoglund combination discloses all elements recited in claim 11 from which claim 13 depends. Accordingly, claim 13 should be found invalid under 35 U.S.C. § 103 based on Garfinkel and Hoglund.

14. Claim 14

The non-transitory tangible computer-readable storage medium of claim 11 further comprising program instructions for managing pestware-related events after the computer becomes configured to run native applications.

A PHOSITA would have known that the Garfinkel disclosed program instructions (e.g., VMware) executing on a computer are loaded from a non-transitory tangible computer readable storage medium, such as a hard drive.

EX1003, ¶275. The rest of claim 14 is identical to claim 4. *Id.* at ¶271. For the reasons discussed re claim 4, Garfinkel discloses claim 14. EX1003, ¶¶262, 271-276. As shown, the Garfinkel-Hoglund combination discloses all elements recited in claim 11 from which claim 14 depends. Accordingly, claim 14 should be found invalid under 35 U.S.C. § 103 based on Garfinkel and Hoglund.

15. Claim 15

The non-transitory tangible computer-readable storage medium of claim 11, wherein the monitoring includes monitoring the boot sequence while boot drivers are initiated.

A PHOSITA would have known that the Garfinkel disclosed program instructions (e.g., VMware) executing on a computer are loaded from a non-transitory tangible computer readable storage medium, such as a hard drive.

EX1003, ¶275. The rest of claim 15 is identical to claim 5. *Id.* at ¶271. For the reasons discussed re claim 5, Garfinkel discloses claim 15. EX1003, ¶¶262, 271-276. As shown, the Garfinkel-Hoglund combination discloses all elements recited in claim 11 from which claim 15 depends. Accordingly, claim 15 should be found invalid under 35 U.S.C. § 103 based on Garfinkel and Hoglund.

16. Claim 16

The method of claim 2, further comprising modifying, by the pestware management engine, the behavior rules utilized by a kernel-level monitor, wherein the kernel-level monitor manages pestware-related events in the first and second periods in the boot sequence.

Garfinkel discloses an IDS (corresponding to “pestware management engine”) operated as part of the VMM that monitors the hosted machine (guest OS). EX1003, ¶¶283; EX1006, p.7. The IDS also manages and operates the policy modules that enforce policies on hardware and software events of the hosted machine. EX1003, ¶¶283; EX1006, pp.8-9. The VMM shadows and inspects the page tables of the physical machine and inspects call tables which are at the kernel level (corresponding to “the kernel-level monitor manages pestware-related events in the first and second periods in the boot sequence”) based on the policies in the IDS. EX1003, ¶¶283-284; EX1006, p.10. Further, Garfinkel discloses:

The lie detector module works by directly inspecting hardware and kernel state, and by querying the host system through user-level programs (e.g. ps, ifconfig, netstat) via a remote shell. If it detects conflicts between these two views (i.e. the system is lying), it reports the presence of malicious tampering.

EX1006, p.9.

Therefore, Garfinkel discloses monitors at the kernel level looking at behavioral events and inconsistencies. EX1003, ¶¶191

Additionally, as explained regarding claim 1[g], Garfinkel discloses that the different modules enable different detections and different levels of intrusion containment throughout the hosted machine. EX1003, ¶¶288-290; EX1006, pp.8-10, Table-1. For example, the rootkit “cdoor” would only be detected by raw socket detector, whereas the user level rootkit “lrk5” can be detected by four different modules. *Id.* Table 1 of Garfinkel below also discloses that prevention (P) and detection (D) are separate, and that an event may be detected by one module (see columns) and prevented by another.

Name	Description	nic	raw	sig	int	lie	mem
cdoor	Stealth user level remote backdoor		D				
t0rn	Precompiled user level rootkit			D		D	
Ramen	Linux Worm			D			
lrk5	Source based user level rootkit	P		D	D	D	
knark-0.59	LKM based kernel backdoor/rootkit			D		D	P
adore-0.42	LKM based kernel backdoor/rootkit			D		D	P
dsniff 2.4	All-purpose packet sniffer for switched networks	P					
SUCKIT	/dev/kmem patching based kernel backdoor			D		D	P

Table 1. Results of Livewire policy modules against common attacks. Within the grid, “P” designates a prevented attack and “D” a detected attack.

Garfinkel discloses that policy modules may be used alone or together with others and that the policy modules can be turned on and off or managed such that they work together. EX1003, ¶¶288-291; EX1006, pp.8-10, Table-1. A PHOSITA would have understood that running a policy module as a standalone module would impose a different set of rules than operating them “in concert with other policy modules” (corresponding to “modifying the set of behavior rules”). EX1003, ¶¶287. Therefore, a PHOSITA would have understood that changing the

different modes of the policy engine to incorporate different prevention and detection capabilities corresponds to this limitation. EX1003, ¶¶290-296.

Therefore, Garfinkel discloses the limitations of claim 16. EX1003, ¶297.

As shown, the Garfinkel-Hoglund combination discloses all elements recited in claims 1 and 2 from which claim 16 depends. Accordingly, claim 16 should be found invalid under 35 U.S.C. § 103 based on Garfinkel and Hoglund.

17. Claim 17

The method of claim 1, further comprising modifying the set of behavior rules based on the scanned registry so as to prevent the pestware-related events.

As discussed with respect to limitation 1[g], Garfinkel discloses modifying the behavior rules contained within its policy modules such as the Memory Access Enforcer module. EX1003, ¶¶300; EX1006, pp.8. Specifically, as new types of attacks (i.e., pestware-related events) are identified researchers need to modify the rules to address them. *Id.*

Further, Hoglund teaches that rootkits often need to insert registry keys or modify the windows registry so that the malware can get further access (e.g., memory write capability). EX1003, ¶¶301; EX1007, pp.38,78-79,96-97. Hoglund discloses that to defend against such intrusions, the registry should be hooked and monitored and discloses a scanning tool (“RootkitRevealer”) that parses files and the Windows registry for malicious files (corresponding to “scanning data in a registry of the computer”). EX1003, ¶¶303; EX1007, pp.347,359. A PHOSITA

would have been motivated to combine the scanning functionality during boot of Garfinkel with the registry scanning of Hoglund for hosted Windows machines. EX1003, ¶¶304.

A PHOSITA would then have been motivated to block newly identified behavior based on findings in the registry in order to prevent the behavior of the rootkit. *Id.*

Therefore, Garfinkel discloses the limitations of claim 17. EX1003, ¶305. As shown, the Garfinkel-Hoglund combination discloses all elements recited in claim 1 from which claim 17 depends. Accordingly, claim 17 should be found invalid under 35 U.S.C. § 103 based on Garfinkel and Hoglund.

18. Claim 18

The method of claim 1, wherein the pestware-related invents include events associated with at least one of loading pestware, executing pestware, and one or more behaviors of pestware.

Garfinkel discloses detecting via its memory protection enforcer several memory-related events that are either the loading of pestware or the executing of pestware. EX1003, ¶309; EX1006, pp.8,10. Garfinkel disclosed blocking the execution and installation of malware in this manner (i.e., when the malware executes, it attempts to install a more permanent copy of itself, which is blocked).

Our Memory Protection Enforcer (“mem”) was able to detect and prevent the installation of all of our kernel backdoors. knark and adore were stopped

by blocking their attempt to modify sys call table. SUCKIT was stopped by blocking its attempt to modify the interrupt dispatch table.

EX1006, p.10.

A PHOSITA would have understood that the blocking of the rootkits show that the detected events have been associated with the execution and loading of malware. EX1003, ¶¶310-312; EX1006, pp.8-10. That is, if an event triggers the Memory Protection Enforcer to block the execution or loading of the rootkit, the event that triggered this action is associated with that rootkit as shown in Table 1 of Garfinkel, for example. EX1003, ¶¶310; EX1006, Table-1.

Therefore, Garfinkel discloses the limitations of claim 18. EX1003, ¶313. As shown, the Garfinkel-Hoglund combination discloses all elements recited in claim 1 from which claim 18 depends. Accordingly, claim 18 should be found invalid under 35 U.S.C. § 103 based on Garfinkel and Hoglund.

19. Claim 19

The system of claim 6, wherein the pestware-related invents include events associated with at least one of loading pestware, executing pestware, and one or more behaviors of pestware.

Claim 19 is identical to claim 18 except that it depends from Claim 6 instead of claim 1. For the reasons discussed re claim 18, Garfinkel discloses claim 19 limitations. EX1003, ¶¶316-317. As shown, the Garfinkel-Hoglund combination discloses all elements recited in claim 6 from which claim 19 depends.

Accordingly, claim 19 should be found invalid under 35 U.S.C. § 103 based on Garfinkel and Hoglund.

20. Claim 20

The non-transitory tangible computer-readable storage medium of claim 11, wherein the pestware-related invents include events associated with at least one of loading pestware, executing pestware, and one or more behaviors of pestware.

Claim 20 is identical to claim 18 except that it depends from Claim 11 instead of claim 1. For the reasons discussed re claim 18, Garfinkel discloses claim 20 limitations. EX1003, ¶¶320-321. As shown, the Garfinkel-Hoglund combination discloses all elements recited in claim 11 from which claim 20 depends. Accordingly, claim 20 should be found invalid under 35 U.S.C. § 103 based on Garfinkel and Hoglund.

VI. NO BASIS FOR DISCRETIONARY DENIAL

There is no basis to deny institution under 35 U.S.C. § 325(d).

Petitioner stipulates that, if the Board institutes inter partes review, Petitioner will not seek resolution in the District Court of any ground of invalidity for the '505 Patent that utilizes Garfinkel and Hoglund—the prior art references relied upon in the grounds of the instant petition. Petitioner's proposal is comprehensive and addresses the concern of narrowing the issues, as this stipulation ensures there is no overlap between the '505 patent invalidity theories

before the Board and at issue in the District Court. Petitioner's proposed stipulation is far more restrictive than a *Sand Revolution* stipulation, which excludes only the same grounds advanced in the IPR, allowing the Petitioner to rely on the same references in the two tribunals so long as the reliance is not identical. *Sand Revolution II, LLC v. Cont'l Intermodal Group – Trucking LLC*, IPR2019-01393, Paper 24 at 11-12 (PTAB June 16, 2020) (informative). Accordingly, this factor weighs heavily in favor of institution.

There is parallel court litigation, so *Fintiv* factors are addressed. First, no stay has been entered in the parallel litigation. Second, trial is currently set for August 19, 2024. Trial is currently set for five different cases on that same date in the same Court against Webroot, of which Trend Micro's case is just one. Trial will not proceed on all five cases on the same date, and it is still unclear which, if any, parties will proceed to trial on that date and when trial would be scheduled for the remaining defendants. Thus, it is unclear the exact date Trend Micro's case will go to trial, and it is possible for this inter partes review to reach final written decision before then. Third, it is early in the parallel litigation as the parties are still in fact discovery, which is not set to close until January 18, 2024, after which the parties still have expert discovery, dispositive motions, and pretrial disclosures before reaching trial. Fourth, Petitioner has identified the Garfinkel and Hoglund prior art in Defendant's invalidity contentions filed in the parallel litigation.

Finally, Petitioner and Webroot are parties in the parallel litigation as identified in the Mandatory Notices.

Discretionary denial is strongly disfavored here because the “same or substantially the same art or arguments” were not presented to the Office, and the Office “erred in a manner material” to patentability by not asserting the Garfinkel and Hoglund references. *See Advanced Bionics, LLC v. MED-EL Elektromedizinische Gerate GmbH*, IPR2019-01469, Paper 6, 8-9 (Feb. 13, 2020). Analysis of this issue is guided by the “*Becton*” factors. *Advanced Bionics* at 9-11.

Garfinkel and Hoglund were not cited, individually or in combination, in any of the office communications. *See* EX1002. Regarding *Becton* factors (a) (material differences between asserted art and prior art) and (b) (cumulative nature of the asserted art and prior art), Garfinkel and Hoglund are materially different from the combination of references asserted in Office communications because Garfinkel and Hoglund disclose the independent claim limitations that the data received from remote devices includes information about the behavior of the object and the mask is generated based on normal behavior of the object determined from the received data including behavior information. Thus, the individual references of Garfinkel and Hoglund are not merely cumulative of references the Office

relied. Regarding *Becton* factor (d) (extent of overlap), the arguments made in *supra* do not overlap with the arguments made during examination.

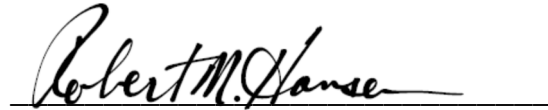
Even if the Board finds that the “same or substantially the same” art or arguments were presented to the Office, discretionary denial under § 325(d) is still not proper because the Office has made a “material error” in lieu of the Garfinkel and Hoglund references.

Regarding *Becton* factors (c) (evaluation of art during examination), neither of Garfinkel nor Hoglund were used in any claim rejections, weighing strongly against exercising discretion. *See Celco Partnership v. Huawei Device Co.*, IPR2020-01117, Paper 10, 13 (Feb. 3, 2021). Regarding *Becton* factor (e) (examiner error) and (f) (other facts), this petition specifically points out and explains how Garfinkel and Hoglund discloses all limitations of the claims.

VI. CONCLUSION

Inter partes review of claims 1-20 of U.S. Patent No. 8,856,505 is requested based on the grounds described above.

Respectfully submitted,
The Marbury Law Group, PLLC

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

Robert M. Hansen, Reg. No. 43,656
Scott B. Amankwatia, Reg. No. 78,226
Matthew J. Anderson, Reg. No. 78,257

Attorneys for Petitioner

Date: September 7, 2023

The Marbury Law Group, PLLC
11800 Sunrise Valley Drive
15th Floor
Reston, VA 20191
Tel: (703) 391-2900

CERTIFICATION OF SERVICE (37 C.F.R. §§ 42.6(e), 42.105(a))

The undersigned hereby certifies that the above-captioned **PETITION FOR INTER PARTES REVIEW OF U.S. PATENT NO. 8,856,505**, Petitioner's power of attorney, and all associated exhibits were served in their entireties on September 7, 2023, on the following party via Express Mail:

**Sprinkle IP Law Group
1301 W. 25th Street, Suite 400
Austin, TX 78705**

A handwritten signature in black ink, reading "Robert M. Hansen", written over a horizontal line.

Robert M. Hansen, Reg. No. 43,656
Attorney for Petitioner

Date: September 7, 2023

The Marbury Law Group, PLLC
11800 Sunrise Valley Drive
15th Floor
Reston, VA 20191
Tel: (703)-391-2900

CERTIFICATE OF COMPLIANCE WITH TYPE-VOLUME LIMITATION

Pursuant to 37 C.F.R. § 42.24 *et seq.*, the undersigned certifies that this Petition complies with the 14,000-word type-volume limitation. The Petition contains **9,653** words, excluding the table of contents, table of authorities, exhibit list, mandatory disclosures, words in copies of reference figures, signature block, certificates and attached exhibits of the Petition.

The Marbury Law Group, PLLC

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

Robert M. Hansen, Reg. 43,656
Attorney for Petitioner

Date: September 7, 2023

The Marbury Law Group, PLLC
11800 Sunrise Valley Drive
15th Floor
Reston, VA 20191
Tel: 703-391-2900