

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

INTERNATIONAL BUSINESS MACHINES CORPORATION,

Petitioner,

v.

CROGA INNOVATIONS LTD.,

Patent Owner.

Case No. IPR2025-00379

U.S. Patent 11,178,104

DECLARATION OF DR. WENKE LEE

TABLE OF CONTENTS

	<u>Page</u>
I. OVERVIEW OF THE TECHNOLOGY.....	2
A. Background Technology as Stated in the '104 Patent	2
B. Sandboxed Computing Environments Were Well Known Before the '104 Patent Priority Date	4
C. Cloud Computing Was Well Known Before the '104 Patent Priority Date	7
D. Isolation of Computing Environments and Networks Using Firewalls and Proxy Servers Was Well Known Before the '104 Patent Priority Date	7
E. Memory Partitioning Was Well Known Before the '104 Patent Priority Date	9
F. Virtual Private Networks and VPN Tunnels Were Well Known Before the '104 Patent Priority Date	10
G. Authentication of Computing Environments Was Well Known Before the '104 Patent Priority Date	13
II. THE '104 PATENT.....	16
A. Overview	16
B. Prosecution History	22
1. Non-Final Office Action Dated May 21, 2020	23
2. Office Action Response Dated August 14, 2020	25
3. Final Office Action Dated September 9, 2020	27
4. Office Action Response and RCE Dated March 8, 2021	29
5. Non-Final Office Action Dated March 23, 2021	31
6. Office Action Response Dated September 14, 2021	32
7. Notice of Allowance Dated October 5, 2021	32
C. Priority Date	33
D. Challenged Claims	33
III. LEVEL OF ORDINARY SKILL IN THE ART	33
IV. CLAIM CONSTRUCTION	37

A.	“sandboxed computing environment” (all challenged claims)	38
1.	Proper Construction Based on a POSITA’s Understanding	38
2.	Alternative Construction Based on Patent Owner’s Position.....	38
B.	“authentication device” (all challenged claims).....	40
V.	INVALIDITY GROUNDS.....	42
A.	Ground 1: Claims 1-3, 8, 10, 11 and 17-19 Are Obvious in View of Hoy and Pratt.....	42
1.	Combination of Hoy and Pratt	42
a.	Summary of Hoy	42
b.	Summary of Pratt.....	48
c.	A POSITA Would Have Been Motivated to Combine the Authentication Process of Pratt with the Computer System of Hoy	52
d.	A POSITA Would Have Had a Reasonable Expectation of Success in Combining the Authentication Process of Pratt with the Computer System of Hoy	55
2.	Independent Claim 1	56
3.	Dependent Claim 2	83
4.	Dependent Claim 3	85
5.	Dependent Claim 8	87
6.	Independent Claim 10.....	90
7.	Dependent Claim 11	96
8.	Dependent Claim 17	97
9.	Independent Claim 18.....	98
10.	Dependent Claim 19	99
B.	Ground 2: Claims 1-3, 8, 10, 11 and 17-19 Are Obvious in View of Hoy, Pratt, and Krasin.....	100
1.	Combination of Hoy and Pratt	100
2.	Combination of Hoy and Pratt, with Krasin	100

a.	Summary of Krasin.....	100
b.	A POSITA Would Have Been Motivated to Combine the Sandbox Environment of Krasin with the Computer System of Hoy/Pratt.....	104
c.	A POSITA Would Have Had a Reasonable Expectation of Success in Combining the Sandbox Environment of Krasin with the Computer System of Hoy/Pratt	109
3.	Claim Analysis.....	110
C.	Ground 3: Claims 1-3, 8, 10, 11, and 17-19 Are Anticipated by Benoit	112
1.	Summary of Benoit	112
2.	Independent Claim 1	114
3.	Dependent Claim 2	138
4.	Dependent Claim 3	141
5.	Dependent Claim 8	143
6.	Independent Claim 10.....	145
7.	Dependent Claim 11	153
8.	Dependent Claim 17	154
9.	Independent Claim 18.....	155
10.	Dependent Claim 19	156
D.	Ground 4: Claims 1-3, 8, 10, 11, and 17-19 Are Obvious in View of Benoit and Hunt	157
2.	Combination of Benoit and Hunt.....	158
a.	Summary of Hunt	158
b.	A POSITA Would Have Been Motivated to Combine Sandbox Implementation Details of Hunt with the Computer System of Benoit	162
c.	A POSITA Would Have Had a Reasonable Expectation of Success in Combining the Sandbox Implementation Details of Hunt with the Computer System of Benoit.....	165

3.	Claim Analysis.....	166
a.	“a first memory space” and “a second memory space” in elements 1c, 1d, 18b, and 18d.....	167
b.	“isolate the isolated computing environment from the workspace using an internal isolation firewall” (elements 1f, 10d, and 18f).....	168
c.	Dependent Claim 8	173
d.	The Combination Would Not Have Changed the Principle of Operation for Either Benoit or Hunt.....	177
E.	Ground 5: Claims 1-3, 8, 10, 11 and 17-19 Are Obvious in View of Benoit, Hunt, and Pratt.....	179
1.	Combination of Benoit and Hunt.....	179
2.	Combination of Benoit and Hunt, with Pratt	179
b.	A POSITA Would Have Had a Reasonable Expectation of Success in Combining the Authentication Process of Pratt with the Computer System of Benoit/Hunt	182
3.	Claim Analysis.....	183
VI.	BACKGROUND AND QUALIFICATIONS.....	186
VII.	COMPENSATION.....	189
VIII.	MATERIALS AND OTHER INFORMATION CONSIDERED.....	189
IX.	UNDERSTANDING OF THE LAW	189
X.	RESERVATION OF RIGHTS.....	189
	APPENDIX 1: CURRICULUM VITAE OF WENKE LEE.....	191
	APPENDIX 2: MATERIALS CONSIDERED IN THE PREPARATION OF THIS DECLARATION.....	242
	APPENDIX 3: CHALLENGED CLAIMS.....	243
	APPENDIX 4: UNDERSTANDING OF THE LAW	248

I, Wenke Lee, declare as follows:

1. My name is Wenke Lee. I am a Regents' Professor and the John P. Imlay Jr. Chair in the School Cybersecurity and Privacy of the College of Computing at the Georgia Institute of Technology. I was the executive director of the Institute for Information Security & Privacy from 2015 to 2021. I have prepared this report as an expert witness retained by International Business Machines Corporation. In this report, I give my opinions as to whether certain claims of U.S. Patent No. 11,178,104 ("the '104 patent") are invalid. I provide technical bases for these opinions as appropriate.

2. This report contains statements of my opinions formed to date and the bases and reasons for those opinions. I may offer additional opinions based on further review of materials in this case, including opinions and/or testimony of other expert witnesses. I make this declaration based upon my own personal knowledge and, if called upon to testify, would testify competently to the matters contained herein.

3. I have reviewed each of the following documents.

EXHIBIT	DESCRIPTION
1001	U.S. Patent No. 11,178,104 to Coleman et al. ("the '104 Patent")
1002	Prosecution history of the '104 Patent
1004	U.S. Patent Publication No. 2013/0318594 to Hoy et al., published on November 28, 2013 ("Hoy")

EXHIBIT	DESCRIPTION
1005	U.S. Patent Publication No. 2015/0096031 to Benoit et al., published on April 2, 2015 (“ Benoit ”)
1006	U.S. Patent No. 10,798,077 to Pratt et al., filed on January 23, 2015 (“ Pratt ”)
1007	U.S. Patent Publication No. 2015/0278513 to Krasin et al., published on October 1, 2015 (“ Krasin ”)
1008	U.S. Patent Publication No. 2012/0017213 to Hunt et al., published on January 19, 2012 (“ Hunt ”)
1009	Selected pages from MICROSOFT COMPUTER DICTIONARY, FIFTH EDITION (Microsoft Press, 2002)
1010	<i>Plaintiff Croga Innovations Ltd.’s P.R. 4-2 Disclosure</i> , in <i>Croga Innovations Ltd. v. International Business Machines Corporation</i> , No. 2:23-cv-00634-JRG (U.S. District Court, Eastern District of Texas)

I. OVERVIEW OF THE TECHNOLOGY

A. Background Technology as Stated in the ’104 Patent

4. The ’104 patent pertains to “a sandbox based network isolation system” to protect assets on a computer network from harmful data residing on the Internet. Ex-1001, Abstract. The Background section of the ’104 patent describes that the Internet provides access to systems from which a host computer system may retrieve beneficial and harmful data. Harmful data includes virus software, which the ’104 patent refers to as “malware.” A computer user may download malware to a host computer system unintentionally, without the user’s knowledge. Ex-1001, 1:13-21.

5. The ’104 patent refers to a host computer system that has downloaded malware as an “infected host computer system.” The Background of the ’104

patent explains that an infected host computer system may result in security losses, reduction of efficiency, and malfunctions. Security losses may include data loss, the installation of additional malware onto the infected host computer system, and the loss of command and control. Reduction of efficiency may result from additional processor demand, additional memory usage, and additional network access by the executing malware. Ex-1001, 1:22-34.

6. The Background of the '104 patent describes that malware may compromise the security and privacy of the user of the infected host computer system. Malware may attack information located on the infected host computer system, information stored on a local area network, and information communicated via a local area network. According to the '104 patent, malware may use an infected host computer as a vehicle to stage an attack on other network resources. Such an attack is often undetected by the user of the host computer system and network administrators. Ex-1001, 1:35-45.

7. The Background of the '104 patent further describes that an infected host computer system may lose command and control. For example, malware may perform a reconnaissance of the infected host computer system, determine system privileges, access local files, access security tokens, and attempt infection or access of other network resources. Ex-1001, 1:46-54.

8. While the Background of the '104 patent outlines the problems that can arise when a host computer system becomes infected with malware, the patent does not discuss the many solutions that existed before the stated priority date of the patent (September 26, 2017) to protect network assets from harmful data residing on the Internet. Indeed, the network security field was crowded with solutions for protecting network assets before September 26, 2017, including sandbox-based network isolation solutions, and these solutions were well known to people of ordinary skill in the art in the network security field.

B. Sandboxed Computing Environments Were Well Known Before the '104 Patent Priority Date

9. Sandbox-based network isolation solutions were well known more than a decade before the stated priority date of the '104 patent, such that the term “sandbox” received its own entry in technical computer dictionaries. For example, the Microsoft Computer Dictionary, Fifth Edition (Microsoft Press, 2002), which I am informed was cited and produced by the owner of the '104 patent as part of claim construction proceedings in related litigation, gives a definition of “sandbox” as a Java Virtual Machine “security area for downloaded (remote or untrusted) applets, an area in which such applets are confined and prevented from accessing system resources.” Ex-1009 (Microsoft Computer Dictionary at 463); Ex-1010 (Ex. A to Croga’s P.R. 4-2 Disclosure).

10. **Sandboxing** thus was a common security feature in computing platforms before the stated priority date of the '104 patent (September 26, 2017) to restrict applications to performing only certain activities and data access, e.g., access to memory and file systems authorized by the operating system. Sandboxing has been used for network security and fault isolation (e.g., limiting the impact of runtime errors of untrusted code to isolated segments of memory) since at least 1993.¹ All major computing platforms, including Windows, MacOS, iOS, Android, and Linux, have long implemented sandboxing technologies to isolate applications from each other and from the operating systems to provide runtime security protection, including before the stated priority date of the '104 patent.

11. **Virtual machines** are one way to implement “sandbox”-like functionality. Virtual machines have been known for decades and well before the stated priority date of the '104 patent, including multiple virtual machines running on the same physical computer. The concept of virtualization and the implementation of virtual machines date as far back as IBM products in the late 1960s (e.g., IBM virtual machine operating systems on IBM System/370). Before the stated priority date of the '104 patent, people of ordinary skill in the field of

¹ Robert Wahbe, Steven Lucco, Thomas E. Anderson, and Susan L. Graham, *Efficient software-based fault isolation*, Proceedings of the Fourteen ACM Symposium on Operating Systems Principles, 1993.

network security considered virtual machines to be a form of sandboxing because a virtual machine isolates the host system on a physical computer from the guest system in the virtual machine (and also isolates one virtual machine from another). That is, the guest system is separated from the host system and has no direct access to the memory and other resources of the host system. Before the stated priority date of the '104 patent, it was well known to people of ordinary skill in the field of network security how to create and run virtual machines on a physical machine using a hypervisor or virtual machine monitor, as well as how to monitor the performance and security of the virtual machines.

12. A sandbox does not require its own operating system, and this was also true before the stated priority date of the '104 patent. As I described above, a sandbox is a limited-purpose isolated computing environment that has restricted access to the host system. While a virtual machine virtualizes an entire computer system, a sandbox is restricted to run only the specified applications or code snippet in the sandboxed environment. A virtual machine may be used as a sandboxed computing environment, but in many instances, this is not necessary or desirable, as running and creating a virtual machine incurs significant overhead, and for many use-cases, a sandbox is sufficient, as it is not needed to virtualize an entire computing system. People of ordinary skill in the field of network security

before the stated priority date of the '104 patent understood these benefits and drawbacks of sandboxes and virtual machines.

C. Cloud Computing Was Well Known Before the '104 Patent Priority Date

13. Before the stated priority date of the '104 patent, people of ordinary skill in the network security field were also highly familiar with **cloud computing**, which included such well-known services as Google's Gmail and Google Docs. The concept of cloud computing dates to the 1960s in the form of remote job submissions and time-sharing of mainframe computers. Cloud providers, including entities that make Internet-facing computing resources available to customers, emerged in the early 2000s, with Amazon, Microsoft, and Google as prime examples.

D. Isolation of Computing Environments and Networks Using Firewalls and Proxy Servers Was Well Known Before the '104 Patent Priority Date

14. **Firewalls** were developed well before the stated priority date of the '104 patent because of the need to control what data could be allowed to go from a local network onto the Internet and come from the Internet into the network. For example, firewalls in 1989 could already “screen packets based not only on destinations, but on sources or source-destination pairs,” and with Cisco firewall systems in 1989 “one could deny access to all but one host on a particular

network.”² That is, packet filtering rules, i.e., rules for blocking or admitting packets, were available before 1989, and firewalls were developed to implement such rules. Firewall technologies continued to evolve over the following decades. For example, before the stated priority date of the ’104 patent, there were session-filtering firewalls that inspected information from multiple packets of a session, and application-level gateways and proxies that examined application-specific data.

15. Among the types of firewalls that were developed well before the stated priority date of the ’104 patent were:

- **enterprise firewalls**, which were usually deployed at the entry point to an enterprise network to inspect traffic between the computers in the enterprise network and the Internet;
- **host-based firewalls**, which were deployed at each computer to inspect traffic to and from the computer; and
- **internal isolation firewalls**, which were deployed on a host and could prevent an isolated environment, e.g., a sandbox, from communicating with processes outside of the isolated environment.

² Jeffrey C. Mogul, *Simple and Flexible Datagram Access Controls for Unix-based Gateways*, WRL RESEARCH REPORT 89/4 (Western Research Laboratory, Mar. 1989).

16. Before the stated priority date of the '104 patent, people of ordinary skill in the network security field were also highly familiar with **web proxies** and **network proxies**, including proxies that could not only inspect web and network traffic between an internal network and the Internet, but that could also act as an Internet site or destination to the internal network and vice versa (as the internal network to an Internet site or destination). The concept of a network proxy dates back to the 1970s, and web proxies have as long a history as the Internet itself.³

17. Using firewalls and proxies, **isolation of computing environments and networks** could be accomplished by people of ordinary skill in the network security field before the stated priority date of the '104 patent by inspecting all traffic and blocking unwanted, unauthorized, and suspicious traffic to and from the computing environment or network.

E. Memory Partitioning Was Well Known Before the '104 Patent Priority Date

18. Before the stated priority date of the '104 patent, people of ordinary skill in the network security field were highly familiar with memory management techniques (and bookkeeping data) that divide memory into partitions or spaces for

³ Ari Luotonen and Kevin Altis, *World-Wide Web Proxies*, Apr. 1994, available at <https://web.archive.org/web/20161009061935/http://courses.cs.vt.edu/~cs4244/spring.09/documents/Proxies.pdf>.

use by different processes. Such **memory partitioning** is a concept of operating systems that date back to the 1960s, e.g., IBM's CP-67 operating system.

F. Virtual Private Networks and VPN Tunnels Were Well Known Before the '104 Patent Priority Date

19. Before the stated priority date of the '104 patent, people of ordinary skill in the network security field were also highly familiar with virtual private networks. A **virtual private network (VPN)** extends a private network (e.g., an enterprise network) across multiple other networks, e.g., the Internet, so that when a computer in the private network (e.g., a home or enterprise computer) is communicating with another computer (e.g., an enterprise computer at another company site) across other networks or the Internet, the traffic is traveling as if it were within the same private network. That is, the traffic is protected from other networks or the Internet.

20. This could be accomplished using a **VPN tunnel**, which people of ordinary skill in the art were highly familiar with before the stated priority date of the '104 patent. Such a VPN tunnel authenticates both the sending computing environment and the receiving computing environment, encrypts traffic from the sender, and guarantees that the encrypted traffic is delivered to the intended receiver and that only the receiver can decrypt the traffic. The history of VPNs

and VPN tunnels dates back decades, e.g., Microsoft's Point-to-Point Tunneling Protocol in the 1990s.⁴

21. Before the stated priority date of the '104 patent, people of ordinary skill in the network security field knew that VPN tunnels require systems to authenticate themselves to the VPN termination point in order to gain access to the termination point. That is how a VPN stays private—by making sure that a system trying to gain access to a VPN termination point is authorized to do so by presenting the proper authentication credentials. Before the stated priority date of the '104 patent, there were several protocols through which a computing environment attempting to communicate through a VPN tunnel could be authenticated:

- **Internet Protocol Security (IPsec)** included protocols to authenticate a network sender and receiver, negotiate cryptographic keys, and use the keys to authenticate and encrypt the IP traffic between the sender and receiver. The history of IPsec started in 1992 when the Internet Engineering Task Force (IETF) formed a working group.⁵ The original

⁴ K. Hamzeh et al., *Point-to-Point Tunneling Protocol (PPTP)*, Request for Comments: 2637, Network Working Group, Internet Engineering Task Force, July 1999.

⁵ IETF Datatracker, IP Security Protocol, Group history, available at <https://datatracker.ietf.org/wg/ipsec/history/>.

Requests for Comments (RFCs) 1825⁶, 1826⁷ and 1827⁸ pertaining to IPsec were published in 1995.

- **Secure Sockets Layer or SSL** (now Transport Layer Security or TLS) aimed to secure network- or Internet-facing applications, such as emails and web applications. SSL included protocols to authenticate the sender and receiver of a network session, set up cryptographic keys, and encrypt the traffic to protect data authenticity, confidentiality, and integrity. The history of TLS dates to IETF RFC 2246⁹ in 1999, an upgrade of SSL 3.0.
- **Hypertext Transfer Protocol Secure (HTTPS)** is an extension of the Hypertext Transfer Protocol (HTTP). HTTPS authenticates websites and encrypts HTTP traffic using TLS.

⁶ R. Atkinson, *Security Architecture for the Internet Protocol*, IETF Network Working Group, Request for Comments 1825 (Aug. 1995), available at <https://datatracker.ietf.org/doc/html/rfc1825>.

⁷ R. Atkinson, *IP Authentication Header*, IETF Network Working Group, Request for Comments 1826 (Aug. 1995), available at <https://datatracker.ietf.org/doc/html/rfc1826>.

⁸ R. Atkinson, *IP Encapsulating Security Payload (ESP)*, IETF Network Working Group, Request for Comments 1827 (Aug. 1995), available at <https://datatracker.ietf.org/doc/html/rfc1827>.

⁹ T. Dierks and C. Allen, *The TLS Protocol Version 1.0*, IETF Network Working Group, Request for Comments 2246 (Jan. 1999), available at <https://datatracker.ietf.org/doc/html/rfc2246>.

For example, HTTPS can authenticate an application website (e.g., Gmail) to a web client (e.g., a web browser on a user's computer) using a trusted public-key certificate and encrypts traffic in both directions. The history of HTTPS dates back to 1994, when Netscape Communications created HTTPS using SSL.

G. Authentication of Computing Environments Was Well Known Before the '104 Patent Priority Date

22. The history of authentication in the context of computing dates back decades, e.g., password-based authentication started in at least the 1960s, and cryptographic-based authentication (e.g., using symmetric or public key cryptography) started in the 1970s.

23. As discussed above, IPsec, SSL (TLS), and HTTPS all included an **authentication** step in their protocols well before the stated priority date of the '104 patent. The goal of these protocols was and is to protect the authenticity, confidentiality, and integrity of network connections, and the protection mechanisms specified in these protocols are based on cryptographic keys and algorithms. Therefore, when these protocols are used to protect a network connection, the first step is for both parties involved in the network connection to authenticate each other, i.e., to be sure they are communicating with their intended counterpart, and negotiate and agree on the cryptographic keys and algorithms.

24. This was the case with IPsec, SSL, and HTTPS before the stated priority date of the '104 patent. For IPsec, “[t]he set of security services offered includes ... data origin authentication”¹⁰ Among other parts of the IPsec specification, “the IPsec firewall function makes use of the cryptographically-enforced authentication and integrity provided for all IPsec traffic to offer better access control”¹¹ The authentication service in IPsec can be provided through the use of the Authentication Header (AH) traffic security protocol, which “offers integrity and data origin authentication, with optional (at the discretion of the receiver) anti-replay features.”¹² The Authentication Header protocol is described in the IETF Network Working Group Request for Comments 4302, describes the Authentication Header format and processing, among other things.¹³

25. Similarly, SSL includes the SSL handshake protocol, which “allows the server and client to authenticate each other and to negotiate an encryption algorithm and cryptographic keys before the application protocol transmits or

¹⁰ S. Kent and K. Seo, *Security Architecture for the Internet Protocol*, IETF Network Working Group, Request for Comments 4301 § 2.1 (Dec. 2005), available at <https://datatracker.ietf.org/doc/html/rfc4301>.

¹¹ *Id.*

¹² *Id.* § 3.2

¹³ S. Kent, *IP Authentication Header*, IETF Network Working Group, Request for Comments 4302 (Dec. 2005), available at <https://datatracker.ietf.org/doc/html/rfc4302>.

receives its first byte of data.”¹⁴ In SSL, “identity can be authenticated using asymmetric, or public key, cryptography.”¹⁵ Thus, “when an SSL client and server first start communicating, they agree on a protocol version, select cryptographic algorithms, optionally authenticate each other, and use public key encryption techniques to generate shared secrets.”¹⁶ Among other things, the SSL handshake protocol includes code for both server and client authentication.¹⁷

26. Before the stated priority date of the ’104 patent, people of ordinary skill in the network security field were highly familiar with using a trusted **authentication server** or device to perform authentication tasks on behalf of a computing environment. Of course, the identity of the authentication device must be verified and trusted in the first place, and people of ordinary skill knew that this could be accomplished by configuring traffic from a computer to go through a network or web proxy, which could act as an authentication device or could contact another verified and trusted authentication device.

¹⁴ A. Freier et al., The Secure Sockets Layer (SSL) Protocol Version 3.0, IETF Request for Comments 6101 § 1 (Aug. 2011), available at <https://datatracker.ietf.org/doc/html/rfc6101>.

¹⁵ *Id.*

¹⁶ *Id.* § 5.5.

¹⁷ *Id.*, Appx. A.4.2, A.5.

II. THE '104 PATENT

A. Overview

27. The '104 patent does not explain what problems, if any, existed in the prior solutions to protect network assets from harmful data residing on the Internet. Nor does the '104 patent explain what problems are solved by the alleged invention disclosed in the patent, beyond a high-level statement that the alleged invention protects computer assets from malware. In my opinion, a person of ordinary skill in the art in the network security field, reading the application for the '104 patent as of its stated priority date (September 26, 2017), would have been left wondering what had allegedly been invented.

28. As described in the '104 patent, the alleged invention uses “a sandbox based network isolation system” to protect network assets. Ex-1001, Abstract, 1:58-60, 4:25-27. An example of the sandbox-based network isolation system is shown in Figure 2 of the patent, reproduced below.

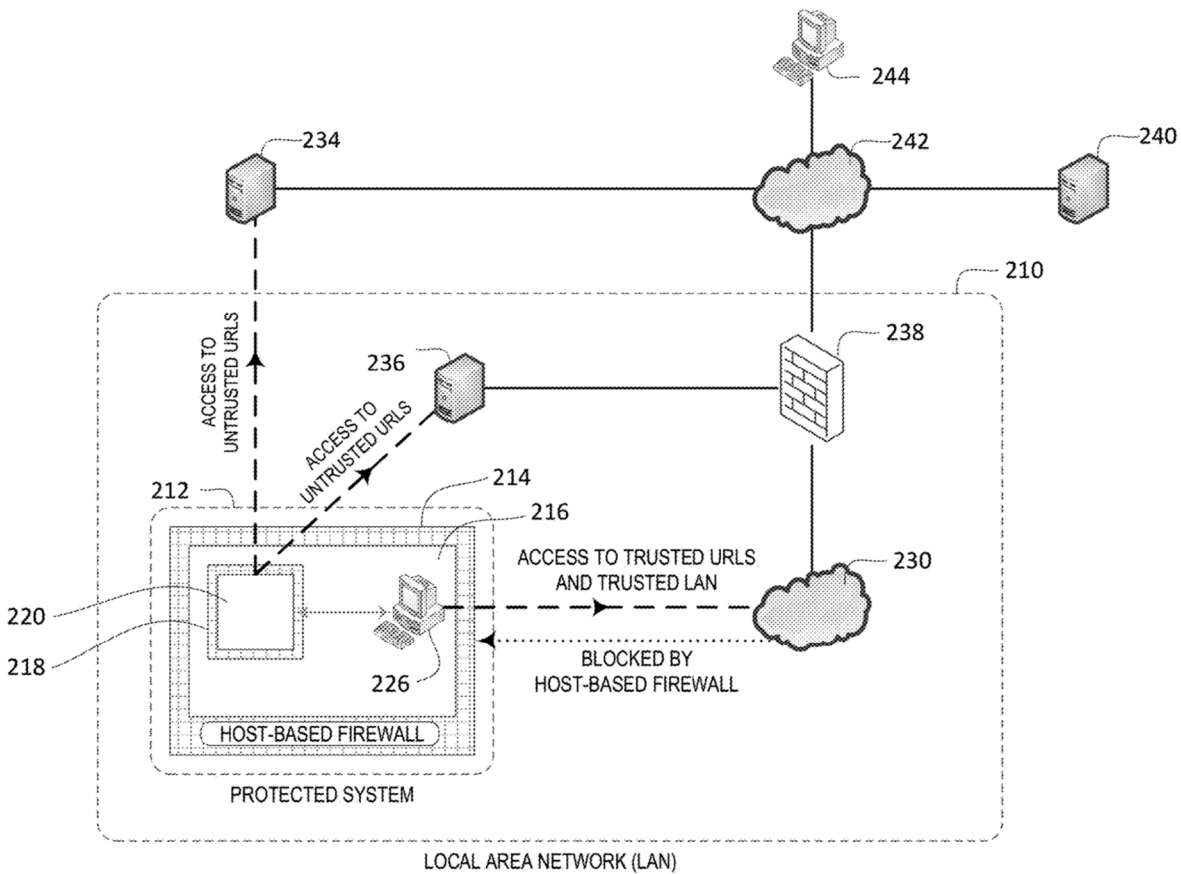


FIG. 2

29. As discussed in the Detailed Description section of the '104 patent, Figure 2 depicts a host computer system 212 connected to a local area network ("LAN") 210. Ex-1001, 10:65-11:1. The LAN 210 may include a router, a border firewall 238, and a LAN-based web proxy 236. Ex-1001, 11:7-9. The host computer system 212 may send traffic to a trusted network 230 within the LAN 210 via an operating system 226. Ex-1001, 11:9-11. The host computer system 212 may also send traffic to an untrusted network destination on the Internet 242 using one or more of the border firewall 238, the LAN-based web proxy 236, a

first Internet-based web proxy 234, and a second Internet-based web proxy 240.

Ex-1001, 11:11-21.

30. According to the '104 patent, the host computer system 212 may include a host-based firewall 214, the operating system 226, a sandboxed computing environment 220, an internal isolation firewall 218, and a workspace 216. Ex-1001, 11:22-26. The '104 patent admits that the host computer system 212 itself is not new; it “may be any host computer system, such as a cell phone, a pager, a personal computer (PC),” etc. Ex-1001, 11:26-32.

31. The '104 patent states that the host computer system 212 may isolate itself from communications with untrusted network destinations. Ex-1001, 11:49-50. The host computer system 212 may do this using one or more of the host-based firewall 214, the border firewall 238, the LAN-based web proxy 236, the first Internet-based web proxy 234, the second internet-based web proxy 240, the internal isolation firewall 218, or a segregation of untrusted memory space. Ex-1001, 11:51-56.

32. The '104 patent admits, either explicitly or implicitly, that none of these components for isolating communications was new. The patent states that “the host-based firewall 214 may be implemented using software that is commercially available, for example, such as Symantec Endpoint Protection, MacAfee [sic] Host Based Security Systems, etc.” Ex-1001, 12:7-11, 16:12-16.

The patent incorporates the other firewalls and various web proxies as conventional components well-known to people of ordinary skill in the art in the network security field. And they were indeed well-known. Network firewalls were already well-known back in the 1980s. And by 1995, companies like Network Translation Inc. were selling commercially successful firewall products like the PIX (Private Internet eXchange) Firewall. Web proxies also were well-known back in the early 1990s. One such web proxy was the CERN Proxy, which was launched by the European Center for High Energy Physics (CERN) in 1994. While network firewalls and web proxies have advanced since then, nothing in the '104 patent indicates that the firewalls and various web proxies disclosed in the patent are anything other than conventional firewalls and web proxies that existed as of the stated priority date of the patent (September 26, 2017).

33. According to the '104 patent, the sandboxed computing environment 220 within the host computer system 212 may be enforced via a sandbox container process. Ex-1001, 12:29-31. The sandbox container process may be a security mechanism used to separate resources associated with the sandboxed computing environment 220 from other applications and processes that might be running on the workspace 216 of the host computer system 212. Ex-1001, 12:31-35. For example, an application or process operating within the sandboxed computing environment 220 may be allowed to access memory associated with the sandboxed

computing environment 220. Ex-1001, 12: 40-44. The memory associated with the sandboxed computing environment 220 may be separate from memory that enables storage and operation of the workspace 216, which runs processes that are not restricted by the sandbox container process. Ex-1001, 12:44-50. As described in the '104 patent, the operating system 226 of the host computer system 212 may include a set of resources to enable operation of the workspace 216 and the sandbox container process. Ex-1001, 12:50-53.

34. Again, the '104 patent admits, either explicitly or implicitly, that none of these components was new. The patent admits that “[t]he operating system 226 may be Windows-based, Linux-Based, Macintosh-based, and/or the like.” Ex-1001, 11:40-42. Likewise, the patent admits that “the sandboxed computing environment 220 and/or the sandbox container process may be implemented using software that is commercially available, for example, such as Invincea Endpoint Protection, Tanium Endpoint Platform, Bromium Advanced Endpoint Security, etc.” Ex-1001, 16:49-54.

35. According to the '104 patent, the LAN-based web proxy 236, as well as the Internet-based web proxies 234 and 240, may enable isolation of the host computer system 212 from untrusted network destinations. Ex-1001, 12:54-56, 13:8-10; *see also id.*, 14:5-7 (same for remote host computer system 244). At the same time, the proxies may allow processes and applications, such as a browser

process or an email process, operating within the sandboxed computing environment 220 of the host computer system 212 to access an untrusted network destination. Ex-1001, 12:56-61, 13:10-15; *see also id.*, 14:15-20. The sandbox container process may authenticate with one or more of the proxies on behalf of the processes and applications operating within the sandboxed computing environment. Ex-1001, 12:62-66, 13:28-32; *see also id.*, 15:11-16. When the sandbox container process has successfully authenticated with one or more of the proxies, the processes and applications operating within the sandboxed computing environment 220 may be allowed to access the untrusted network destination. Ex-1001, 13:2-7, 13:36-42; *see also id.*, 15:19-15:24.

36. The '104 patent provides little description of the authentication process for the sandbox container. The LAN-based web proxy server and the Internet-based web proxy servers are the only structures described in the specification of the '104 patent that correspond to the function of authenticating a computing environment. The patent states only that the sandbox container process may authenticate with the LAN-based web proxy 236 and/or the Internet-based web proxies 234 and 240 “using preconfigured credentials stored, for example, within an encrypted configuration file.” Ex-1001, 12:66-13:2, 13:32-36; *see also id.*, 15:16-19. Performing authentication using preconfigured credentials stored in an encrypted configuration file was highly conventional and well-known to people

of ordinary skill in the art in the network security field before the stated priority date of the '104 patent (September 26, 2017).

37. Figure 3 of the '104 patent depicts a similar host computer system that is connected to a wide area network (WAN) 300 in addition to the Internet.

Ex-1001, 18:60-64. Figure 4 of the '104 patent depicts an enterprise network configuration with multiple trusted networks 410A and 410B that include a similar host computer system. Ex-1001, 22:35-39, 22:48-51. The sandbox-based network isolation systems in these examples are similar to the system of Figure 2.

B. Prosecution History

38. The '104 patent was filed with twenty claims in the U.S. Patent and Trademark Office ("USPTO") as U.S. Patent Application No. 16/141,378 ("the '378 application") on September 25, 2018, in the names of Glenn Coleman, Peter Martz, and Kenneth Moritz as inventors and L3 Technologies, Inc. as the applicant ("Applicant"). Ex-1002 (Filing Receipt at 1). The '378 application asserted the benefit of U.S. Provisional Patent Application No. 62/563,237 ("the '237 provisional application"), filed on September 26, 2017. *Id.*

39. As filed, claim 1 recited as follows:

1. A host computer system comprising a processor and memory, the processor configured to:

implement a workspace, wherein the workspace is configured to enable operation of a first set of one or more applications or processes via a first memory space;

implement an isolated computing environment, the isolated computing environment being configured to enable operation of a second set of one or more applications or processes via a second memory space, wherein the isolated computing environment is configured to access an Internet-based cloud service via at least one application of the second set of applications;

isolate the isolated computing environment from the workspace using an internal isolation firewall; and

communicate with at least one proxy server in order to access the Internet-based cloud service, wherein the isolated computing environment is configured to authenticate with an authentication device in order to allow communication between the at least one application and the Internet-based cloud service.

Ex-1002 (Application at 31).

1. Non-Final Office Action Dated May 21, 2020

40. On May 21, 2020, the USPTO rejected all twenty claims in the '378 application in view of the prior art. Ex-1002 (May 21, 2020 Office Action at 3-12). Specifically:

- Claims 1, 2, 4-9 and 19 were rejected under 35 U.S.C. § 102(a)(1) as unpatentable over U.S. Patent Publication No. 2013/0318594 to Hoy et al. ("Hoy");

- Claims 10, 11 and 14-18 were rejected under 35 U.S.C. § 102(a)(1) as unpatentable over U.S. Patent Publication No. 2015/0156203 to Giura et al. (“Giura”);
- Claims 3 and 20 were rejected under 35 U.S.C. § 103 as unpatentable over Hoy in view of Giura.
- Claim 12 was rejected under 35 U.S.C. § 103 as unpatentable over Hoy in view of U.S. Patent Publication No. 2017/0034174 to Jagers; and
- Claim 13 was rejected under 35 U.S.C. § 103 as unpatentable over Hoy in view of Jagers and U.S. Patent Publication No. 2012/0054744 to Singh et al.

41. The USPTO also rejected dependent claims 5-7 and 12 in the '378 application under 35 U.S.C. § 112(b) as indefinite for failing to particularly point out and distinctly claim the subject matter of the alleged invention. Ex-1002 (May 21, 2020 Office Action at 2-3). According to the patent examiner, dependent claims 5-7 were indefinite because the limitation “to determine whether to send web traffic to the local network-based proxy server or the Internet-based proxy server” required at least two proxy servers, and the parent claim (dependent claim 4) recited only one. *Id.* Similarly, dependent claim 12 was indefinite because for essentially reciting that “1” equals “2.” *Id.*

42. Additionally, the USPTO rejected all twenty claims in the '378 application on the ground of non-statutory double patenting as unpatentable over claims 1-20 of U.S. Patent No. 10,554,475 and over claims 1-20 of U.S. Patent No. 10,558,798. Ex-1002 (May 21, 2020 Office Action at 13-14). The USPTO further provisionally rejected all twenty claims on the ground of non-statutory double patenting as unpatentable over the claims of co-pending U.S. Patent Application Nos. 16/059,381, 16/109,275, 16/141,535, 16/138,696, 16/144,609, 16/144,761, 16/146,149, 16/142,810, 16/146,490, 16/163,796, 16/163,884 and 16/164,342. Ex-1002 (May 21, 2020 Office Action at 14).

2. Office Action Response Dated August 14, 2020

43. On August 14, 2020, Applicant filed a *Reply Pursuant to 37 CFR § 1.111* that canceled claim 18 and amended claims 1-17 and 19. Ex-1002 (Aug. 14, 2020 Reply at 2-6). Among other things, Applicant amended independent claims 1 and 19 to state that the recited “isolated computing environment” uses “an operating system associated with the host computing system and compris[es] a sandboxed computing environment.” Ex-1002 (Aug. 14, 2020 Reply at 2, 6). Applicant also amended independent claim 10 to state, among other things, that the server is configured to “implement an isolated computing environment comprising a sandboxed computing environment.” Ex-1002 (Aug. 14, 2020 Reply at 4). Additionally, Applicant amended dependent claims 5-7 to depend directly from

independent claim 1 instead of dependent claim 4, and amended dependent claim 12 to state that “the proxy is a first proxy.” Ex-1002 (Aug. 14, 2020 Reply at 3, 5).

44. Applicant’s remarks indicated that the amendments to claims 5-7 and 12 addressed the rejections for indefiniteness under 35 U.S.C. § 112(b). Ex-1002 (Aug. 14, 2020 Reply at 7). Applicant also requested that the patent examiner hold the rejections for double patenting in abeyance until the claims are allowed. *Id.*

45. Regarding the rejections in view of Hoy, Applicant argued that Hoy does not disclose “a sandbox computing environment,” noting that Hoy does not mention the term “sandbox.” Ex-1002 (Aug. 14, 2020 Reply at 8). Applicant also argued that Hoy does not teach or suggest an “isolated computing environment using an operating system associated with the hosting computing system,” asserting that the virtual machine environment in Hoy uses a virtual guest operating system that is allegedly not associated with the host operating system architecture. *Id.* According to Applicant, these arguments showed the patentability of independent claims 1 and 19. Ex-1002 (Aug. 14, 2020 Reply at 9).

46. Regarding the rejections in view of Giura, Applicant argued that Giura does not teach or suggest implementing “an isolated computing environment comprising a sandboxed computing environment,” noting that Giura is silent respecting the term “sandboxed.” Ex-1002 (Aug. 14, 2020 Reply at 9). Applicant

also argued that Giura does not teach or suggest isolating “the isolated computing environment from the workspace using an internal isolation firewall,” noting that the firewall disclosed in Giura separates virtual browser servers from an enterprise network instead of isolating a computing environment from a workspace. Ex-1002 (Aug. 14, 2020 Reply at 9-10). According to Applicant, the firewall disclosed in Giura handles only enterprise network traffic, such as incoming and outgoing enterprise network traffic related to the virtual browser servers, and not traffic between a computing environment and a workspace. Ex-1002 (Aug. 14, 2020 Reply at 10). For Applicant, these arguments showed the patentability of independent claim 10. *Id.*

47. Applicant’s remarks did not provide separate arguments for the dependent claims, asserting that the dependent claims were patentable because the independent claims were patentable. Ex-1002 (Aug. 14, 2020 Reply at 9-10).

3. Final Office Action Dated September 9, 2020

48. On September 9, 2020, the USPTO rejected all claims then pending in the ’378 application in view of the prior art. Ex-1002 (Sept. 9, 2020 Office Action at 3-9). Specifically, claims 1-17, 19 and 20 were rejected under 35 U.S.C. § 103 as unpatentable over Hoy in view of the Comodo Internet Security Software Version 7.0 User Guide (Version 7.0.111114, 2014) (“Comodo”). Ex-1002 (Sept. 9, 2020 Office Action at 3). Although the patent examiner based the rejection on

35 U.S.C. § 103 (obviousness), the examiner found that Hoy taught all of the limitations of claims 1, 10 and 19. Ex-1002 (Sept. 9, 2020 Office Action at 3-5, 8). Significantly, although Hoy does not use the term “sandbox,” the examiner found that Hoy teaches a sandboxed environment due through least the following disclosure in Hoy:

Because it is exposed to potentially malicious software as it browses the Internet, the virtual guest system browser and its virtual operating system may be a hardened version that is exceptionally resistant to corruption, or it may have enhanced anomaly detection to aid in noting when dangerous data has been encountered by the browser. The virtual guest system also has no connection whatsoever to the LAN, and cannot access any address on the LAN.

Ex-1002 (Sept. 9, 2020 Office Action at 5); Ex-1004 (Hoy) ¶ 0027. The examiner stated that this disclosure “is essentially a definition of sandboxed environment as known in the art.” Ex-1002 (Sept. 9, 2020 Office Action at 5).

49. As a backup position “[t]o exclude any doubt,” the patent examiner cited Comodo, which the examiner noted “explicitly teaches a sandboxed computing environment (An innovative sandboxed environment to run programs and browse the Internet isolated from your real computer. Comodo, page 10).” Ex-1002 (Sept. 9, 2020 Office Action at 5).

50. Responding to Applicant’s assertion that the virtual machine environment in Hoy uses a virtual guest operating system that is not associated

with the host operating system architecture, the Examiner cited the following disclosure in paragraph 26 of Hoy: “The browser is activated by a desk-top shortcut or application on the host system that allows the user to link to the isolated Internet access of the virtual guest system by launching the virtual guest web browser, which operates transparently to the user in the virtual guest system.” Ex-1002 (Sept. 9, 2020 Office Action at 3); Ex-1004 (Hoy) ¶ 0026. The examiner stated that this disclosure “explicitly and clearly disclosed” the limitation of an “isolated computing environment using an operating system associated with the hosting computing system.” Ex-1002 (Sept. 9, 2020 Office Action at 3).

51. Additionally, the USPTO repeated its prior rejections on the ground of non-statutory double patenting. Ex-1002 (Sept. 9, 2020 Office Action at 9-12).

4. Office Action Response and RCE Dated March 8, 2021

52. On March 8, 2021, Applicant filed a *Request for Continued Examination* and a *Reply Pursuant to 37 CFR § 1.114* that amended independent claims 1, 10 and 19. Ex-1002 (Mar. 8, 2021 Reply at 2, 4, 6). Among other things, Applicant simplified the limitation “the isolated computing environment using an operating system associated with the host computing system” in independent claims 1 and 19 as “the isolated computing environment using a host operating system.” Ex-1002 (Mar. 8, 2021 Reply at 2, 6). Applicant also added the “using a host operating system” language to independent claim 10.

Additionally, Applicant amended independent claims 1, 10 and 19 to recite the limitation of “authentica[ing] the isolated computing environment with an authentication device” and to clarify that communication with the Internet cloud service or resource is allowed “when the isolated computing environment has been authenticated.” Ex-1002 (Mar. 8, 2021 Reply at 2, 4, 6). This limitation had been in the prior versions of independent claims 1 and 19 but had been phrased differently.

53. In its remarks, Applicant requested that the patent examiner hold the rejections for double patenting in abeyance until the claims are allowed. Ex-1002 (Mar. 8, 2021 Reply at 7).

54. Regarding the rejections in view of Hoy and Comodo, the only feature argued by Applicant was the limitation of “authentica[ing] “the isolated computing environment with an authentication device.” Ex-1002 (Mar. 8, 2021 Reply at 7-8). Notably, Applicant did not dispute the examiner’s conclusion that Hoy teaches a sandboxed environment. Nor did Applicant dispute the examiner’s conclusion that Hoy discloses an isolated computing environment using an operating system associated with the hosting computing system.

55. On the “authenticate” limitation, Applicant’s analysis of Hoy was brief, noting only that Hoy is silent with respect to authenticating the isolated computing environment and does not contain the term “authentication device.”

Ex-1002 (Mar. 8, 2021 Reply at 7-8). Applicant did not address the teaching in Hoy that “[t]he VPN transmits data packets that are encrypted according to an IPSEC or SSL VPN tunnel construct protocol.” Ex-1004 (Hoy) ¶ 0054. As discussed later in this declaration, both the Internet Protocol Security (IPsec) standard and the Secure Sockets Layer (SSL) standard contemplate the authentication of computing environments that are sending web traffic.

56. Applicant’s analysis of Comodo included asserting that the disclosure in Comodo of comparing a program against a list of known or recognized applications is not equivalent to authenticating an isolated computing environment with an authentication device. Ex-1002 (Mar. 8, 2021 Reply at 8). As discussed elsewhere in this declaration, however, the process of authenticating an isolated computing environment with an authentication device was highly conventional and well known to people of ordinary skill in the art in the network security field before the stated priority date of the ’104 patent (September 26, 2017).

5. Non-Final Office Action Dated March 23, 2021

57. On March 23, 2021, the USPTO rejected all claims then pending in the ’378 application on the ground of non-statutory double patenting but stated that the pending claims were allowable over the prior art. Ex-1002 (Mar. 23, 2021 Office Action at 3-5). The patent examiner did not explain his reasoning for finding the pending claims to be allowable, other than a short statement that

“[n]one of the cited references teaches claims limitations as currently amended.”

Ex-1002 (Mar. 23, 2021 Office Action at 3-5).

6. Office Action Response Dated September 14, 2021

58. On September 14, 2021, Applicant filed a *Terminal Disclaimer To Obviate a Double Patenting Rejection over a “Prior” Patent and a Reply Pursuant to 37 CFR § 1.111*. Ex-1002 (Sept. 14, 2021 Terminal Disclaimer and Reply). In its Reply, Applicant noted its submission of an electronic terminal disclaimer to disclaim claims 1-20 of U.S. Patent No. 10,554,475 and claims 1-20 of U.S. Patent No. 10,558,798. Ex-1002 (Sept. 14, 2021 Reply at 7). Applicant also requested withdrawal of the double patenting rejections based on co-pending applications because (1) certain of the co-pending applications had a later U.S. filing date than the '378 application; and (2) the patent examiner had not demonstrated how each claim element may be found in the remaining co-pending applications. Ex-1002 (Sept. 14, 2021 Reply at 8).

7. Notice of Allowance Dated October 5, 2021

59. After ministerial prosecution activities, the USPTO issued a *Notice of Allowance and Fee(s) Due* on October 5, 2021, noting the filing of the terminal disclaimer on September 20, 2021 and withdrawing all previous rejections. Ex-1002 (Oct. 5, 2021 Notice of Allowance at 2). The '378 application thereafter issued as the '104 patent.

C. Priority Date

60. As noted above, the '378 application asserted the benefit of the '237 provisional application, filed on September 26, 2017. I have been asked to treat September 26, 2017, as the effective filing or priority date of the '104 patent. I have not reviewed the '237 provisional application to determine that the '104 patent is in fact entitled to that priority date.

D. Challenged Claims

61. I understand that Petitioner International Business Machines Corporation ("Petitioner" or "IBM") is challenging the validity of claims 1-3, 8, 10, 11, and 17-19 of the '104 patent in the *Petition for Inter Partes Review* to which this Declaration will be attached. Those claims are reproduced in Appendix 3 to this declaration. While the Petition and this Declaration are directed to the challenged claims, I have considered all claims 1-19 of the '104 patent, as well as the specification of the '104 patent and portions of the '104 patent prosecution history, in forming my opinions.

III. LEVEL OF ORDINARY SKILL IN THE ART

62. I have analyzed the '104 patent and determined that the field of the patent is network security. As described in the Background of the '104 patent, an infected host computer system may result in security losses, reduction of efficiency, and malfunctions. Ex-1001, 1:22-34. The Background of the '104 patent further describes that malware may compromise the security and privacy of

the user of the infected host computer system. Ex-1001, 1:35-45. To address these security issues, the '104 patent describes using a sandbox container process as a security mechanism to separate a sandboxed computing environment from other applications and processes that may be running on a workspace of a host computer system. Ex-1001, 6:25-29, 12:31-35. According to the '104 patent, a sandbox-based network isolation system may prevent malware from establishing command and control on a computer system, may prevent malware from moving from a first computer system to a second computer system, and may increase the security posture of an enterprise network. Ex-1001, 23:35-55. Indeed, sandbox-based network isolation systems and their benefits were well known to people of ordinary skill in the art in the network security field before the stated priority date of the patent (September 26, 2017).

63. I have been advised that the claims of a patent are reviewed from the point of view of a hypothetical person of ordinary skill in the art ("POSITA"). In determining the characteristics of a POSITA in the field of the '104 patent at the time of the claimed invention, I considered several things, including various prior art techniques relating to network security, the types of problems that such techniques gave rise to, prior-art solutions to those problems, and the rapidity with which innovations were made. I also considered the sophistication of the technologies involved, as well as the educational background and experience of

those actively working in the field at the time. I also considered the level of education that would be necessary to understand the '104 patent. Finally, I placed myself back at the time of the alleged invention and considered the technology available at that time, the students whom I have trained, and the engineers, programmers, and other professionals that I have worked with and managed in the field of network security.

64. Based on these considerations, I came to the conclusion that a POSITA in the field of the '104 patent at the time of the alleged invention would have been a person with a bachelor's degree in computer science, computer engineering, electrical engineering, or a similar field of study, plus at least two years of professional experience working in the field of network security, or a person with a master's degree in computer science, computer engineering, electrical engineering, or a similar field of study plus a specialization in network security.¹⁸ A POSITA would have had experience or the ability to work with computer network architectures and services, computer virtualization, virtual computer systems, computer and network security, and virtualization and security standards. A person with less education but more relevant practical experience or

¹⁸ Any reference herein to what a person of ordinary skill in the art or POSITA would have known, understood, or been motivated to do, and the like, refers to the time of the alleged invention.

technical training may also meet this standard, while advanced education might substitute for experience.

65. In view of my education and experience, as summarized in my curriculum vitae (Appendix 1 to this Declaration), I meet and exceed this definition of a POSITA.

66. In arriving at my opinions and conclusions in this Declaration, I have considered the issues from the perspective of a POSITA at the time of the alleged invention.

67. Before the stated priority date of the '104 patent (September 26, 2017), a POSITA would have known and had the skills necessary to, among other things, implement a sandbox-based network isolation solution that results in sandboxed computing environment, as well as to implement a separate memory space for such a sandboxed computing environment, since such solutions were well known and conventional to people of ordinary skill in the art in the network security field at the time. A POSITA would also have known and had the skills necessary to implement other conventional components of network security solutions, such as host-based firewalls, internal firewalls, network firewalls, and web proxy servers, since such components were well known and conventional to people of ordinary skill in the art in the network security field at the time. Likewise, a POSITA would have known and had the skills necessary to perform

common computer-security tasks, like isolating a computing environment using an internal firewall, as well as performing authentication of a computing environment using preconfigured credentials stored in an encrypted configuration file, since such tasks were well known and conventional to people of ordinary skill in the art in the network security field at the time. These are just some of the computer security components and tasks that a POSITA would have known and found conventional before the stated priority date of the '104 patent.

IV. CLAIM CONSTRUCTION

68. For purposes of this Declaration, I have considered the claim language, specification, and portions of the prosecution history to determine the meaning of the claim language as it would have been understood by a POSITA at the time of the invention. I am informed that the “plain and ordinary meaning” or *Phillips* standard has traditionally been applied in district court litigation, where a claim term is given its plain and ordinary meaning in light of the specification and the prosecution history, from the viewpoint of a POSITA.

69. I have applied the *Phillips* standard to my analysis in this *inter partes* review proceeding. Unless otherwise stated, I have applied the plain and ordinary meaning to claim terms.¹⁹

¹⁹ In this Declaration, I do not offer an opinion on whether any claim terms meet the statutory requirements of 35 U.S.C. § 112.

70. In this section of my Declaration, I have been asked to offer my opinions about the understanding of a POSITA regarding particular claim terms in the '104 Patent. Those opinions are below.

A. “sandboxed computing environment” (all challenged claims)

1. Proper Construction Based on a POSITA’s Understanding

71. Independent claims 1, 10, and 18 all recite a “sandboxed computing environment.” Based on my review of the claims, specification, and prosecution history of the '104 patent, it is my opinion that a POSITA would have understood the claim term “sandboxed computing environment” to mean “computing environment that does not require its own separate operating system.” The specification of the '104 patent states that “the sandbox container process may permit isolation of the sandboxed computing environment without requiring the sandboxed computing environment to boot a separate operating system, load separate libraries, etc.” Ex-1001, 7:10-15. In my opinion, a POSITA, having reviewed the claims, specification, and prosecution history of the '104 patent, would have construed a “sandboxed computing environment” as I have done above.

2. Alternative Construction Based on Patent Owner’s Position

72. In the alternative, I am informed that a petitioner in an *inter partes* review proceeding may in some circumstances rely on positions taken by a patent owner regarding claim scope in related infringement litigation. Here, I am

informed that the '104 patent is presently the subject of a patent infringement lawsuit brought by the assignee, Croga Innovations Ltd. (“Patent Owner” or “Croga”), against Petitioner IBM and captioned *Croga Innovations Ltd. v. International Business Machines Corporation*, No. 2:23-cv-00634-JRG (U.S. District Court, Eastern District of Texas (the “Related Litigation”).

73. I am further informed that Croga served its preliminary proposed constructions of claim terms in the '104 patent in the Related Litigation on November 26, 2024. I have reviewed Exhibit A to Croga’s preliminary proposed constructions, which provides both a proposed construction and an identification of extrinsic evidence for the phrase “sandboxed computing environment.” Ex-1010. In Exhibit A, Croga found “[n]o construction necessary” for the phrase “sandboxed computing environment,” but also identified the definition of “sandbox” in the Microsoft Computer Dictionary, Fifth Edition (Microsoft Press, 2002). *Id.* That dictionary defines “sandbox” as follows:

Java Virtual Machine security area for downloaded (remote or untrusted) applets, an area in which such applets are confined and prevented from accessing system resources. Confinement to the sandbox prevents downloaded applets from carrying out potentially dangerous operations, maliciously or otherwise. They have to “play” inside the sandbox, and any attempt to “escape” is thwarted by the Java Security Manager.

Ex-1009 (Microsoft Computer Dictionary at 463, which I am informed was cited and produced by Croga as part of claim construction proceedings in the Related Litigation). As noted above, the fact that the word “sandbox” received its own entry in computer dictionaries more than a decade before the stated priority date of the ’104 patent indicates how well-known sandboxed computing environments were to people of ordinary skill in the art.

74. Based on Exhibit A to Croga’s preliminary proposed constructions in the Related Litigation, it appears that Croga’s position in the Related Litigation is that “sandboxed computing environment” broadly means an area in which software is confined and prevented from accessing system resources. While I do not believe that a POSITA, having reviewed the claims, specification, and prosecution history of the ’104 patent, would have construed a “sandboxed computing environment” in this way, I have been asked to apply this alternative claim construction in addition to the proper construction to show that the prior art renders the challenged claims of the ’104 patent anticipated or obvious under either construction.

B. “authentication device” (all challenged claims)

75. Independent claims 1, 10 and 18 all recite an “authentication device.” Based on my review of the claims, specification, and prosecution history of the ’104 patent, it is unclear to me what the outer boundary of the term “authentication device” in the ’104 patent would have been to a POSITA. As discussed above, the

'104 patent provides little description of the authentication process for the sandbox container.

76. The specification of the '104 patent does make clear that, at a minimum, an authentication device can be a LAN-based web proxy server or an Internet-based web proxy server. These are the only structures described in the specification of the '104 patent that correspond to the function of authenticating a computing environment. Ex-1001, 2:16-24 (“The isolated applications may communicate with the Internet destinations via an authentication device (e.g., a proxy server). ... There may be more than one (e.g., two) authentication devices, which may include, for example, an Internet-based proxy server and a local network-based proxy server.”).

77. Based on this disclosure in the specification, it is my opinion that a POSITA would have understood that the scope of the term “authentication device” includes, at a minimum, a proxy server that performs the function of authenticating a computing environment. Because my opinions in this Declaration show that the challenged claims of the '104 patent would have been anticipated or obvious under any claim construction for “authentication device” that includes a proxy server within its scope, I do not believe that it is necessary here to determine the outer boundary of the term “authentication device.”

V. INVALIDITY GROUNDS

78. Based on my review and analysis of the materials cited herein, my opinions regarding the understanding of a POSITA at the time of the alleged invention of the '104 patent, and my training and experience, it is my opinion that claims 1-3, 8, 10, 11 and 17-19 of the '104 patent are anticipated or obvious on the following grounds:

Ground	Claims	Basis	Prior Art
1	1-3, 8, 10, 11, 17-19	§ 103 (obvious)	Hoy and Pratt
2	1-3, 8, 10, 11, 17-19	§ 103 (obvious)	Hoy, Krasin, and Pratt
3	1-3, 8, 10, 11, 17-19	§ 102 (anticipated)	Benoit
4	1-3, 8, 10, 11, 17-19	§ 103 (obvious)	Benoit and Hunt
5	1-3, 8, 10, 11, 17-19	§ 103 (obvious)	Benoit, Hunt, and Pratt

A. Ground 1: Claims 1-3, 8, 10, 11 and 17-19 Are Obvious in View of Hoy and Pratt

79. It is my opinion that a POSITA would have found claims 1-3, 8, 10, 11 and 17-19 of the '104 patent to be obvious based on the teachings of Hoy in view of Pratt.

1. Combination of Hoy and Pratt

a. Summary of Hoy

80. Hoy was filed on January 27, 2012, and was published on November 28, 2013, almost four years before the stated priority date of the '104 patent

(September 26, 2017). Ex-1004 (Hoy). I am informed that Hoy therefore qualifies as prior art to the '104 patent.

81. Hoy teaches a networked computer system 1 linked to the Internet 5 via a firewall 3 and a router 2 of a conventional configuration that was well known in the art. Ex-1004 (Hoy) ¶ 0038. The networked computer system 1 is shown in Figure 1 of Hoy, reproduced below.

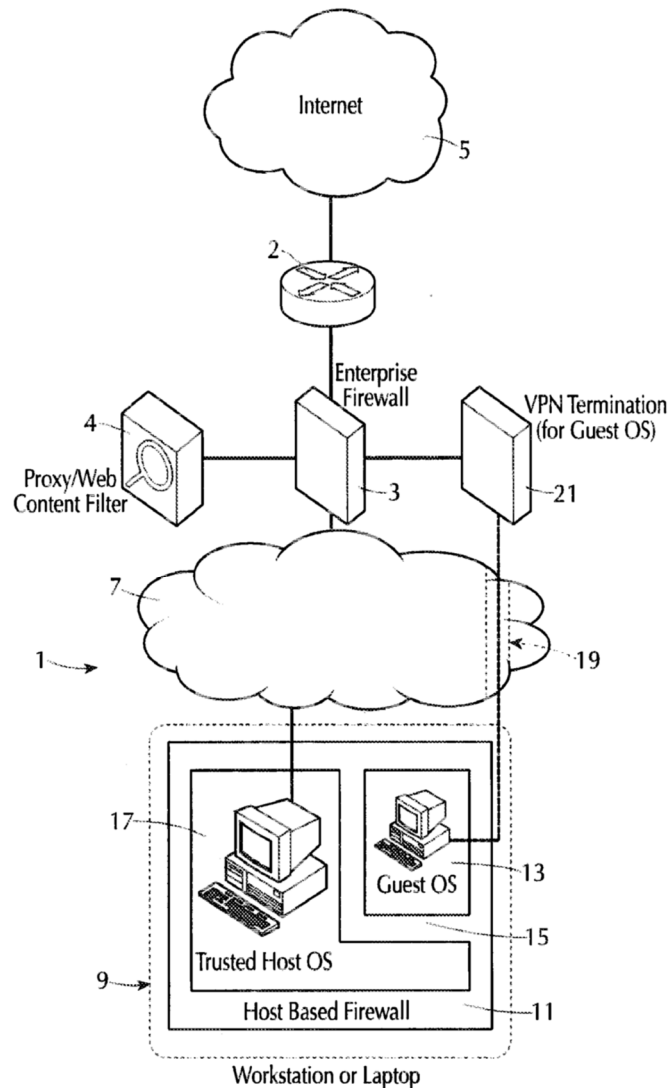


FIG. 1

82. As shown in Figure 1 and as described in the specification of Hoy, a proxy/web-content filter 4 is connected with and cooperates with firewall hardware 3 to control the type of data permitted to pass from the Internet 5 into the local system 1, filtering pornographic material or data not intended for the system 1, and also blocks communications from users on the trusted local area network (LAN) 7 trying to reach disallowed sites. Ex-1004 (Hoy) ¶ 0038. The LAN 7 transmits electrical data signals between a plurality of computer systems each identified by a unique LAN address. *Id.* The computers include at least one workstation 9. *Id.*

83. Hoy teaches that the workstation 9 can be virtually any host computer system, including a PC or a laptop. Ex-1004 (Hoy) ¶ 0039. The host system 9 has an operating system that allows the host system 9 to function, e.g., a Windows or Linux operating system, as was well known in the art. *Id.* The host system 9 also has a host-based firewall 11 that restricts egress from the computer 9. Ex-1004 (Hoy) ¶ 0040. Like the host-based firewall 214 in the '104 patent, the host-based firewall 11 in Hoy may be implemented using Symantec Endpoint Protection or McAfee Host-Based Security Systems. *Id.* The host-based firewall 11 blocks all communications from the computer 9 except for communications to the IP addresses of other computers on the trusted network 7. Ex-1004 (Hoy) ¶ 0041.

This firewall 11 is therefore a preventive barrier to any access of the Internet from host system 9 through its ports or directly to the firewall 3 via the LAN. *Id.*

84. Host system 9 also has software that creates a hypervisor on the computer system 9. Ex-1004 (Hoy) ¶ 0042. Hoy teaches that the hypervisor may be one of a variety of types of hypervisor, either software that executes directly as a separate operating system on the hardware of the computer 9, or “as a software module that operates based on the underlying operating system, e.g., Windows 7, of the computer 9.” *Id.* In either case, the hypervisor creates a virtual machine operating on the host computer, which is illustrated in Figure 1 of Hoy as the virtual guest system or guest OS 13. *Id.*

85. In addition to forming the virtual guest system 13, the hypervisor also provides an additional internal host-supported firewall 15 that separates and restricts interaction between the virtual guest system 13 and the trusted-host operating system 17. Ex-1004 (Hoy) ¶¶ 0043-0047. The virtual guest system 13 may run a browser and related supportive software, such as Adobe or Windows Media Player, that cooperates with the browser when certain types of files are accessed on the Internet, and potentially an email program such as Microsoft Outlook. Ex-1004 (Hoy) ¶ 0048. File transfer to and from the virtual guest system is restricted. Ex-1004 (Hoy) ¶¶ 0049-0051.

86. In operation, the host OS 17 displays a desktop to the user that includes an icon corresponding to the Internet browser that runs in the virtual guest system 13. Ex-1004 (Hoy) ¶ 0052. When a user clicks this icon, it initiates the browser program running in the virtual guest system 13. *Id.* The virtual guest system 13 permits access to ports 80 and 443, which function to access the Internet. The traffic between the Internet and the browser of the virtual guest system 13 is routed through a sealed “virtual conduit” 19 passing through the LAN. *Id.*

87. Hoy explains that the virtual conduit 19 is a software-generated Virtual Private Network (VPN) tunnel that encrypts data passing to and from the Internet and ensures that the data are delivered to the virtual guest system 13 at one end and the VPN termination point 21 at the other end, without the possibility that the data can reach any other location in the LAN except those two locations. Ex-1004 (Hoy) ¶ 0053. The virtual guest system 13 uses Network Address Translation to support a captive VPN session to the VPN termination point 21. *Id.* Hoy discloses that the VPN 19 is a tamper-proof conduit through the LAN 7. *Id.* The VPN 19 “can be analogized to a pipe full of toxin passing through an otherwise clean water supply.” *Id.*

88. Hoy teaches that the VPN transmits data packets that “are encrypted according to an IPSEC or SSL VPN tunnel construct protocol.” Ex-1004 (Hoy) ¶

0054. The VPN transmits encrypted data over the LAN from the virtual guest system 13 on the host computer 9 to the VPN termination point 21, where the data is decrypted for transmission. *Id.* The VPN termination point is physically connected with the firewall device 3 outside of the LAN and adds data defining a unique IP address for the specific virtual guest system 13, e.g., 10.1.xxx.xxx, or 255.255.2xx.xxx, to the data transmitted through the firewall 3 to the Internet. *Id.*

89. Similarly, data packets received from the Internet directed to the virtual guest system 13 pass through the firewall 3 and other standard prior-art security measures, such as the filter 4. Ex-1004 (Hoy) ¶ 0055. These packets are then encrypted at the VPN termination point and sent encrypted through the LAN to the host computer 9. *Id.* At the host computer 9, the packets are received and passed to the virtual guest system 13, where the data is decrypted and displayed by the browser. *Id.*

90. Hoy provides additional teachings pertaining to Internet access by software installed on the host machine 17, Ex-1004 (Hoy) ¶ 0057, the email program for the workstation 9, *id.* ¶ 0058, the functions of the host-based firewall 11 depending on whether the computer 9 is at home on the LAN or roaming, *id.* ¶¶ 0059-0065, the configuration of the LAN firewall 3, *id.* ¶¶ 0066-0071, an embodiment without a LAN, *id.* ¶¶ 0072-0078, and an embodiment

in which the host and guest systems each have browser programs installed, *id.* ¶¶ 0079-0083.

b. Summary of Pratt

91. Pratt was filed on January 23, 2015, more than two-and-a-half years before the stated priority date of the '104 patent (September 26, 2017). Ex-1006 (Pratt). I am informed that Pratt therefore qualifies as prior art to the '104 patent.

92. Pratt teaches a system 100 including a device 110, a network 140, and a service or resource provider 150, with the device 110 having an operating environment 120 that may be securely authenticated to the service or resource provider 150. Ex-1006 (Pratt), 2:15-22. The device 110 may be any computerized device capable of executing the operating environment 120 and an interceptor module 114. Ex-1006 (Pratt), 2:23-25. The system 100 is shown in Figure 1 of Pratt, reproduced below.

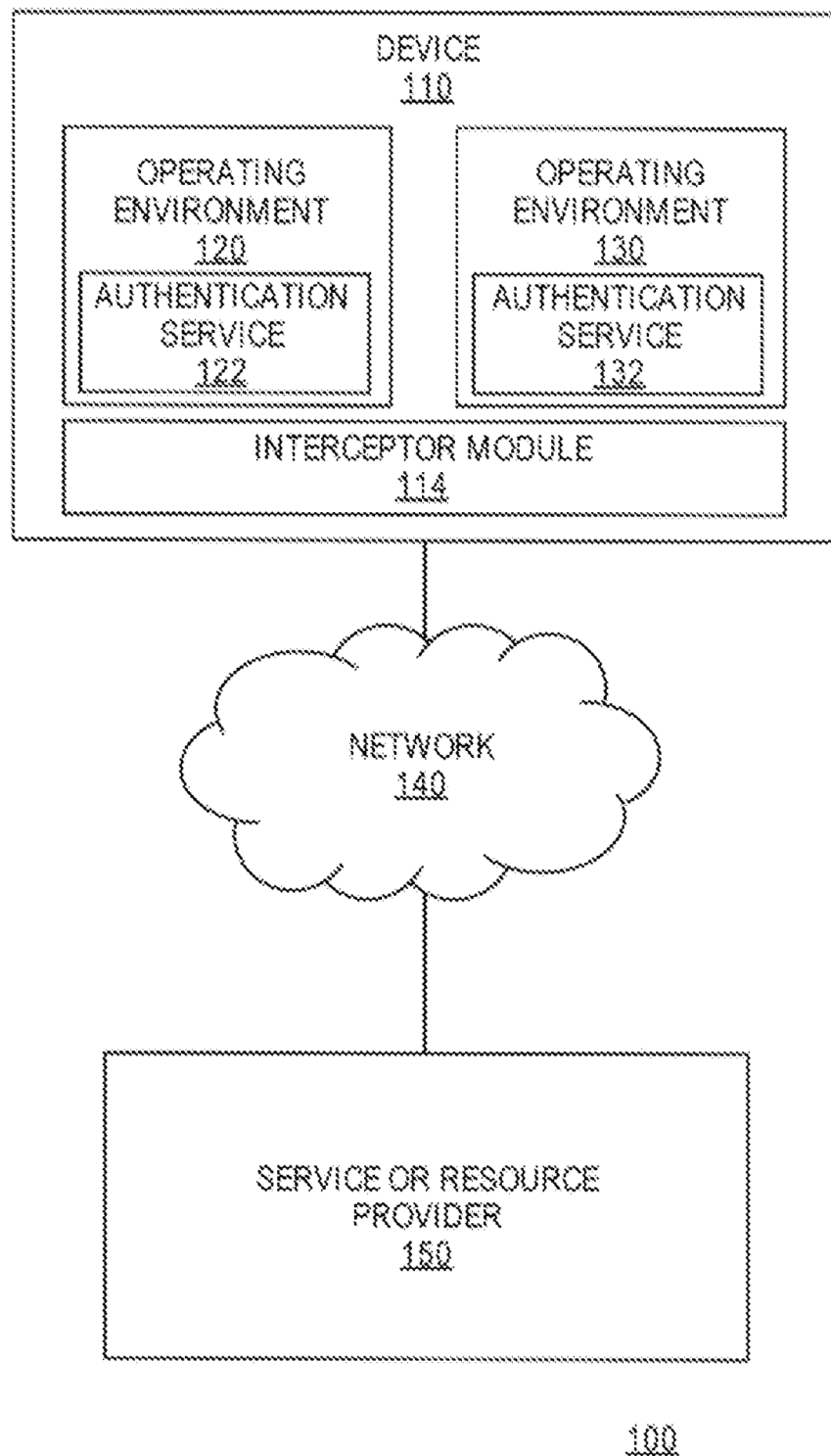


FIG. 1

93. Pratt teaches that the operating environment 120 is an isolated computing environment that can comprise a sandboxed computing environment and use either a host operating system or a guest operating system. Ex-1006 (Pratt), 2:29-36. In particular, after introducing the operating environment 120, Pratt explains as follows:

The term “operating environment” broadly refers to any kind of isolated software computer environment. Non-limiting, illustrative examples of an operation environment include a virtual machine, a container, a **sandbox environment**, a host operating system, and a guest operating system. A host operating system or a guest operating system may execute within each of a virtual machine, a container, and a sandbox environment.

Id. (emphasis added).

94. Figure 1 of Pratt depicts two operating environments—120 and 130. Ex-1006 (Pratt), 2:37-38. Pratt teaches that the operating environment 120 represents an operating environment wherein malicious code may be inadvertently present or introduced. Ex-1006 (Pratt), 2:38-40. In contrast, the operating environment 130 may correspond to an operating environment deemed more secure or trustworthy than the untrusted operating environment 120. Ex-1006 (Pratt), 2:40-43.

95. Pratt teaches that each operating environment may comprise an authentication service, such as the authentication services 122 and 132 shown in

Figure 1. Ex-1006 (Pratt), 2:59-63. These authentication services are components capable of providing authentication data to an operating system. Ex-1006 (Pratt), 2:63-65. Pratt highlights that, as of its filing date (January 23, 2015), there were “many types of schemas and approaches for performing authentication” of an operating environment. Ex-1006 (Pratt), 2:65-3:2.

96. Pratt discloses that authentication of the operating environments is needed for the service or resource provider 150 to provide a service or resource to the operating environments. Ex-1006 (Pratt), 3:49-52. Pratt teaches that the provider 150 may be a proxy server, such that the operating environments are authenticated with a proxy. Ex-1006 (Pratt), 3:52-54. In operation, the authentication data for the provider 150 is provided as appropriate by the trusted operating environment 130 to the untrusted operating environment 120, which then authenticates itself to the provider 150. Ex-1006 (Pratt), 5:43-53. In this way, the untrusted operating environment 120 can authenticate itself to the provider 150, which again may be a proxy server, without exposing any malicious code to the authentication mechanism in the trusted operating environment 130. Ex-1006 (Pratt), 5:54-58.

**c. A POSITA Would Have Been Motivated to
Combine the Authentication Process of Pratt
with the Computer System of Hoy**

97. In my opinion, a POSITA would have been motivated to combine the authentication process of Pratt with the computer system of Hoy. Both Hoy and Pratt are directed to the field of network security. As explained in Pratt, in order to gain access to a computerized resource or service, it may be necessary for a requesting party to present an authentication credential to allow the identity of the requesting party to be authenticated. Ex-1006 (Pratt), 1:11-14. Only a party that presents valid authentication credentials may gain access to the desired resources. Ex-1006 (Pratt), 1:17-19. Such access may be facilitated by a cryptographic ticket provided to a requesting party that submits valid credentials to a network service. Ex-1006 (Pratt), 1:22-29.

98. Pratt recognizes that a requesting party may be an isolated software computer environment akin to the virtual guest system 13 in Hoy. Ex-1006 (Pratt), 2:29-36, 4:4-8. A provider or proxy that restricts access would send a message to the operating environment to authenticate itself. Ex-1006 (Pratt), 2:29-36, 4:8-13. Hoy teaches transmitting data packets using the IPsec or SSL protocols, which include authenticating the computing environments that are sending the data packets, but Hoy does not discuss the details of the specific protocols or authentication more generally, indicating that Hoy's disclosure assumed that this

was within the knowledge of a POSITA. Ex-1004 (Hoy) ¶ 0054. And indeed a POSITA would have agreed that the IPsec and SSL protocols included authentication as the first step in protecting network connections. As a POSITA would have appreciated, Pratt provides an authentication process for an isolated computing environment like the virtual guest system 13 of Hoy so that it can authenticate itself to a proxy or another network device that requires such authentication.

99. In my opinion, the ability for Hoy's isolated guest system to authenticate itself would have motivated a POSITA to combine the authentication process of Pratt with the computer system of Hoy. This combination would simply involve a POSITA combining prior-art elements according to known methods to yield a predictable result—the proxy or other network device determines that Hoy's virtual guest system 13 is authentic and grants access.

100. A POSITA would have been further motivated to combine the authentication process of Pratt with the computer system of Hoy because of the stated advantages of Pratt's process. Pratt teaches that “[a]dvantageously, operating environment 120 is able to authenticate itself to provider 150 without exposing ... any malicious code which may be inadvertently introduced or present within operating environment 120 to the authentication mechanism capable of authenticating to provider 150.” Ex-1006 (Pratt), 5:54-58. In my opinion, a

POSITA would have been motivated by this stated advantage (allowing an untrusted operating environment like Hoy's virtual guest system 13 to authenticate itself without exposing any malicious code within the virtual guest system 13) to combine the authentication process of Pratt with the computer system of Hoy.

101. Additionally, Pratt teaches that “[a]ny malicious code present in operating environment 120 are prevented from abusing access to such an authentication mechanism themselves or communicating the authentication mechanism to other malicious parties.” Ex-1006 (Pratt), 5:59-62. The benefit of preventing such abuse by malicious code present in an untrusted operating environment (like Hoy's virtual guest system 13) would have further motivated a POSITA to combine the authentication process of Pratt with the computer system of Hoy.

102. Furthermore, a POSITA would have been familiar with software-generated Virtual Private Network (VPN) tunnels of the type disclosed in Hoy and would have known that such VPN tunnels require systems to authenticate themselves to the VPN termination point in order to gain access to the termination point. That is how a VPN (virtual private network) stays private—by making sure that a system trying to gain access to a VPN termination point is authorized to do so by presenting the proper authentication credentials, and this was true back at the time that Hoy was filed (January 27, 2012). Pratt makes clear that this commonly-

known authentication process can be extended to untrusted isolated computing environments operating within a computerized device that is also operating a trusted computing environment. The ability to authenticate an isolated guest system in order to establish the VPN tunnel would have motivated a POSITA to combine the authentication process of Pratt with the computer system of Hoy. This combination would simply involve a POSITA using a known technique (the authentication process of Pratt) in the same way to improve a similar device (Hoy) to yield a predictable result—the proxy or other network device determines that Hoy’s virtual guest system 13 is authentic, thus establishing the VPN tunnel so that the virtual guest system can communicate.

d. A POSITA Would Have Had a Reasonable Expectation of Success in Combining the Authentication Process of Pratt with the Computer System of Hoy

103. In my opinion, a POSITA would have had a reasonable expectation of success in combining the authentication process of Pratt with the computer system of Hoy, given the similarities in the teachings and systems of the two references, and given that technologies for authenticating computing environments were well known at the time. A POSITA would have expected to be successful in making the combination because the combination involves using conventional and well-known computer programming methods to yield a predictable result. Implementing an authentication process for an isolated computing environment,

like the virtual guest system 13 of Hoy, would not have required specialized skills beyond what a POSITA would have had.

104. Furthermore, a POSITA would have been motivated and would have had a reasonable expectation of success in combining Hoy and Pratt, as explained above, because the combination would not have changed the principle of operation for any of the teachings of the references. In fact, Hoy implicitly contemplates the authentication of the virtual guest system 13 because Hoy teaches transmitting data packets using the IPsec or SSL protocols. These protocols include the authentication of the computing environments that are sending the data packets. Additionally, Hoy teaches software-generated Virtual Private Network (VPN) tunnels, and a POSITA would have known that such VPN tunnels require systems to authenticate themselves to the VPN termination point. Rather than changing Hoy's principle of operation, the combination with Pratt would actually reinforce Hoy's principle of operation by using a known technique (the authentication process of Pratt) to achieve what is already implicit in Hoy (the authentication of the virtual guest system 13).

2. Independent Claim 1

105. It is my opinion that the combination of Hoy and Pratt renders the limitations of claim 1 obvious.

[1pre] A host computer system:

106. Hoy discloses a **host computer system** in the form of the workstation or laptop 9, and therefore teaches the preamble of claim 1. In particular, Hoy teaches that “[t]he workstation 9 can be virtually any **host computer system**, including a PC or a laptop or any of a variety of other computer systems with electronically accessible data storage, a user visible display, and input devices such as a keyboard and a mouse.” Ex-1004 (Hoy) ¶ 0039; *see*

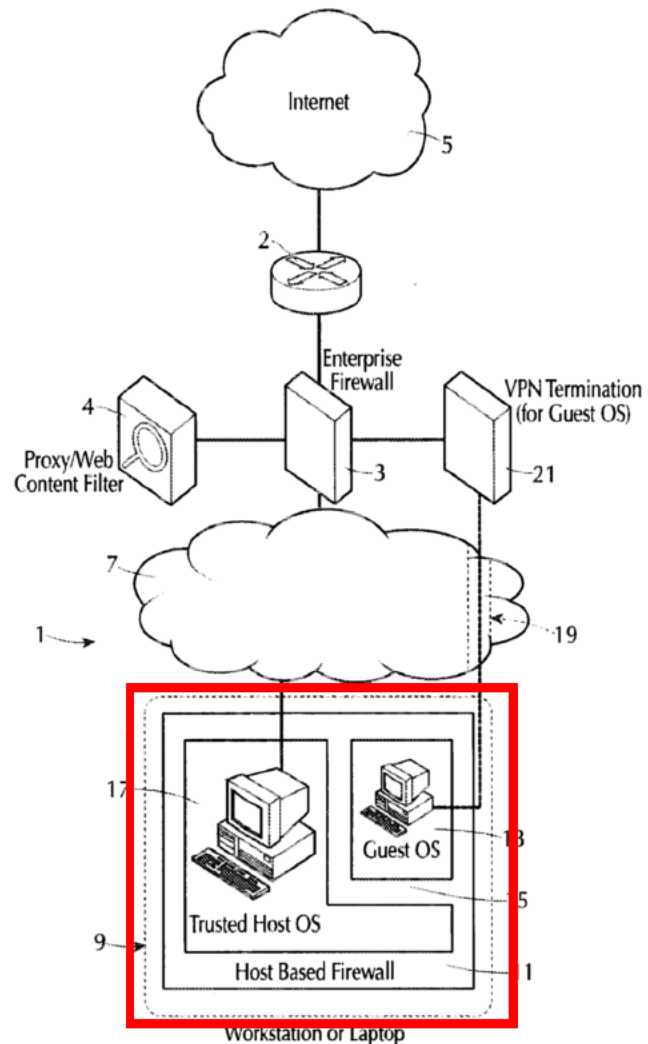


FIG. 1

aspect of the invention, a method of safely browsing the Internet comprises providing a host computer system operating according to a stored operating system and one or more stored program applications, and having operating thereon a firewalled virtual computer system with which the host computer system has

restricted communication of data.”), Fig. 1 (reproduced above with the host computer system highlighted).

107. Accordingly, to the extent that the preamble of claim 1 is limiting, Hoy teaches it.

[1a] a memory; and

108. Hoy discloses a **memory** for the host computer system and therefore teaches element 1a. In particular, Hoy teaches that the host computer system (the workstation 9) “may be a PC sold by Dell, with 4 GB **RAM**” Ex-1004 (Hoy) ¶ 0039. A POSITA at the time that Hoy was filed (January 27, 2012) would have understood the acronym “RAM” to mean random-access memory. Hoy also teaches that the host computer system may comprise “a **computer memory** area (such as a respective desktop) or virtual data storage accessible to the virtual guest OS 13” and “a computer memory area (such as a respective desktop) or data storage accessible to the host OS 17.” Ex-1004 (Hoy) ¶ 0047.

109. Accordingly, Hoy teaches element 1a.

[1b] a processor configured to:
--

110. Hoy discloses a host computer system in the form of the workstation or laptop 9, which necessarily would have included a **processor**, and therefore teaches element 1b. In particular, Hoy teaches that “[t]he workstation 9 can be virtually any host computer system, including a PC or a laptop or any of a variety

of other computer systems with electronically accessible data storage, a user visible display, and input devices such as a keyboard and a mouse.” Ex-1004 (Hoy) ¶ 0039. At the time when Hoy was filed, a POSITA would have understood that such a PC or laptop would necessarily have included a processor in order to operate. Additionally, Hoy expressly teaches that “additional **processor** demand” would be incurred by malware executing on a computer, which provides an explicit teaching of a processor.

111. Accordingly, Hoy teaches element 1b.

[1c] implement a workspace, wherein the workspace is configured to enable operation of a first set of one or more applications or processes via a first memory space;

112. Hoy teaches element 1c.

113. The ’104 patent does not explicitly define a “workspace,” which has many meanings in the network security field. As used in the ’104 patent, however, a workspace broadly appears to be a computer environment in which one or more applications may run. *See, e.g.*, Ex-1001, 1:60-63.

114. Hoy discloses the implementation of a **workspace** in the form of a trusted host operating system 17 that runs a standard operating system, such as Windows 7, and software applications, such as a browser and an email program. Ex-1004 (Hoy) ¶¶ 0043, 0057, 0058, 0080. The trusted host OS workspace is functionally illustrated in Figure 1 of Hoy, reproduced below with the trusted host

OS workspace highlighted. Ex-1004 (Hoy), Fig. 1. In addition to running an operating system and software applications like Internet Explorer and Mozilla Firefox, the trusted host operating system 17 of Hoy can display a desktop to a user, direct the transfer of files, and access websites that are on a whitelist of predetermined trusted sites, Ex-1004 (Hoy) ¶¶ 0047, 0052, 0057, 0080.

Accordingly a POSITA would have understood that the trusted host operating system 17 of Hoy is a

workspace as that term is used in the '104 patent and that this workspace is **configured to enable operation of a first set of one or more applications or processes**, such as Internet Explorer and Mozilla Firefox.

115. Hoy further discloses that the trusted host operating system 17 (the workspace) is configured to enable the operation of the applications or processes **via a first memory space**. In particular, Hoy teaches that, unless malicious

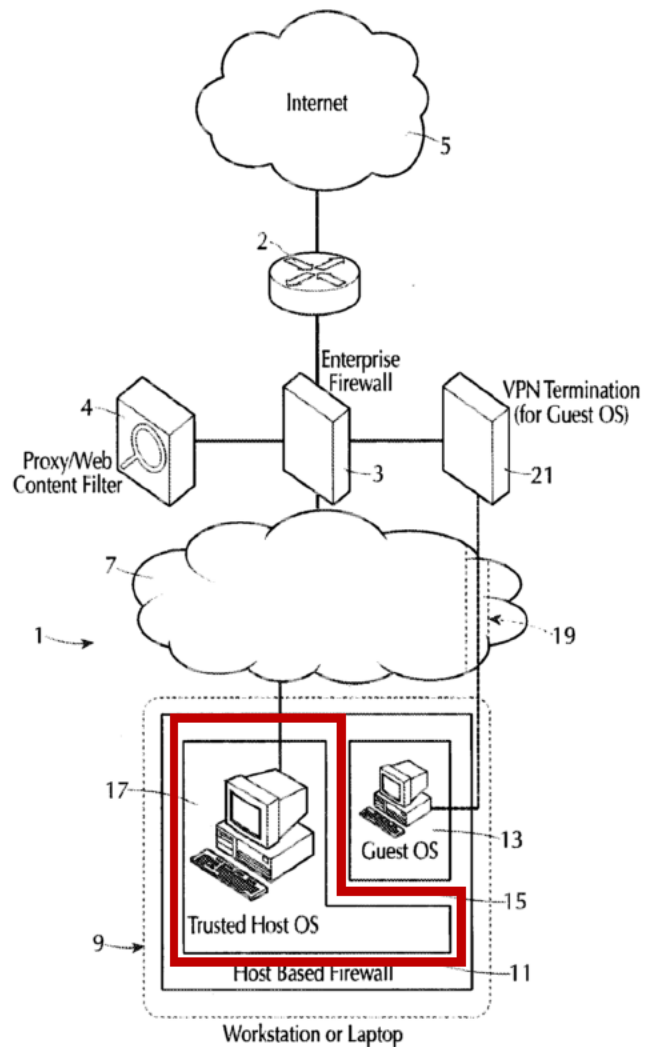


FIG. 1

software is isolated on a host computer, such software “allows its creators to simply capture and reuse directly from random access memory of the infected host computer, or from a network data stream, the security token or hash of the user, ... and use that token or hash to gain privileged access to other resources on the LAN.” Ex-1004 (Hoy) ¶ 0009. Hoy thereafter discloses a solution in which “[t]he guest is separated and completely isolated by an internal firewall from the host, and the guest cannot access any of the resources of the host computer”

Ex-1004 (Hoy) ¶ 0019.

116. A POSITA would have recognized that isolating the guest from accessing the resources of the host, as taught in Hoy, would have included isolating the guest from accessing the random access memory space of the host operating system (since memory is a resource), thus creating a first memory space for the host operating system and a second memory space for the virtual guest system. That is how internal isolation firewalls, like the internal host-supported firewall 15 taught in Hoy, worked as of Hoy’s filing date. A POSITA would have agreed that the internal isolation firewall, which runs outside of the virtual guest system, would restrict the activities of the virtual guest system, such as detecting that the virtual guest system is attempting to access the memory space of the operating system and preventing such access per policy. It would have been obvious to a POSITA that without the random access memory of Hoy’s host computer 9 being

partitioned into first and second memory spaces, the virtual guest system 13 would be able to access the memory being used by the host operating system 17, and vice versa, which is the problem that Hoy was trying to solve. Accordingly, Hoy teaches a workspace (the host operating system 17) that is configured to enable operation of applications or processes via a first memory space.

117. Additionally, Hoy teaches “a computer memory area (such as a respective desktop) or data storage accessible to the host OS 17.” Ex-1004 (Hoy) ¶ 0047. A POSITA would have interpreted Hoy’s reference to “a computer memory area” broadly, encompassing both volatile random access memory and non-volatile memory like solid-state and hard drives. Such an interpretation is supported by the ’104 patent itself, which recognizes that:

Memory 14 may include volatile and/or non-volatile memory. Memory 14 may include read-only memory (ROM), random access memory (RAM), a register, cache memory, semiconductor memory devices, magnetic media such as, but not limited to, internal hard disks and removable disks, magneto-optical media, and/or optical media such as CD-ROM disks, and/or digital versatile disks (DVDs), and/or the like.

Ex-1001, 9:4-11. Accordingly, Hoy’s disclosure of “a computer memory area ... accessible to the host OS 17” further teaches a workspace (the host operating system 17) that is configured to enable operation of applications or processes via a first memory space. Ex-1004 (Hoy) ¶ 0047.

118. Accordingly, Hoy teaches element 1c.

[1d] implement an isolated computing environment, the isolated computing environment using a host operating system and comprising a sandboxed computing environment that uses a second memory space to enable operation of a second set of one or more applications or processes,

119. Hoy teaches element 1d.

“implement an isolated computing environment”

120. Hoy discloses the implementation of an isolated computing environment in the form of a virtual guest system 13. This virtual guest system is functionally illustrated in Figure 1 of Hoy, reproduced at right with the virtual guest system highlighted. Ex-1004 (Hoy), Fig. 1.

121. Hoy teaches that the virtual guest system 13 is separated from the trusted host operating system 17 by an internal host-supported firewall 15 and that interaction between the two systems is restricted. Ex-1004 (Hoy) ¶ 0043.

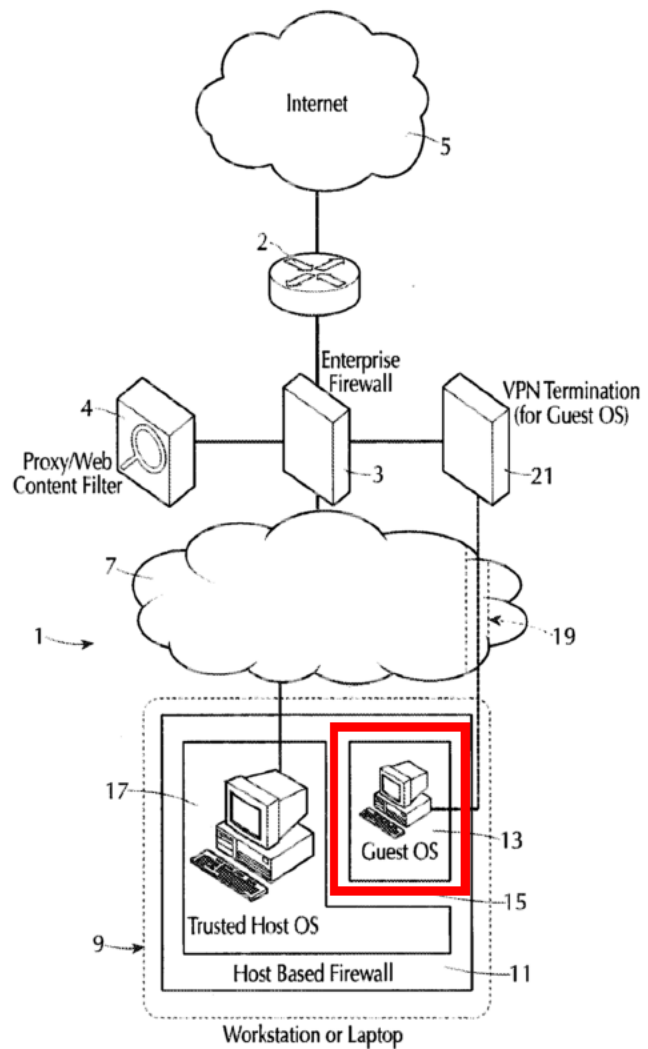


FIG. 1

Specifically, “data from sites that are opened by the virtual guest system browser remains **isolated** in the virtual system behind the internal firewall 15, and can only be moved by ... limited types of transfer” Ex-1004 (Hoy) ¶ 0083. The virtual guest system 13 thus is an isolated computing environment—in the words of Hoy, “a secure isolated service that can access the Internet, but not the internal trusted LAN sites.” Ex-1004 (Hoy) ¶ 0014; *see also id.* ¶ 0019 (“The guest is separated and completely isolated by an internal firewall from the host, and the guest cannot access any of the resources of the host computer, except that the guest can initiate cut, copy and paste operations that reach the host, and the guest can also request print of documents.”), ¶ 0020 (“[T]he virtual computer system is isolated against any other communication over the local area network.”), ¶ 0026 (“the isolated Internet access of the virtual guest system”), ¶ 0031 (“the virtual guest machine, which is configured by the hypervisor as essentially a blank PC with a browser and isolated from the real host computer”).

“the isolated computing environment using a host operating system”

122. Hoy also discloses that the virtual guest system 13 (an isolated computing environment) uses a host operating system. As the patent examiner found in the final office action dated September 9, 2020, the virtual machine environment in Hoy uses an operating system that is associated with the host operating system architecture, citing the following disclosure in paragraph 26 of

Hoy: “The browser is activated by a desk-top shortcut or application on the host system that allows the user to link to the isolated Internet access of the virtual guest system by launching the virtual guest web browser, which operates transparently to the user in the virtual guest system.” Ex-1002 (Sept. 9, 2020 Office Action at 3); Ex-1004 (Hoy) ¶ 0026. The examiner stated that this disclosure “explicitly and clearly disclosed” the limitation of an “isolated computing environment using an operating system associated with the hosting computing system.” Ex-1002 (Sept. 9, 2020 Office Action at 3). I agree. While the Applicant later simplified the limitation “the isolated computing environment using an operating system associated with the host computing system” as “the isolated computing environment using a host operating system,” in its *Request for Continued Examination* and a *Reply Pursuant to 37 CFR § 1.114* dated March 8, 2021, Ex-1002 (Mar. 8, 2021 Reply at 2, 6), this simplification would not have changed the meaning of the claim language to a POSITA. For a POSITA, an “operating system associated with the host computing system” would have meant the same thing as a “host operating system.” The two phrases are interchangeable.

123. Additionally, Hoy discloses that the virtual guest system 13 is created by a hypervisor on the host workstation 9. Ex-1004 (Hoy) ¶ 0042. Hoy makes clear that this hypervisor “may be one of a variety of types of hypervisor, either software that executes directly as a separate operating system on the hardware of

computer 9, or as **a software module that operates based on the underlying operating system, e.g., Windows 7, of the computer 9.**” *Id.* A POSITA would have understood this teaching in Hoy to be a disclosure of an isolated computing environment using a host operating system. Among other things, a POSITA would have known that a hypervisor is responsible for creating, monitoring, and terminating a virtual machine; that is, a virtual machine depends on a hypervisor. When a hypervisor is a software module that operates based on the underlying host operating system, it depends on the host operating system. Therefore, by the chain of dependency, a virtual machine (or, in Hoy’s teaching, a virtual guest system) depends on the host operating system, which a POSITA would have understood in the context of Hoy as using the host operating system.

124. Hoy does refer to the virtual guest system 13 in several places as a “guest OS,” but the disclosures discussed above would have made clear to a POSITA that this “guest OS” operates based on the underlying host operating system, and that the standard operating system running the workstation 9 supports and cooperates with the hypervisor. Ex-1004 (Hoy) ¶¶ 0042-0043. Indeed, it is the host operating system 17 that initiates the browser program running in the virtual guest system 13 through the user’s activation of an icon included in the host OS desktop. Ex-1004 (Hoy) ¶ 0052.

“and comprising a sandboxed computing environment”

125. Hoy further discloses that the virtual guest system 13 comprises a sandboxed computing environment. As the patent examiner found in the final office action dated September 9, 2020, Hoy teaches a sandboxed computing environment through at least the following disclosure in Hoy:

Because it is exposed to potentially malicious software as it browses the Internet, the virtual guest system browser and its virtual operating system may be a hardened version that is exceptionally resistant to corruption, or it may have enhanced anomaly detection to aid in noting when dangerous data has been encountered by the browser. The virtual guest system also has no connection whatsoever to the LAN, and cannot access any address on the LAN.

Ex-1002 (Sept. 9, 2020 Office Action at 5); Ex-1004 (Hoy) ¶ 0027. The examiner stated that this disclosure “is essentially a definition of sandboxed environment as known in the art.” Ex-1002 (Sept. 9, 2020 Office Action at 5). I agree.

126. Moreover, under the proper construction of a “sandboxed computing environment” as discussed in the claim-construction section above, a POSITA would have understood that the virtual guest system 13 of Hoy is a computing environment that does not require its own separate operating system. While Hoy does refer to the virtual guest system 13 in several places as a “guest OS,” I do not see any requirement in Hoy that the virtual guest system 13 actually boot a separate operating system. The main requirement for Hoy’s hypervisor is that it

creates “a virtual machine operating on the host computer that has a virtual configuration different from the real hardware and software configuration of the computer 9.” Ex-1004 (Hoy) ¶ 0042. A POSITA would have known that such a virtual configuration could be accomplished with the same operating system as the host. Indeed, Hoy explicitly teaches that the hypervisor may be “a software module that operates based on the underlying operating system, e.g., Windows 7, of the computer 9.” *Id.* A POSITA would have known that a hypervisor can create a virtual guest system that does not run a full-fledged guest operating system but rather runs a “stripped down” guest operating system that uses the host operating system to perform activities such as file access and network communications; for example, requests by the virtual guest system to perform these activities are passed to the hypervisor in the host operating system, which can grant or deny such requests. Similarly, Pratt explicitly teaches that “[a] host operating system or a guest operating system may execute within each of a virtual machine, a container, and a sandbox environment,” further confirming the understanding of a POSITA. Ex-1006 (Pratt), 2:34-36.

127. The virtual guest system 13 of Hoy also meets the definition of a sandboxed computing environment under the alternative construction that I discuss in the claim-construction section above (“an area in which software is confined and prevented from accessing system resources”). In particular, Hoy teaches that

“[t]he guest is separated and completely isolated by an internal firewall from the host, and the guest cannot access any of the resources of the host computer, except that the guest can initiate cut, copy and paste operations that reach the host, and the guest can also request print of documents.” Ex-1004 (Hoy) ¶ 0019. A POSITA would have understood that, if the guest cannot access any of the resources of the host computer, then software operating in the guest likewise is prevented from accessing those system resources.

“that uses a second memory space to enable operation of a second set of one or more applications or processes”

128. Hoy discloses that the virtual guest system 13 uses a second memory space to enable operation of a second set of one or more applications or processes.

129. Hoy teaches that “[t]he guest is separated and completely isolated by an internal firewall from the host.” Ex-1004 (Hoy) ¶ 0019. As discussed above for element 1c, a POSITA would have understood this to mean that any applications that might be running on the guest system are running in the memory space of the virtual guest machine or sandbox, and that the memory space of the virtual guest system is separate from the host’s memory space. A POSITA would have recognized that isolating the guest from accessing the resources of the host, as taught in Hoy, would have included isolating the guest from accessing the random access memory space of the host operating system (since memory is a resource), thus creating a first memory space for the host operating system and a second

memory space for the virtual guest system. That is how internal isolation firewalls, like the internal host-supported firewall 15 taught in Hoy, worked as of Hoy's filing date, and a POSITA would have known this. Accordingly, Hoy teaches that the virtual guest system 13 uses a second memory space to operate applications.

130. Additionally, Hoy teaches "a computer memory area (such as a respective desktop) or virtual data storage accessible to the virtual guest OS 13." Ex-1004 (Hoy) ¶ 0047. Hoy distinguishes between this computer memory area and the computer memory area accessible to the host operating system 17, which is isolated from the virtual guest system 13 as discussed for element 1d. *Id.* As discussed above for element 1c, a POSITA would have interpreted Hoy's reference to "a computer memory area" broadly, encompassing both volatile random access memory and non-volatile memory like solid-state and hard drives. *See also* Ex-1001, 9:4-11. Accordingly, Hoy's disclosure of "a computer memory area ... accessible to the virtual guest OS 13" further teaches an isolated computing environment (the virtual guest system 13) that **uses a second memory space** to operate applications. Ex-1004 (Hoy) ¶ 0047.

131. As also discussed above for this element 1d, the virtual guest system 13 operates a **set of applications or processes** in a manner that is isolated from the host operating system 17. Hoy teaches that these applications or processes operate in the virtual guest system and include "a browser and possibly

some related supportive software, such as Adobe or Windows Media Player that cooperate with the browser when certain types of files are accessed on the Internet, and potentially an email program such as Microsoft Outlook for certain configurations.” Ex-1004 (Hoy) ¶ 0048; *see also id.* ¶ 0052 (“the browser program running in the virtual guest system 13”). Accordingly, Hoy teaches using the second memory space to enable operation of a second set of one or more applications or processes.

132. Accordingly, Hoy teaches element 1d.

[1e] wherein the isolated computing environment is configured to access an Internet-based cloud service via at least one application of the second set of one or more applications or processes;
--

133. Hoy discloses that the virtual guest system 13 (an **isolated computing environment**) is **configured to access an Internet-based cloud service via at least one application** operating on the virtual guest system (**the second set of one or more applications or processes**), and therefore teaches element 1e. The Internet-based cloud service is illustrated in Figure 1 of Hoy literally as a cloud with the word “**Internet**” inside. Ex-1004 (Hoy), Fig. 1.

134. Hoy explains that “[t]he Internet provides access to a large number of computer systems,” which a POSITA would have understood as **Internet-based cloud services** at the time that Hoy was filed (January 27, 2012). Hoy discloses that these computer systems may include a website, an Internet service that permits

the downloading of files, an Internet service that permits the uploading of files, websites that provide access to online software certificates, websites that provide access to online software help, an email service accessible through a browser application, etc. Ex-1004 (Hoy) ¶¶ 0031, 0050, 0051, 0057, 0058, 0075. A POSITA would have recognized all of these websites and services as Internet-based cloud services at the time that Hoy was filed.

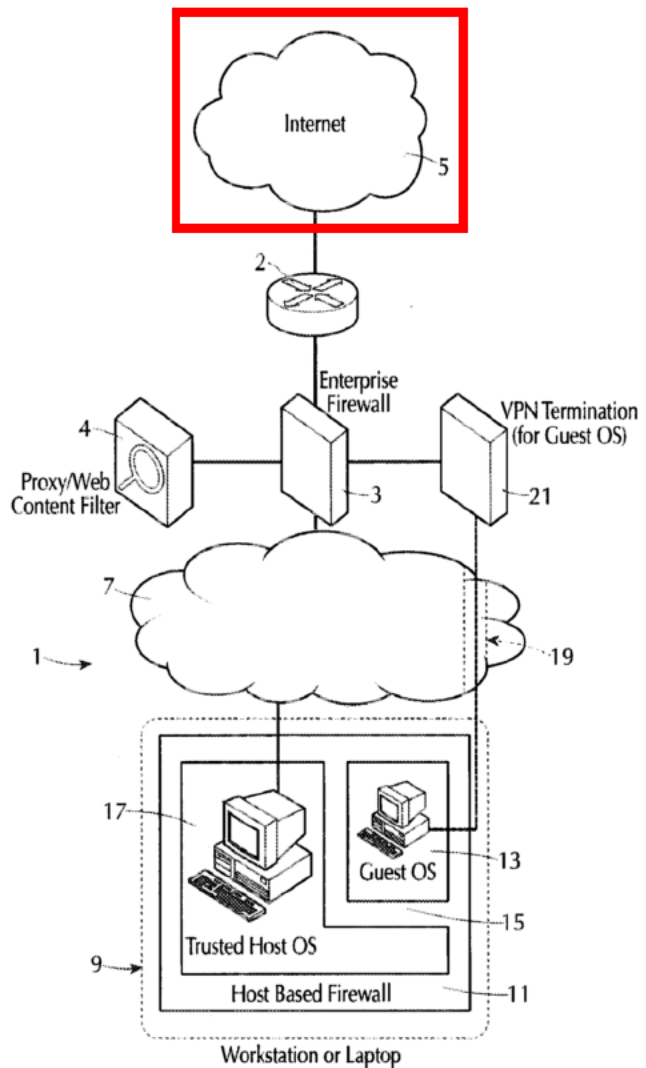


FIG. 1

135. To access these Internet-based cloud services, Hoy discloses that the virtual guest system 13 has an **application or process** in the form of an Internet browser or email program (at least one application of the second set of one or more applications or processes). *Id.*; see also *id.* ¶¶ 0048, 0052.

136. Accordingly, Hoy teaches element 1e.

[1f] isolate the isolated computing environment from the workspace using an internal isolation firewall;

137. Hoy teaches element 1f.

138. As discussed above for element 1d, Hoy discloses **isolating** the virtual guest system 13 (an **isolated computing environment**) from the host operating system 17 (a **workspace**). Hoy discloses that isolation is accomplished **using an internal isolation firewall** in the form of an internal host-supported firewall 15. The internal firewall 15 is in addition to the host-based firewall 11 and is functionally

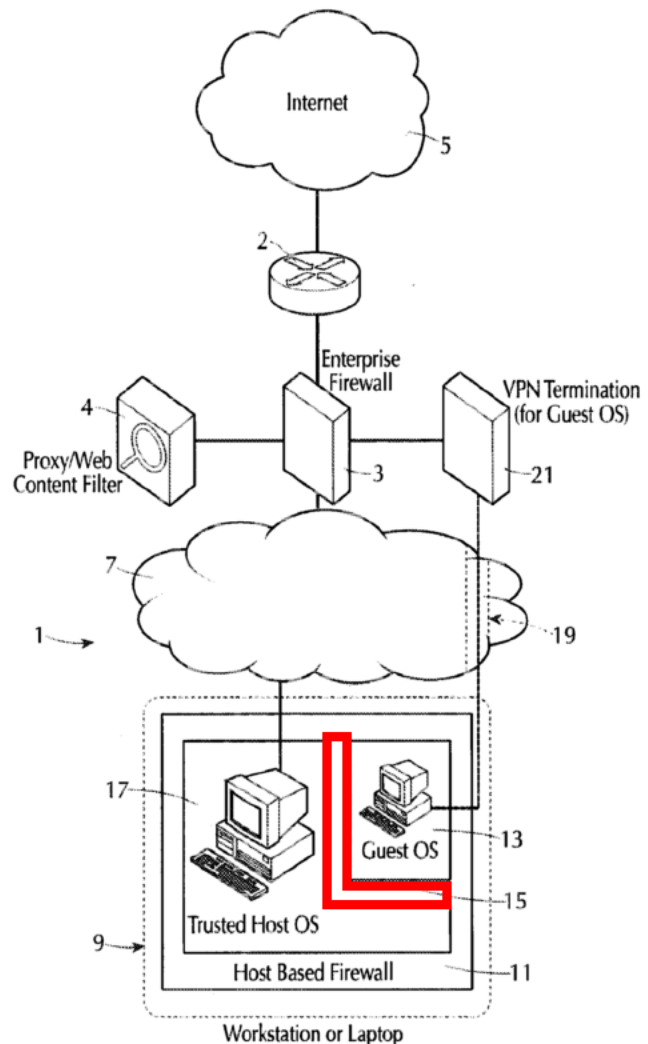


FIG. 1

illustrated in Figure 1 of Hoy, reproduced above with the internal firewall 15 highlighted. Ex-1004 (Hoy), Fig. 1.

139. Hoy teaches that the internal firewall 15 “separates and restricts interaction between virtual guest system 13 and the trusted-host operating system 17,” thus isolating the virtual guest system 13 from the host operating

system 17. Ex-1004 (Hoy) ¶ 0043. In particular, “data from sites that are opened by the virtual guest system browser remains isolated in the virtual system behind the internal firewall 15, and can only be moved by the limited types of transfer described above, i.e., cut, copy, paste, print, etc.” Ex-1004 (Hoy) ¶ 0083; *see also id.* ¶ 0019 (“The guest is separated and completely isolated by an internal firewall from the host, and the guest cannot access any of the resources of the host computer.”), ¶ 0021 (“a firewalled virtual computer system with which the host computer system has restricted communication of data”), ¶ 0030 (“Communications across this internal firewall between the guest system and the host system are limited”), ¶ 0074 (“[T]he firewall 15 of the first embodiment restricts communication between its host and guest systems.”), ¶ 0083 (“[T]he other aspects of the firewall 15 described previously are fully maintained to keep the systems isolated.”).

140. Accordingly, Hoy teaches element 1f.

[1g] authenticate the isolated computing environment with an authentication device; and

141. Hoy in combination with Pratt teaches element 1g.

142. Hoy discloses several devices that could perform the function of authenticating the virtual guest system 13 (an isolated computing environment), including the router 2, the firewall 3, the proxy/web-content filter 4, a cloud

service on the Internet 5, and the Virtual Private Network (VPN) termination point 21. Ex-1004 (Hoy), Fig. 1. Hoy also discloses that “[t]he VPN transmits data packets that are encrypted according to an IPSEC or SSL VPN tunnel construct protocol.” Ex-1004 (Hoy) ¶ 0054. Both the Internet Protocol Security (IPsec) standard and the Secure Sockets Layer (SSL) standard contemplate the authentication of computing environments that are sending the data packets, but Hoy does not discuss the details of the specific protocols or authentication more generally.

143. Pratt provides authentication details, which as discussed above, a POSITA would have been motivated to combine with the computer system of Hoy with a reasonable expectation of success. As discussed above, Pratt teaches a device 110 having an isolated operating environment 120 that may be securely authenticated to a service or resource provider 150. Ex-1006 (Pratt), 2:15-22 (“System 100 may be used to securely **authenticate operating environment 120** to service or resource provider 150”). Like the virtual guest system 13 in Hoy, the operating environment 120 in Pratt is an isolated computing environment that can comprise a sandboxed computing environment and use a host operating system. Ex-1006 (Pratt), 2:29-36. And like Hoy’s virtual guest system 13, Pratt’s

operating environment 120 is an operating environment in which malicious code may be inadvertently introduced. Ex-1006 (Pratt), 2:38-40.

144. In Pratt, each operating environment (the untrusted operating environment 120 and a separate trustworthy operating environment 130) may comprise an “authentication service,” such as the authentication services 122 and 132 shown in Figure 1 of Pratt (reproduced at right). Ex-1006 (Pratt), 2:59-63 (“[O]perating environment 120 comprises authentication service 122 and

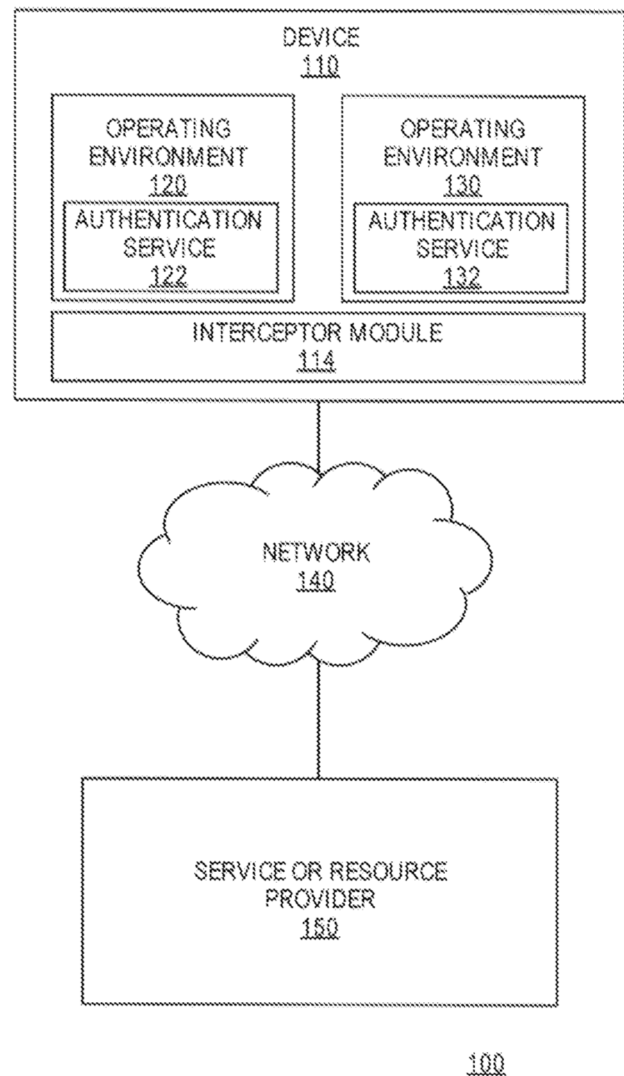


FIG. 1

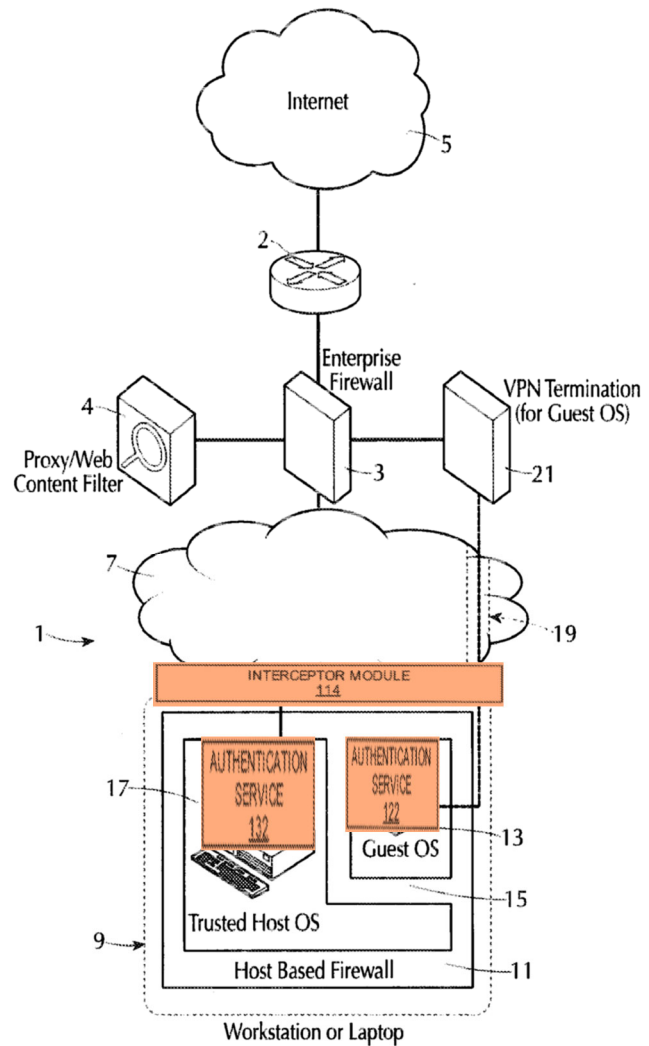
operating environment 130 comprises authentication service 132.”). Each of these authentication services is “a component capable of providing authentication data to an operating system.” Ex-1006 (Pratt), 2:63-65. Pratt highlights that, as of its filing date (January 23, 2015), there were “many types of schemas and approaches for performing authentication” of an operating environment. Ex-1006 (Pratt),

2:65-3:2. Indeed, authenticating operating environments, both isolated and non-isolated, was well known and conventional to people of ordinary skill in the network security field at the time.

145. As a POSITA would have expected, authentication of Pratt's operating environments is needed before the service or resource provider 150 will provide its service or resource. Ex-1006 (Pratt), 3:49-52. In operation, an interceptor module 114 intercepts a message from the provider 150 that requires the untrusted operating environment 120 to authenticate itself. Ex-1006 (Pratt), 4:14-16. Instead of notifying the untrusted operating environment of the intercepted message, the interceptor module 114 notifies the trusted operating environment 130. Ex-1006 (Pratt), 4:40-41. Through its authentication service 132, the trusted operating environment 130 provides the authentication data for the provider 150 to the untrusted operating environment 120, which then "authenticates itself to provider 150 using the authentication data." Ex-1006 (Pratt), 5:1-3, 5:43-53. In this way, "operating environment 120 is able to authenticate itself to provider 150 without exposing ... any malicious code which may be inadvertently introduced or present within operating environment 120 to the authentication mechanism" Ex-1006 (Pratt), 5:54-58.

146. As discussed above, a POSITA would have been motivated to combine the authentication process of Pratt with the computer system of Hoy with

a reasonable expectation of success. To accomplish this, a POSITA would have configured the virtual guest system 13 of Hoy to authenticate itself in a manner similar to the untrusted operating environment 120 of Pratt. Both the virtual guest system 13 in Hoy and the untrusted operating environment 120 in Pratt are isolated, sandboxed computing environments, and Hoy's computer system would be adapted to include the interceptor module 114 and authentication services 122 and 132 of Pratt in an arrangement similar to what is shown in Figure 1 of Pratt.



host operating system 17 is modified to include Pratt's authentication service 132.

As in Pratt, the interceptor module 114 of the combined system intercepts the authentication messages and notifies the authentication service 132 in the host operating system 17, which then provides authentication data as appropriate to the virtual guest system 13 so that it can authenticate itself to the requesting entity. As in Pratt, the virtual guest system 13 can authenticate itself to the requesting entity without exposing any malicious code to the authentication service 132 in the host operating system 17.

148. The combination of Hoy and Pratt involves using conventional and well-known computer programming methods to yield a predictable result. The implementation of Pratt's authentication process in Hoy's computer system would not have required specialized skills beyond what a POSITA would have had.

149. The combination also would not have changed the principle of operation for any of the teachings of the references. As noted above, Hoy implicitly contemplates the authentication of the virtual guest system 13 because Hoy teaches transmitting data packets using the IPsec or SSL protocols, which contemplate the authentication of the computing environments that are sending the data packets. Additionally, Hoy already teaches that "[t]he host monitors the guest browser at all times and screens any destination resource or address that the guest browser tries to access." Ex-1004 (Hoy) ¶ 0082. It would have been

straightforward for a POSITA to have added Pratt's interceptor module 114, which would similarly monitor the guest browser's communications with entities beyond the trusted LAN 7 and screen them for authentication requests.

150. The combination of Hoy and Pratt thus teaches **authenticating an isolated computing environment** (the virtual guest system 13) **with an authentication device**. The authentication device in the combination can be whatever entity is ultimately receiving the authentication data from the virtual guest system 13 and performing the function of authenticating that computing environment, be it the router 2, the firewall 3, the proxy/web-content filter 4, a cloud service on the Internet 5, or the Virtual Private Network (VPN) termination point 21. As discussed above, the scope of "authentication device" in the '104 patent includes, at a minimum, a proxy server that performs the function of authenticating a computing environment. In the Hoy-Pratt system, the authentication device in the combination would be the proxy/web-content filter 4 in Hoy, as Pratt explicitly discloses that the provider 150 requesting authentication may be a proxy server, such that the operating environments are authenticated with a proxy. Ex-1006 (Pratt), 3:52-54.

151. Accordingly, Hoy in combination with Pratt teaches element 1g.

[1h] communicate with a proxy server to access the Internet-based cloud service to allow communication between the at least one application and the Internet-based cloud service when the isolated computing environment has been authenticated.

152. Hoy in combination with
Pratt teaches element 1h.

153. Hoy teaches
communicating with the proxy/web-
content filter 4 (**a proxy server**) to
access an Internet-based cloud
service to allow communication
between at least one application
(e.g., a browser or an email program)
operating on the virtual guest
system 13 **and the Internet-based**
cloud service. The proxy/web-
content filter 4 is functionally

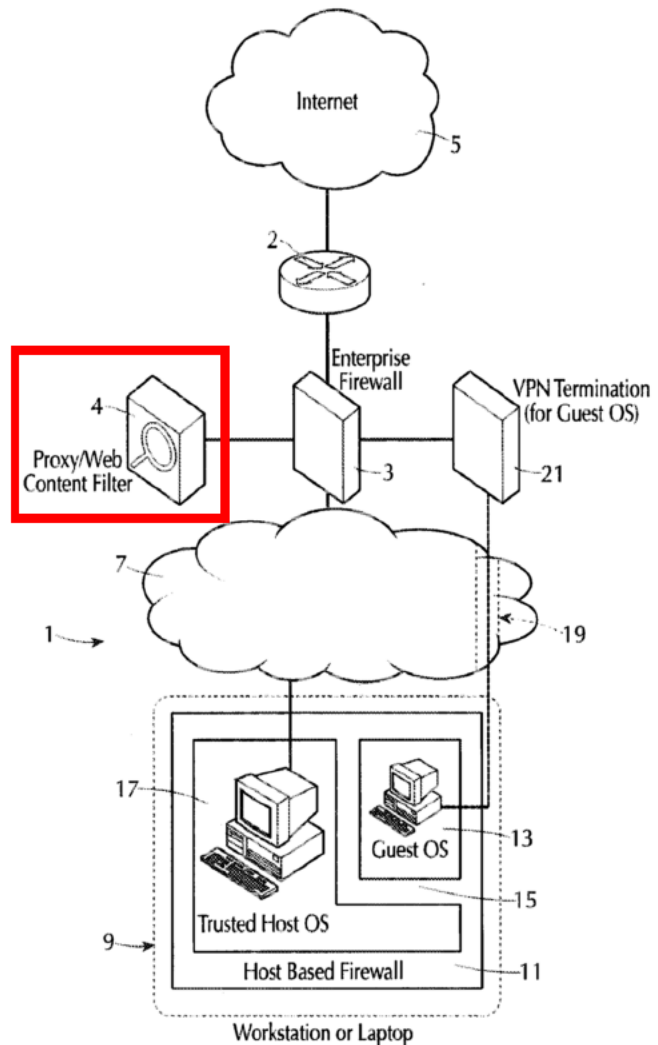


FIG. 1

illustrated in Figure 1 of Hoy, reproduced above with the proxy/web content filter highlighted. Ex-1004 (Hoy), Fig. 1.

154. In particular, Hoy discloses that “[a] **proxy/web-content filter 4** is connected with and cooperates with the firewall hardware 3 to control the type of

data permitted to pass from the Internet 5 into the local system 1, filtering e.g., pornographic material or data not intended for the system 1.” Ex-1004 (Hoy) ¶ 0038. Hoy teaches that the proxy/web-content filter 4 may also be used to screen undesirable or inappropriate outgoing data. Ex-1004 (Hoy) ¶ 0073. The workstation or laptop 9 of Hoy thus communicates with the proxy/web-content filter 4 to access the Internet and the cloud services thereon, discussed in connection with element 1e. If the proxy/web-content filter 4 permits the data to pass, then communication is allowed between the requesting application operating on the workstation’s virtual guest system 13 and the Internet-based cloud service.

155. While Hoy uses the terminology “proxy/web-content filter,” a POSITA at the time that Hoy was filed would have known that proxy/web-content filters were commonly and conventionally implemented on servers. Indeed, that is what is shown in Figure 1 of Hoy, where the proxy/web-content filter is shown being implemented on its own server.

156. Hoy in combination with Pratt discloses that the communication between the requesting application and the Internet-based cloud service is allowed when the virtual guest system 13 (the isolated computing environment) has been **authenticated**. Pratt explains that authentication of an isolated operating environment like the virtual guest system 13 in Hoy might be needed for a service or resource provider like the Internet-based cloud services in Hoy to provide its

services or resources to an application running in the isolated operating environment and seeking access. Ex-1006 (Pratt), 3:49-52. As discussed above for element 1g, Pratt teaches a system and method for authenticating the isolated operating environment, after which communication between the application running in the isolated operating environment and the provider is allowed. Pratt also teaches that the provider may be a proxy server, such that the operating environments communicate and are authenticated with a proxy. Ex-1006 (Pratt), 3:52-54.

157. Accordingly, Hoy in combination with Pratt teaches element 1h.

3. Dependent Claim 2

[2pre] The host computer system of claim 1,
--

158. As discussed above, Hoy in combination with Pratt discloses the host computer system of independent claim 1, and therefore teaches the preamble of dependent claim 2.

[2a] wherein the processor is configured to access the Internet-based cloud service via one or more of a local area network (LAN), a wide area network (WAN), a cellular network, a Wi-Fi network, a digital subscribers line (DSL) network, a cloud computing network, a data center, or a satellite network.

159. Hoy teaches element 2a.

160. As discussed above for element 1e, Hoy teaches that the virtual guest system 13 implemented by the workstation's processor is **configured to access an Internet-based cloud service**.

161. Hoy additionally teaches that the access is done **via one or more of a local area network (LAN), a wide area network (WAN), or a Wi-Fi network**.

In particular, Hoy discloses a trusted LAN and the Internet (which a POSITA would have recognized as a type of WAN), as functionally illustrated in Figure 1

of Hoy, reproduced at right with the trusted LAN 7 and the Internet 5 highlighted. Ex-1004 (Hoy), Fig. 1.

Hoy explains: "the traffic between the

Internet and the browser of the virtual

guest system 13 is routed through a

sealed 'virtual conduit' 19 passing

through the LAN." Ex-1004 (Hoy) ¶

0052. The data packets in the

outgoing traffic are encrypted and

then "transmitted over the LAN from

the virtual guest system 13 on the

host computer 9 in this encrypted

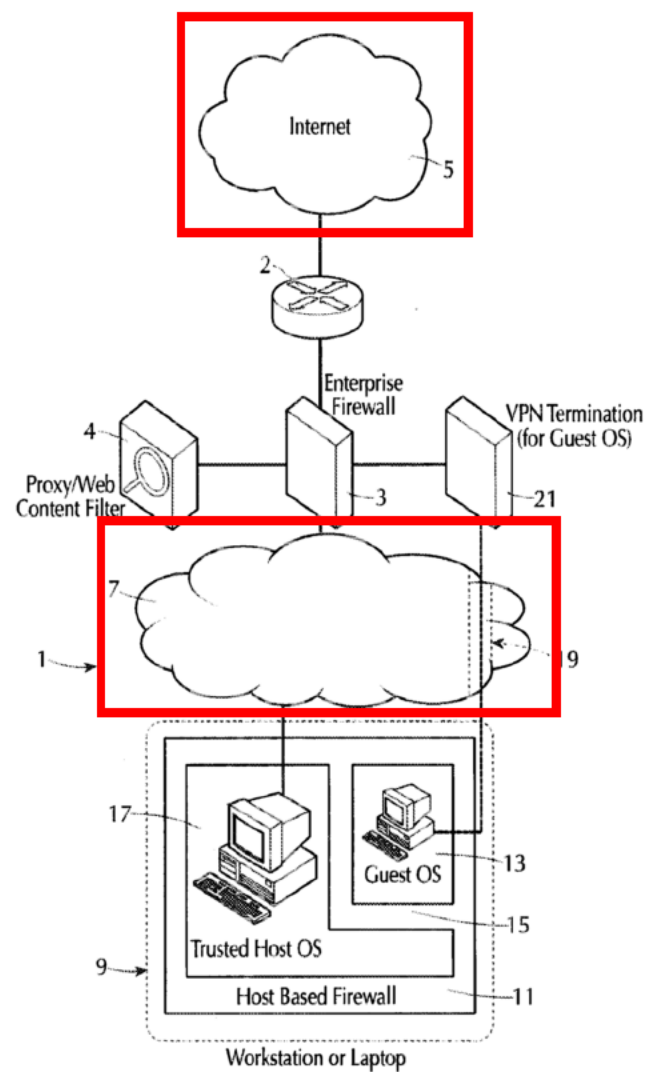


FIG. 1

form to the VPN termination point 21, where it is decrypted for transmission to the Internet.” Ex-1004 (Hoy) ¶ 0054. The data packets in the incoming traffic are “encrypted at the VPN termination point and then sent encrypted through the LAN to the host computer 9.” Ex-1004 (Hoy) ¶ 0055. Hoy thus teaches that the virtual guest system 13 implemented by the workstation’s processor is configured to access the Internet-based cloud service via the LAN and the Internet (a WAN).

162. Hoy further teaches that the access can be done via a **Wi-Fi network**. In particular, Hoy discloses that “if the system is a laptop connecting to the Internet in a hotel room or via a **Wi-Fi network**, the computer system 9 can still connect to the Internet without requiring the VPN.” Ex-1004 (Hoy) ¶ 0064.

163. Accordingly, Hoy teaches element 2a.

4. Dependent Claim 3

[3pre] The host computer system of claim 1,
--

164. As discussed above, Hoy in combination with Pratt discloses the host computer system of independent claim 1, and therefore teaches the preamble of dependent claim 3.

[3a] wherein the Internet-based cloud service comprises one or more of an Infrastructure as a Service (IaaS), a Software as a Service (SaaS), or a Platform as a Service (PaaS).

165. Hoy teaches element 3a.

166. As discussed above for element 1e, Hoy teaches an **Internet-based cloud service**. Hoy discloses that these services may include an Internet service that permits the downloading of files, an Internet service that permits the uploading of files, websites that provide access to online software certificates, websites that provide access to online software help, and an email service accessible through a browser application, etc. Ex-1004 (Hoy) ¶¶ 0031, 0050, 0051, 0057, 0058, 0075.

167. A POSITA would have recognized (1) the Internet service that permits the downloading of files and (2) the Internet service that permits the uploading of files as examples of **Infrastructure as a Service (IaaS)**. IaaS is a type of cloud service that offers computing, storage, or networking resources over the Internet. Both (1) the Internet service that permits the downloading of files and (2) the Internet service that permits the uploading of files would have been understood by a POSITA to be cloud services that offer storage resources over the Internet.

168. A POSITA would have further recognized (3) websites that provide access to online software certificates, (4) websites that provide access to online software help, and (5) an email service accessible through a browser application as examples of **Software as a Service (SaaS)**. SaaS is a type of cloud service that offers software resources over the Internet. A POSITA would have understood that such software resources could include the provision of online software

certificates, the provision of online software help, and the provision of an email service. Additionally, an email service accessible through a browser application would also have been recognized by a POSITA as an example of IaaS, since such email services at the time normally stored the user's email on a cloud server.

169. Accordingly, Hoy teaches element 3a.

5. Dependent Claim 8

[8pre] The host computer system of claim 1,

170. As discussed above, Hoy in combination with Pratt discloses the host computer system of independent claim 1, and therefore teaches the preamble of dependent claim 8.

[8a] wherein the internal isolation firewall is configured to segregate the first memory space from the second memory space, and

171. Hoy teaches element 8a.

172. As discussed above for elements 1c, 1d, and 1f of claim 1, Hoy teaches a first memory space (see element 1c), a second memory space (see element 1d), and an internal host-supported isolation firewall 15 (see element 1f). Hoy's **internal host-supported firewall** (also called a "hypervisor firewall") **is configured to segregate the first memory space from the second memory space**. Ex-1004 (Hoy) ¶ 0043. In particular, Hoy discloses that "[c]ommunications between the trusted host OS 17 and the guest OS 13 pass

through the hypervisor firewall 15, and are limited to ... [t]ransfer of files, when directed by the host computer OS 17, between a computer memory area (such as a respective desktop) or virtual data storage accessible to the virtual guest OS 13 and a computer memory area (such as a respective desktop) or data storage accessible to the host OS 17.” Ex-1004 (Hoy) ¶¶ 0043, 0047.

173. To a POSITA, this disclosure in Hoy would have made clear that the files accessible by virtual guest system 13 are not accessible by the host operating system 17, and vice versa, and that, therefore, the memory space for storing the files of virtual guest system 13 is segregated from the memory space for storing the files of the host operating system 17. Otherwise, the files would not need to be transferred between the two systems. Additionally, Hoy makes clear that the segregation is performed by the internal firewall 15, which is the disclosed gatekeeper for the transfer of files between the virtual guest system 13 and the host operating system 17. Hoy explains that the “hypervisor also provides an additional internal host-supported firewall indicated at 15 that separates and restricts interaction between virtual guest system 13 and the trusted operating system.” Ex-1004 (Hoy) ¶ 0043.

174. Accordingly, Hoy teaches element 8a.

[8b] wherein the processor is configured to enforce policies for communication between the first memory space and the second memory space using the internal isolation firewall.
--

175. Hoy teaches that the workstation's processor is **configured to enforce policies for communication between the first memory space** (accessible by the virtual guest system 13) **and the second memory space** (accessible by the host operating system 17) **using the internal isolation firewall** 15, and therefore teaches element 8b. Again, Hoy discloses that “[c]ommunications between the trusted host OS 17 and the guest OS 13 pass through the hypervisor firewall 15, and are limited” to four particular functions, including the transfer of files as discussed above. Ex-1004 (Hoy) ¶¶ 0043-0047. These four functions, listed in paragraphs 0044-0047 of Hoy, are policies for communication enforced by the processor using the internal isolation firewall, as recited in claim 8 of the '104 patent.

176. Additionally, Hoy discloses that communications specifically between the memory space for the virtual guest system 13 and the memory space for the host operating system 17 are limited to the transfer of files, and only when directed by the host operating system 17, which a POSITA would also have understood as policies for communication enforced by the processor using the internal isolation firewall, as recited in claim 8.

177. Accordingly, Hoy teaches element 8b.

6. Independent Claim 10

178. It is my opinion that the combination of Hoy and Pratt renders the limitations of claim 10 obvious. Claim 10 largely repeats the limitations of claim 1. I incorporate by reference the relevant analysis above for the limitations repeated from claim 1 and do not re-state that analysis in the following discussion, it being understood that the analysis would be the same.

[10pre] A cloud computing system comprising:

179. Hoy discloses a cloud computing system in the form of the networked computer system 1, and therefore teaches the preamble of claim 10. Ex-1004 (Hoy), Fig. 1; *see also id.* ¶ 0038 (“Referring to FIG. 1, a networked computer system generally indicated at 1 is linked to the Internet 5 via a firewall 3 and a router 2 of conventional configuration that is well-known in the art.”). As discussed above for element 1e, the workstation 9 in the networked computer system is configured to access an Internet-based cloud service, making the networked computer system 1 a cloud computing system.

180. Accordingly, to the extent that the preamble of claim 10 is limiting, Hoy teaches it.

[10a] a cloud computing network comprising a server,

181. Hoy teaches element 10a.

182. Hoy discloses a cloud computing network in the form of the networked computer system 1, with the system comprising a server in the form of the workstation 9, and therefore teaches element 10a of claim 10 of the '104 patent. Ex-1004 (Hoy), Fig. 1; *see also id.* ¶ 0038 (“Referring to FIG. 1, a networked computer system generally indicated at 1 is linked to the Internet 5 via a firewall 3 and a router 2 of

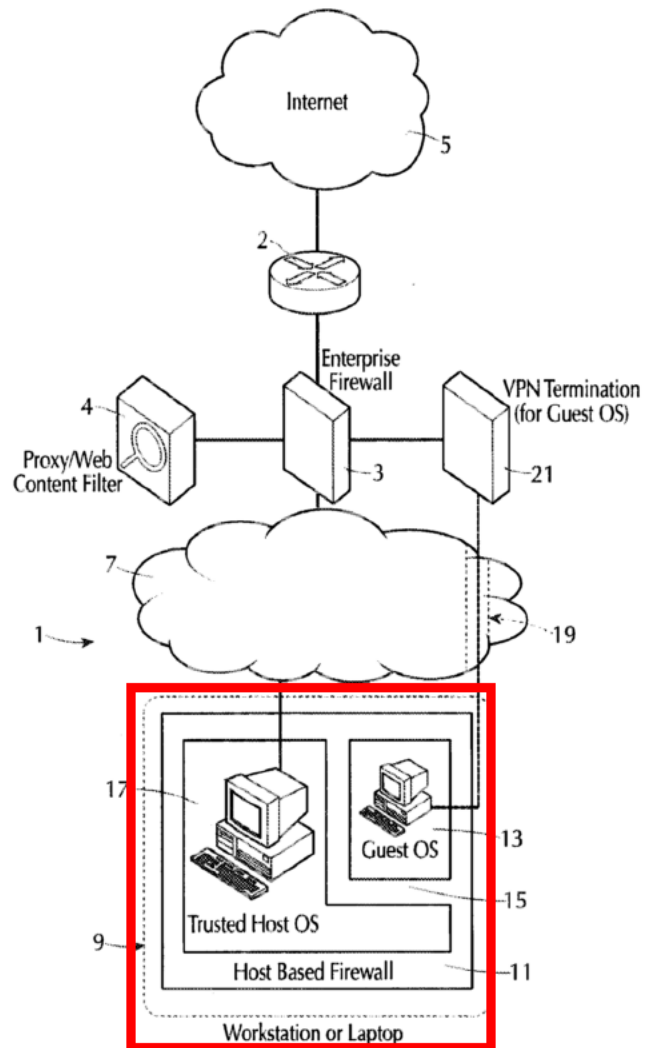


FIG. 1

conventional configuration that is well-known in the art.”). The workstation 9 is functionally illustrated in Figure 1 of Hoy, reproduced above with the workstation 9 highlighted.

183. A POSITA would have recognized the workstation 9 as a type of server since it can provide information to other devices on the trusted local area

network (LAN) 7. Indeed, Hoy explicitly refers to “servers or devices in the LAN.” Ex-1004 (Hoy) ¶ 0016.

184. As discussed above for element 1e, the workstation 9 in the networked computer system is configured to access an Internet-based cloud service, making the networked computer system 1 a cloud computing network comprising a server.

185. Accordingly, Hoy teaches element 10a.

[10b] wherein the server is configured to: implement a workspace configured to enable operation of a first set of one or more applications or processes;
--

186. As discussed above for element 10a, Hoy teaches a server in the form of the workstation 9. As discussed above for element 1c, Hoy teaches implementing a workspace in the form of the trusted host operating system 17 that is configured to enable operation of a first set of one or more applications or processes—a standard operating system, such as Windows 7, and software applications, such as a browser and an email program. Ex-1004 (Hoy) ¶¶ 0043, 0057, 0058, 0080. For the reasons discussed above for element 1c, Hoy discloses element 10b.

[10c] implement an isolated computing environment using a host operating system and comprising a sandboxed computing environment and being configured to enable operation of a second set of one or more applications or processes;

187. As discussed above for element 1d, Hoy discloses implementing an isolated computing environment in the form of a virtual guest system 13, as well as the remaining limitations in this element 10c. For the reasons discussed above for element 1d, Hoy discloses element 10c.

[10d] isolate the isolated computing environment from the workspace using an internal isolation firewall; and

188. As discussed above for element 1f, Hoy discloses isolating the virtual guest system 13 (an isolated computing environment) from the host operating system 17 (a workspace) using an internal isolation firewall in the form of an internal host-supported firewall 15. For the reasons discussed above for element 1f, Hoy discloses element 10d.

[10e] authenticate the isolated computing environment with an authentication device; and

189. As discussed above for element 1g, Hoy in combination with Pratt teaches element 10e, with the authentication details provided by Pratt as discussed above.

[10f] a proxy configured to control communication between the server and an Internet resource,

190. As discussed above for element 1h, Hoy teaches a proxy/web-content filter 4 controlling communications between the workstation 9 (a server) and resources on the Internet 5, and thus discloses element 10f. In particular, the “proxy/web-content filter 4 is connected with and cooperates with the firewall hardware 3 to control the type of data permitted to pass from the Internet 5 into the local system 1,” which includes the workstation 9. Ex-1004 (Hoy) ¶ 0038. For the reasons discussed above for element 1h, Hoy discloses element 10f.

[10g] wherein the proxy is configured to allow communication between the second set of one or more applications or processes running in the isolated computing environment and the Internet resource when the isolated computing environment has been authenticated and

191. Hoy in combination with Pratt teaches element 10g.

192. As discussed above for element 1h, Hoy teaches that the proxy/web-content filter 4 is configured to allow communication between one or more one applications or processes (e.g., a browser or an email program) running in the virtual guest system 13 (an isolated computing environment) and an Internet-based resource. As further discussed above for element 1h, Hoy in combination with Pratt discloses that the communication between the requesting application and the Internet-based resource is allowed when the virtual guest system 13 (the isolated

computing environment) has been authenticated. For the reasons discussed above for element 1h of claim 1, Hoy in combination with Pratt discloses element 10g.

[10h] to block communication between the first set of one or more applications or processes running in the workspace and the Internet resource.

193. Hoy discloses that the proxy/web-content filter 4 is also configured to block communication from applications or processes running in the workspace and Internet resources, and thus teaches element 10h. In particular, Hoy describes that the “proxy/web-content filter 4 ... blocks communications from users on the LAN trying to reach disallowed sites, e.g., requests to access www.onlinecasino.com.” Ex-1004 (Hoy) ¶ 0038; *see also id.* ¶ 0073 (“Proxy content filter 4 may optionally be used to screen undesirable or inappropriate incoming or outgoing data ...”).

194. Additionally, Hoy discloses that the “proxy/web-content filter 4 cooperates with the firewall hardware 3,” which in turn “is configured to block all host-based egress for all ports, protocols and destinations” with limited exceptions. Ex-1004 (Hoy) ¶¶ 0038, 0068. Through the cooperation of the proxy/web-content filter 4 and the firewall hardware 3, applications or processes running in the host operating system 17 (the workspace) are therefore blocked from communicating with Internet resources.

195. Accordingly, Hoy teaches element 10h.

7. Dependent Claim 11

[11pre] The cloud computing system of claim 10,

196. As discussed above, Hoy in combination with Pratt discloses the host computer system of independent claim 10, and therefore teaches the preamble of dependent claim 11.

[11a] wherein the server is configured to perform at least one processing task on behalf of a client device.

197. Hoy teaches element 11a. As discussed above for elements 1e and 3a, Hoy teaches an Internet-based cloud service, including file services (upload/download), online software certificates, online software help, and online email service, etc. Ex-1004 (Hoy) ¶¶ 0031, 0050, 0051, 0057, 0058, 0075. These are examples of where **the server is configured to perform at least one processing task on behalf of a client device**. The '104 patent describes the server performing a task for the client in the same manner:

The server may perform a processing task on behalf of a client device. For example, the server may receive input data from the client device and may send data to the client device. The processing task may include one or more of an Infrastructure as a Service (IaaS), a Software as a Service (SaaS), and/or a Platform as a Service (PaaS).

Ex-1001, 2:57-62. In each of Hoy's examples, the server receives input data from the client device, e.g., file upload.

198. Additionally, Hoy teaches an example “where the LAN includes a server inside the LAN for email.” A POSITA would have understood that such an email server would necessarily have been configured to perform processing tasks (e.g., receiving, storing, and providing access to email messages) on behalf of a client device (e.g., a user computer). A POSITA would also have understood that Hoy’s workstation 9 could be such an email server, since the workstation 9 is part of the LAN 7, with the email server configured to have the workstation’s host-based firewall 11, a virtual guest system 13, an internal host-supported firewall 15, a host operating system 17, etc. As a POSITA would have recognized, such a configuration would give the email server the ability to isolate untrusted emails sent by untrusted devices on the Internet 5 from trusted emails sent by trusted devices on the LAN 7, and thus keep malware sent by untrusted email from spreading to trusted devices on the LAN.

199. Accordingly, Hoy teaches element 11a.

8. Dependent Claim 17

[17pre] The cloud computing system of claim 11,
--

200. As discussed above, Hoy in combination with Pratt discloses the cloud computing system of dependent claim 11, and therefore teaches the preamble of dependent claim 17.

[17a] wherein the at least one processing task performed on behalf of the client device comprises a processing task for an Infrastructure as a Service (IaaS), a Software as a Service (SaaS), or a Platform as a Service (PaaS).

201. Hoy teaches element 17a.

202. As discussed above for element 3a, Hoy teaches services including an Internet service that permits the downloading of files, an Internet service that permits the uploading of files, websites that provide access to online software certificates, websites that provide access to online software help, and an email service accessible through a browser application, etc. Ex-1004 (Hoy) ¶¶ 0031, 0050, 0051, 0057, 0058, 0075. As further discussed above for element 3a, a POSITA would have recognized these services as examples of Infrastructure as a Service (IaaS), Software as a Service (SaaS), or both.

203. Accordingly, Hoy teaches element 17a.

9. Independent Claim 18

204. Independent claim 18 recites a method implemented in a host computer system that is the method counterpart of the host computer system of claim 1 with the same elements phrased in the form of method steps. Claim 18 contains no additional limitations beyond the fact that it recites a method instead of a system. For the reasons stated above in connection with claim 1, Hoy in combination with Pratt discloses the method recited in claim 18.

10. Dependent Claim 19

[19pre] The method of claim 18,

205. As discussed above, Hoy in combination with Pratt discloses the cloud computing system of independent claim 18, and therefore teaches the preamble of dependent claim 19.

[19a] wherein the Internet-based cloud service comprises one or more of an Infrastructure as a Service (IaaS), a Software as a Service (SaaS), or a Platform as a Service (PaaS).

206. Hoy teaches element 19a.

207. As discussed above for element 3a, Hoy teaches services including an Internet service that permits the downloading of files, an Internet service that permits the uploading of files, websites that provide access to online software certificates, websites that provide access to online software help, and an email service accessible through a browser application, etc. Ex-1004 (Hoy) ¶¶ 0031, 0050, 0051, 0057, 0058, 0075. As further discussed above for element 3a, a POSITA would have recognized these services as examples of Infrastructure as a Service (IaaS), Software as a Service (SaaS), or both.

208. Accordingly, Hoy teaches element 19a.

209. For the foregoing reasons, it is my opinion that a POSITA would have found claims 1-3, 8, 10, 11 and 17-19 of the '104 patent to be obvious based on the teachings of Hoy in view of Pratt.

B. Ground 2: Claims 1-3, 8, 10, 11 and 17-19 Are Obvious in View of Hoy, Pratt, and Krasin

210. It is my opinion that a POSITA would have found claims 1-3, 8, 10, 11 and 17-19 of the '104 patent to be obvious based on the teachings of Hoy in view of Pratt, and further in view of Krasin.

1. Combination of Hoy and Pratt

211. For the reasons discussed above in connection with Ground 1, it is my opinion that a POSITA would have been motivated to combine the authentication process of Pratt with the computer system of Hoy, and that POSITA would have had a reasonable expectation of success in making the combination.

2. Combination of Hoy and Pratt, with Krasin

a. Summary of Krasin

212. Krasin was filed on April 6, 2012, and was published on October 1, 2015, almost two years before the stated priority date of the '104 patent (September 26, 2017). Ex-1007 (Krasin). I am informed that Krasin therefore qualifies as prior art to the '104 patent.

213. At a high level, Krasin teaches a server system that can create in-process sandbox environments to host and control execution of third-party applications and data. Ex-1007 (Krasin) ¶ 0005. Krasin discloses that sandbox environments protect the rest of the server system, including other third party applications, from the executing applications. *Id.* Krasin explicitly teaches that “a

sandbox environment can be created by any suitable computer system capable of hosting applications or other software.” Ex-1007 (Krasin) ¶ 0045.

214. In a particular example, Krasin discloses a system to execute untrusted code modules, such as an application code. Ex-1007 (Krasin) ¶ 0044. Krasin’s system can be used to execute untrusted stand-alone applications in a cluster or server environment, and safely execute email attachments. *Id.* The system 400 is shown in Figure 4A of Krasin, reproduced below.

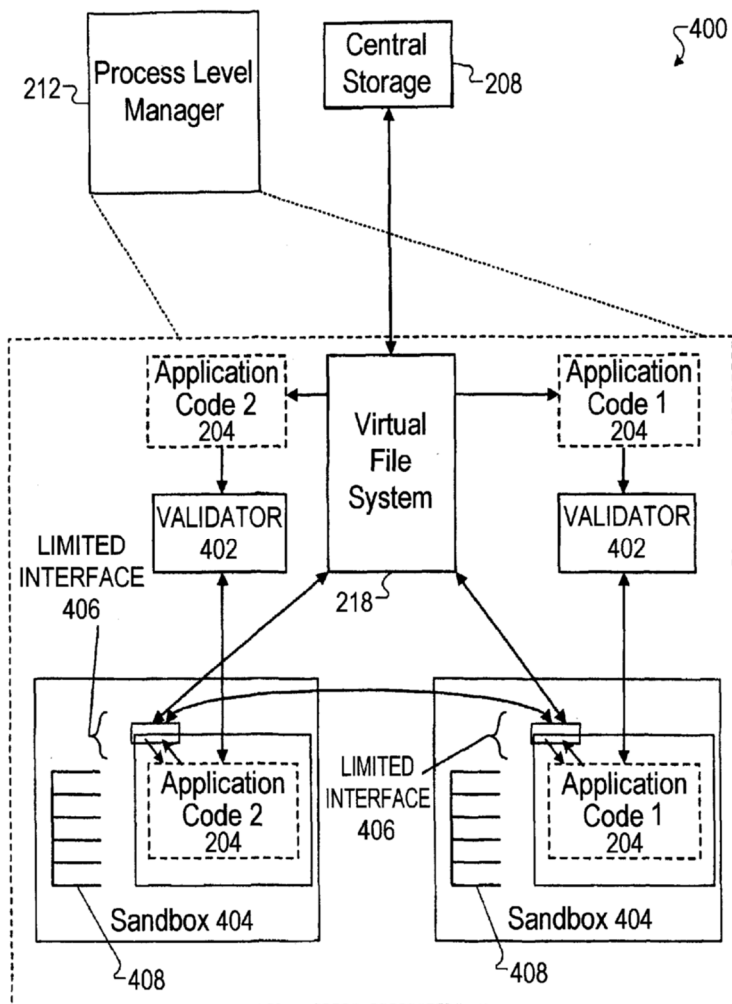


FIG.4A

215. Krasin teaches that, during execution, a sandbox environment provides a limited interface between the application code 204 operating in the sandbox environment, and other software entities and hardware resources.

Ex-1007 (Krasin) ¶ 0047. The interface moderates all external requests made by the application code, as well as the way in which these requests are made. *Id.*

Ultimately, the system ensures that untrusted modules cannot read or write data to which they have not been explicitly granted access: “By loading each module into different sandbox environments, the system can prevent a module in one sandbox environment from accessing data in another sandbox environment.” Ex-1007 (Krasin) ¶ 0058.

216. Krasin also teaches that the sandbox environment can dynamically allocate resources to the application code. Ex-1007 (Krasin) ¶ 0048. For example, a memory buffer can be allocated to the sandbox environment by a process-level manager. *Id.* Before allocating the memory buffer, the process-level manager can determine that there is available system memory on the machine hosting the sandbox environment.

217. In addition to the system shown in Figure 4A, Krasin discloses a system for hosting “workers” in sandbox environments, shown in Figure 4B. Ex-1007 (Krasin) ¶ 0077. In Krasin’s nomenclature, a “worker” is a software process created from application code to complete an assigned work assignment.

Ex-1007 (Krasin) ¶ 0041. The system 450 can be used to ensure that each worker in the system is able to execute separately, with access limited only to appropriate input and output files. Ex-1007 (Krasin) ¶ 0077. The system is shown in Figure 4B of Krasin, reproduced below.

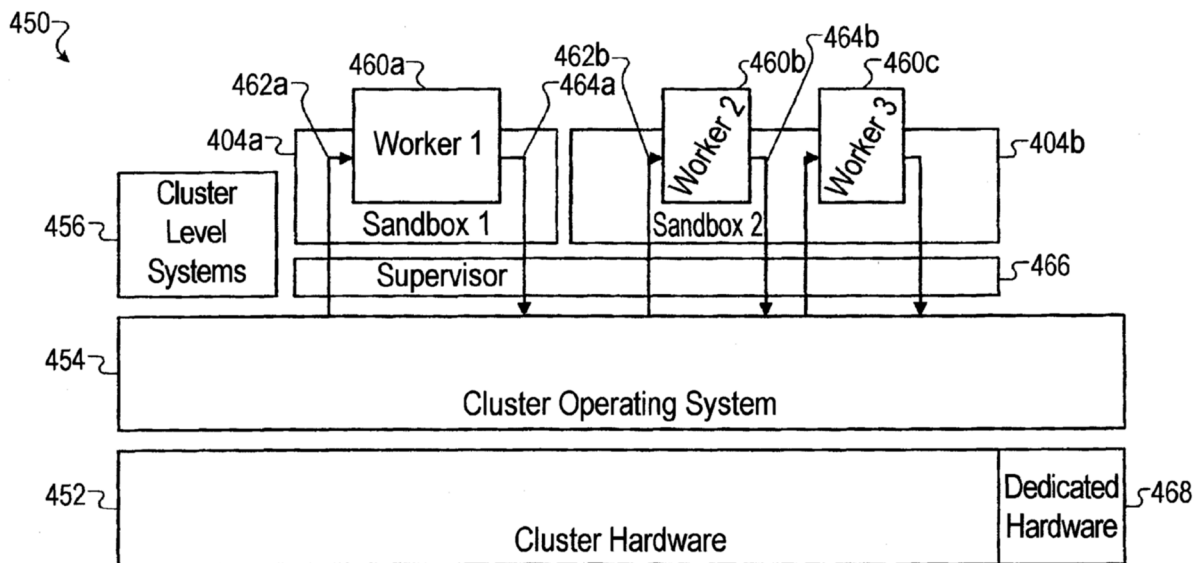


FIG.4B

218. In particular, Krasin teaches a cluster operating system that facilitates communication between cluster-level entities, which include various managers, a cluster cache, and a virtual file system. Ex-1007 (Krasin) ¶ 0078. Communications between the cluster-level workers may be subject to a range of security restrictions, encryption, and logging. *Id.*

219. Krasin also teaches that the workers execute in the sandbox environments and are subject to tight control on available input and output.

Ex-1007 (Krasin) ¶ 0079. The sandbox environments are created by the process-

level manager and provide execution space and limited communications functionality to the workers. *Id.* Krasin teaches that there is an advantage or benefit to implementing the system using sandbox environments:

In some implementations, the sandbox environments 404 are light weight computational structures that require less resource overhead than, for example, virtual machines. Virtual machines, unlike a sandbox environment, emulate physical computer hardware or a full physical machine. Virtual machines often also instantiate a full or partial operating system, which the sandbox environments 404 need not do. Additionally, virtual machines often provide full, or nearly full, system APIs to hosted applications where sandbox environments may provide limited versions APIs that do not support all functions of the API.

Id.; see also *id.* ¶ 0087 (“The use of sandbox environments for sandboxing as opposed to, for example, virtual machines, permits use of worker 460 that have been designed and programmed using imperative algorithms.”).

b. A POSITA Would Have Been Motivated to Combine the Sandbox Environment of Krasin with the Computer System of Hoy/Pratt

220. In my opinion, a POSITA would have been motivated to combine the sandbox environment of Krasin with the computer system of Hoy (adapted to include the authentication process of Pratt as discussed for Ground 1). As a preliminary matter, and as I explained above for element 1d in Ground 1, Hoy discloses a virtual guest system 13 comprising a sandboxed computing

environment that uses a host operating system. Indeed, the Examiner for the '104 patent reached the same conclusion in the final office action dated September 9, 2020. Ex-1002 (Sept. 9, 2020 Office Action at 5).

221. To exclude any doubt, however, I have been asked to opine on the validity of the challenged claims by further analyzing the combination of Hoy and Pratt in view of Krasin. The Krasin reference shows that a POSITA would have been highly familiar with implementing an isolated computing environment as a sandboxed computing environment using a host operating system well before the stated priority date of the '104 patent (September 26, 2017).

222. A POSITA would have been motivated to combine Hoy and Pratt with Krasin. All three references are directed to the field of network security. As indicated in Krasin, untrusted software modules can cause unwanted side effects on a host process or other parts of a system. Ex-1007 (Krasin) ¶ 0057. Untrusted modules can also attempt to make system calls, which can exploit system vulnerabilities by directly creating or modifying files in the file system, starting processes, engaging in clandestine network communications, etc. *Id.* Additionally, untrusted modules can attempt to read or write data to which they have not been explicitly granted access. Ex-1007 (Krasin) ¶ 0058.

223. Krasin recognizes that sandboxed computing environments can protect the parts of the system outside of the sandbox from untrusted applications

executing in the sandbox, providing a motivation for a POSITA to use such sandboxed computing environments in the computing system of Hoy. Ex-1007 (Krasin) ¶ 0005. The sandboxed computing environments taught in Krasin can provide “a secure and light-weight execution environment for untrusted code,” Ex-1007 (Krasin) ¶ 0014, in addition to the following advantages:

- “The sandbox environments can support efficient use of computing resources by dynamically allocating memory buffers to hosted application and transferring running processes to dedicated hardware.” *Id.*
- “In-process sandbox environments can host multiple third-party applications within one process, supporting efficient use of resources and speed of execution due to decreased number of system calls.” *Id.*
- “[U]ntrusted modules cannot read or write data to which they have not been explicitly granted access. By loading each module into different sandbox environments, the system can prevent a module in one sandbox environment from accessing data in another sandbox environment.” Ex-1007 (Krasin) ¶ 0058.

224. Krasin also recognizes that sandboxed computing environments can use a host operating system, instead of an operating system specifically created for the sandbox. Ex-1007 (Krasin) ¶ 0079. While it is my opinion that Hoy discloses the isolated virtual guest system 13 using a host operating system as discussed above for element 1d of Ground 1, Hoy also teaches that the virtual guest system 13 is a “virtual machine,” which Krasin notes “often also instantiate a full or partial operating system” in addition to the host operating system. *Id.* In my opinion, a POSITA would still consider such a virtual machine to be “using a host operating system” because the virtual machine still “operates based on the underlying operating system.” Ex-1004 (Hoy) ¶ 0042.

225. Krasin, however, explicitly teaches that sandbox environments need not instantiate an additional operating system at all. Ex-1007 (Krasin) ¶ 0079 (“Virtual machines often also instantiate a full or partial operating system, which the sandbox environments 404 need not do.”). Krasin also provides teachings that would have motivated a POSITA to use Krasin’s sandbox environments in place of the virtual machines in Hoy. For example, in addition to the advantages discussed above for the Krasin’s sandbox environments, Krasin also discloses that “the sandbox environments 404 are light weight computational structures that require less resource overhead than, for example, virtual machines.” *Id.* In my opinion, the stated ability for Krasin’s sandbox environments to lessen the resource

overhead would have motivated a POSITA to use these sandbox environments in place of Hoy's virtual machines. This combination would simply involve a POSITA combining prior-art elements according to known methods to yield a predictable result—Krasin's sandbox environment provides the isolated computing environment on Hoy's workstation instead of the virtual guest system in order to enable operation of untrusted applications or processes on the workstation without compromising the security of the overall system.

226. Krasin further provides additional teachings that would have motivated a POSITA to use the sandbox environments in place of the virtual machines in Hoy. Ex-1007 (Krasin) ¶ 0067 (“Note that the system can use a combination of compiler-based techniques and static binary analysis (e.g., analysis of assembly code during validation) to achieve safety with lower execution overhead than dynamically analyzing and rewriting executable code at runtime (as is commonly done in some conventional virtual machine environments).”), ¶ 0087 (“The use of sandbox environments for sandboxing as opposed to, for example, virtual machines, permits use of worker 460 that have been designed and programmed using imperative algorithms.”).

**c. A POSITA Would Have Had a Reasonable
Expectation of Success in Combining the
Sandbox Environment of Krasin with the
Computer System of Hoy/Pratt**

227. In my opinion, a POSITA would have had a reasonable expectation of success in combining the sandbox environment of Krasin with the computer system of Hoy (adapted to include the authentication process of Pratt as discussed for Ground 1), given the similarities in the teachings and systems of the three references, and given that technologies for implementing sandboxed computing environments using a host operating system were well known at the time. A POSITA would have expected to be successful in making the combination because the combination involves using conventional and well-known computer programming methods to yield a predictable result. Implementing a sandboxed computing environment using a host operating system would not have required specialized skills beyond what a POSITA would have had.

228. Furthermore, a POSITA would have been motivated and would have had a reasonable expectation of success in combining Hoy, Pratt, and Krasin, as explained above, because the combination would not have changed the principle of operation for any of the teachings of the references. In fact, Hoy already teaches a sandboxed computing environment that uses a host operating system, as discussed for element 1d and as the Examiner for the '104 patent concluded in the final office action dated September 9, 2020. Both Hoy's virtual guest system and Krasin's

sandbox environments operate through isolation from other computing environments. Rather than changing Hoy's principle of operation, the combination with Krasin would actually reinforce Hoy's principle of operation by using a known technique (Krasin's sandbox environments using a host operating system) to achieve what Hoy already teaches (a sandboxed computing environment using a host OS).

3. Claim Analysis

229. For this Ground 2, the element-by-element analysis of all of the challenged claims is the same as the element-by-element analysis for Ground 1, except that Krasin provides additional disclosure for the limitation in elements 1d, 10c and 18d of an "isolated computing environment using a host operating system and comprising a sandboxed computing environment."

230. As discussed above, Krasin discloses sandboxed computing environments throughout its specification and drawings, including the sandbox environments functionally illustrated in Figures 4A and 4B. Krasin teaches that the workers executing in the sandbox environments are subject to tight control on available input and output, thus creating an isolated computing environment. Ex-1007 (Krasin) ¶ 0079. Krasin also explicitly discloses that its sandbox environments need not instantiate an additional operating system and thus that they use the host operating system. *Id.* ("Virtual machines often also instantiate a full

computing environment that uses “a second memory space to enable operation of a second set of one or more applications or processes.”

233. For the foregoing reasons, it is my opinion that a POSITA would have found claims 1-3, 8, 10, 11 and 17-19 of the '104 patent to be obvious based on the teachings of Hoy in view of Pratt, and further in view of Krasin.

C. Ground 3: Claims 1-3, 8, 10, 11, and 17-19 Are Anticipated by Benoit

234. It is my opinion that a POSITA would have found claims 1-3, 8, 10, 11 and 17-19 to be anticipated by the teachings of Benoit.

1. Summary of Benoit

235. Benoit was filed on September 12, 2014, and was published on April 2, 2015, more than two years before the stated priority date of the '104 patent (September 26, 2017). Ex-1005 (Benoit). I am informed that Benoit therefore qualifies as prior art to the '104 patent.

236. Benoit teaches a computer device as shown in Figure 2, reproduced below.

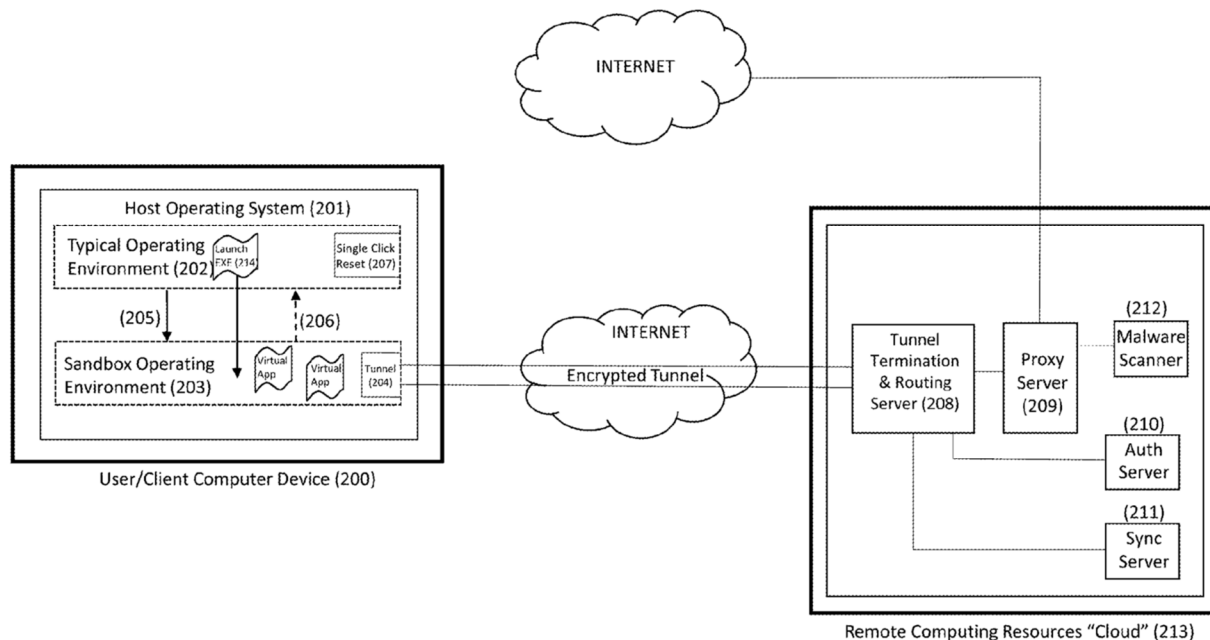


FIG. 2

237. As shown in Figure 2 and as described in the specification of Benoit, the computer device comprises a host operating system (OS), a typical operating environment (OE), a sandbox operating environment (OE), one or more connections between the typical OE and the sandbox OE (connections labeled 205 and 206), and an interaction element (a single-click reset). Ex-1005 (Benoit) ¶ 0041.

238. Connected to the computer device through an encrypted tunnel are remote computing resources (RCR), which Benoit refers to as a “cloud component.” Ex-1005 (Benoit) ¶ 0041. The cloud component comprises a tunnel termination and routing server, a proxy server, an authentication server, a synchronization server, and a malware scanner. *Id.*

239. In operation, an executable file builds the sandbox OE. Ex-1005 (Benoit) ¶ 0042. The system then creates the necessary file associations, configures pre-defined read and write permissions to the host system, and launches a secured tunnel to the cloud. Ex-1005 (Benoit) ¶¶ 0042-0043. The secured tunnel creates an encrypted connection from the system components on the host computing device to the cloud. Ex-1005 (Benoit) ¶ 0044. After the secured tunnel is launched and established, the system can launch a virtualized application inside the sandbox system. Ex-1005 (Benoit) ¶ 0045.

240. After a successful launch of the virtual application, the user is prompted to authenticate in order to access the cloud. Ex-1005 (Benoit) ¶ 0046. Upon a successful authentication, a connection is established, via the secured tunnel, with the proxy server in the cloud. Ex-1005 (Benoit) ¶ 0047. The user can then initiate an Internet-bound data request, which is sent across the secured tunnel. *Id.*

241. Benoit provides additional teachings pertaining to, among other things, a process for removing malware from the computer device. Ex-1005 (Benoit) ¶¶ 0050-0056.

2. Independent Claim 1

242. It is my opinion that Benoit anticipates the limitations of claim 1.

[1pre] A host computer system:

243. Benoit discloses a **host computer system** in the form of the user/client computer device, and therefore teaches the preamble. The user/client computer device is functionally illustrated in Figure 2 of Benoit, reproduced below with the user/client computer device 200 highlighted. Ex-1005 (Benoit), Fig. 2. In the specification, Benoit specifically refers to this user/client computer device as a “**host computing device.**” Ex-1005 (Benoit) ¶¶ 0030, 0031, 0044, 0046, 0048-0051, 0053, 0054.

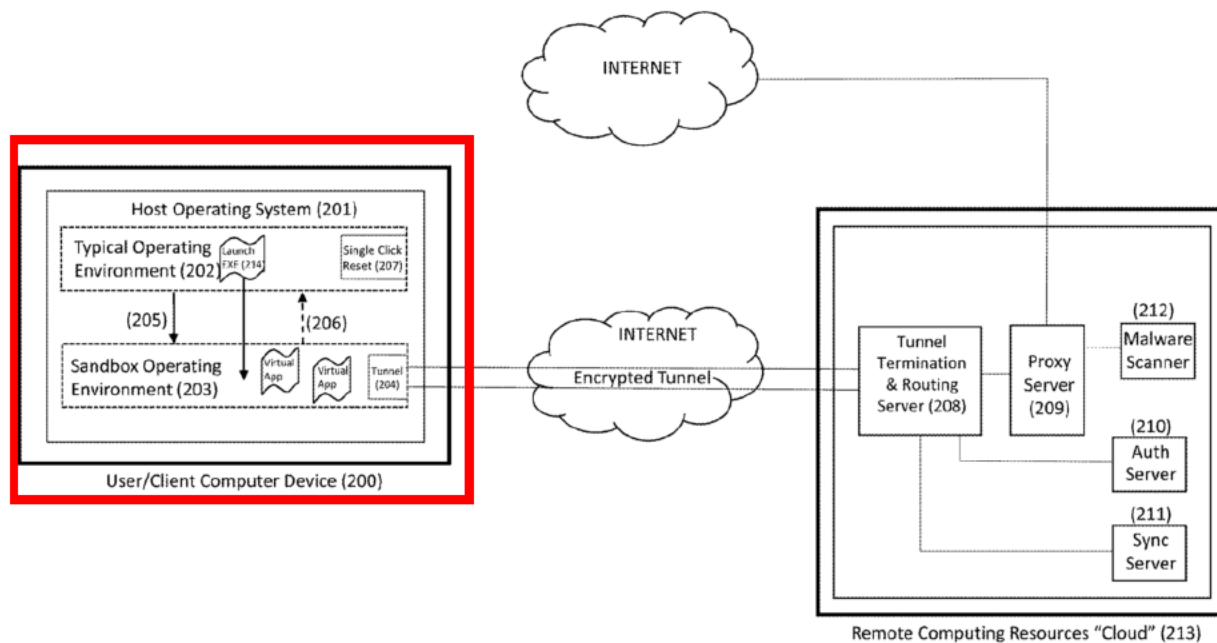


FIG. 2

244. Accordingly, to the extent that the preamble is limiting, Benoit teaches it.

[1a] a memory; and

245. Benoit discloses **a memory** for the host computer system and therefore teaches element 1a. In particular, Benoit teaches that “a computing device 100 appropriate for use with embodiments of the present application may generally be comprised of one or more of ... **Random Access Memory (RAM)** 102” Ex-1005 (Benoit) ¶ 0024; *see also id.* ¶¶ 0007 (“computer-executable code stored in non-volatile memory”), 0061 (“A programmable apparatus includes one or more ... memory devices”), 0065 (“a random access memory (RAM), a read-only memory (ROM),”), 0066 (“Computer program instructions can be stored in a computer-readable memory”). Figure 1 of Benoit is reproduced below with the RAM 102 highlighted. Ex-1005 (Benoit), Fig. 1.

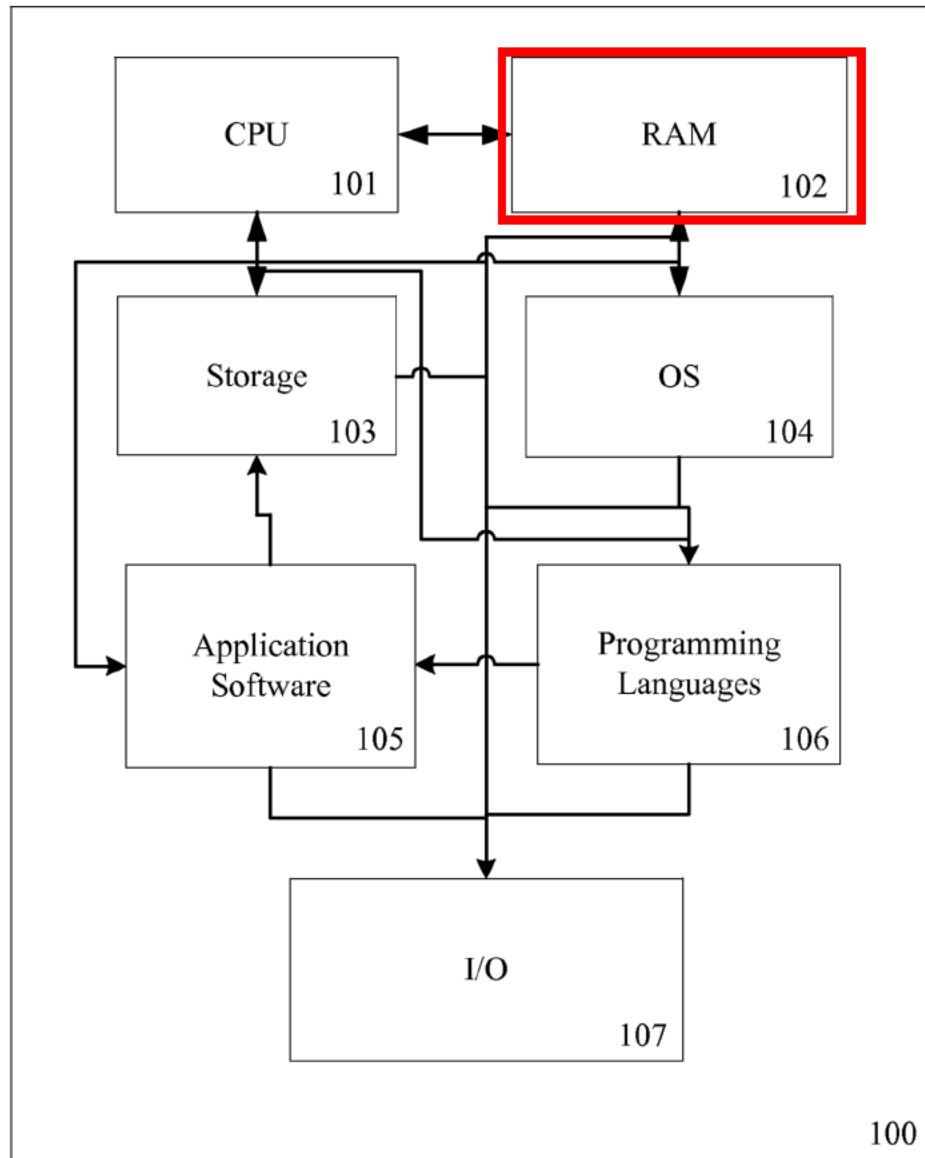


FIG. 1

246. Accordingly, Benoit teaches element 1a.

[1b] a processor configured to:

247. Benoit discloses **a processor** for the host computer system and therefore teaches element 1b. In particular, Benoit teaches that “a computing device 100 appropriate for use with embodiments of the present application may

generally be comprised of one or more of a **Central Processing Unit (CPU)**

101” Ex-1005 (Benoit) ¶ 0024; *see also id.* ¶¶ 0007-0009 (referring to a

“processor”), 0061 (“A programmable apparatus includes one or more

microprocessors”), 0065 (“a random access memory (RAM), a read-only

memory (ROM),”), 0072 (“The multiple programs or threads may be processed

more or less simultaneously to enhance utilization of the processor”). Figure 1

of Benoit is reproduced below with the CPU 101 highlighted. Ex-1005 (Benoit),

Fig. 1.

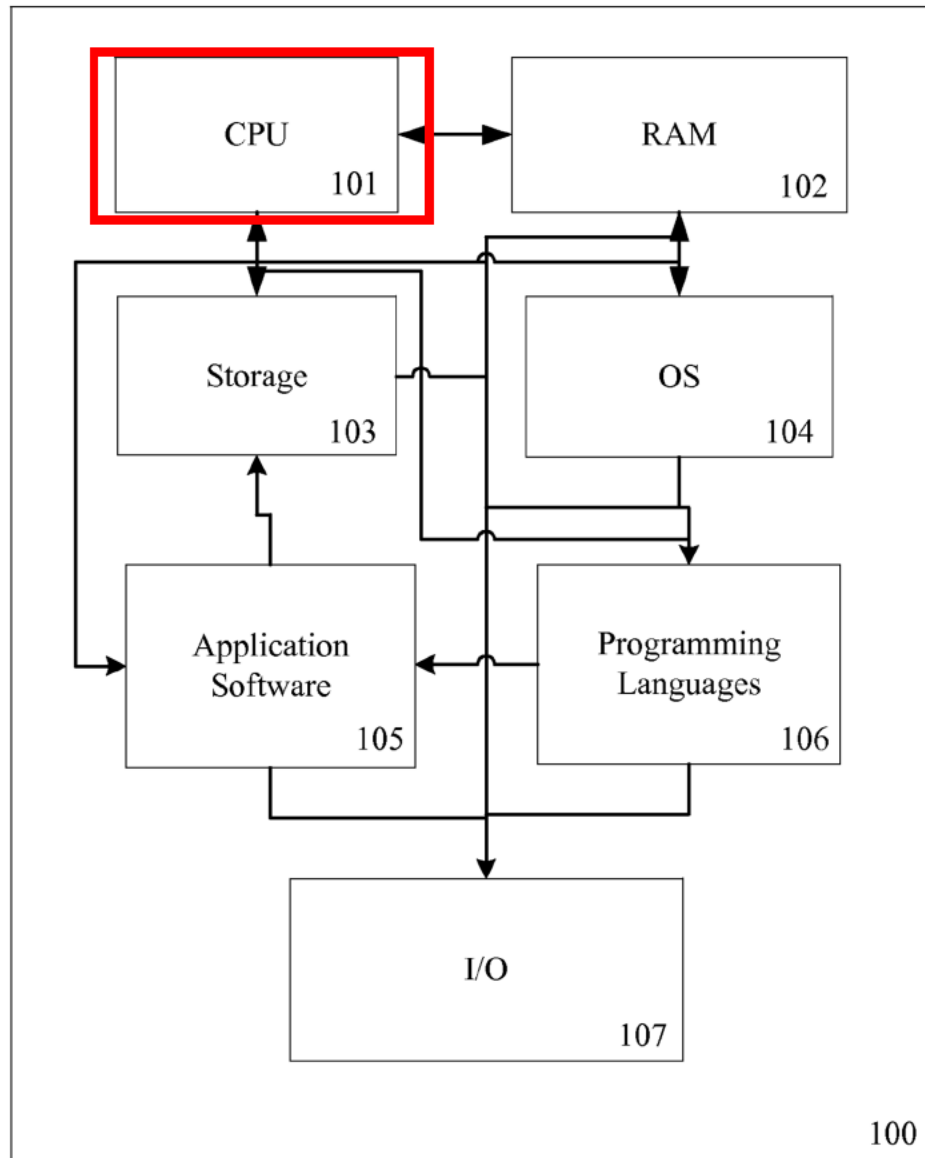


FIG. 1

248. Accordingly, Benoit teaches element 1b.

[1c] implement a workspace, wherein the workspace is configured to enable operation of a first set of one or more applications or processes via a first memory space;

249. Benoit teaches element 1c.

250. As discussed above for element 1c of Ground 1, the '104 patent does not explicitly define a “**workspace**,” which has many meanings in the network security field. As used in the '104 patent, however, a workspace broadly appears to be a computer environment in which one or more applications may run. *See, e.g.,* Ex-1001, 1:60-63.

251. Benoit discloses the implementation of a **workspace** in the form of a typical operating environment (OE) on the user/client computer device. Ex-1005 (Benoit) ¶ 0041. Figure 2 of Benoit is reproduced below with the typical OE 202 highlighted. Ex-1005 (Benoit), Fig. 2.

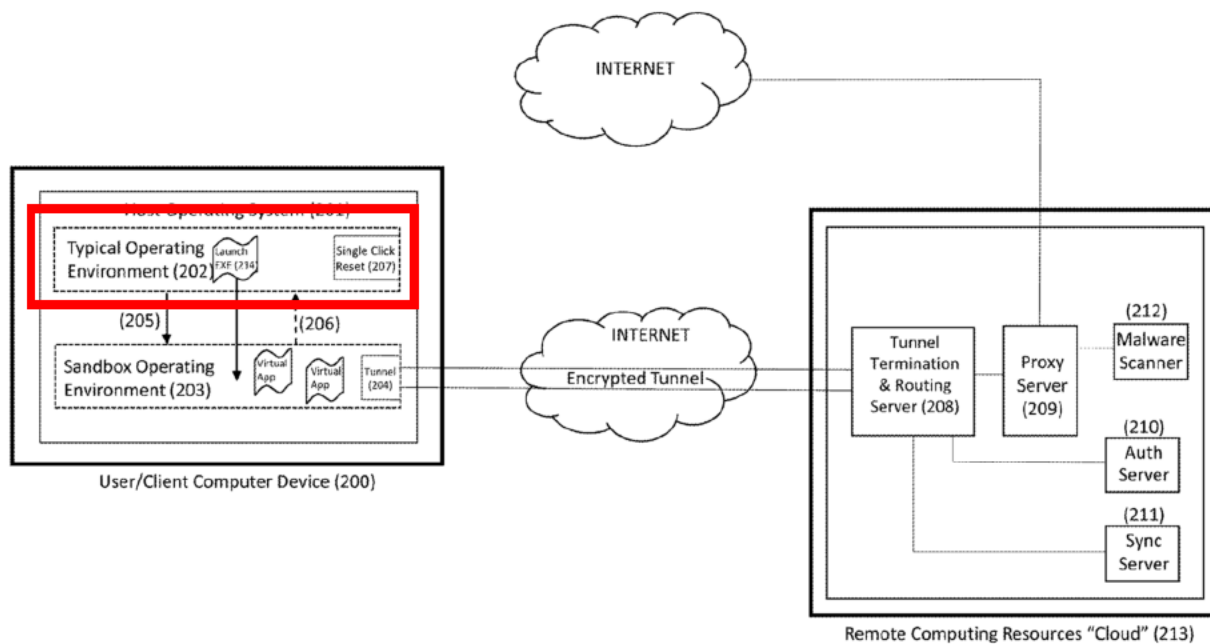


FIG. 2

252. As shown in Figure 2, the typical OE runs **applications** such as a single-click reset process 207 and a launch executable program 214. Ex-1005 (Benoit), Fig. 2; *see also id.* ¶ 0041. Accordingly a POSITA would have

understood that the typical OE of Benoit is a **workspace** as that term is used in the '104 patent and that this workspace is **configured to enable operation of a first set of one or more applications or processes**, such as the single-click reset process and the launch executable program.

253. Benoit further discloses that the typical OE (the workspace) is configured to enable operation of the applications or processes **via a first memory space**. A POSITA would have understood that the typical OE 202 would necessarily have had its own memory space. To a POSITA, a sandbox having “pre-defined read (40) and write (50) permissions to the host system,” as taught in Benoit and shown in Figure 3A (reproduced at right), is a sandbox having its own memory space separate from the memory spaces for other operating environments on the user/client computer device, including the typical OE. A POSITA would have known this because each process on a computer usually has its own memory space, and even though a set of related applications or processes running in one operating environment on the computer might share memory space, their memory space would be different and isolated from the memory space of another set of applications and processes running in a different operating environment on the same computer. Therefore, a POSITA would have understood that the memory space of Benoit’s typical OE is different and isolated from the memory space of Benoit’s sandbox OE. Benoit also teaches that “the sandbox is designed to ‘deny

all' software code from accessing host resources outside of the sandbox," further indicating that the sandbox and the typical OE have separate resources, including separate memory spaces. Ex-1005 (Benoit) ¶ 0032.

254. Accordingly, Benoit teaches element 1c.

[1d] implement an isolated computing environment, the isolated computing environment using a host operating system and comprising a sandboxed computing environment that uses a second memory space to enable operation of a second set of one or more applications or processes,

255. Benoit teaches element 1d.

“implement an isolated computing environment”

256. Benoit discloses the implementation of an isolated computing environment in the form of the **sandbox operating environment** (OE) on the user/client computer device. Ex-1005 (Benoit) ¶ 0041. A POSITA would have understood that Benoit's reference to a sandbox operating environment was a reference to an isolated operating environment. Figure 2 of Benoit is reproduced below with the isolated sandbox OE 203 highlighted. Ex-1005 (Benoit), Fig. 2.

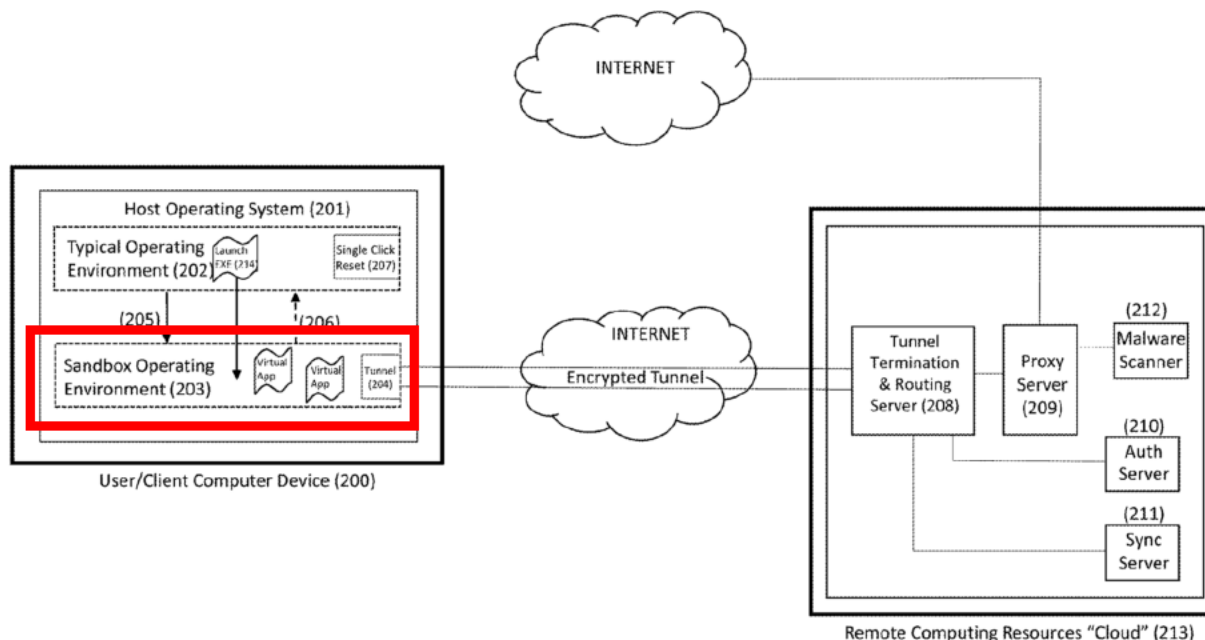


FIG. 2

257. Benoit teaches that the sandbox OE is separated from the typical operating environment (OE) and that interaction between the two operating environments is restricted to certain “connections between the typical OE 202 and the sandbox OE 203 (connections labeled 205, 206).” Ex-1005 (Benoit) ¶ 0041 & Fig. 2. Moreover, after the sandbox is built, the system “configures pre-defined read (40) and write (50) permissions to the host system,” which would have further confirmed to a POSITA that the sandbox OE is an isolated computing environment.

258. As further evidence that the sandbox OE is an isolated computing environment, Benoit teaches that “a sandbox is utilized and configured to reduce the ability for computer code to access the host computing device to the maximum amount possible, while still permitting full functionality and an uninhibited end-

user experience.” Ex-1005 (Benoit) ¶ 0031. The sandbox OE does this because “the sandbox is designed to ‘deny all’ software code from accessing host resources outside of the sandbox.” Ex-1005 (Benoit) ¶ 0032; *see also id.* (“[N]othing occurs outside of the resources belonging to the sandbox (e.g., virtual host resources)”).

“the isolated computing environment using a host operating system”

259. Benoit also discloses that the sandbox operating environment (OE) (an isolated computing environment) uses a host operating system. Benoit describes the user/client computer device as comprising “a host operating system (OS) 201, a typical operating environment (OE) 202, [and] a sandbox operating environment (OE) 203.” Ex-1005 (Benoit) ¶ 0041. Only one operating system is disclosed in Benoit, and that operating system is the host OS 201. In Figure 2 of Benoit (reproduced below with the host OS 201 highlighted), both the typical OE 202 and the sandbox OE 203 are shown as being within the host OS 201. Because Benoit teaches only one operating system (the host OS) and because this host OS is shown as including both the typical OE and the sandbox OE, a POSITA at the time that Benoit was filed (September 12, 2014) would have understood that the sandbox OE was using the host OS.

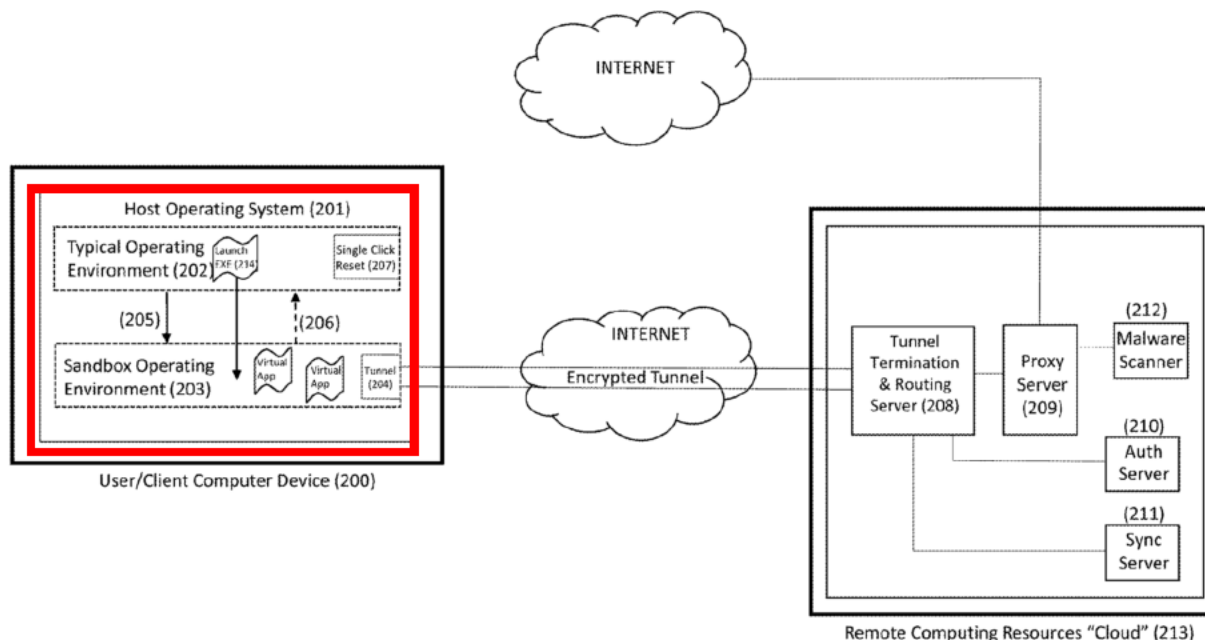


FIG. 2

“and comprising a sandboxed computing environment”

260. As discussed above, Benoit discloses a **sandbox operating environment** (OE) on the user/client computer device. Ex-1005 (Benoit) ¶ 0041 & Fig. 2.

261. Moreover, under the proper construction of a “sandboxed computing environment” as discussed in the claim-construction section above, a POSITA would have understood that Benoit’s sandbox OE is a computing environment that does not require its own separate operating system. As discussed in the immediately preceding section, Benoit discloses that the sandbox operating environment (OE) uses a host operating system. Benoit’s Figure 2 illustrates both the typical OE and the sandbox OE as being within the host OS.

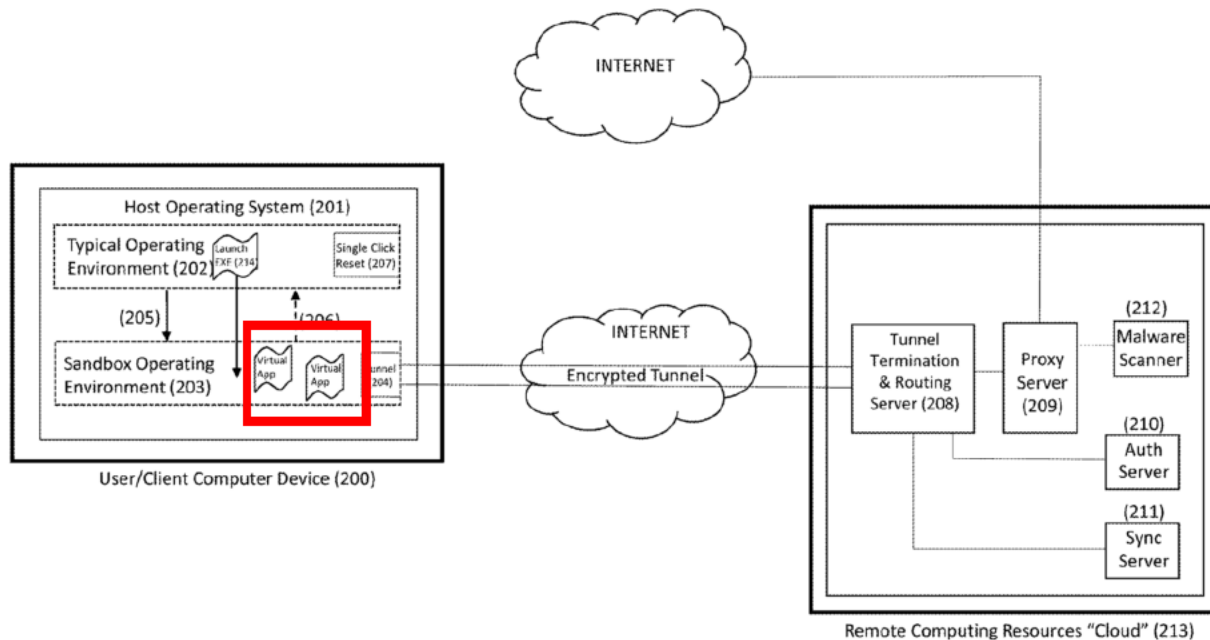
262. Benoit's sandbox OE also meets the definition of a sandboxed computing environment under the alternative construction that I discuss in the claim-construction section above ("an area in which software is confined and prevented from accessing system resources"). Indeed, that is what is illustrated in Figure 2 of Benoit. Ex-1005 (Benoit), Fig. 2. Benoit teaches that the sandbox OE is separated from the typical operating environment (OE) and that interaction between the two operating environments is restricted to certain "connections between the typical OE 202 and the sandbox OE 203 (connections labeled 205, 206)." Ex-1005 (Benoit) ¶ 0041 & Fig. 2. Moreover, after the sandbox is built, the system "configures pre-defined read (40) and write (50) permissions to the host system," which would have further confirmed to a POSITA that the sandbox OE provides an area in which software is confined and restricted in terms of the system resources it can access. Ex-1005 (Benoit) ¶ 0043; *see also id.* ¶ 0032 ("[T]he sandbox is designed to 'deny all' software code from accessing host resources outside of the sandbox.").

"that uses a second memory space to enable operation of a second set of one or more applications or processes"

263. Benoit discloses that the sandbox operating environment (OE) uses a second memory space to enable operation of a second set of one or more applications or processes.

264. As discussed above, a POSITA would have understood that the sandbox OE would necessarily have had its own **memory space**. To a POSITA, a sandbox having “pre-defined read (40) and write (50) permissions to the host system,” as taught in Benoit, is a sandbox having its own memory space separate from the memory spaces for other operating environments on the user/client computer device, including the typical OE. Benoit also teaches that “the sandbox is designed to ‘deny all’ software code from accessing host resources outside of the sandbox,” further indicating that the sandbox and the typical OE have separate resources, including separate memory spaces. Ex-1005 (Benoit) ¶ 0032; *see also id.* (“[N]othing occurs outside of the resources belonging to the sandbox (e.g., virtual host resources)”).

265. Benoit also teaches the **operation of a second set of one or more applications or processes** through “launching one or more **virtual applications** from the sandbox on the host system,” which is functionally illustrated in Figure 2 of Benoit (reproduced below with the virtual applications highlighted). Ex-1005 (Benoit) ¶ 0014 & Fig. 2; *see also id.* ¶¶ 0007 (same), 0008 (“allow execution of executable code associated with a second virtual application within the sandbox”).



266. Accordingly, Benoit teaches element 1d.

[1e] wherein the isolated computing environment is configured to access an Internet-based cloud service via at least one application of the second set of one or more applications or processes;

267. Benoit discloses that the sandbox operating environment (OE) (an **isolated computing environment**) is **configured to access an Internet-based cloud service via at least one application** operating in the sandbox OE (**the second set of one or more applications or processes**), and therefore teaches element 1e. The Internet-based cloud service is illustrated in Figure 2 of Benoit as a set of remote computing resources or “**cloud**” 213 connected to the virtual applications in the sandbox operating environment (OE) 203 via an encrypted tunnel through the Internet. Ex-1005 (Benoit), Fig. 2.

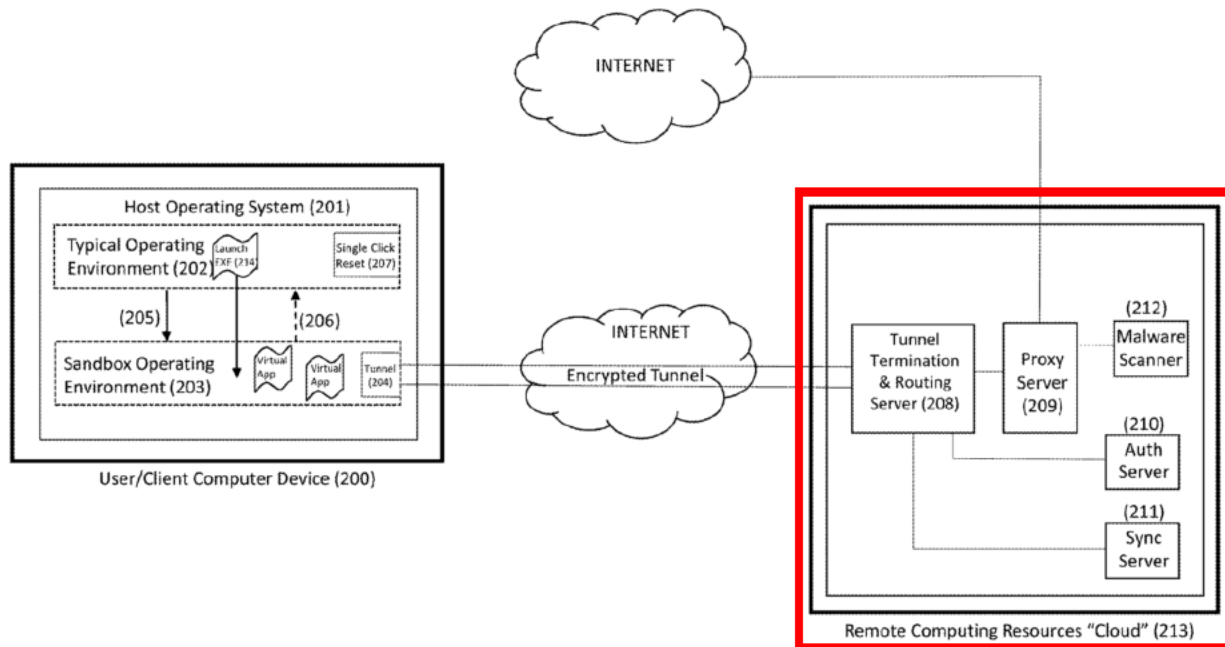


FIG. 2

268. Benoit refers to the remote computing resources using the acronym “RCR.” Ex-1005 (Benoit) ¶ 0032. Benoit discloses that “the RCR (i.e., cloud component) 213, compris[es] one or more of a tunnel termination & routing server 208, a proxy server 209, an authentication server 210, a synchronization server 211 and a malware scanner 212.” Ex-1005 (Benoit) ¶ 0041. A POSITA would have understood each of these servers and the malware scanner as Internet-based cloud services at the time that Benoit was filed.

269. Benoit teaches that the tunnel termination and routing server can launch a secured tunnel from the isolated sandbox OE through the Internet to the cloud and thus “create an encrypted connection from the system components on the host computing device to the components of the system located on the RCR.” Ex-1005 (Benoit) ¶ 0044 & Fig. 2. Benoit teaches that this encrypted cloud connection is established “[a]fter successful launch of the one or more virtual applications” and is at least initially with “the proxy server (110) in the RCR component of the system.” Ex-1005 (Benoit) ¶¶ 0044, 0047. The “proxy server (110)” refers to step 110 in Figure 3B, which depicts the user authentication and data flow processes and makes clear in step 120 that the “user initiates Internet bound traffic via virtual app(s) in sandbox.” Ex-1005 (Benoit), Fig. 3B.

Figure 3B is reproduced at right with steps 110 and 120 highlighted, showing that the user at the user/client computer device can establish an encrypted Internet connection with the cloud through the tunnel termination and routing server, and then initiate

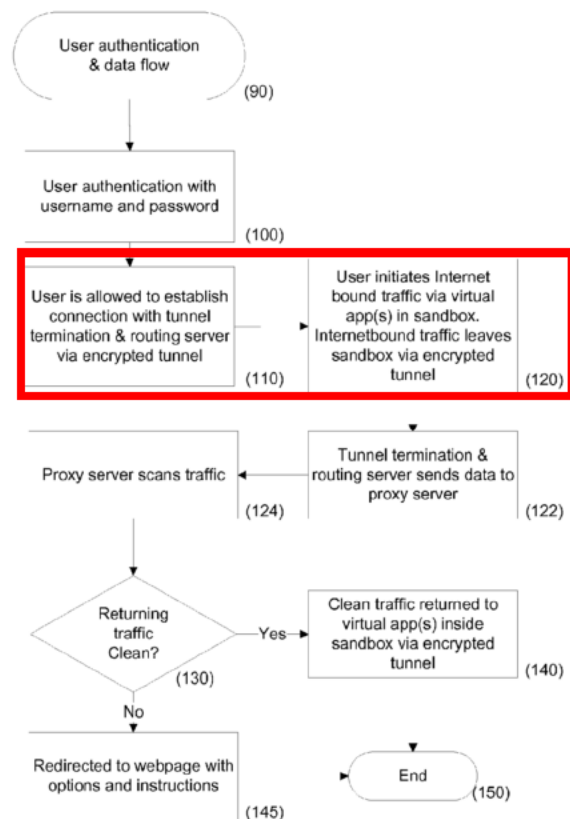


FIG. 3B

Internet-bound traffic via the virtual applications in the isolated sandbox OE. *Id.*

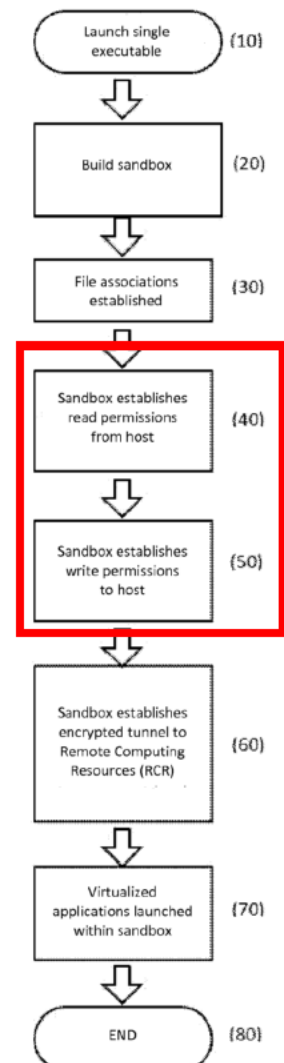
270. Accordingly, Benoit teaches element 1e.

[1f] isolate the isolated computing environment from the workspace using an internal isolation firewall;

271. Benoit teaches element 1f.

272. As discussed above, Benoit discloses a sandboxed computing environment that is **isolated** from another computing environment (a **workspace**) on the same computer device.

273. A POSITA would have understood that Benoit's host operating system implements an internal isolation firewall between the typical operating environment and the sandbox operating environment. To a POSITA, a sandbox having "pre-defined read (40) and write (50) permissions to the host system," as taught in Benoit and shown in Figure 3A (reproduced at right), is a sandbox having its own memory space and isolated from the host system **using an internal isolation firewall** (the pre-defined read and write permissions). That is, a POSITA would have known that to enforce the permissions policy on a sandbox, the host operating system needs a module to monitor sandbox



activities such as access requests and to determine whether to permit or restrict such activities, and that this module would be an internal isolation firewall.

Ex-1005 (Benoit) ¶ 0043; *see also id.* ¶ 0032 (“[T]he sandbox is designed to ‘deny all’ software code from accessing host resources outside of the sandbox.”).

274. Benoit thus teaches **isolating the isolated computing environment** (Benoit’s sandbox OE) **from the workspace** (Benoit’s typical OE) **using an internal isolation firewall** (the pre-defined read and write permissions).

Accordingly, Benoit teaches element 1f.

[1g] authenticate the isolated computing environment with an authentication device; and

275. Benoit discloses **authenticating** the sandbox operating environment (OE) (an **isolated computing environment**) with an authentication server (an **authentication device**), and thus teaches element 1g. In particular, Benoit describes “request[ing] **authentication** from the user to provide access to the remote system via the encrypted connection, which request may also be automated, requiring no user input.” Ex-1005 (Benoit) ¶ 0007. The authentication is performed with an **authentication server**, which is functionally illustrated in Figure 2 of Benoit (reproduced below with the authentication server 210 highlighted). Ex-1005 (Benoit) ¶ 0041 & Fig. 2.

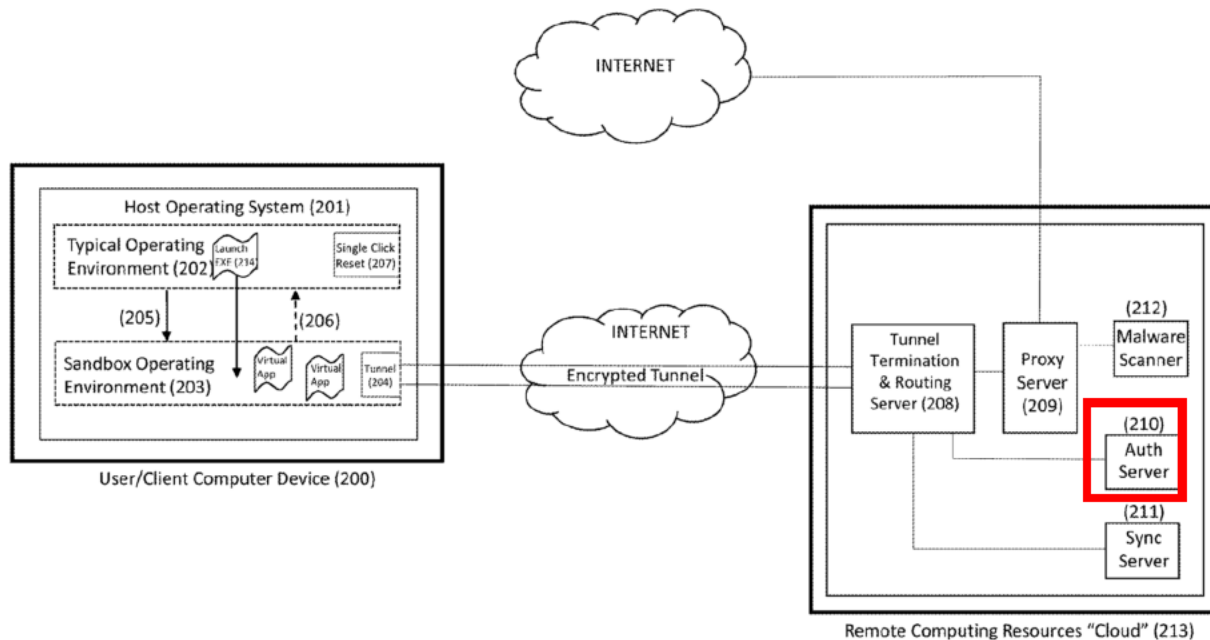


FIG. 2

276. Benoit teaches that, after a successful launch of the one or more virtual applications, shown in Figure 2 within the sandbox OE:

the user will be prompted to authenticate (100) in order to access the RCR components of the system. In some instances, the user may authenticate when they first launch the executable (10), FIG. 3A, and in other instances, the authentication may be automated, requiring no user input.

Upon successful authentication, the user is allowed to establish a connection, via the secured tunnel, with the proxy server (110) in the RCR component of the system.

Ex-1005 (Benoit) ¶¶ 0046-0047. The references to “100” and “110” are to Figure 3B of Benoit, which describes an authentication process before virtual applications in the sandbox OE can send Internet-bound traffic via the encrypted tunnel

between the sandbox and the termination and routing server.

Ex-1005 (Benoit), Fig. 3B. Figure 3B is reproduced at right with steps 100 and 110 highlighted.

277. A POSITA would have understood that an automated authentication request without user input would have been a request to authenticate the sandbox OE. Ex-1005 (Benoit) ¶¶ 0007, 0046. In particular,

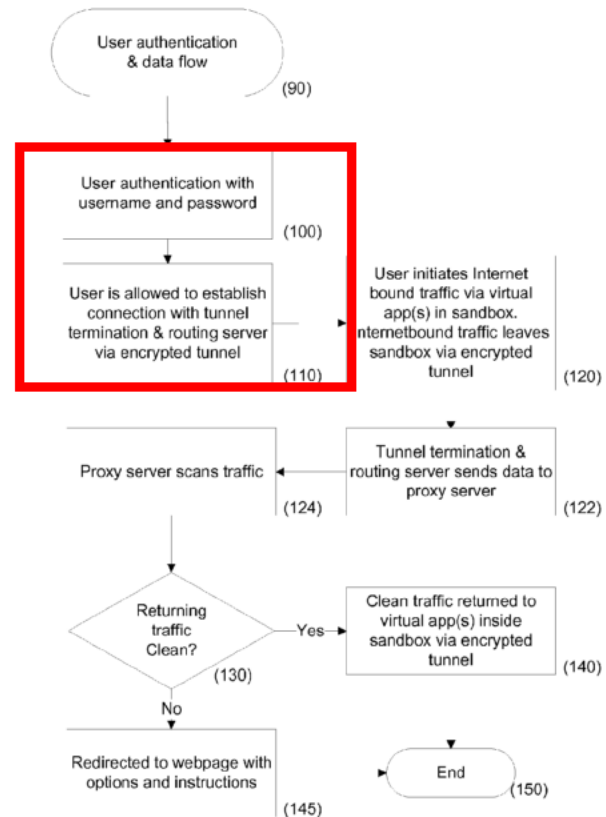


FIG. 3B

Benoit teaches that the disclosed system may “leverage[] a well-established encryption solution known as Secure Shell (SSH)” that “supports a feature known as ‘tunneling’.” Ex-1005 (Benoit) ¶ 0036. Through the SSH protocol, “sandboxed applications can be configured to automatically use the SSH tunnel or other encrypted tunnel/connection.” Ex-1005 (Benoit) ¶ 0037. This tunnel is shown in Figure 2 of Benoit (reproduced below with the tunnel termination 204 highlighted) as terminating inside the sandbox OE 203. A POSITA would have understood that the sandbox OE was being authenticated, since the sandbox OE itself contains the

tunnel termination and it is the sandboxed applications that are automatically configured to use it.

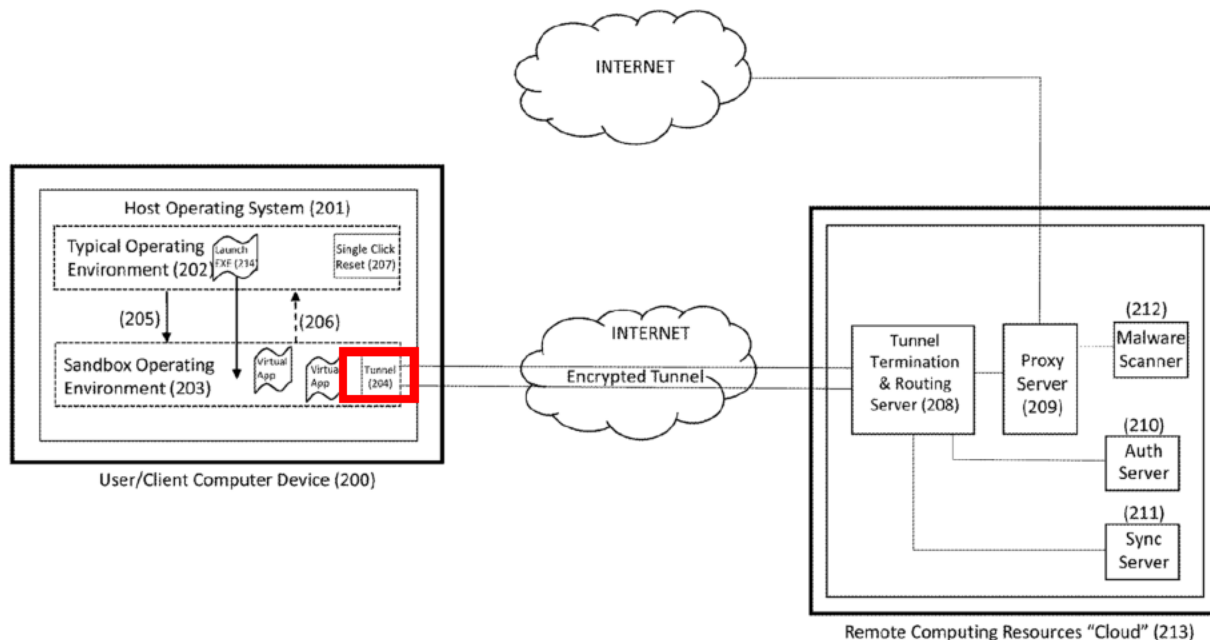


FIG. 2

278. Accordingly, Benoit teaches element 1g.

[1h] communicate with a proxy server to access the Internet-based cloud service to allow communication between the at least one application and the Internet-based cloud service when the isolated computing environment has been authenticated.

279. Benoit discloses communicating with the proxy server to access, via the Internet, the remote computing resources or “cloud” (an Internet-based cloud service) to allow communication between at least one virtual application operating on the sandbox operating environment (OE) (an isolated computing environment) and the cloud when the sandbox OE has been authenticated, and thus teaches

element 1h. Figure 2 of Benoit is reproduced below with the **proxy server 209** highlighted. Ex-1005 (Benoit), Fig. 2.

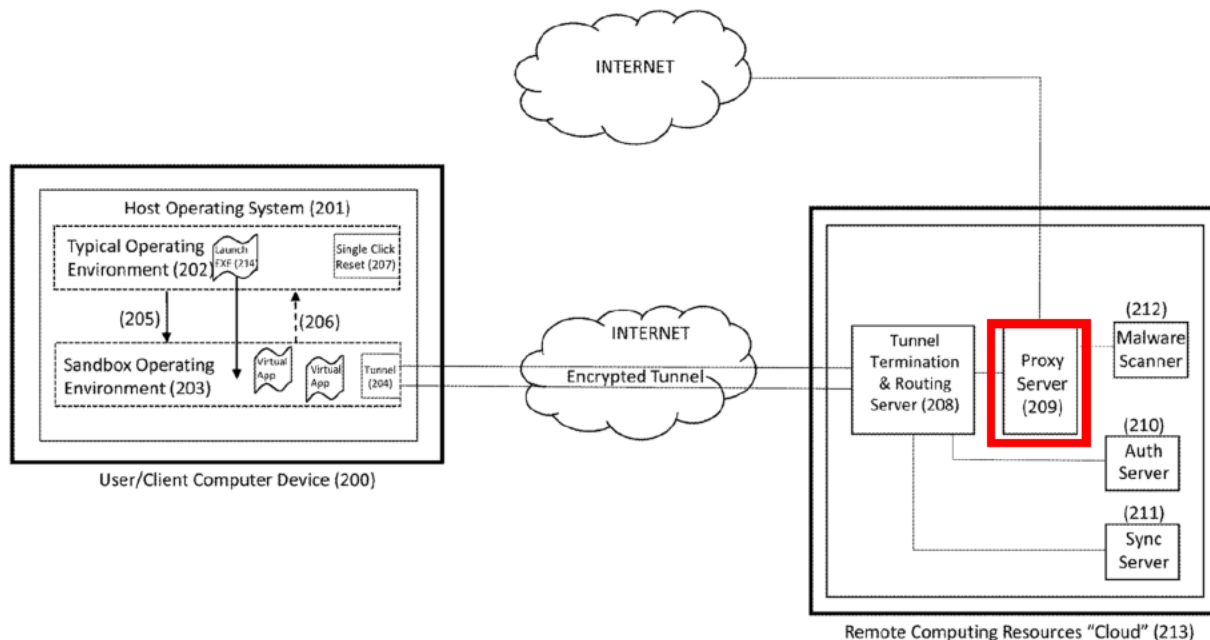


FIG. 2

280. As shown in Figure 2 of Benoit, the sandbox OE 203 **communicates with the proxy server 209 to access services in the cloud 213 via the Internet**, as well as to access other cloud services outside of the particular cloud 213 via a connection from the proxy server 209 to a second cloud labeled “Internet.” Ex-1005 (Benoit), Fig. 2. As a result, Benoit teaches that communication is allowed between the virtual applications operating in the sandbox OE and the cloud or other cloud services. *Id.*; *see also id.* ¶ 0048 (“Traffic returning from the Internet is scanned by a multi-engine signature- and heuristic-based antimalware system (130). This process occurs at the RCR components of the system, as to prevent any malicious code or content from reaching the host computing device of

the user. If the traffic returning from the Internet is free of malicious code or content ('clean'), then the "clean" traffic is returned to the requesting virtual application running on the system components of the user's host computing device (140).").

281. Benoit teaches that the communication between the virtual applications operating in the sandbox OE and the cloud or other cloud services is allowed **when the sandbox OE has been authenticated** as discussed above for element 1g. In particular, Benoit discloses that "[a]fter successful launch of the one or more virtual applications within system components on the host computing device, the user will be prompted to authenticate (100) in order to access the RCR components of the system" and that "authentication may be automated, requiring no user input." Ex-1005 (Benoit) ¶ 0046. Benoit then explains:

[u]pon successful authentication, the user is allowed to establish a connection, via the secured tunnel, **with the proxy server** (110) in the RCR component of the system. The user is then able to initiate an Internet bound data

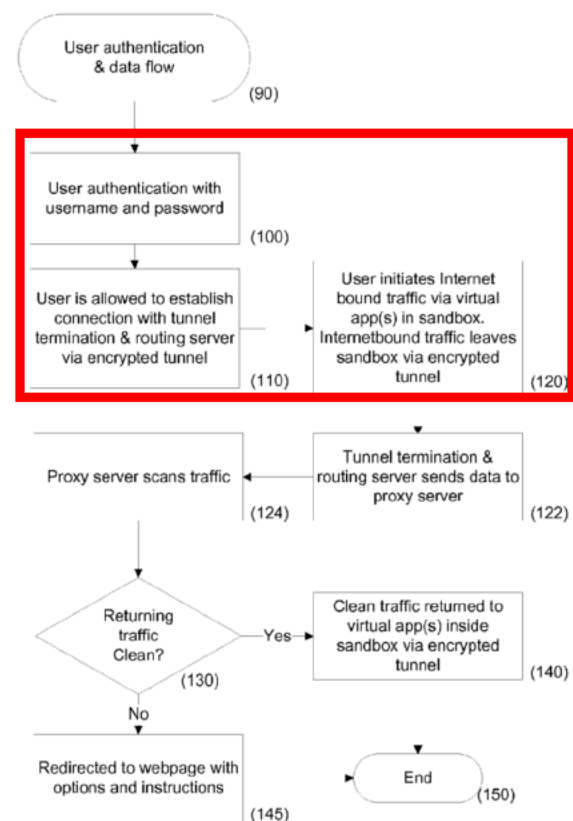


FIG. 3B

request (120), and the data request is sent across the secured (i.e., encrypted) tunnel (120).

Ex-1005 (Benoit) ¶ 0047.

282. Steps 100-120 are shown in Figure 3B of Benoit, reproduced above. These steps disclose that the Internet traffic between the virtual applications operating in the sandbox OE and the cloud or other cloud services is allowed only upon successful authentication of the sandbox OE.

283. Accordingly, Benoit teaches element 1h.

3. Dependent Claim 2

[2pre] The host computer system of claim 1,

284. As discussed above, Benoit discloses the host computer system of independent claim 1, and therefore teaches the preamble of dependent claim 2.

[2a] wherein the processor is configured to access the Internet-based cloud service via one or more of a local area network (LAN), a wide area network (WAN), a cellular network, a Wi-Fi network, a digital subscribers line (DSL) network, a cloud computing network, a data center, or a satellite network.

285. Benoit teaches element 2a.

286. As discussed above for element 1e, Benoit teaches that the sandbox operating environment (OE) implemented by the computer device's processor is **configured to access an Internet-based cloud service.**

287. Benoit additionally teaches that the access is done **via one or more of a local area network (LAN), a wide area network (WAN), or a cloud computing network**. In particular, Benoit discloses the Internet (which a POSITA would have recognized as a type of **WAN**) and remote computing resources or “cloud” (which a POSITA would have recognized as a **cloud computing network**), as functionally illustrated in Figure 2 of Benoit, reproduced below with the Internet and cloud 213 highlighted. Ex-1005 (Benoit), Fig. 2. Indeed, Benoit explicitly recognizes the Internet as a type of WAN. Ex-1005 (Benoit) ¶¶ 0003 (“the Internet and other wide area networks (WAN)”), 0025 (“In an exemplary embodiment according to the present invention, data may be provided to the system, stored by the system and provided by the system to users of the system across ... wide area networks (WANs) (e.g., the Internet).”). Benoit thus teaches that sandbox OE implemented by the computer device’s processor is configured to access the Internet-based cloud service via the Internet (a WAN) and/or via the cloud (for implementations where the sandbox OE is accessing cloud services outside of the particular cloud via the connection from the proxy server to the second cloud labeled “Internet”). Ex-1005 (Benoit), Fig. 2.

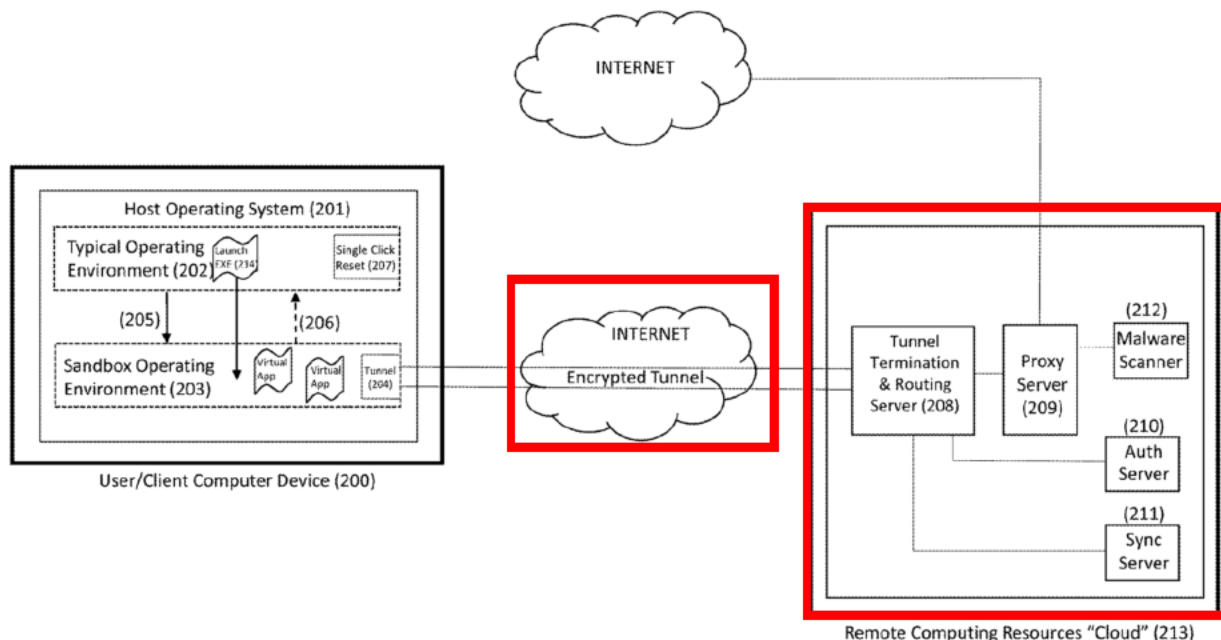


FIG. 2

288. Benoit further teaches that the access can be done via a **LAN**. In particular, Benoit discloses that “[i]n an exemplary embodiment according to the present invention, data may be provided to the system, stored by the system and provided by the system to users of the system across **local area networks (LANs)** (e.g., office networks, home networks)” Ex-1005 (Benoit) ¶ 0025; *see also id.* ¶¶ 0001 (“The present invention generally relates to systems and methods for end-to-end security for computing devices in standalone, LAN, WAN or Internet architectures.”), 0007 (“providing simplified end-to-end security for computing devices in standalone, LAN, WAN or Internet architectures”), 0014 (same), 0025 (“[T]he system may be comprised of numerous servers communicatively connected across one or more LANs and/or WANs.”).

289. Accordingly, Benoit teaches element 2a.

4. Dependent Claim 3

[3pre] The host computer system of claim 1,

290. As discussed above, Benoit discloses the host computer system of independent claim 1, and therefore teaches the preamble of dependent claim 3.

[3a] wherein the Internet-based cloud service comprises one or more of an Infrastructure as a Service (IaaS), a Software as a Service (SaaS), or a Platform as a Service (PaaS).

291. Benoit teaches element 3a.

292. As discussed above for element 1e, Benoit teaches an **Internet-based cloud service**. Benoit also teaches that the encrypted connection between the sandbox operating environment (OE) and the remote computing resources or “cloud” “is a secure shell providing an encryption tunnel between the host system and the remote computing system that supports multiple protocols selected from a group of protocols comprising web, mail, video conferencing, and instant messaging, and all other communication and application protocols.” Ex-1005 (Benoit) ¶ 0010; *see also id.* ¶¶ 0036 (“[A] tunneled SSH connection can support multiple protocols, including, but not limited to, web, mail, video conferencing and instant messaging.”), 0049 (“The user may operate within steps 100-140 throughout the running duration of the system components on host computing devices. For example, web surfing various sites, email, instant messaging, video conferencing, etc.”).

293. A POSITA would have understood that email, video conferencing, and instant messaging are examples of **Software as a Service (SaaS)**. SaaS is a type of cloud service that offers software resources over the Internet. A POSITA would have understood that such software resources could include the provision of email, video conferencing, and instant messaging services.

294. Benoit also teaches “multi-layered malware protection, multi-protocol encrypted tunnel handling, and proxy capabilities for privacy, user flexibility and ease of use,” which are services provided over the Internet by the tunnel termination and routing server, the proxy server, and the malware scanner in the cloud. Ex-1005 (Benoit) ¶¶ 0027, 0041; *see also id.* ¶¶ 0033 (“multi-protocol encrypted tunnel & proxy (MPETP) services”), 0048 (“Traffic returning from the Internet is scanned by a multi-engine signature- and heuristic-based antimalware system (130).”). Benoit further teaches synchronizing the end-user’s custom settings (customer profile) with a synchronization server in the cloud so that the custom settings can be restored to the sandbox if the sandbox becomes infected by malicious software code and needs to be rebuilt. Ex-1005 (Benoit) ¶¶ 0032, 0041. A POSITA would have understood that malware protection, encrypted tunnel handling, proxy, and synchronization services are examples of SaaS.

295. A POSITA would further have understood that the tunnel termination and routing server, the proxy server, the authentication server, the synchronization

server, and the malware scanner in the cloud are examples of **Infrastructure as a Service (IaaS)**. IaaS is a type of cloud service that offers computing, storage, or networking resources over the Internet. The tunnel termination and routing server, the proxy server, and the authentication server would have been understood by a POSITA to be cloud services that offer storage resources over the Internet, while the synchronization server and the malware scanner would have been understood by a POSITA to be cloud services that offer computing resources over the Internet.

296. Accordingly, Benoit teaches element 3a.

5. Dependent Claim 8

[8pre] The host computer system of claim 1,

297. As discussed above, Benoit discloses the host computer system of independent claim 1, and therefore teaches the preamble of dependent claim 8.

[8a] wherein the internal isolation firewall is configured to segregate the first memory space from the second memory space, and

298. Benoit discloses that the **internal isolation firewall is configured to segregate the first memory space from the second memory space**, and thus teaches element 8a.

299. As discussed above for element 1d, a POSITA would have understood that the sandbox OE would necessarily have had its own **memory space**. To a POSITA, a sandbox having “pre-defined read (40) and write (50) permissions to

the host system,” as taught in Benoit, is a sandbox having its own memory space (a second memory space) separate from the memory spaces (the first memory space) for other operating environments on the user/client computer device, including the typical OE. Ex-1005 (Benoit) ¶ 0043 & Fig. 3A.

300. Additionally, a POSITA would have understood the pre-defined read (40) and write (50) permissions to the host system to be a form of **internal isolation firewall that segregates the memory spaces**. That is, a POSITA would have known that the host operating system needs a module to monitor sandbox activities and to determine whether to permit or restrict activities such as accessing a memory space that is separate from the memory space of the sandbox, and this module is an internal isolation firewall. Benoit also teaches that “the sandbox is designed to ‘deny all’ software code from accessing host resources outside of the sandbox,” which would have further indicated to a POSITA that the sandbox OE is isolated by an internal isolation firewall that segregates the sandbox’s resources, including its memory space, from the host’s resources, including its memory space. Ex-1005 (Benoit) ¶ 0032; *see also id.* (“[N]othing occurs outside of the resources belonging to the sandbox (e.g., virtual host resources)”).

301. Accordingly, Benoit teaches element 8a.

[8b] wherein the processor is configured to enforce policies for communication between the first memory space and the second memory space using the internal isolation firewall.
--

302. Benoit discloses that the processor is **configured to enforce policies for communication between the first memory space and the second memory space using the internal isolation firewall**, and thus teaches element 8b. In Benoit, the policies for communication are the “pre-defined read (40) and write (50) permissions to the host system,” which are implemented by the internal isolation firewall. Ex-1005 (Benoit) ¶ 0043 & Fig. 3A. A POSITA would have understood that these policies govern communication between the memory space for the typical OE and the memory space for the sandbox OE.

303. Accordingly, Benoit teaches element 8b.

6. Independent Claim 10

304. It is my opinion that Benoit anticipates the limitations of claim 10. Claim 10 largely repeats the limitations of claim 1. I incorporate by reference the relevant analysis above for the limitations repeated from claim 1 and do not re-state that analysis in the following discussion, it being understood that the analysis would be the same.

[10pre] A cloud computing system comprising:

305. Benoit discloses a cloud computing system in the form of the system illustrated in Figure 2 of Benoit, and therefore teaches the preamble of claim 10.

Ex-1005 (Benoit), Fig. 2; *see also id.* ¶ 0041 (describing the computing system of Figure 2 as having a “cloud component” 213). As discussed above for element 1e, the user/client computer device in the system is configured to access an Internet-based cloud service in the form of the remote computing resources or “cloud,” making the system a cloud computing system.

306. Accordingly, to the extent that the preamble of claim 10 is limiting, Benoit teaches it.

[10a] a cloud computing network comprising a server,

307. Benoit teaches element 10a.

308. Benoit discloses a cloud computing network in the form of the system illustrated in Figure 2 of Benoit, with the system comprising a server in the form of the user/client computer device, and therefore teaches element 10a of claim 10 of the '104 patent. Ex-1005 (Benoit), Fig. 2; *see also id.* ¶ 0041 (describing the computing system of Figure 2 as having a “cloud component” 213). Figure 2 of Benoit is reproduced below with the user/client computer device 200 highlighted.

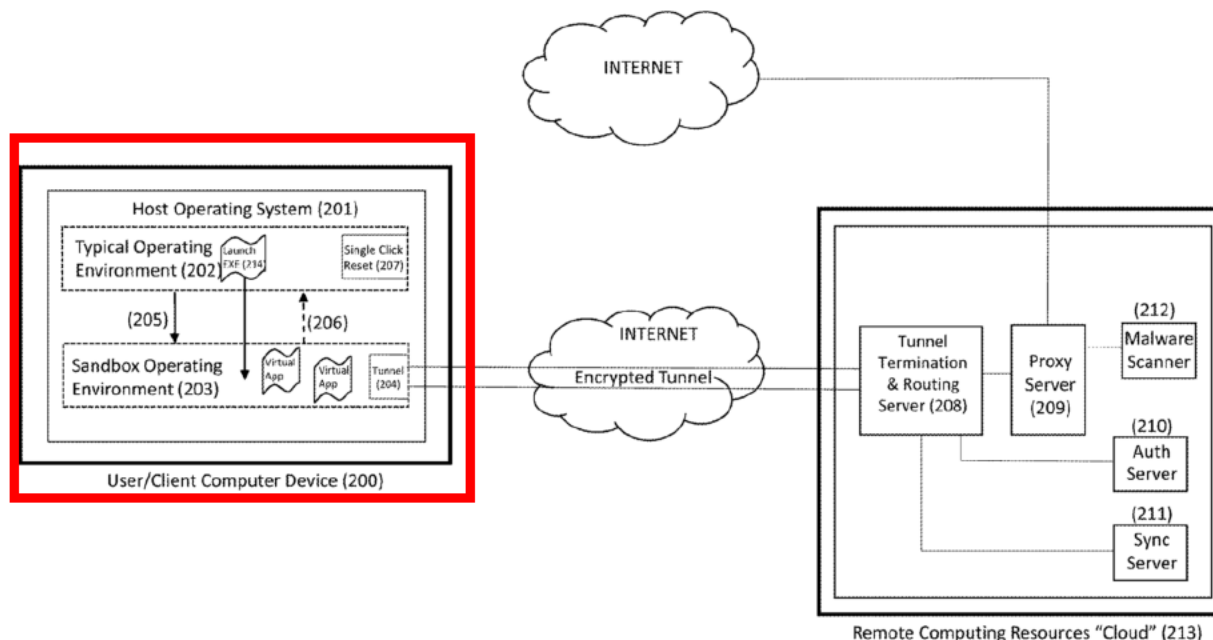


FIG. 2

309. A POSITA would have recognized the user/client computer device as a type of server since it can provide information to other devices on the Internet. Indeed, Benoit explicitly states as follows:

Examples of computing devices usable with embodiments of the present invention include, but are not limited to, personal computers, smart phones, laptops, mobile computing devices, tablet PCs and **servers**. The term computing device may also describe two or more computing devices communicatively linked in a manner as to distribute and share one or more resources, such as clustered computing devices and **server** banks/farms.

Ex-1005 (Benoit) ¶ 0024; *see also id.* ¶ 0025 (“[T]he system may be comprised of numerous servers communicatively connected across one or more LANs and/or WANs.”).

310. As discussed above for element 1e, the user/client computer device is configured to access an Internet-based cloud service, making the system a cloud computing network comprising a server.

311. Accordingly, Benoit teaches element 10a.

[10b] wherein the server is configured to: implement a workspace configured to enable operation of a first set of one or more applications or processes;

312. As discussed above for element 10a, Benoit teaches a server in the form of the user/client computer device. As discussed above for element 1c, Benoit teaches implementing a workspace in the form of the typical operating environment (OE) that is configured to enable operation of a first set of one or more applications or processes, such as a single-click reset process and a launch executable program. Ex-1005 (Benoit), Fig. 2; *see also id.* ¶ 0041. For the reasons discussed above for element 1c, Benoit discloses element 10b.

[10c] implement an isolated computing environment using a host operating system and comprising a sandboxed computing environment and being configured to enable operation of a second set of one or more applications or processes;

313. As discussed above for element 1d, Benoit discloses implementing an isolated computing environment in the form of a sandbox operating environment (OE), as well as the remaining limitations in this element 10c. For the reasons discussed above for element 1d, Benoit discloses element 10c.

[10d] isolate the isolated computing environment from the workspace using an internal isolation firewall; and

314. As discussed above for element 1f, Benoit discloses isolating Benoit's sandbox operating environment (OE) (an isolated computing environment) from Benoit's typical operating environment (OE) (a workspace) using an internal isolation firewall. For the reasons discussed above for element 1f, Benoit discloses element 10d.

[10e] authenticate the isolated computing environment with an authentication device; and

315. As discussed above for element 1g, Benoit teaches element 10e, including an authentication device in the form of the authentication server that authenticates the sandbox operating environment (OE) (an isolated computing environment).

[10f] a proxy configured to control communication between the server and an Internet resource,

316. Benoit discloses element 10f.

317. As discussed above for element 1h, Benoit teaches a proxy server controlling communications between the user/client computer device (a server) and Internet resources in the cloud, as well as other Internet resources outside of the particular cloud via a connection from the proxy server to a second cloud labeled "Internet." Ex-1005 (Benoit), Fig. 2. As shown in Figure 2 of Benoit, the

user/client computer device 200 communicates with the proxy server 209 in order to access Internet resources in the cloud 213, as well as to access other Internet resources outside of the particular cloud 213 via a connection from the proxy server 209 to a second cloud labeled “Internet.” Ex-1005 (Benoit), Fig. 2. Benoit makes clear that the connection is established “via the secured tunnel, with the proxy server (110) in the RCR component of the system.” Ex-1005 (Benoit) ¶ 0047. The reference to “the proxy server (110)” is to step 110 in Figure 3B of Benoit, which makes clear that the proxy server is involved in establishing the connection via the encrypted tunnel. Ex-1005 (Benoit), Fig. 3B.

318. As further shown in Figure 3B of Benoit, the proxy server scans all traffic before it is returned to the virtual applications running inside the sandbox OE on the user/client computer device. This disclosure is further confirmation that the proxy server is configured to control communications between the user/client computer device and the Internet resources. The relevant steps 110-140 are shown in Figure 3B of Benoit, reproduced at right.

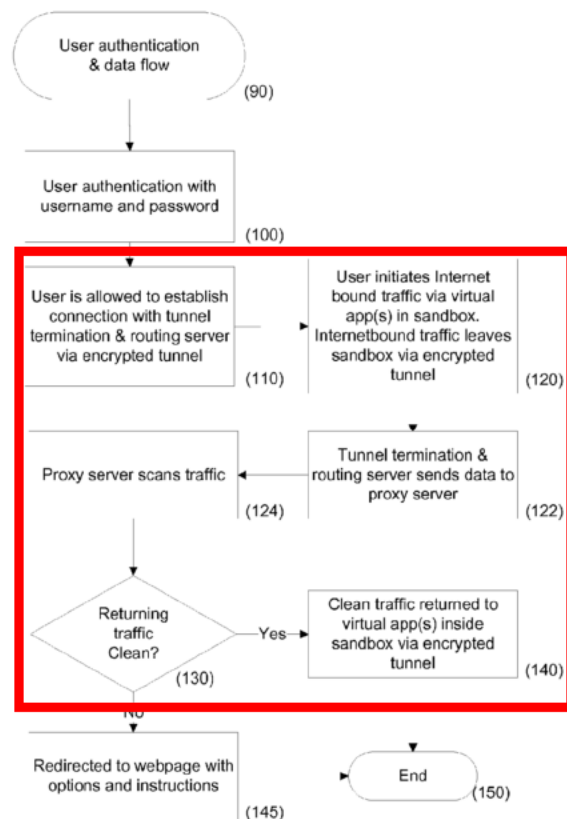


FIG. 3B

319. For these reasons as well as the reasons discussed above for element 1h, Benoit discloses element 10f.

[10g] wherein the proxy is configured to allow communication between the second set of one or more applications or processes running in the isolated computing environment and the Internet resource when the isolated computing environment has been authenticated and

320. Benoit teaches element 10g.

321. As discussed above for element 1h, Benoit teaches that the proxy server is configured to allow communication between one or more one applications or processes running in the sandbox operating environment (OE) (an isolated computing environment) and an Internet-based resource. As further discussed above for element 1h, Benoit discloses that the communication between the requesting application and the Internet-based resource is allowed when the sandbox OE (the isolated computing environment) has been authenticated. For the reasons discussed above for element 1h of claim 1, Benoit discloses element 10g.

[10h] to block communication between the first set of one or more applications or processes running in the workspace and the Internet resource.

322. Benoit discloses that the proxy server is also configured to **block communication from applications or processes running in any of the workspaces** on the user/client computer device **and the Internet** resources, and thus teaches element 10h. In particular, Benoit explains:

Traffic returning from the Internet is scanned by a multi-engine signature- and heuristic-based antimalware system (130). This process occurs at the RCR components of the system, as to prevent any malicious code or content from reaching the host computing device of the user. ... If the traffic returning from the Internet is not free of malicious code or content, then the traffic is **not returned** to the virtual application running on the system components of the user's host computer and the user is redirected to a webpage with further options and instructions (140).

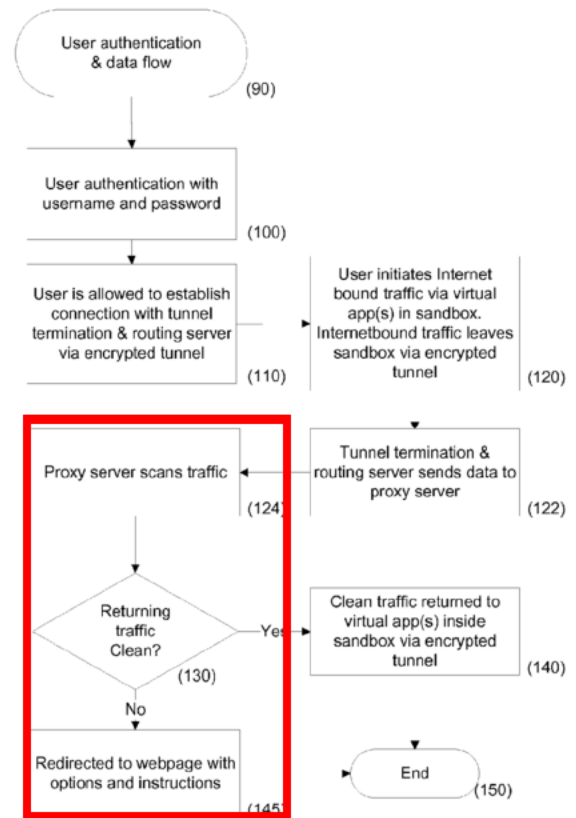


FIG. 3B

Ex-1005 (Benoit) ¶ 0048.

323. Figure 3B of Benoit teaches in step 124 that it is the proxy server which scans and blocks the Internet communications. The relevant steps 124, 130 and 145 are shown in Figure 3B of Benoit, reproduced above. Ex-1005 (Benoit), Fig. 3B.

324. Accordingly, Benoit teaches element 10h.

7. Dependent Claim 11

[11pre] The cloud computing system of claim 10,

325. As discussed above, Benoit discloses the host computer system of independent claim 10, and therefore teaches the preamble of dependent claim 11.

[11a] wherein the server is configured to perform at least one processing task on behalf of a client device.

326. Benoit teaches element 11a. As discussed above for elements 1e and 3a, Benoit teaches an Internet-based cloud service, including mail, video conferencing, and instant messaging. Ex-1005 (Benoit) ¶¶ 0010, 0036, 0049. These are examples of where **the server is configured to perform at least one processing task on behalf of a client device**. The '104 patent describes the server performing a task for the client in the same manner:

The server may perform a processing task on behalf of a client device. For example, the server may receive input data from the client device and may send data to the client device. The processing task may include one or more of an Infrastructure as a Service (IaaS), a Software as a Service (SaaS), and/or a Platform as a Service (PaaS).

Ex-1001, 2:57-62. In each of Benoit's examples, the server receives input data from the client device, e.g., emails, videos, and instant messages. *See also* Ex-1005 (Benoit) ¶ 0025 (“[D]ata may be provided to the system, stored by the system and provided by the system to users of the system across local area

networks (LANs) (e.g., office networks, home networks) or wide area networks (WANs) (e.g., the Internet).”).

327. Accordingly, Benoit teaches element 11a.

8. Dependent Claim 17

[17pre] The cloud computing system of claim 11,

328. As discussed above, Benoit discloses the cloud computing system of dependent claim 11, and therefore teaches the preamble of dependent claim 17.

[17a] wherein the at least one processing task performed on behalf of the client device comprises a processing task for an Infrastructure as a Service (IaaS), a Software as a Service (SaaS), or a Platform as a Service (PaaS).

329. Benoit teaches element 17a.

330. As discussed above for element 3a, Benoit teaches services including mail, video conferencing, and instant messaging that a POSITA would have recognized as examples of Software as a Service (SaaS). Ex-1005 (Benoit) ¶¶ 0010, 0036, 0049.

331. As also discussed above for element 3a, Benoit teaches “multi-layered malware protection, multi-protocol encrypted tunnel handling, and proxy capabilities for privacy, user flexibility and ease of use,” which are services provided over the Internet by the tunnel termination and routing server, the proxy server, and the malware scanner in the cloud that a POSITA would have

recognized as examples of Infrastructure as a Service (IaaS). Ex-1005 (Benoit) ¶¶ 0027, 0041; *see also id.* ¶¶ 0033, 0048.

332. As additionally discussed above for element 3a, Benoit teaches synchronizing the end-user's custom settings (customer profile) with a synchronization server in the cloud so that the custom settings can be restored to the sandbox if the sandbox becomes infected by malicious software code and needs to be rebuilt. Ex-1005 (Benoit) ¶¶ 0032, 0041. A POSITA would have recognized such malware protection, encrypted tunnel handling, proxy, and synchronization services as examples of SaaS.

333. As further discussed above for element 3a, a POSITA would further have understood the tunnel termination and routing server, the proxy server, the authentication server, the synchronization server, and the malware scanner in the cloud as examples of Infrastructure as a Service (IaaS).

334. Accordingly, Benoit teaches element 17a.

9. Independent Claim 18

335. Independent claim 18 recites a method implemented in a host computer system that is the method counterpart of the host computer system of claim 1 with the same elements phrased in the form of method steps. Claim 18 contains no additional limitations beyond the fact that it recites a method instead of

a system. For the reasons stated above in connection with claim 1, Benoit discloses the method recited in claim 18.

10. Dependent Claim 19

[19pre] The method of claim 18,

336. As discussed above, Benoit discloses the cloud computing system of independent claim 18, and therefore teaches the preamble of dependent claim 19.

[19a] wherein the Internet-based cloud service comprises one or more of an Infrastructure as a Service (IaaS), a Software as a Service (SaaS), or a Platform as a Service (PaaS).

337. Benoit teaches element 19a.

338. As discussed above for element 3a, Benoit teaches services including mail, video conferencing, and instant messaging that a POSITA would have recognized as examples of Software as a Service (SaaS). Ex-1005 (Benoit) ¶¶ 0010, 0036, 0049.

339. As also discussed above for element 3a, Benoit teaches “multi-layered malware protection, multi-protocol encrypted tunnel handling, and proxy capabilities for privacy, user flexibility and ease of use,” which are services provided over the Internet by the tunnel termination and routing server 208, the proxy server 209, and the malware scanner 212 in the cloud 213 that a POSITA would have understood as examples of Infrastructure as a Service (IaaS). Ex-1005 (Benoit) ¶¶ 0027, 0041; *see also id.* ¶¶ 0033, 0048.

340. As additionally discussed above for element 3a, Benoit teaches synchronizing the end-user's custom settings (customer profile) with a synchronization server in the cloud so that the custom settings can be restored to the sandbox if the sandbox becomes infected by malicious software code and needs to be rebuilt. Ex-1005 (Benoit) ¶¶ 0032, 0041. A POSITA would have recognized such malware protection, encrypted tunnel handling, proxy, and synchronization services as examples of SaaS.

341. As further discussed above for element 3a, a POSITA would further have recognized the tunnel termination and routing server, the proxy server, the authentication server, the synchronization server, and the malware scanner in the cloud as examples of Infrastructure as a Service (IaaS).

342. Accordingly, Benoit teaches element 19a.

343. For the foregoing reasons, it is my opinion that a POSITA would have found claims 1-3, 8, 10, 11 and 17-19 of the '104 patent to be anticipated by Benoit.

D. Ground 4: Claims 1-3, 8, 10, 11, and 17-19 Are Obvious in View of Benoit and Hunt

344. It is my opinion that a POSITA would have found claims 1-3, 8, 10, 11 and 17-19 to be obvious based on the teachings of Benoit in view of Hunt.

2. Combination of Benoit and Hunt

a. Summary of Hunt

345. Hunt was filed on July 13, 2010, and was published on January 19, 2012, more than five years before the stated priority date of the '104 patent (September 26, 2017). Ex-1008 (Hunt). I am informed that Hunt therefore qualifies as prior art to the '104 patent.

346. Hunt teaches a secure application execution system as shown in Figure 1, reproduced below.

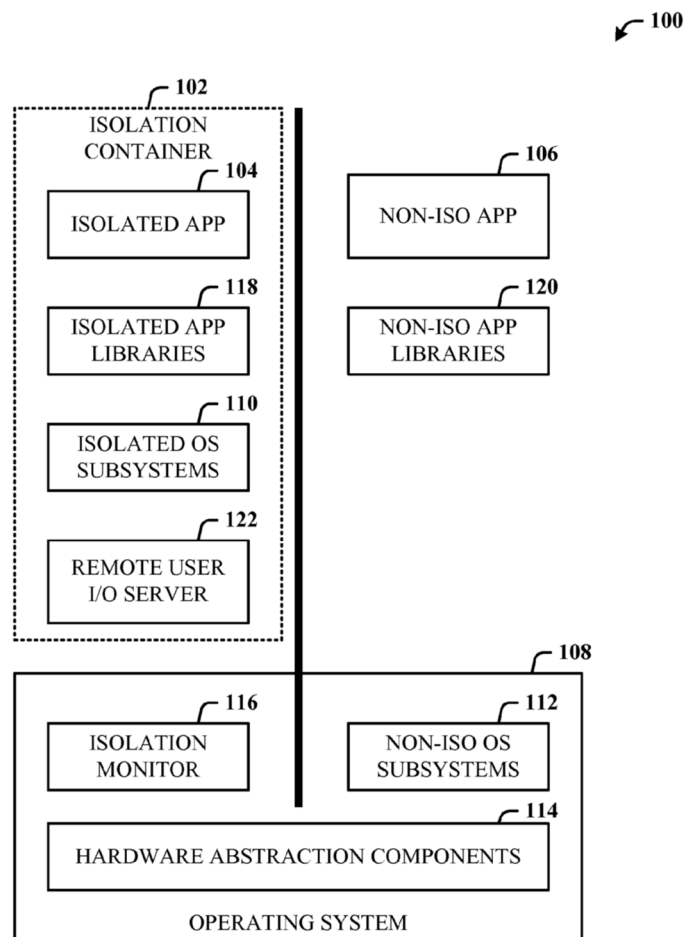


FIG. 1

347. As shown in Figure 1 and as described in the specification of Hunt, the system includes an isolation container, in which an application runs in isolation from a non-isolated application. Ex-1008 (Hunt) ¶ 0024. Hunt teaches that the isolated application and the non-isolated application both run in association with a single operating system (OS). *Id.* Isolated OS subsystems of the isolation container provide services to the isolated application through isolated application libraries, and non-isolated OS subsystems of the OS provide equivalent services to the non-isolated application through non-isolated application libraries. Ex-1008 (Hunt) ¶¶ 0024-0025.

348. Hunt teaches that the OS includes hardware abstractions for both the isolated application and the non-isolated application. Ex-1008 (Hunt) ¶¶ 0025, 0029. Additionally, the OS includes an isolation monitor that provides the interface for services from the OS to the isolation container. Ex-1008 (Hunt) ¶ 0025. The separation of the state related to the isolated application from the state related to the non-isolated application is represented by the black bar in Figure 1 of Hunt that extends between the isolated application 104 and the non-isolated application 106, and down into the OS 108 between the isolation monitor 116 and the non-isolated OS subsystem 112. *Id.* Although illustrated as a distinct, separate component in Figure 1, Hunt explains that the isolation monitor functionality can alternatively be implemented by modifying portions of the OS to run the isolating

functions of the isolation monitor for the isolated application in addition to running non-isolated functions for the non-isolated application. Ex-1008 (Hunt) ¶ 0028.

349. Hunt’s isolation container uses the host OS. In particular, Hunt teaches that the isolated application and the non-isolated application use basic computation services provided by the OS. Ex-1008 (Hunt) ¶ 0030. The basic computation services can include virtual memory management, thread creation, and thread synchronization. *Id.* As discussed above, the OS can also include the hardware abstractions for both the isolated and non-isolated applications. Ex-1008 (Hunt) ¶¶ 0025, 0029. The code within the isolation container—including the isolated application, the isolated application libraries, the isolated OS subsystems, and the remote user I/O server—interfaces to the kernel of the operating system through the isolation monitor. Ex-1008 (Hunt) ¶ 0031.

350. The isolation monitor “employs a collection of rules that map the approval or denial of requests to access resources to an application manifest.” Ex-1008 (Hunt) ¶ 0029. The isolation monitor thus is an internal isolation wall, which Hunt calls an “application firewall.” Ex-1008 (Hunt) ¶ 0053. Hunt describes “rules that confine untrusted applications,” teaches that “[a] collection of such rules forms an application firewall,” and explains that “[t]he rules map to approving or denying application manifest requests.” *Id.*; *see also id.* ¶ 0055 (“[A]n application firewall can impose information flow rules, ruling out particular

pipes, or specifying ALLOW or DENY lists of applications that may connect to a given pipe endpoint.”). Such an application firewall describes the workings of Hunt’s isolation monitor.

351. In operation, certain “rich functionality” in an operating system kernel is moved from the kernel into the isolated OS subsystems of an isolation container. Ex-1008 (Hunt) ¶ 0094. Communications between the rich functionality in the isolated OS subsystems and the operating system kernel are implemented via the isolation monitor. *Id.* The isolation is performed “using an application firewall of rules that control interaction between the functionality and the kernel.” *Id.*

352. In addition to isolating the sandboxed isolation container, the isolation monitor also isolates abstractions like virtual memory for use by the sandbox environment. Ex-1008 (Hunt) ¶ 0005. These abstractions create a memory address space specific to the isolation container. Ex-1008 (Hunt) ¶¶ 0031, 0066. Hunt teaches that “[i]solation is enforced by a combination of virtual memory hardware and a highly restricted kernel API exposed by the isolation monitor 116.” Ex-1008 (Hunt) ¶ 0037. Hunt then describes an exemplary computation interface used for creating the sandboxed isolation container that involves allocating virtual memory between a free memory space and a protected memory space. Ex-1008 (Hunt) ¶ 0092.

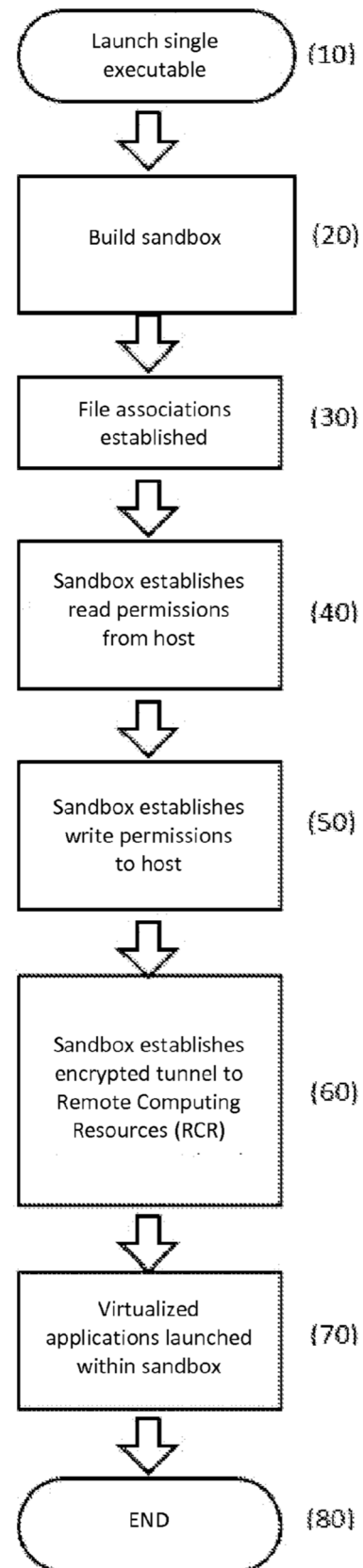
**b. A POSITA Would Have Been Motivated to
Combine Sandbox Implementation Details of
Hunt with the Computer System of Benoit**

353. In my opinion, a POSITA would have been motivated to combine Hunt's sandbox implementation, including the firewall and memory allocation processes, with the computer system of Benoit. Both Benoit and Hunt are directed to the field of network security. As explained in Hunt, "[s]andboxing is a security technique for isolating the execution of untested code and untrusted applications." Ex-1008 (Hunt) ¶ 0001. Like Krasin (discussed earlier in this declaration), Hunt describes the benefits of using a low-overhead sandbox environment instead of a high-overhead sandbox solution in the form of a virtual machine. Ex-1008 (Hunt) ¶¶ 0001-0003, 0005. Hunt discloses that the reduction in overhead taught in Hunt is achieved by "tak[ing] basic computing services already provided by the host operating system, such as virtual memory and threads, and safely isolat[ing] these abstractions for use by the sandbox environment." Ex-1008 (Hunt) ¶ 0005.

354. Both Benoit and Hunt thus disclose a sandboxed computing environment, and as discussed above, both references also teach that this sandboxed computing environment uses a host operating system and is isolated from another computing environment on the same computer device. Benoit,

however, does not expressly use the word “firewall” to describe the isolation of the typical operating environment from the sandbox operating environment.

355. As a preliminary matter, it is my opinion that a POSITA, at the time when Benoit was filed, would have found that Benoit’s host operating system would have necessarily implemented an internal isolation firewall between the typical operating environment and the sandbox operating environment, and that these two environments would necessarily have had their own memory spaces. To a POSITA, a sandbox having “pre-defined read (40) and write (50) permissions to the host system,” as taught in Benoit and shown in Figure 3A (reproduced at right), is a sandbox having its own memory space and isolated from the host system using an internal isolation firewall (the pre-defined read and write permissions).



356. To exclude any doubt, however, I have been asked to opine on the validity of the challenged claims by further analyzing the combination of Benoit and Hunt. In my opinion, Hunt demonstrates that a POSITA would have been highly familiar with implementing an isolated computing environment as a sandboxed computing environment having its own memory space and isolated from the host system using an internal isolation firewall, well before the stated priority date of the '104 patent (September 26, 2017). Hunt simply provides the implementation details for the sandboxed computing environment, including the firewall and memory allocation processes, details that a POSITA would have straightforwardly used to implement Benoit's sandbox operating environment with a reasonable expectation of success. This combination would simply involve a POSITA combining prior-art elements according to known methods to yield a predictable result—Benoit's sandbox operating environment would have its own memory space and be isolated using an internal isolation firewall.

357. A POSITA would have been further motivated to combine the sandbox implementation details of Hunt with the computer system of Benoit because of the stated advantages of Hunt's sandbox architecture. Hunt teaches that the disclosed architecture both reduces computing overhead and safely isolates abstractions like virtual memory and threads for use by a sandbox environment. Ex-1008 (Hunt) ¶ 0005. Hunt teaches that “[t]he significant reduction in resources

and overhead provided by the disclosed architecture now makes it possible to sandbox every application.” Ex-1008 (Hunt) ¶ 0021. In my opinion, a POSITA would have been further motivated by these stated advantages to combine the sandbox implementation details of Hunt with the computer system of Benoit with a reasonable expectation of success.

c. A POSITA Would Have Had a Reasonable Expectation of Success in Combining the Sandbox Implementation Details of Hunt with the Computer System of Benoit

358. In my opinion, a POSITA would have had a reasonable expectation of success in combining the sandbox implementation details of Hunt with the computer system of Benoit, given the similarities in the teachings and systems of the two references, and given that technologies for implementing sandboxed computing environments were well known at the time. A POSITA would have expected to be successful in making the combination because the combination involves using conventional and well-known computer programming methods to yield a predictable result. Implementing an internal isolation firewall and a separate memory space for a sandboxed computing environment, like the sandbox operating environment (OE) of Benoit, would not have required specialized skills beyond what a POSITA would have had.

359. Furthermore, a POSITA would have been motivated and would have had a reasonable expectation of success in combining Benoit and Hunt, as

explained above, because the combination would not have changed the principle of operation for any of the teachings of the references. In fact, Benoit already teaches a sandboxed computing environment that uses its own memory space and is isolated from the typical operating environment (OE) using an internal isolation firewall, as discussed above for elements 1d and 1f. Both Benoit's sandbox OE and Hunt's isolation container operate through isolation from other computing environments. Rather than changing Benoit's principle of operation, the combination with Hunt would actually reinforce Benoit's principle of operation by using a known technique (the sandbox implementation details of Hunt) to achieve what Benoit already teaches (a sandboxed computing environment that uses its own memory space and is isolated using an internal isolation firewall).

3. Claim Analysis

360. For this Ground 4, the element-by-element analysis of all of the challenged claims is the same as the element-by-element analysis for Ground 3, except that Hunt provides additional disclosure for:

- The limitation in elements 1c, 1d, 18b, 18d of “a first memory space” and “a second memory space”;
- The limitation in elements 1f, 10d, 18f of “isolat[ing] the isolated computing environment from the workspace using an internal isolation firewall;” and

- Dependent claim 8.

a. “a first memory space” and “a second memory space” in elements 1c, 1d, 18b, and 18d

361. Benoit in combination with Hunt discloses that the typical OE (the workspace) is configured to enable operation of applications or processes via a **first memory space**, while the sandbox OE (the isolated computing environment) uses a **second memory space**. As discussed above for elements 1c, 1d, 18b, and 18d, a POSITA would have understood that Benoit’s typical OE and sandbox OE have their own spaces. To exclude any doubt, however, I have been asked to opine on the validity of the challenged claims by further analyzing the combination of Benoit and Hunt.

362. Hunt provides the details of a sandbox implementation, which as discussed above, a POSITA would have been motivated to combine with the computer system of Benoit with a reasonable expectation of success. As discussed above, Hunt teaches that, in addition to isolating the sandboxed isolation container, the isolation monitor also isolates abstractions like virtual memory for use by the sandbox environment. Ex-1008 (Hunt) ¶ 0005. These abstractions create a memory address space specific to the isolation container. Ex-1008 (Hunt) ¶¶ 0031, 0066. Hunt teaches that “[i]solation is enforced by a combination of virtual memory hardware and a highly restricted kernel API exposed by the isolation monitor 116.” Ex-1008 (Hunt) ¶ 0037.

363. Hunt then describes an exemplary computation interface used for creating the sandboxed isolation container that involves allocating virtual memory from a free memory space to a protected memory space. Ex-1008 (Hunt) ¶ 0092 (in particular, DKVirtualMemoryAllocate and DKVirtualMemoryFree deal with memory space allocation). A POSITA would have understood that the free memory space would have been memory space available to Hunt’s non-isolated applications operating outside the isolation container (a **first memory space**). A POSITA would have also understood that the protected memory space would have been memory space available to Hunt’s isolated applications operating inside the isolation container (a **second memory space**).

364. Accordingly, Benoit in combination with Hunt teaches elements 1c, 1d, 18b, and 18d.

b. “isolate the isolated computing environment from the workspace using an internal isolation firewall” (elements 1f, 10d, and 18f)

365. Benoit in combination with Hunt teaches the limitation in elements 1f, 10d, and 18f of “isolat[ing] the isolated computing environment from the workspace using an internal isolation firewall.”

366. As discussed above, both Benoit and Hunt disclose a sandboxed computing environment that is isolated from another computing environment on the same computer device. Benoit, however, does not expressly use the word

“firewall” to describe the isolation of the typical operating environment from the sandbox operating environment. As discussed above, it is my opinion that a POSITA would have understood that Benoit’s host operating system would have necessarily implemented an internal isolation firewall between the typical operating environment and the sandbox operating environment.

367. To exclude any doubt, however, I have been asked to opine on the validity of the challenged claims by further analyzing the combination of Benoit and Hunt. In my opinion, Hunt demonstrates that a POSITA would have been highly familiar with implementing an isolated computing environment as a sandboxed computing environment having its own memory space and isolated from the host system using an internal isolation firewall, well before the stated priority date of the ’104 patent (September 26, 2017). This combination would simply involve a POSITA combining prior-art elements according to known methods to yield a predictable result—Benoit’s sandbox operating environment would have its own memory space and be isolated using an internal isolation firewall.

368. Hunt discloses isolating the isolation container (an isolated computing environment) from the rest of the application execution system, including the non-isolated applications (a workspace). Hunt teaches that this isolation is accomplished using an internal isolation firewall in the form of an isolation

monitor. Figure 1 of Hunt is reproduced at right with the isolation monitor 116 highlighted. Ex-1008 (Hunt), Fig. 1.

369. Hunt’s isolation monitor “employs a collection of rules that map the approval or denial of requests to access resources to an application manifest.” Ex-1008 (Hunt) ¶ 0029. Hunt’s isolation monitor thus is an internal isolation wall, which Hunt calls

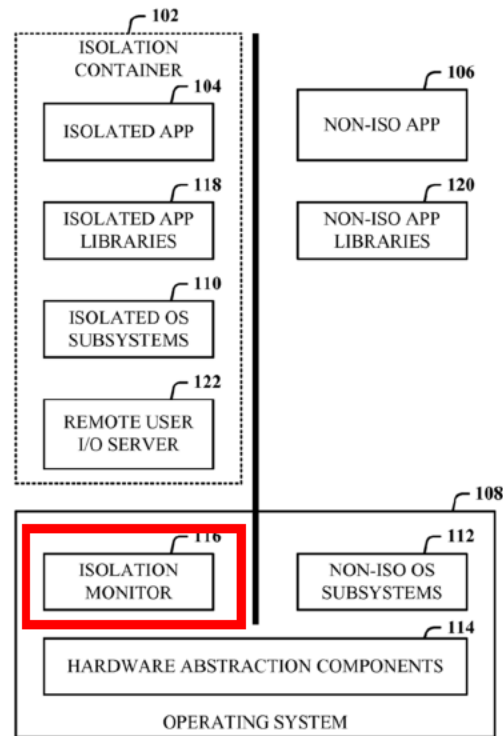


FIG. 1

an “application firewall.” Ex-1008 (Hunt) ¶ 0053. Hunt describes “rules that confine untrusted applications,” teaches that “[a] collection of such rules forms an application firewall,” and explains that “[t]he rules map to approving or denying application manifest requests.” *Id.*; *see also id.* ¶ 0055 (“[A]n application firewall can impose information flow rules, ruling out particular pipes, or specifying ALLOW or DENY lists of applications that may connect to a given pipe endpoint.”). Such an application firewall describes the workings of Hunt’s isolation monitor.

370. Hunt teaches that, in operation, certain “rich functionality” in an operating system kernel is moved from the kernel into the isolated OS subsystems of an isolation container. Ex-1008 (Hunt) ¶ 0094. Communications between the rich functionality in the isolated OS subsystems and the operating system kernel are implemented via the isolation monitor. *Id.* The isolation is performed “using an application firewall of rules that control interaction between the functionality and the kernel.” *Id.* The isolation monitor thus is an internal isolation firewall that isolates the isolation container (an isolated computing environment) from the rest of the application execution system, including the non-isolated applications (a workspace).

371. As discussed above, a POSITA would have been motivated to combine the sandbox implementation details of Hunt, including the firewall and memory allocation processes, with the computer system of Benoit and have a reasonable expectation of success in doing so. To accomplish this, a POSITA would have implemented Hunt’s isolation monitor between Benoit’s typical operating environment (OE) and sandbox operating environment (OE). Hunt’s isolation monitor would be adapted to provide the interface for services from Benoit’s host operating system to Benoit’s sandbox OE, akin to the function that the isolation monitor performs in Hunt. Ex-1008 (Hunt) ¶ 0025. The isolation monitor would also employ a collection of rules that govern the approval or denial

of requests by Benoit's sandbox OE to access resources, again akin to the function that the isolation monitor performs in Hunt. Ex-1008 (Hunt) ¶ 0029. The result would be a separation of the state of the isolated applications in the Benoit's sandbox OE from the state of the non-isolated applications in Benoit's typical OE. Ex-1008 (Hunt) ¶ 0025.

372. Below is Figure 2 of Benoit, modified to show Hunt's isolation monitor positioned to provide the interface for communications between the host operating system and the typical OE, on the one hand, and the sandbox OE, on the other. Hunt's adapted isolation monitor governs the approval or denial of requests by Benoit's sandbox OE to access resources on the user/client computer device. The result is a separation of the state of the isolated applications in the Benoit's sandbox OE from the state of the non-isolated applications in Benoit's typical OE.

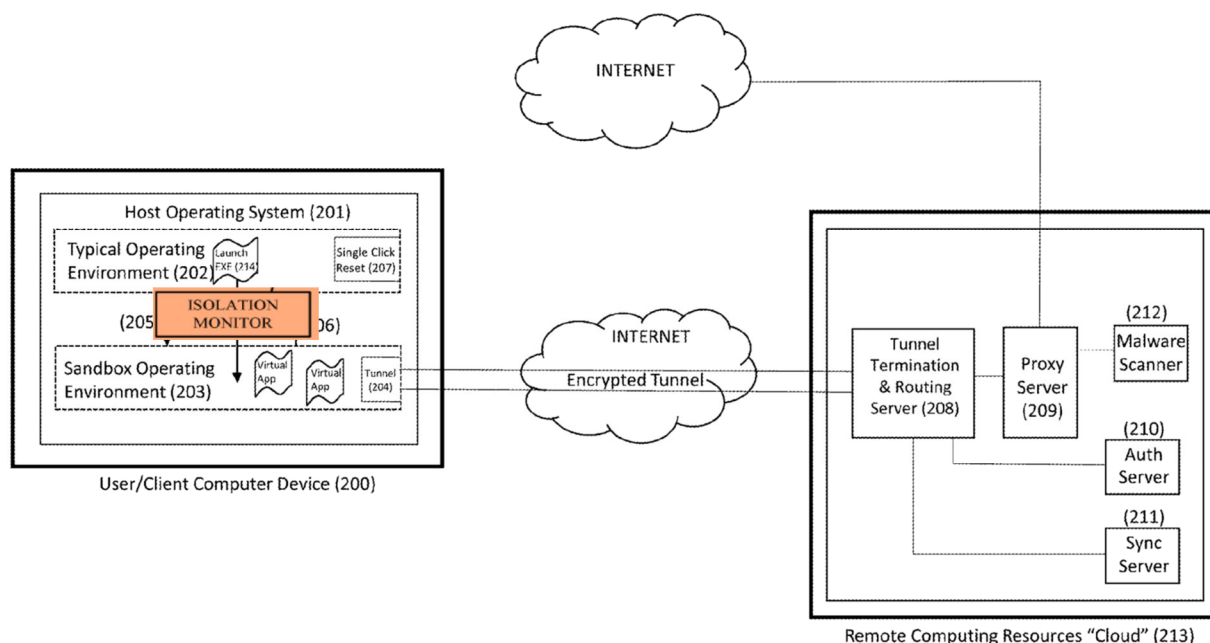


FIG. 2

373. The combination of Benoit and Hunt involves using conventional and well-known computer programming methods to yield a predictable result. The implementation of Hunt's isolation monitor in Benoit's computer system would not have required specialized skills beyond what a POSITA would have had.

374. The combination of Benoit and Hunt thus teaches isolating the isolated computing environment (Benoit's sandbox OE) from the workspace (Benoit's typical OE) **using an internal isolation firewall** (Hunt's isolation monitor). Accordingly, Benoit in combination with Hunt teaches elements 1f, 10d, and 18f.

c. Dependent Claim 8

[8pre] The host computer system of claim 1,

375. As discussed above, Benoit in combination with Hunt discloses the host computer system of independent claim 1, and therefore teaches the preamble of dependent claim 8.

[8a] wherein the internal isolation firewall is configured to segregate the first memory space from the second memory space, and

376. Benoit in combination with Hunt discloses that **the internal isolation firewall is configured to segregate the first memory space from the second memory space**, and thus teaches element 8a.

377. As discussed above, a POSITA would have understood that Benoit's sandbox OE would necessarily have had its own memory space. Additionally, a POSITA would have understood Benoit's pre-defined read (40) and write (50) permissions to the host system to be a form of internal isolation firewall that segregates the memory spaces. Benoit also teaches that "the sandbox is designed to 'deny all' software code from accessing host resources outside of the sandbox," which would have further indicated to a POSITA that the sandbox OE is isolated by an internal isolation firewall that segregates the sandbox's resources, including its memory space, from the host's resources, including its memory space. Ex-1005 (Benoit) ¶ 0032; *see also id.* ("[N]othing occurs outside of the resources belonging to the sandbox (e.g., virtual host resources) ...").

378. To exclude any doubt, however, I have been asked to opine on the validity of the challenged claims by further analyzing the combination of Benoit and Hunt. In my opinion, Hunt demonstrates that a POSITA would have been highly familiar with implementing an isolated computing environment as a sandboxed computing environment having its own memory space **segregated from the memory space for the host system using an internal isolation firewall**, well before the stated priority date of the '104 patent (September 26, 2017). This combination would simply involve a POSITA combining prior-art elements according to known methods to yield a predictable result—Benoit's

sandbox operating environment (OE) would have its own memory space segregated from the memory space for the typical operating environment (OE) using an internal isolation firewall.

379. In particular, as discussed above, Hunt teaches that, in addition to isolating the sandboxed isolation container, the isolation monitor also isolates abstractions like virtual memory for use by the sandbox environment. Ex-1008 (Hunt) ¶ 0005. These abstractions create a memory address space (a second memory space) specific to the isolation container. Ex-1008 (Hunt) ¶¶ 0031, 0066. Hunt teaches that “[i]solation is enforced by a combination of virtual memory hardware and a highly restricted kernel API exposed by the isolation monitor 116.” Ex-1008 (Hunt) ¶ 0037.

380. A POSITA would have recognized Hunt’s isolation monitor to be a form of internal isolation firewall, as discussed above, and further would have understood based on this disclosure that the isolation monitor segregates the memory space for the isolation container (a second memory space) from the memory space for the rest of the application execution system (a first memory space), including the non-isolated applications. A POSITA would have used Hunt’s isolation monitor to segregate the memory space for the sandbox OE from the memory space for the typical OE of Benoit’s user/client computer device.

381. Accordingly, Benoit in combination with Hunt teaches element 8a.

[8b] wherein the processor is configured to enforce policies for communication between the first memory space and the second memory space using the internal isolation firewall.
--

382. Benoit in combination with Hunt discloses that **the processor is configured to enforce policies for communication between the first memory space and the second memory space using the internal isolation firewall**, and thus teaches element 8b. In Benoit, the policies for communication are the “pre-defined read (40) and write (50) permissions to the host system.” Ex-1005 (Benoit) ¶ 0043 & Fig. 3A.

383. In Hunt, the **policies for communication** are the “collection of rules that map the approval or denial of requests to access resources to an application manifest.” Ex-1008 (Hunt) ¶ 0029. A POSITA would have understood that such resources include **memory spaces**. Hunt describes “rules that confine untrusted applications,” teaches that “[a] collection of such rules forms an application firewall,” and explains that “[t]he rules map to approving or denying application manifest requests.” Ex-1008 (Hunt) ¶ 0053; *see also id.* ¶¶ 0055 (“[A]n application firewall can impose information flow rules, ruling out particular pipes, or specifying ALLOW or DENY lists of applications that may connect to a given pipe endpoint.”), 0094 (“an application firewall of rules that control interaction between the functionality and the kernel”).

384. Hunt's isolation monitor thus is an internal isolation firewall that enforces policies for communication between the memory space for the isolation container (a second memory space) and the memory space for the rest of the application execution system (a first memory space), including the non-isolated applications. A POSITA would have used Hunt's isolation monitor to enforce policies for communication between the memory space for the sandbox OE (a second memory space) and the memory space for the typical OE (a first memory space) of Benoit's user/client computer device.

385. Accordingly, Benoit in combination with Hunt teaches element 8b.

d. The Combination Would Not Have Changed the Principle of Operation for Either Benoit or Hunt

386. As discussed above, Benoit discloses a sandbox operating environment (OE) on the user/client computer device. Ex-1005 (Benoit) ¶ 0041 & Fig. 2. Hunt also discloses a sandboxed computing environment in the form of Hunt's isolation container. Ex-1008 (Hunt) ¶ 0024 & Fig. 1. Indeed, the title of Hunt is "Ultra-Low Cost Sandboxing for Application Appliances." Ex-1008 (Hunt), Title. Hunt teaches that the isolation container is subject to "rules that confine untrusted applications ('disallow this downloaded game from touching any of my data')" and is restricted in terms of the system resources it can access. Ex-1008 (Hunt) ¶ 0053.

387. As discussed above, a POSITA would have been motivated to combine the sandbox implementation details of Hunt, including the firewall and memory allocation processes, with the computer system of Benoit. Benoit's sandbox OE in the combination would continue to be an isolated computing environment. Indeed, the sandboxed computing environment taught in Hunt is implemented inside "an isolation container 102 in which an isolated application 104 (denoted Isolated App) runs in isolation from a non-isolated application 106 (denoted NON-ISO App)." Ex-1008 (Hunt) ¶ 0024.

388. The sandbox OE in the combination would continue to use a host operating system. In Hunt, the isolation container uses the host OS via the isolation monitor. In particular, Hunt teaches that "[t]o reduce overheads, the architecture takes basic computing services already provided by the host operating system, such as virtual memory and threads, and safely isolates these abstractions for use by the sandbox environment." Ex-1008 (Hunt) ¶ 0005; *see also id.* ¶ 0018 ("The disclosed architecture takes the abstractions provided by the host operating system and converts (refactors) these abstractions for use in and by the sandbox environment."). The interface to the host operating system services can be via a kernel interface implemented inside the isolation monitor. Ex-1008 (Hunt) ¶ 0095. Hunt thus teaches that the isolation container uses the host OS via the isolation monitor.

389. For the foregoing reasons, it is my opinion that a POSITA would have found claims 1-3, 8, 10, 11 and 17-19 of the '104 patent to be obvious based on the teachings of Benoit in view of Hunt.

E. Ground 5: Claims 1-3, 8, 10, 11 and 17-19 Are Obvious in View of Benoit, Hunt, and Pratt

390. It is my opinion that a POSITA would have found claims 1-3, 8, 10, 11 and 17-19 of the '104 patent to be obvious based on the teachings of Benoit in view of Hunt, and further in view of Pratt.

1. Combination of Benoit and Hunt

391. For the reasons discussed above in connection with Ground 4, it is my opinion that a POSITA would have been motivated to combine Hunt's sandbox implementation, including the firewall and memory allocation processes, with the computer system of Benoit, and that POSITA would have had a reasonable expectation of success in making the combination.

2. Combination of Benoit and Hunt, with Pratt

392. In my opinion, a POSITA would have been motivated to combine the authentication process of Pratt with the computer system of Benoit (adapted to include sandbox implementation of Hunt as discussed for Ground 4). As a preliminary matter, and as I explained above for element 1g in Ground 3, Benoit teaches authenticating the sandbox operating environment (OE) (an isolated computing environment) with an authentication server (an authentication device).

The combination of Benoit and Hunt thus renders obvious all of the challenged claims.

393. To exclude any doubt, however, I have been asked to opine on the validity of the challenged claims by further analyzing the combination of Benoit and Hunt in view of Pratt. As discussed above, Pratt provides an authentication process for an isolated computing environment like the sandbox OE of Benoit so that it can authenticate itself to a proxy or another network device that requires such authentication.

394. A POSITA would have been motivated to combine Benoit and Hunt with Pratt. As discussed earlier in this Declaration, all three references are directed to the field of network security.

395. In my opinion, the ability for Benoit's sandbox OE to authenticate itself would have motivated a POSITA to combine the authentication process of Pratt with the computer system of Benoit/Hunt. This combination would simply involve a POSITA combining prior-art elements according to known methods to yield a predictable result—the proxy or other network device determines that Benoit's sandbox OE is authentic and grants access.

396. A POSITA would have been further motivated to combine the authentication process of Pratt with the computer system of Benoit/Hunt because of the stated advantages of Pratt's process, discussed in connection with Ground 1.

Pratt teaches that “[a]dvantageously, operating environment 120 is able to authenticate itself to provider 150 without exposing ... any malicious code which may be inadvertently introduced or present within operating environment 120 to the authentication mechanism capable of authenticating to provider 150.” Ex-1006 (Pratt), 5:54-58. In my opinion, a POSITA would have been motivated by this stated advantage (allowing an untrusted operating environment like Benoit’s sandbox OE to authenticate itself without exposing any malicious code within the sandbox OE) to combine the authentication process of Pratt with the computer system of Benoit/Hunt.

397. Additionally, Pratt teaches that “[a]ny malicious code present in operating environment 120 are prevented from abusing access to such an authentication mechanism themselves or communicating the authentication mechanism to other malicious parties.” Ex-1006 (Pratt), 5:59-62. The benefit of preventing such abuse by malicious code present in an untrusted operating environment (like Benoit’s sandbox OE) would have further motivated a POSITA to combine the authentication process of Pratt with the computer system of Benoit/Hunt.

**b. A POSITA Would Have Had a Reasonable
Expectation of Success in Combining the
Authentication Process of Pratt with the
Computer System of Benoit/Hunt**

398. In my opinion, a POSITA would have had a reasonable expectation of success in combining the authentication process of Pratt with the computer system of Benoit (adapted to include the sandbox implementation details of Hunt as discussed for Ground 5), given the similarities in the teachings and systems of the three references, and given that technologies for authenticating computing environments were well known at the time. A POSITA would have expected to be successful in making the combination because the combination involves using conventional and well-known computer programming methods to yield a predictable result. Implementing an authentication process for an isolated computing environment, like the sandbox operating environment (OE) of Benoit, would not have required specialized skills beyond what a POSITA would have had.

399. Furthermore, a POSITA would have been motivated and would have had a reasonable expectation of success in combining Benoit, Hunt, and Pratt, as explained above, because the combination would not have changed the principle of operation for any of the teachings of the references. In fact, Benoit already teaches the authentication of the sandbox OE as discussed above for element 1g. Both Benoit's sandbox OE and Pratt's operating environment 120 are intended to protect

other parts of the computer device from malware and other malicious behavior, and authentication of these isolated computing environments prior to allowing communication with an Internet-based cloud service furthers this purpose. Rather than changing Benoit's principle of operation, the combination with Pratt would actually reinforce Benoit's principle of operation by using a known technique (the authentication process of Pratt) to achieve what Benoit already teaches (the authentication of the sandbox OE).

3. Claim Analysis

400. For this Ground 5, the element-by-element analysis of all of the challenged claims is the same as the element-by-element analysis for Grounds 3 and 4, except that Pratt provides additional disclosure for the limitation in elements 1g, 10e, and 18g of "authentica[ting] the isolated computing environment with an authentication device."

401. As discussed above, Pratt teaches a device having an isolated operating environment that may be securely authenticated to a service or resource provider. Ex-1006 (Pratt), 2:15-22 ("System 100 may be used to securely authenticate operating environment 120 to service or resource provider 150 ..."). Like the sandbox operating environment (OE) in Benoit, the operating environment 120 in Pratt is an isolated computing environment that can comprise a sandboxed computing environment and use a host operating system. Ex-1006

(Pratt), 2:29-36. And like Benoit's sandbox OE, Pratt's operating environment 120 is an operating environment in which malicious code may be inadvertently introduced. Ex-1006 (Pratt), 2:38-40.

402. I discuss the details of Pratt's authentication process in connection with Ground 1, above. To accomplish the combination of Pratt's authentication process with the Benoit/Hunt computer system, a POSITA would have configured the sandbox OE of Benoit to authenticate itself in a manner similar to the untrusted operating environment 120 of Pratt. Both the sandbox OE in Benoit and the untrusted operating environment 120 in Pratt are isolated, sandboxed computing environments, and the computer system of Benoit/Hunt would be adapted to include the interceptor module and authentication services of Pratt in an arrangement similar to what is shown in Figure 1 of Pratt.

403. Below is Figure 2 of Benoit, modified to show the interceptor module of Pratt positioned to intercept authentication messages originating from the remote computing resources or "cloud" or from the second cloud labeled "Internet." Benoit's sandbox OE is modified to include Pratt's authentication service 122, while Benoit's typical OE is modified to include Pratt's authentication service 132. The figure is also modified to show Hunt's isolation monitor positioned to provide the interface for communications between the host operating system and the typical OE, on the one hand, and the sandbox OE, on the other.

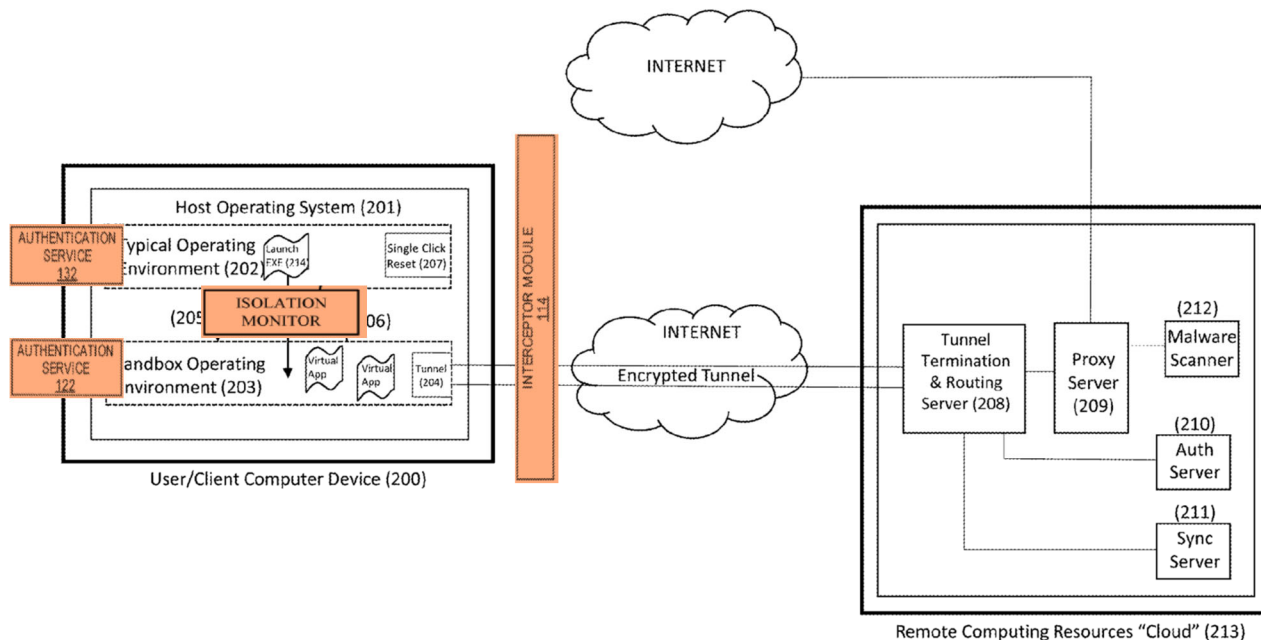


FIG. 2

404. As in Pratt, the interceptor module of the combined system intercepts the authentication messages and notifies the authentication service 132 in the typical OE, which then provides authentication data as appropriate to the sandbox OE so that it can authenticate itself to the requesting entity. As in Pratt, the sandbox OE can authenticate itself to the requesting entity without exposing any malicious code to the authentication service 132 in the typical OE.

405. The combination of Benoit, Hunt, and Pratt involves using conventional and well-known computer programming methods to yield a predictable result. The implementation of Pratt's authentication process in the computer system of Benoit/Hunt would not have required specialized skills beyond what a POSITA would have had.

406. The combination also would not have changed the principle of operation for any of the teachings of the references. As noted above for element 1g, Benoit already contemplates the authentication of the sandbox OE with an authentication server. It would have been straightforward for a POSITA to have added Pratt's interceptor module, which would monitor the sandbox OE's communications with the cloud and screen them for authentication requests.

407. For the foregoing reasons, it is my opinion that a POSITA would have found claims 1-3, 8, 10, 11 and 17-19 of the '104 patent to be obvious based on the teachings of Benoit in view of Hunt, and further in view of Pratt.

VI. BACKGROUND AND QUALIFICATIONS

408. This section contains a summary of my educational background, career history, publications, and other relevant qualifications. My full curriculum vitae is attached as Appendix 1 to this declaration.

409. I am a Regents' Professor and the John P. Imlay Jr. Chair in the College of Computing at the Georgia Institute of Technology. I have been a network security researcher since 1994. My research interests are systems and software security, network security, machine learning, and applied cryptography. I served as the Executive Director of the Institute for Information Security & Privacy (IISP) from 2015-2021 and as the Director of the Georgia Tech

Information Security Center (GTISC) from 2012-2015. I have graduated 31 Ph.D. students and have supervised 10 postdoctoral fellows.

410. Beyond my research activities, I have served on and chaired the program committees of many top security conferences, such as the IEEE Symposium on Security and Privacy and the ACM Conference on Computer and Communications Security. I have also served on the editorial boards for two leading journals: the IEEE Transactions on Dependable and Secure Computing (TDSC) from 2010 to 2011 and the ACM Transactions on Information and System Security (TISSEC) from 2005 to 2011.

411. I am a named inventor on twenty-one patents, eighteen of which are directed to detecting intrusions, malware, or responding to network attacks (as well as detecting them). I have also served as an expert and have provided testimony in multiple patent litigations. Consequently, I have a general familiarity with principles of patent law respecting anticipation and non-obviousness.

412. I have a Bachelor of Science degree in Computer Science from the Sun Yat-Sen University in Guangzhou, China, a Master of Science degree in Computer Science from The City College of New York, and a Ph.D. in Computer Science from Columbia University.

413. I was a software analyst and a senior software analyst at Intergraph Corp. from 1990 to 1994, and a summer research intern at AT&T Labs - Research

in 1996 and IBM T.J. Watson Research Center (1997). I was a tenure-track assistant professor in the Department of Computer Science at North Carolina State University from 1999 to 2001. I have been on the faculty of the Georgia Institute of Technology since 2001.

414. I have won many awards and honors for my research contributions, including a Fellow of the Association for Computing Machinery (ACM), a Fellow of the Institute of Electrical and Electronics Engineers (IEEE), the ACM Special Interest Group on Security, Audit, and Control (SIGSAC) Outstanding Innovation Award (2019), the National Science Foundation (NSF) Career Award (2002), and eight Best Paper and Test of Time awards from the top academic conferences in security.

415. I have taught undergraduate-level and graduate-level courses since 1999. These courses include Introduction to Information Security, Network Security, and Information Security Labs: System and Network Defenses. All of the relevant technologies of the '104 patent and the prior art references discussed above, including firewalls, intrusion detection, sandboxing, and virtual machines, are either taught in these courses or used in course projects.

416. I have published more than 200 papers in academic journals and conferences. According to Google Scholar, my h-index is 104, and the total number of citations of all my papers is more than 49,900. These papers cover most

of the technologies relevant to the '104 patent, including intrusion detection, malware analysis, sandboxing, operating system protection, and virtual machine monitoring.

VII. COMPENSATION

417. For my efforts in connection with the preparation of this declaration, I have been compensated at my standard rate for this type of consulting activity. My compensation is in no way contingent on the results of this proceeding or any other proceedings relating to the '104 patent.

VIII. MATERIALS AND OTHER INFORMATION CONSIDERED

418. I have considered information from various sources in forming my opinions. I have reviewed and considered each of the exhibits listed in the attached Appendix 2 (Materials Considered in the Preparation of This Declaration) in forming my opinions.

IX. UNDERSTANDING OF THE LAW

419. I am not an attorney. In forming my opinions in this Declaration, I applied the relevant legal principles provided to me by counsel, which are summarized in Appendix 4.

X. RESERVATION OF RIGHTS

420. My opinions are based upon the information that I have considered to date. I am unaware of any evidence of secondary considerations with respect to the '104 patent that would render any of the challenged claims non-obvious. I

reserve the right, however, to supplement my opinions in the future to respond to any arguments raised by the owner of the '104 patent and to take into account new information that becomes available to me.

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code.

Executed on December 29, 2024.

By:

DocuSigned by:

30EAC476E44B486...
Wenke Lee

APPENDIX 1: CURRICULUM VITAE OF WENKE LEE

Wenke Lee
Regents' Professor and John P. Imlay Jr. Chair
School of Cybersecurity and Privacy and School of Computer Science
College of Computing
Georgia Institute of Technology
Atlanta, Georgia 30332-0280
wenke@cc.gatech.edu
<http://www.cc.gatech.edu/~wenke/>

Contents

EDUCATIONAL BACKGROUND	3
EMPLOYMENT HISTORY	3
CURRENT FIELDS OF INTEREST	3
I. TEACHING	5
A. Courses Taught	5
B. Curriculum Development	8
C. Postdoctoral Fellows Supervised	9
D. Ph.D. Students Supervised	9
II. RESEARCH AND CREATIVE SCHOLARSHIP	12
A. Theses	12
B. Published Journal Papers	12
C. Published Books and Parts of Books	13
D. Edited Proceedings and Books	14
E. Conference Presentations (refereed)	14
E.1 with Proceedings (refereed)	14
F. Workshop Presentations (refereed)	30
F.1 with Proceedings	30
G. Other	31
H. Research Proposals and Grants	32
I. Honors and Awards	39
III. SERVICE	41
A. Professional Activities	41
A.1 Membership and Activities in Professional Societies	41
A.2 Journal Edit-oral Board Activities	41
A.3 Conference Committee Activities	41
B. On-campus Georgia Tech Committees	45
IV. NATIONAL AND INTERNATIONAL PROFESSIONAL RECOGNITION	47
A. Patents	47

V. OTHER CONTRIBUTIONS	49
A. Seminar Presentations	49

EDUCATIONAL BACKGROUND

Degree	Year	University	Field
Ph.D.	1999	Columbia University	Computer Science
M.S.	1990	The City College of New York	Computer Science
B.S.	1988	Sun Yat-Sen University, China	Computer Science

EMPLOYMENT HISTORY

Title	Organization	Years
Regents' Professor	The University System of Georgia	4/2024–present
John P. Imlay Jr. Chair	College of Computing	3/2016–present
Director	Institute for Information Security & Privacy (IISP)	7/2015–7/2021
	Georgia Institute of Technology	
Director	Georgia Tech Information Security Center (GTISC)	8/2012–6/2015
	Georgia Institute of Technology	
Professor	College of Computing	2/2009–present
	Georgia Institute of Technology	
Associate Professor	College of Computing	2/2005–2/2009
	Georgia Institute of Technology	
Assistant Professor	College of Computing	8/2001–2/2005
	Georgia Institute of Technology	
Assistant Professor	Department of Computer Science	8/1999–8/2001
	North Carolina State University	
Research Assistant	Department of Computer Science	1994–1999
	Columbia University	
Research Staff Member	IBM T.J. Watson Research Center	5/1997–8/1997
Member of Technical Staff	AT&T Research	5/1996–8/1996
Senior Software Analyst	Intergraph Corporation	1991–1994
Software Analyst	Intergraph Corporation	1990–1991

CURRENT FIELDS OF INTEREST

Information Security

Main research thrusts are: “Malware analysis and detection” – developing “live” and “safe” environment to uncover malware behaviors and machine-learning based techniques to automatically generate detection models and forensic analysis rules; “Adversarial Machine Learning” – studying the vulnerabilities of machine learning based systems due to adversarial manipulations and developing countermeasures; “Software debloating and protocol subsetting” – developing program and protocol analysis and rewriting techniques to remove unnecessary features and code from program binaries; “Privacy-preserving biometric-based authentication and search” – researching efficient methods to

protect the privacy of biometric data while enabling important security applications including remote authentication and video surveillance; “Adaptive analysis framework for advanced persistent threats” – developing game-theoretic approaches to model and analyze advanced persistent threats.

I. TEACHING

A. Courses Taught

<u>Semester/Year</u>	<u>Course</u>	<u>Number of Students</u>	<u>Comments</u>
Fall 2023	OMS CS/Cybersecurity 6262 Network Security	252	Online
Fall 2023	OMS CS/Cybersecurity 6035 Introduction to Information Security	772	Online
Fall 2023	CS 6264 Information Security Labs: System and Network Defenses	23	At CoC
Fall 2023	OMS CS/Cybersecurity 6264 Information Security Labs: System and Network Defenses	39	Online
Summer 2023	OMS CS/Cybersecurity 6262 Network Security	231	Online
Summer 2023	OMS CS/Cybersecurity 6035 Introduction to Information Security	395	Online
Summer 2023	OMS CS/Cybersecurity 6264 Information Security Labs: System and Network Defenses	29	Online
Spring 2023	OMS CS/Cybersecurity 6262 Network Security	295	Online
Spring 2023	CS 6262 Network Security	105	At CoC
Spring 2023	OMS CS/Cybersecurity 6035 Introduction to Information Security	710	Online
Fall 2022	CS 4235/6035 Introduction to Information Security	184	At CoC
Fall 2022	OMS CS/Cybersecurity 6262 Network Security	251	Online
Fall 2022	OMS CS/Cybersecurity 6035 Introduction to Information Security	622	Online
Fall 2022	OMS CS/Cybersecurity 6264 Information Security Labs: System and Network Defenses	18	Online
Summer 2022	OMS CS/Cybersecurity 6262 Network Security	258	Online
Summer 2022	OMS CS/Cybersecurity 6035 Introduction to Information Security	334	Online
Spring 2022	OMS CS/Cybersecurity 6262 Network Security	300	Online
Spring 2022	OMS CS/Cybersecurity 6035 Introduction to Information Security	635	Online
Spring 2022	OMS CS/Cybersecurity 6264 Information Security Labs: System and Network Defenses	17	Online
Fall 2021	CS 6262 Network Security	40	At CoC
Fall 2021	OMS CS/Cybersecurity 6262 Network Security	360	Online
Fall 2021	OMS CS/Cybersecurity 6035 Introduction to Information Security	980	Online
Fall 2021	OMS CS/Cybersecurity 6264 Information Security Labs: System and Network Defenses	17	Online
Summer 2021	OMS CS/Cybersecurity 6262 Network Security	200	Online
Summer 2021	OMS CS/Cybersecurity 6035 Introduction to Information Security	460	Online
Spring 2021	OMS CS/Cybersecurity 6264 Information Security Labs: System and Network Defenses	25	Online
Spring 2021	OMS CS/Cybersecurity 6262 Network Security	300	Online
Spring 2021	OMS CS/Cybersecurity 6035 Introduction to Information Security	800	Online

<u>Semester/Year</u>	<u>Course</u>	<u>Number of Students</u>	<u>Comments</u>
Fall 2020	CS 6262 Network Security	40	At CoC
Fall 2020	OMS CS/Cybersecurity 8803 Information Security Labs: System and Network Defenses	25	Online
Fall 2020	OMS CS/Cybersecurity 6262 Network Security	260	Online
Fall 2020	OMS CS/Cybersecurity 6035 Introduction to Information Security	850	Online
Summer 2020	OMS CS/Cybersecurity 8803 Information Security Labs: System and Network Defenses	20	Online
Summer 2020	OMS CS/Cybersecurity 6262 Network Security	150	Online
Summer 2020	OMS CS/Cybersecurity 6035 Introduction to Information Security	450	Online
Spring 2020	OMS CS/Cybersecurity 8803 Information Security Labs: System and Network Defenses	40	Online
Spring 2020	OMS CS/Cybersecurity 6262 Network Security	360	Online
Spring 2020	OMS CS/Cybersecurity 6035 Introduction to Information Security	884	Online
Fall 2019	CS 6262 Network Security	43	At CoC
Fall 2019	OMS CS/Cybersecurity 6262 Network Security	156	Online
Fall 2019	OMS CS/Cybersecurity 6035 Introduction to Information Security	770	Online
Summer 2019	OMS CS/Cybersecurity 6262 Network Security	129	Online
Summer 2019	OMS CS/Cybersecurity 6035 Introduction to Information Security	456	Online
Spring 2019	OMS CS/Cybersecurity 6262 Network Security	275	Online
Spring 2019	OMS CS/Cybersecurity 6035 Introduction to Information Security	938	Online
Fall 2018	CS 6262 Network Security	47	At CoC
Fall 2018	OMSCS 6262 Network Security	154	Online
Fall 2018	OMSCS 6035 Introduction to Information Security	525	Online
Summer 2018	OMSCS 6262 Network Security	191	Online
Summer 2018	OMSCS 6035 Introduction to Information Security	338	Online
Spring 2018	OMSCS 6262 Network Security	220	Online
Spring 2018	OMSCS 6035 Introduction to Information Security	428	Online
Fall 2017	CS 4235/6035 Introduction to Information Security	120	At CoC
Fall 2017	OMSCS 6262 Network Security	300	Online
Fall 2017	OMSCS 6035 Introduction to Information Security	350	Online
Summer 2017	OMSCS 6262 Network Security	300	Online
Summer 2017	OMSCS 6035 Introduction to Information Security	350	Online
Spring 2017	OMSCS 6262 Network Security	300	Online
Spring 2017	OMSCS 6035 Introduction to Information Security	350	Online

<u>Semester/Year</u>	<u>Course</u>	<u>Number of Students</u>	<u>Comments</u>
Fall 2016	CS 4235/6035 Introduction to Information Security	80	At CoC
Fall 2016	OMSCS 6262 Network Security	300	Online
Fall 2016	OMSCS 6035 Introduction to Information Security	350	Online
Summer 2016	OMSCS 6035 Introduction to Information Security	350	Online
Spring 2016	OMSCS 6035 Introduction to Information Security	350	Online
Fall 2015	CS 4235/6035 Introduction to Information Security	80	At CoC
Fall 2015	OMSCS 6035 Introduction to Information Security	230	Online
Fall 2014	CS 4235/6035 Introduction to Information Security	80	At CoC
Spring 2013	CS 4235 Introduction to Information Security	50	At CoC
Fall 2012	CS 4235 Introduction to Information Security	41	At CoC
Spring 2012	CS 6262 Network Security	35	At CoC
Fall 2010	CS 3210 Operating System Design	25	At CoC
Spring 2010	CS 6262 Network Security	45	At CoC
Fall 2009	CS 4237 Computer and Network Security	15	At CoC
Spring 2009	CS 4237 Computer and Network Security	15	At CoC
Fall 2008	CS 6262 Network Security	55	At CoC
Spring 2008	CS 8803ANS Advanced Systems and Network Security	25	At CoC
Fall 2007	CS 4237 Computer and Network Security	20	At CoC
Spring 2007	CS 6262 Network Security	45	At CoC
Fall 2006	CS 6265 Information Security Lab (w/ Mustaque)	18	At CoC
Fall 2006	CS 4803 Computer and Network Security	30	At CoC
Spring 2006	CS 6262 Network Security	45	At CoC
Fall 2005	CS 4803 Computer and Network Security	30	At CoC
Spring 2005	CS 6262 Network Security	30	At CoC
Fall 2004	CS 4803K Computer and Network Security	30	At CoC
Spring 2004	CS 6262 Network Security	30	At CoC
Fall 2003	CS 4803K Computer and Network Security	35	At CoC
Spring 2003	CS 8803K Advanced Systems and Network Security	16	At CoC
Fall 2002	CS 6262 Network Security	28	At CoC
Spring 2002	CS 6262 Network Security	30	New at CoC
Fall 2000	CSC 591Q Special Topics: Network Security	65	At NC State
Fall 1999	CSC 591Q Special Topics: Network Security	45	At NC State
Summer 1998	CS 3139 Data Structures and Algorithms	20	At Columbia

B. Curriculum Development

CS 6264/OMS CS/Cybersecurity 6264: Information Security Labs: System and Network Defenses. This graduate-level course helps students develop both in-depth knowledge and hands-on skills in a number of important cybersecurity areas, including software security, malware, and threat analysis, end-point security, network security, web security, mobile security, and machine learning-based security analytics. The lecture materials of each topic area are drawn from the latest research papers and prototypes, and a comprehensive project is designed to help students master each area. The lecture materials explain the design principles of cutting-edge security tools, and the projects are designed to let students extend these tools. Most of the tools are in the open-source, and therefore, students can continue to build on and use these tools beyond this course. Recorded videos for on-line offering as part of OMSCS as well as OMS Cybersecurity.

CS 6262/OMS CS/Cybersecurity 6262: Network Security. Graduate-level course in network security with topics including large-scale attacks and impacts, penetration testing and security assessments, security of Internet protocols (IP, TCP, DNS, and BGP), advanced web security, advanced malware analysis, advanced network monitoring, Internet-scale threat analysis, bitcoins and cryptocurrencies, big data and security, cloud security, and attack-tolerant systems. Completely revamped in 2016 with new course materials drawing from research papers and projects developed by my Ph.D. students. Recorded videos for on-line offering via Udacity as part of OMSCS as well as OMS Cybersecurity.

CS 4235/OMS CS/Cybersecurity 6035: Introduction to Information Security. Cross-listed undergraduate and graduate introductory course in information security. It teaches the basic concepts, principles, and fundamental approaches to secure computers and networks. Its main topics include: security basics, security management and risk assessment, software security, operating systems security, database security, cryptography algorithms and protocols, network authentication and secure network applications, malware, network threats and defenses, web security, mobile security, legal and ethical issues, and privacy. Completely revamped in 2015 with new materials covering the up-to-date threat models, attack methods, and new technologies and policy considerations. Recorded videos for on-line offering via Udacity as part of OMS CS as well as OMS Cybersecurity.

CS 6262: Network Security. Graduate-level course focusing on the fundamental principles as well as advanced techniques in network security. This course covers cryptography and its application to network and operating system security; authentication; security for electronic mail; Java security; Firewalls and intrusion detection.

CS 4237: Computer and Network Security. New undergraduate course (first offered as CS 4803 in Fall 2003) focusing on the fundamental principles and techniques in computer and network security.

CS 8803: Advanced Systems and Network Security. Graduate course on advanced topics such as network and host-based intrusion detection, botnet detection, software security, malware analysis, and virtual machine monitoring. Students are required to study research papers and work on research projects.

C. Postdoctoral Fellows Supervised

1. **Bo Feng:** 2022 - 2023; Tenure-track Assistant Professor, Zhejiang University, Hangzhou, Zhejiang, China
2. **Erkam Uzun:** 2021 - 2022; Software Engineer, ByteDance/TikTok
3. **Yisroel Mirsky:** 2019 - 2021; Tenure-track Lecturer in the Department of Software and Information Systems Engineering, Ben-Gurion University of the Negev, Israel
4. **Guangliang Yang:** 2019 - 2021; Tenure-track Assistant Professor in the Department of Computer Science, Fudan University, Shanghai, China
5. **Hong Hu:** 2017-2019; Tenure-track Assistant Professor in the College of Information Sciences and Technology, Pennsylvania State University
6. **Sangho Lee:** 2015-2018; Researcher in Microsoft Research (MSR) Redmond
7. **Tielei Wang:** 2011 - 2014; Start-up in China
8. **Simon Pak Ho Chung:** 2011 -2014; Research Scientist in SCS at Georgia Tech
9. **Roberto Perdisci:** 2009 - 2010; Professor with tenure in the Department of Computer Science, University of Georgia
10. **Daniel Xiapu Luo:** 2008 - 2010; Professor with tenure at Hong Kong Polytechnic University

D. Ph.D. Students Supervise

1. **Xinzhou Qin:** Graduated in Summer 2005; Senior Engineering Manager, VMware
2. **Yian Huang:** Graduated in Summer 2006; Senior Engineering Manager, LinkedIn
3. **Prahlad Fogla:** Graduated in Summer 2007; Software Engineer, Google
4. **Guofei Gu:** Graduated in Summer 2008; Professor with tenure in the Department of Computer Science, Texas A&M University
5. **Bryan Payne:** Graduated in Summer 2010; Director, AWS Vulnerability Management, Amazon Web Services (AWS)
6. **Monirul Sharif:** Graduated in Fall 2010; Director of Engineering, Google
7. **Kapil Singh:** Graduated in Summer 2011; Principal Research Scientist, IBM T.J. Watson Research Center
8. **Manos Antonakakis:** Graduated in Spring 2012; Associate Professor with tenure in ECE at Georgia Tech
9. **Martim Carbone:** Graduated in Summer 2012; GPU Software Engineer, Apple

10. **Junjie Zhang:** Graduated in Summer 2012; Associate Professor with tenure in the Department of Computer Science and Engineering, Wright State University
11. **Long Lu:** Graduated in Summer 2013; Chief Digital Safety & Security Officer, NIO; Associate Professor with tenure in the College of Computer and Information Science, Northeastern University
12. **Brendan Dolan-Gavitt:** Graduated in Summer 2014; Associate Professor with tenure in the Department of Computer Science and Engineering, NYU Tandon School of Engineering
13. **Yacin Nadji:** Graduated in Summer 2015; Senior Security Researcher, Corelight
14. **Xinyu Xing:** Graduated in Summer 2015; Associate Professor of Computer Science, Northwestern University
15. **Chengyu Song:** Graduated in Summer 2016; Associate Professor with tenure in the Department of Computer Science and Engineering, UC Riverside
16. **Byoungyoung Lee:** Graduated in Summer 2016; Associate Professor in the Department of Electrical and Computer Engineering, Seoul National University, South Korea
17. **Yeongjin Jang:** Graduated in Summer 2017; Principal Engineer, Software Vulnerability Research, Samsung Research America (SRA)
18. **Kangjie Lu:** Graduated in Summer 2017; Associate Professor with tenure in the Department of Computer Science and Engineering, University of Minnesota
19. **Wei Meng:** Graduated in Summer 2017; Associate Professor with tenure in the Department of Computer Science and Engineering, The Chinese University of Hong Kong
20. **Yizheng Chen:** Graduated in Fall 2017; Tenure-track Assistant Professor in the Department of Computer Science, The University of Maryland, College Park, Maryland
21. **Yang Ji:** Graduated in Fall 2019; Manager of Research, Palo Alto Networks
22. **Ruian Duan:** Graduated in Fall 2019; Staff Security Researcher, Palo Alto Networks
23. **Erkam Uzun:** Graduated in Summer 2021; Software Engineer, ByteDance/TikTok
24. **Chenxiong Qian:** Graduated in Summer 2021; Tenure-track Assistant Professor in the Department of Computer Science, The University of Hong Kong
25. **Carter Yagemann:** Graduated in Summer 2022; Tenure-track Assistant Professor in the Department of Computer Science and Engineering, The Ohio State University
26. **Joey Allen:** Graduated in Summer 2023; Senior Staff Researcher, Palo Alto Networks
27. **Moses Ike:** Graduated in Summer 2023; Senior Member of Technical Staff, Sandia National Laboratories
28. **Evan Downing:** Graduated in Fall 2023; Trail of Bits

29. **Feng Xiao:** Graduated in Fall 2023; Google
30. **Matthew Landen:** Graduated in Summer 2024; Applied Physics Lab, The Johns Hopkins University
31. **Yongheng Chen:** Graduated in Summer 2024; Google
32. **Haoran Wang:** In Progress
33. **Ding Zhang:** In Progress
34. **Zhang Yang:** In Progress
35. **Yupeng Yang:** In Progress
36. **Jizhou Chen:** In Progress
37. **Matthew Lau :** In Progress
38. **Shenglong Yao:** In Progress
39. **Leyan Pan:** In Progress

II. RESEARCH AND CREATIVE SCHOLARSHIP

A. Theses

Ph.D. Thesis

Title: A Data Mining Framework for Constructing Features and Models for Intrusion Detection Systems.

Date Completed: June 1999

Advisor: Professor Salvatore J. Stolfo.

University: Columbia University

B. Published Journal Papers (refereed)

1. Matthew Lau, Ismaila Seck, Athanasios P. Meliopoulos, Wenke Lee, and Eugène Ndiaye. Revisiting Non-separable Binary Classification and its Applications in Anomaly Detection. *Transactions on Machine Learning Research*. Vol 2024.
2. Dinuka Sahabandu, Shana Moothedath, Joey Allen, Linda Bushnell, Wenke Lee, and Radha Poovendran. RL-ARNE: A Reinforcement Learning Algorithm for Computing Average Reward Nash Equilibrium of Nonzero-Sum Stochastic Games. *IEEE Transactions on Automatic Control*. Vol 69(11), 2024.
3. Shana Moothedath, Dinuka Sahabandu, Joey Allen, Andrew Clark, Linda Bushnell, Wenke Lee, and Radha Poovendran. Dynamic Information Flow Tracking for Detection of Advanced Persistent Threats: A Stochastic Game Approach. *IEEE Transactions on Automatic Control*. Vol 69(10), 2024.
4. Shana Moothedath, Dinuka Sahabandu, Joey Allen, Linda Bushnell, Wenke Lee, and Radha Poovendran. Stochastic Dynamic Information Flow Tracking game using supervised learning for detecting advanced persistent threats. *Automatica*. Vol 159, 2024.
5. Yisroel Mirsky, Ambra Demontis, Jaidip Kotak, Ram Shankar, Deng Gelei, Liu Yang, Xiangyu Zhang, Maura Pintor, Wenke Lee, Yuval Elovici, and Battista Biggio. The Threat of Offensive AI to Organizations. *Computer Security*. Vol 124, 2023.
6. Yisroel Mirsky and Wenke Lee. The Creation and Detection of Deepfakes: A Survey. *ACM Computing Surveys*. 54(1), 2022.
7. Kangjie Lu, Meng Xu, Chengyu Song, Taesoo Kim, and Wenke Lee. Stopping Memory Disclosures via Diversification and Replicated Execution. *IEEE Transactions on Dependable and Secure Computing (TDSC)*. 18(1). 2021.
8. Shana Moothedath, Dinuka Sahabandu, Joey Allen, Andrew Clark, Linda Bushnell, Wenke Lee, and Radha Poovendran. A Game-Theoretic Approach for Dynamic Information Flow Tracking to Detect Multistage Advanced Persistent Threats. *IEEE Transactions on Automatic Control*. 65(12): 5248-5263, 2020.
9. Junjie Zhang, Roberto Perdisci, Wenke Lee, Unum Sarfraz, and Xiapu Luo. Building a Scalable System for Stealthy P2P-Botnet Detection. *IEEE Transactions on Information Forensics and Security*, January 2014.

10. Roberto Perdisci, Davide Ariu, Prahlad Fogla, Giorgio Giacinto, and Wenke Lee. McPAD: A Multiple Classifier System for Accurate Payload-Based Anomaly Detection. *Computer Networks*, Vol. 53, 2009.
11. Roberto Perdisci, Andrea Lanzi, and Wenke Lee. Classification of Packed Executables for Accurate Computer Virus Detection. *Pattern Recognition Letters* Vol. 29, No. 14 (October 2008).
12. Prahlad Fogla and Wenke Lee. q-Gram Matching Using Tree Models. *IEEE Transactions on Knowledge and Data Engineering*, Vol. 18, No. 4 (April 2006).
13. W. Fan, M. Miller, S. Stolfo, Wenke Lee, and P. Chan. Using Artificial Anomalies to Detect Unknown and Known Network Intrusions. *Knowledge and Information Systems*, Vol. 6, No. 5 (September 2004), Springer.
14. Yongguang Zhang, Wenke Lee, and Yian Huang. Intrusion Detection Techniques for Mobile Wireless Networks. *ACM/Kluwer Wireless Networks Journal (ACM WINET)*, 9(5), September 2003.
15. Joao B.D. Cabrera, Lundy Lewis, Xinzhou Qin, Wenke Lee, and Raman K. Mehra. Proactive Intrusion Detection and Distributed Denial of Service Attacks - A Case Study in Security Management. *Journal of Network and Systems Management*. 10(2), June 2002.
16. Wenke Lee, Wei Fan, Matt Miller, Sal Stolfo, and Erez Zadok. Toward Cost-Sensitive Modeling for Intrusion Detection and Response. *Journal of Computer Security*, 10(1,2), 2002.
17. Wenke Lee and Sal Stolfo. A Framework for Constructing Features and Models for Intrusion Detection Systems. *ACM Transactions on Information and System Security*, 3(4), November 2000.
18. Wenke Lee, Sal Stolfo, and Kui Mok. Adaptive Intrusion Detection: A Data Mining Approach. *Artificial Intelligence Review*, Kluwer Academic Publishers, 14(6):533-567, December 2000.
19. Wenke Lee and Gail E. Kaiser. Interfacing Oz with the PCTE OMS: A Case Study of Integrating a Legacy System with a Standard Object Management System. *Journal of Systems Integration*, Kluwer Academic Publishers, 9(4):329-358, December 1999.

C. Published Books and Parts of Books

1. Xinzhou Qin and Wenke Lee. Discovering Novel Attack Strategies from INFOSEC Alerts. *Data Warehousing and Data Mining Techniques for Cyber Security*. Anoop Singhal (eds), Springer, 2007.
2. Yongguang Zhang and Wenke Lee. Security in Mobile Ad-Hoc Networks. *Ad Hoc Networks: Technologies and Protocols*. P. Mohapatra and S. Krishnamurthy (eds), Springer, 2004.

3. Xinzhou Qin, Wenke Lee, Lundy Lewis, Joao B.D. Cabrera. Using MIB II Variables for Network Intrusion Detection. *Applications of Data Mining in Computer Security*. D. Barbara and S. Jajodia (eds), Kluwer Academic Publishers, May 2002.
4. Joao B.D. Cabrera, Lundy Lewis, Xinzhou Qin, Wenke Lee, Raman K. Mehra. Proactive Intrusion Detection - A Study on Temporal Data Mining. *Applications of Data Mining in Computer Security*. D. Barbara and S. Jajodia (eds), Kluwer Academic Publishers, May 2002
5. Wenke Lee, Sal Stolfo, and Kui Mok. Algorithms for Mining System Audit Data. *Data Mining, Rough Sets, and Granular Computing*. T. Y. Lin, Y. Y. Yao, and L. A. Zadeh (eds), Physica-Verlag, 2002
6. Wenke Lee and Naser Barghouti. Jadve: An Extensible Data Visualization Environment. In *Object-Oriented Applications Frameworks*, M. Fayad, D. Schmidt, and R. Johnson (eds), John Wiley & Sons, 1999.

D. Edited Proceedings and Books

1. *Botnet Detection: Countering the Largest Security Threat (Advances in Information Security)*, Wenke Lee, Cliff Wang, and David Dagon (Eds.), Springer, 2007.
2. *Proceedings of the 4th International Symposium on Recent Advances in Intrusion Detection (RAID 2001)*, Wenke Lee, Ludovic Me, and Andreas Wespi (Eds.), Lecture Notes in Computer Science, Vol. 2212, Springer, 2001.

E. Conference Presentations (refereed)

E.1. Conference Presentations with Proceedings

1. Feng Xiao, Zhongfu Su, Guangliang Yang, and Wenke Lee. Jasmine: Scale up JavaScript Static Security Analysis with Computation-based Semantic Explanation. In *Proceedings of the IEEE Symposium on Security and Privacy*. 2024.
2. Joey Allen, Zheng Yang, Feng Xiao, Matthew Landen, Roberto Perdisci, and Wenke Lee. WE-BRR: A Forensic System for Replaying and Investigating Web-Based Attacks in The Modern Web. In *Proceedings of the 33rd USENIX Security Symposium (USENIX)*. 2024.
3. Yupeng Yang, Yongheng Chen, Rui Zhong, Jizhou Chen, and Wenke Lee. Towards Generic Database Management System Fuzzing. In *Proceedings of the 33rd USENIX Security Symposium (USENIX)*. 2024.
4. Matthew Lau, Leyan Pan, Stefan Davidov, Athanasios P. Meliopoulos, and Wenke Lee. Geometric Implications of Classification on Reducing Open Space Risk. *Tiny Papers @ The Twelfth International Conference on Learning Representations (ICLR)*. 2024.
5. Matthew Lau, Fahad Alsaeed, Kayla Thames, Nano Suresettakul, Saman A. Zonouz, Wenke Lee, and Athanasios P. Meliopoulos. Physics-Assisted Explainable Anomaly Detection in Power Systems. In *Proceedings of the European Conference on Artificial Intelligence*. 2024.

6. Zheng Yang, Joey Allen, Matthew Landen, Roberto Perdisci, and Wenke Lee. TRIDENT: Towards Detecting and Mitigating Web-based Social Engineering Attacks. In *Proceedings of the 32nd USENIX Security Symposium (USENIX)*. Anaheim, California. 2023.
7. Yisroel Mirsky, George Macon, Michael Brown, Carter Yagemann, Matthew Pruett, Evan Downing, Sukarno Mertoguno, and Wenke Lee. VulChecker: Graph-based Vulnerability Localization in Source Code. In *Proceedings of the 32nd USENIX Security Symposium (USENIX)*. Anaheim, California. 2023.
8. Carter Yagemann, Simon Chung, Brendan Saltaformaggio, and Wenke Lee. PUMM: Preventing Use-After-Free Using Execution Unit Partitioning. In *Proceedings of the 32nd USENIX Security Symposium (USENIX)*. Anaheim, California. 2023.
9. Yongheng Chen, Rui Zhong, Yupeng Yang, Hong Hu, Dinghao Wu, and Wenke Lee. FUZZ: Redesign of Parallel Fuzzing using Microservice Architecture. In *Proceedings of the 32nd USENIX Security Symposium (USENIX)*. Anaheim, California. 2023.
10. Moses Ike, Kandy Phan, Keaton Sadoski, Romuald Valme, and Wenke Lee. SCAPHY: Detecting Modern ICS Attacks by Correlating Behaviors in SCADA and PHYsical. In *Proceedings of the 2023 IEEE Symposium on Security and Privacy*. San Francisco, California. 2023.
11. Matthew Landen, Keywhan Chung, Moses Ike, Sarah Mackay, Jean-Paul Watson, and Wenke Lee. DRAGON: Deep Reinforcement Learning for Autonomous Grid Operation and Attack Detection. In *Proceedings of the Annual Computer Security Applications Conference (ACSAC)*. 2022.
12. ChangSeok Oh, Sangho Lee, Chenxiong Qian, Hyungjoon Koo, and Wenke Lee. DeView: Confining Progressive Web Applications by Debloating Web APIs. In *Proceedings of the Annual Computer Security Applications Conference (ACSAC)*. 2022.
13. Feng Xiao, Zheng Yang, Joey Allen, Guangliang Yang, Grant Williams, and Wenke Lee. Understanding and Mitigating Remote Code Execution Vulnerabilities in Cross-platform Ecosystem. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 2022.
14. Carter Yagemann, Simon Chung, Brendan Saltaformaggio, and Wenke Lee. Automated Bug Hunting with Data-Driven Symbolic Root Cause Analysis. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 2021.
15. Carter Yagemann, Mohammad Nouredine, Wajih Hassan, Simon Chung, Adam Bates, and Wenke Lee. Validating the Integrity of Audit Logs Against Execution Repartitioning Attacks. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 2021.
16. Erkam Uzun, Simon P. Chung, Vladimir Kolesnikov, Alexandra Boldyreva, and Wenke Lee. Fuzzy Labeled Private Set Intersection with Applications to Private Real-Time Biometric Search. In *Proceedings of the 2021 USENIX Security Symposium*. 2021.

17. Evan Downing, Kyuhong Park, Yisroel Mirsky, and Wenke Lee. DeepReflect: Discovering Malicious Functionality through Binary Reconstruction. In *Proceedings of the 2021 USENIX Security Symposium*. 2021.
18. Carter Yagemann, Matthew Pruett, Simon P. Chung, Kennon Bittick, Brendan Saltaformaggio, and Wenke Lee. ARCUS: Symbolic Root Cause Analysis of Exploits in Production Systems. In *Proceedings of the 2021 USENIX Security Symposium*. 2021.
19. Feng Xiao, Jianwei Huang, Yichang Xiong, Guangliang Yang, Hong Hu, Guofei Gu, and Wenke Lee. Abusing Hidden Properties to Attack the Node.js Ecosystem. In *Proceedings of the 2021 USENIX Security Symposium*. 2021.
20. Kyuhong Park, Burak Sahin, Yongheng Chen, Jisheng Zhao, Evan Downing, Hong Hu, and Wenke Lee. Identifying Behavior Dispatchers for Malware Analysis. In *Proceedings of the 16th ACM ASIA Conference on Computer and Communications Security (ACM AsiaCCS 2021)*.
21. Erkam Uzun, Carter Yagemann, Simon P. Chung, Vladimir Kolesnikov, and Wenke Lee. Cryptographic Key Derivation from Biometric Inferences for Remote Authentication. In *Proceedings of the 16th ACM ASIA Conference on Computer and Communications Security (ACM AsiaCCS 2021)*.
22. Dongsong Yu, Guangliang Yang, Guozhu Meng, Xiaorui Gong, Xiu Zhang, Xiaobo Xiang, Xiaoyu Wang, Yue Jiang, Kai Chen, Wei Zou, Wenke Lee, and Wenchang Shi. SEPAL: Towards a Large-scale Analysis of SEAndroid Policy Customization. In *Proceedings of The Web Conference 2021 (WWW 2021)*.
23. Yongheng Chen, Rui Zhong, Hong Hu, Hangfan Zhang, Yupeng Yang, Dinghao Wu, and Wenke Lee. One Engine to Fuzz 'em All: Generic Language Processor Testing with Semantic Validation. In *Proceedings of the 41st IEEE Symposium on Security and Privacy (Oakland)*. 2021.
24. Ruian Duan, Omar Alrawi, Ranjita Pai Kasturi, Ryan Elder, Brendan Saltaformaggio, and Wenke Lee. Towards Measuring Supply Chain Attacks on Package Managers for Interpreted Languages. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*. 2021.
25. Kangjie Lu, Meng Xu, Chengyu Song, Taesoo Kim, and Wenke Lee. Stopping Memory Disclosures via Diversification and Replicated Execution. *IEEE Transactions on Dependable and Secure Computing (TDSC)*. 18(1), 2021.
26. Carter Yagemann, Simon P. Chung, Erkam Uzun, Sai Ragam, Brendan Saltaformaggio, and Wenke Lee. On the Feasibility of Automating Stock Market Manipulation. In *Proceedings of the Annual Computer Security Applications Conference (ACSAC)*. 2020.
27. Chenxiong Qian, Hyungjoon Koo, ChangSeok Oh, Taesoo Kim, and Wenke Lee. Slimium: Debloating the Chromium Browser with Feature Subsetting. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 2020.

28. Joey Allen, Zheng Yang, Matthew Landen, Raghav Bhat, Harsh Grover, Andrew Chang, Yang Ji, Roberto Perdisci, and Wenke Lee. Mnemosyne: An Effective and Efficient Postmortem Watering Hole Attack Investigation System. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 2020.
29. Rui Zhong, Yongheng Chen, Hong Hu, Hangfan Zhang, Wenke Lee, and Dinghao Wu. SQUIRREL: Testing Database Management Systems with Language Validity and Coverage Feedback. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 2020.
30. Dinuka Sahabandu, Joey Allen, Shana Moothedath, Linda Bushnell, Wenke Lee, and Radha Poovendran. Quickest Detection of Advanced Persistent Threats: A Semi-Markov Game Approach. In *Proceedings of the ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS)*. 2020.
31. D. Sahabandu, S. Moothedath, J. Allen, A. Clark, L. Bushnell, Wenke Lee, and R. Poovendran. Dynamic Information Flow Tracking Games for Simultaneous Detection of Multiple Attackers. In *Proceedings of the IEEE Conference on Decision and Control (CDC)*, Nice, France, December 2019.
32. S. Misra, S. Moothedath, H. Hosseini, J. Allen, L. Bushnell, Wenke Lee, and R. Poovendran. Learning Equilibria in Stochastic Information Flow Tracking Games with Partial Knowledge. In *Proceedings of the IEEE Conference on Decision and Control (CDC)*, Nice, France, December 2019.
33. Dinuka Sahabandu, Shana Moothedath, Joey Allen, Linda Bushnell, Wenke Lee, and Radha Poovendran. Stochastic Dynamic Information Flow Tracking Game with Reinforcement Learning. In *Proceedings of the 2019 Conference on Decision and Game Theory for Security*, Stockholm, Sweden, October 2019.
34. Chenxiong Qian, Hong Hu, Mansour Alharthi, Pak Ho Chung, Taesoo Kim, and Wenke Lee. RAZOR: A Framework for Post-deployment Software Debloating. In *Proceedings of the 28th USENIX Security Symposium*. Santa Clara, CA, August 2019.
35. Dinuka Sahabandu, Shana Moothedath, Linda Bushnell, Radha Poovendran, Joey Allen, Wenke Lee, and Andrew Clark. A Game Theoretic Approach for Dynamic Information Flow Tracking with Conditional Branching. In *Proceedings of the 2019 American Control Conference (ACC)*. Philadelphia, PA, July 2019.
36. Ruian Duan, Ashish Bijlani, Yang Ji, Omar Alrawi, Yiyuan Xiong, Moses Ike, Brendan Saltaformaggio, and Wenke Lee. Automating Patching of Vulnerable Open-Source Software Versions in Application Binaries. In *Proceedings of the 2019 Network and Distributed System Security Symposium (NDSS)*. San Diego, CA, February 2019.
37. Joey Allen, Matthew Landen, Sanya Chaba, Yang Ji, Simon Pak Ho Chung, and Wenke Lee. Improving Accuracy of Android Malware Detection with Lightweight Contextual Awareness. In *Proceedings of the 34th Annual Computer Security Applications Conference (ACSAC)*. December, 2018.

38. Dinuka Sahabandu, Baicen Xiao, Andrew Clark, Sangho Lee, Wenke Lee, and Radha Poovendran. DIFT Games: Dynamic Information Flow Tracking Games for Advanced Persistent Threats. In *Proceedings of The 57th IEEE Conference on Decision and Control (CDC)*. Miami Beach, FL, December 2018.
39. Shana Moothedath, Dinuka Sahabandu, Andrew Clark, Sangho Lee, Wenke Lee, and Radha Poovendran. Multi-Stage Dynamic Information Flow Tracking Game. In *Proceedings of The 9th Conference on Decision and Game Theory for Security (GameSec)*. Seattle, WA, October 2018.
40. Hong Hu, Chenxiong Qian, Carter Yagemann, Simon Pak Ho Chung, Bill Harris, Taesoo Kim, and Wenke Lee. Enforcing Unique Code Target Property for Control-Flow Integrity. In *Proceedings of The 25th ACM Conference on Computer and Communications Security (CCS 2018)*. Toronto, Canada, October 2018.
41. Andrea Possemato, Andrea Lanzi, Simon Pak Ho Chung, Wenke Lee, and Yanick Fratantonio. ClickShield: Are You Hiding Something? Towards Eradicating Clickjacking on Android. In *Proceedings of The 25th ACM Conference on Computer and Communications Security (CCS 2018)*. Toronto, Canada, October 2018.
42. Yang Ji, Sangho Lee, Mattia Fazzini, Joey Allen, Evan Downing, Taesoo Kim, Alessandro Orso, and Wenke Lee. Enabling Refinable Cross-Host Attack Investigation with Efficient Data Flow Tagging and Tracking. In *Proceedings of the 27th USENIX Security Symposium*. Baltimore, MD, August 2018
43. Wei Meng, Chenxiong Qian, Shuang Hao, Kevin Borgolte, Giovanni Vigna, and Christopher Kruegel, and Wenke Lee. Rampart: Protecting Web Applications from CPU-Exhaustion Denial-of-Service Attacks. In *Proceedings of the 27th USENIX Security Symposium*. Baltimore, MD, August 2018
44. Erkam Uzun, Simon Pak Ho Chung, Irfan Essa, and Wenke Lee. rtCaptcha: A Real-Time CAPTCHA Based Liveness Detection System. In *Proceedings of The 2018 Network and Distributed System Security Symposium (NDSS)*. San Diego, CA, February 2018.
45. Antonio Bianchi, Yanick Fratantonio, Aravind Machiry, Christopher Kruegel, Giovanni Vigna, Simon Pak Ho Chung, and Wenke Lee. Broken Fingers: On the Usage of the Fingerprint API in Android. In *Proceedings of the 2018 Network and Distributed System Security Symposium (NDSS)*. San Diego, CA, February 2018.
46. Ruian Duan, Ashish Bijlani, Meng Xu, Taesoo Kim, and Wenke Lee. Identifying Open-Source License Violation and 1-day Security Risk at Large Scale. In *Proceedings of The 24th ACM Conference on Computer and Communications Security (CCS 2017)*. Dallas, Texas, October 2017.
47. Yang Ji, Sangho Lee, Evan Downing, Weiren Wang, Mattia Fazzini, Taesoo Kim, Alessandro Orso, and Wenke Lee. RAIN: Refinable Attack Investigation with On-demand Inter-Process Information Flow Tracking. In *Proceedings of The 24th ACM Conference on Computer and Communications Security (CCS 2017)*. Dallas, Texas, October 2017.

48. Ren Ding, Chenxiong Qian, Chengyu Song, Bill Harris, Taesoo Kim, and Wenke Lee. Efficient Protection of Path-Sensitive Control Security. In *Proceedings of the 26th USENIX Security Symposium*. Vancouver, BC, Canada, August 2017.
49. Meng Xu, Kangjie Lu, Taesoo Kim, and Wenke Lee. Bunshin: Compositing Security Mechanisms through Diversification. In *Proceedings of the 2017 USENIX Annual Technical Conference*. Santa Clara, CA, July 2017.
50. Yanick Fratantonio, Chenxiong Qian, Pak Chung, and Wenke Lee. Cloak and Dagger: From Two Permissions to Complete Control of the UI Feedback Loop. In *Proceedings of The 2017 IEEE Symposium on Security and Privacy*. San Jose, CA, May 2017 (**Distinguished Practical Paper Award**).
51. Kangjie Lu, Marie-Therese Walter, David Pfaff, Stefan Nuernberger, Wenke Lee, and Michael Backes. Unleashing Use-Before-Initialization Vulnerabilities in the Linux Kernel Using Targeted Stack Spraying. In *Proceedings of The 2017 Network and Distributed System Security Symposium (NDSS)*. San Diego, CA, February 2017.
52. Le Guan, Jun Xu, Shuai Wang, Xinyu Xing, Lin Lin, Heqing Huang, Peng Liu, and Wenke Lee. From Physical to Cyber: Escalating Protection for Personalized Auto Insurance. In *Proceedings of The 14th ACM Conference on Embedded Networked Sensor Systems (SenSys 2016)*. Stanford, CA, November 2016.
53. Kangjie Lu, Chengyu Song, Taesoo Kim, and Wenke Lee. UniSan: Proactive Kernel Memory Initialization to Eliminate Data Leakages. In *Proceedings of The 23rd ACM Conference on Computer and Communications Security (CCS 2016)*. Vienna, Austria, October 2016.
54. Yizheng Chen, Panagiotis Kintis, Manos Antonakakis, Yacin Nadji, David Dagon, Wenke Lee, and Michael Farrell. Financial Lower Bounds of Online Advertising Abuse - A Four Year Case Study of the TDSS/TDL4 Botnet. In *Proceedings of The 13th Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA 2016)*. Sebastian, Spain, July 2016.
55. Chengyu Song, Hyungon Moon, Monjur Alam, Insu Yun, Byoungyoung Lee, Taesoo Kim, and Wenke Lee, Yunheung Paek. HDFI: Hardware-Assisted Data-Flow Isolation. In *Proceedings of The 2016 IEEE Symposium on Security and Privacy*. San Jose, CA, May 2016.
56. Wei Meng, Byoungyoung Lee, Xinyu Xing, and Wenke Lee. TrackMeOrNot: Enable Flexible Control on Web Tracking. In *Proceedings of The 25th International World Wide Web Conference (WWW), April 2016*. San Diego, CA, February 2016.
57. Wei Meng, Ren Ding, Simon P. Chung, Steven Han, and Wenke Lee. The Price of Free: Privacy Leakage in Personalized Mobile In-Apps Ads. In *Proceedings of The 2016 Network and Distributed System Security Symposium (NDSS)*. San Diego, CA, February 2016.
58. Kangjie Lu, Wenke Lee, Stefan Nrnberger, and Michael Backes. How to Make ASLR Win the Clone Wars: Runtime Re-Randomization. In *Proceedings of The 2016 Network and Distributed System Security Symposium (NDSS)*. San Diego, CA, February 2016.

59. Chengyu Song, Byoungyoung Lee, Kangjie Lu, William Harris, Taesoo Kim and Wenke Lee. Enforcing Kernel Security Invariants with Data Flow Integrity. In *Proceedings of The 2016 Network and Distributed System Security Symposium (NDSS)*. San Diego, CA, February, 2016.
60. Meng Xu, Yeongjin Jang, Xinyu Xing, Taesoo Kim, and Wenke Lee. UCognito: Private Browsing without Tears. In *Proceedings of The 22nd ACM Conference on Computer and Communications Security (CCS)*. Denver, CO, October 2015.
61. Kangjie Lu, Chengyu Song, Byoungyoung Lee, Simon P. Chung, Taesoo Kim, and Wenke Lee. ASLR-Guard: Stopping Address Space Leakage for Code Reuse Attacks. In *Proceedings of The 22nd ACM Conference on Computer and Communications Security (CCS)*. Denver, CO, October 2015.
62. Byoungyoung Lee, Chengyu Song, Taesoo Kim, and Wenke Lee. Type Casting Verification: Stopping an Emerging Attack Vector. In *Proceedings of The 24th USENIX Security Symposium*. Washington, D.C., August 2015. (Awarded the **Internet Defense Prize** by Facebook and USENIX)
63. Xinyu Xing, Wei Meng, Byoungyoung Lee, Udi Weinsberg, Anmol Sheth, Roberto Perdisci, and Wenke Lee. Unraveling the Relationship Between Ad-Injecting Browser Extensions and Malvertising. In *Proceedings of The 24th International World Wide Web Conference (WWW)*. Florence, Italy, May 2015.
64. Chengyu Song, Chao Zhang, Tielei Wang, Wenke Lee, and David Melski. Exploiting and Protecting Dynamic Code Generation. In *Proceedings of The 2015 Network and Distributed System Security Symposium (NDSS)*. San Diego, CA, February, 2015.
65. Byoungyoung Lee, Chengyu Song, Yeongjin Jang, Tielei Wang, Taesoo Kim, Long Lu, and Wenke Lee. Preventing Use-after-free with Dangling Pointers Nullification. In *Proceedings of The 2015 Network and Distributed System Security Symposium (NDSS)*. San Diego, CA, February, 2015.
66. Kangjie Lu, Zhichun Li, Vasileios P. Kemerlis, Zhenyu Wu, Long Lu, Cong Zheng, Zhiyun Qian, Wenke Lee, and Guofei Jiang. Checking More and Alerting Less: Detecting Privacy Leakages via Enhanced Data-flow Analysis and Peer Voting. In *Proceedings of The 2015 Network and Distributed System Security Symposium (NDSS)*. San Diego, CA, February, 2015.
67. Wei Meng, Xinyu Xing, Anmol Sheth, Udi Weinsberg, and Wenke Lee. Your Online Interests Pwned! A Pollution Attack Against Targeted Advertising. In *Proceedings of the 21st ACM Conference on Computer and Communications Security (CCS)*. Scottsdale, AZ, November 2014.
68. Yeongjin Jang, Chengyu Song, Simon Chung, Tielei Wang, and Wenke Lee. A11y Attacks: Exploiting Accessibility in Operating Systems. In *Proceedings of the 21st ACM Conference on Computer and Communications Security (CCS)*. Scottsdale, AZ, November 2014.

69. Billy Lau, Pak Ho Chung, Chengyu Song, Yeongjin Jang, Wenke Lee, and Alexandra Boldyreva. Mimesis Aegis: A Mimicry Privacy Shield - A System's Approach to Data Privacy on Public Cloud. In *Proceedings of the 23rd USENIX Security Symposium*. San Diego, CA, August 2014.
70. Tielei Wang, Yeongjin Jang, Yizheng Chen, Pak Ho Chung, Billy Lau, and Wenke Lee. On the Feasibility of Large-Scale Infections of iOS Devices. In *Proceedings of the 23rd USENIX Security Symposium*. San Diego, CA, August 2014.
71. Byoungyoung Lee, Long Lu, Tielei Wang, Taesoo Kim, and Wenke Lee. From Zygote to Morula: Fortifying Weakened ASLR on Android. In *Proceedings of the 35th IEEE Symposium on Security and Privacy (Oakland)*, San Jose, CA, May 2014.
72. Xinyu Xing, Wei Meng, Dan Doozan, Nick Feamster, Wenke Lee, and Alex C. Snoeren. Exposing Inconsistent Web Search Results with Bobble. In *Proceedings of The 2014 Passive and Active Measurement Conference (PAM)*, Los Angeles, CA, March 2014.
73. Yeongjin Jang, Simon P. Chung, Bryan D. Payne, and Wenke Lee. Gyrus: A Framework for User-Intent Monitoring of Text-Based Networked Applications. In *Proceedings of the 21st Annual Network and Distributed System Security Symposium (NDSS)*, San Diego, CA, February 2014.
74. Yacin Nadji, Manos Antonakakis, Roberto Perdisci, and Wenke Lee. Beheading Hydras: Performing Effective Botnet Takedowns. In *Proceedings of the 20th ACM Conference on Computer and Communications Security (CCS)*, Berlin, Germany, November 2013.
75. Brendan Dolan-Gavitt, Tim Leek, Josh Hodosh, and Wenke Lee. Tappan Zee (North) Bridge: Mining Memory Accesses for Introspection. In *Proceedings of the 20th ACM Conference on Computer and Communications Security (CCS)*, Berlin, Germany, November 2013.
76. Xinyu Xing, Wei Meng, Dan Doozan, Alex C. Snoeren, Nick Feamster, and Wenke Lee. Take this Personally: Pollution Attacks on Personalized Services. In *Proceedings of the 22nd USENIX Security Symposium*, Washington, D.C., August 2013.
77. Tielei Wang, Kangjie Lu, Long Lu, Simon Chung, and Wenke Lee. Jekyll on iOS: When Benign Apps Become Evil. In *Proceedings of the 22nd USENIX Security Symposium*, Washington, D.C., August 2013.
78. Yacin Nadji, Manos Antonakakis, Roberto Perdisci, and Wenke Lee. Connected Colors: Unveiling the Structure of Criminal Networks. In *Proceedings of the 16th International Symposium on Research in Attacks, Intrusions and Defenses (RAID)*, St. Lucia, October 2013 (to appear).
79. Junjie Zhang, Yinglian Xie, Fang Yu, David Soukal, and Wenke Lee. Intention and Origination: An Inside Look at Large-Scale Bot Queries. In *Proceedings of The 20th Annual Network and Distributed System Security Symposium (NDSS)*, San Diego, CA, February 2013.

80. Charles Lever, Manos Antonakakis, Bradley Reaves, Patrick Traynor and Wenke Lee. The Core of the Matter: Analyzing Malicious Traffic in Cellular Carriers. In *Proceedings of The 20th Annual Network and Distributed System Security Symposium (NDSS)*, San Diego, CA, February 2013.
81. Long Lu, Zhichun Li, Zhenyu Wu, Wenke Lee, and Guofei Jiang. CHEX: Statically Vetting Android Apps for Component Hijacking Vulnerabilities. In *Proceedings of the 19th ACM Conference on Computer and Communications Security (CCS)*, Raleigh, NC. October 2012.
82. Martim Carbone, Matthew Conover, Bruce Montague, and Wenke Lee. Secure and Robust Monitoring of Virtual Machines through Guest-Assisted Introspection. In *Proceedings of the 15th International Symposium on Research in Attacks, Intrusions, and Defenses (RAID)*. Amsterdam, The Netherlands. September, 2012.
83. Manos Antonakakis, Roberto Perdisci, Yacin Nadji, Nikolaos Vasiloglou, Saeed Abu-Nimeh, Wenke Lee, and David Dagon. From Throw-Away Traffic to Bots: Detecting the Rise of DGA-Based Malware. In *Proceedings of the 21st USENIX Security Symposium*. Bellevue, WA. August 2012.
84. Kapil Singh, Helen Wang, Alexander Moshchuk, Collin Jackson, and Wenke Lee. Practical End-to-End Web Content Integrity. In *Proceedings of the 21st International World Wide Web Conference (WWW)*, Lyon, France, April 2012.
85. Xiapu Luo, Peng Zhou, Junjie Zhang, Roberto Perdisci, Wenke Lee, and Rocky K.C. Chang. Exposing Invisible Timing-Based Traffic Watermarks with BACKLIT. In *Proceedings of The 27th Annual Computer Security Applications Conference (ACSAC 2011)*, Orlando, FL, December 2011.
86. Yacin Nadji, Manos Antonakakis, Roberto Perdisci, and Wenke Lee. Understanding the Prevalence and Use of Alternative Plans in Malware with Network Games. In *Proceedings of The 27th Annual Computer Security Applications Conference (ACSAC 2011)*, Orlando, FL, December 2011.
87. Long Lu, Roberto Perdisci, and Wenke Lee. SURF: Detecting and Measuring Search Poisoning. In *Proceedings of The 18th ACM Conference on Computer and Communications Security (CCS)*. Chicago, IL, October 2011.
88. Manos Antonakakis, Roberto Perdisci, Wenke Lee, Nick Vasiloglou, and David Dagon. Detecting Malware Domains at the Upper DNS Hierarchy. In *Proceedings of The 20th USENIX Security Symposium*, San Francisco, CA August 2011.
89. Junjie Zhang, Roberto Perdisci, Wenke Lee, Unam Sarfraz, and Xiapu Luo. Detecting Stealthy P2P Botnets Using Statistical Traffic Fingerprints. In *Proceedings of The 41st Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2011)*, Hong Kong, China, June 2011.
90. Xiapu Luo, Peng Zhou, Edmond W. W. Chan, Rocky K. C. Chang, and Wenke Lee. A Combinatorial Approach to Network Covert Communications with Applications in Web Leaks. In

91. Brendan Dolan-Gavitt, Tim Leek, Michael Zhivich, Jon Giffin, and Wenke Lee. Virtuoso: Narrowing the Semantic Gap in Virtual Machine Introspection. In *Proceedings of The 2010 IEEE Symposium on Security and Privacy*, Oakland, CA, May 2011.
92. Junjie Zhang, Jay Stokes, Christian Seifert, and Wenke Lee. ARROW: Generating Signatures to Detect Drive-By Downloads. In *Proceedings of The 20th International World Wide Web Conference (WWW)*, Hyderabad, India, March 2011.
93. Junjie Zhang, Xiapu Luo, Roberto Perdisci, Guofei Gu, Wenke Lee, and Nick Feamster. Boosting the Scalability of Botnet Detection Using Adaptive Traffic Sampling. In *Proceedings of The 6th ACM Symposium on Information, Computer and Communications Security (ASIACCS)*, Hong Kong, March 2011.
94. Xiapu Luo, Peng Zhou, Edmond W.W. Chan, Wenke Lee, Rocky K. C. Chang, and Roberto Perdisci. HTTPoS: Sealing Information Leaks with Browser-side Obfuscation of Encrypted Flows. In *Proceedings of The 18th Annual Network and Distributed System Security Symposium (NDSS)*, San Diego, CA, February 2011.
95. Qing Hui, Xiapu Luo, and Wenke Lee. Control of Low-Rate Denial-of-Service Attacks on Web Servers and TCP Flows. In *Proceedings of The 49th IEEE Conference on Decision and Control (CDC)*, Atlanta, GA, December 2010.
96. Long Lu, Vinod Yegneswaran, Phil Porras, and Wenke Lee. BLADE: An Attack-Agnostic Approach for Preventing Drive-By Malware Infections. In *Proceedings of The 17th ACM Conference on Computer and Communications Security (CCS)*, Chicago, IL, October 2010.
97. Xiapu Luo, Junjie Zhang, Roberto Perdisci, and Wenke Lee. On the Secrecy of Spread-Spectrum Flow Watermarks. In *Proceedings of The 15th European Symposium on Research in Computer Security (ESORICS)*, Athens, Greece, September 2010.
98. Manos Antonakakis, David Dagon, Xiapu Luo, Roberto Perdisci, and Wenke Lee. A Centralized Monitoring Infrastructure for Improving DNS Security. In *Proceedings of The 13th International Symposium on Recent Advances in Intrusion Detection (RAID)*, Ottawa, Ontario, Canada, September 2010.
99. Manos Antonakakis, Roberto Perdisci, David Dagon, Wenke Lee, and Nick Feamster. Building a Dynamic Reputation System for DNS. In *Proceedings of The 19th USENIX Security Symposium*, Washington, DC, August 2010.
100. Kapil Singh, Samrit Sangal, Nehil Jain, Patrick Traynor, and Wenke Lee. Evaluating Bluetooth as a Medium for Botnet Command and Control. In *Proceedings of The 7th Conference on Detection of Intrusions and Malware Vulnerability Assessment (DIMVA)*, Bonn, Germany, July 2010.

101. Kapil Singh, Alexander Moshchuk, Helen J. Wang, and Wenke Lee. On the Incoherencies in Web Browser Access Control Policies. In *Proceedings of The 2010 IEEE Symposium on Security and Privacy*, Oakland, CA, May 2010.
102. Roberto Perdisci, Wenke Lee, and Nick Feamster. Behavioral Clustering of HTTP-based Malware and Signature Generation using Malicious Network Traces. In *Proceedings of The 7th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, San Jose, CA, April 2010.
103. Roberto Perdisci, Iginio Corona, David Dagon, and Wenke Lee. Detecting Malicious Flux Service Networks through Passive Analysis of Recursive DNS Traces. In *Proceedings of The 25th Annual Computer Security Applications Conference (ACSAC 2009)*, Honolulu, HI, December 2009.
104. Guofei Gu, Vinod Yegneswaran, Phillip Porras, Jennifer Stoll, and Wenke Lee. Active Botnet Probing to Identify Obscure Command and Control Channels. In *Proceedings of The 25th Annual Computer Security Applications Conference (ACSAC 2009)*, Honolulu, HI, December 2009.
105. Monirul Sharif, Wenke Lee, Weidong Cui, and Andrea LANZI. Secure In-VM Monitoring Using Hardware Virtualization. In *Proceedings of The 16th ACM Conference on Computer and Communications Security (CCS 2009)*, Chicago, IL, November, 2009.
106. Martim Carbone, Weidong Cui, Long Lu, Wenke Lee, Marcus Peinado, and Xuxian Jiang. Mapping Kernel Objects to Enable Systematic Integrity Checking. In *Proceedings of The 16th ACM Conference on Computer and Communications Security (CCS 2009)*, Chicago, IL, November, 2009.
107. Kapil Singh, Sumeer Bhola, and Wenke Lee. xBook: Redesigning Privacy Control in Social Networking Platforms. In *Proceedings of The 18th USENIX Security Symposium*, Montreal, Canada, August, 2009.
108. Roberto Perdisci, Manos Antonakakis, Xiapu Luo, and Wenke Lee. WSEC DNS: Protecting Recursive DNS Resolvers from Poisoning Attacks. In *Proceedings of The 39th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2009)*, Lisbon, Portugal, June 2009.
109. Monirul Sharif, Andrea LANZI, Jon Giffin, and Wenke Lee. Automatic Reverse Engineering of Malware Emulators. In *Proceedings of The 2009 IEEE Symposium on Security and Privacy*, Oakland, CA, May 2009. **(Best Student Paper Award)**
110. Andrea LANZI, Monirul Sharif, and Wenke Lee. K-Tracer: A System for Extracting Kernel Malware Behavior. In *Proceedings of The 16th Annual Network and Distributed System Security Symposium (NDSS 2009)*, San Diego, CA, February 2009.
111. David Dagon, Manos Antonakakis, Kevin Day, Xiapu Luo, Christopher P. Lee, and Wenke Lee. Recursive DNS Architectures and Vulnerability Implications. In *Proceedings of The 16th Annual Network and Distributed System Security Symposium (NDSS 2009)*, San Diego, CA, February 2009.

112. Roberto Perdisci, Andrea Lanzi, and Wenke Lee. McBoost: Boosting Scalability in Malware Collection and Analysis Using Statistical Classification of Executables. In *Proceedings of The 24th Annual Computer Security Applications Conference (ACSAC 2008)*, Anaheim, CA, December 2008.
113. Artem Dinaburg, Paul Royal, Monirul Sharif, and Wenke Lee. Ether: Malware Analysis via Hardware Virtualization Extensions. In *Proceedings of the 15th ACM Conference on Computer and Communications Security (CCS 2008)*, Alexandria, VA, October 2008.
114. David Dagon, Manos Antonakakis, Paul Vixie, Tatuya Jinmei, and Wenke Lee. Increased DNS Forgery Resistance Through 0x20-Bit Encoding. In *Proceedings of the 15th ACM Conference on Computer and Communications Security (CCS 2008)*, Alexandria, VA, October 2008.
115. Monirul Sharif, Vinod Yegneswaran, Hassen Saidi, Phillip Porras, and Wenke Lee. Eureka: A Framework for Enabling Static Malware Analysis. In *Proceedings of the 13th European Symposium on Research in Computer Security (ESORICS)*, Malaga, Spain, October 2008.
116. Guofei Gu, Roberto Perdisci, Junjie Zhang, and Wenke Lee. BotMiner: Clustering Analysis of Network Traffic for Protocol- and Structure-Independent Botnet Detection. In *Proceedings of The 17th USENIX Security Symposium (Security'08)*, San Jose, CA, July 2008.
117. Kapil Singh, Abhinav Srivastava, Jon Giffin, and Wenke Lee. Evaluating Email's Feasibility for Botnet Command and Control. In *Proceedings of the 38th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2008)*, Anchorage, Alaska, June 2008.
118. Bryan D. Payne, Martim Carbone, Monirul Sharif, and Wenke Lee. Lares: An Architecture for Secure Active Monitoring Using Virtualization. In *Proceedings of the 2008 IEEE Symposium on Security and Privacy*, Oakland, CA, May 2008.
119. Guofei Gu, Alvaro A. Cardenas, and Wenke Lee. Principled Reasoning and Practical Applications of Alert Fusion in Intrusion Detection Systems. In *Proceedings of the ACM Symposium on InformAction, Computer and Communications Security (ASIACCS'08)*, Tokyo, Japan, March 2008.
120. David Dagon, Niels Provos, Chris Lee, and Wenke Lee. Corrupted DNS Resolution Paths: The Rise of a Malicious Resolution Authority. In *Proceedings of The 15th Annual Network and Distributed System Security Symposium (NDSS 2008)*, San Diego, CA, February 2008.
121. Guofei Gu, Junjie Zhang, and Wenke Lee. BotSniffer: Detecting Botnet Command and Control Channels in Network Traffic. In *Proceedings of The 15th Annual Network and Distributed System Security Symposium (NDSS 2008)*, San Diego, CA, February 2008.
122. Monirul Sharif, Andrea Lanzi, Jonathon Giffin, and Wenke Lee. Impeding Malware Analysis using Conditional Code Obfuscation. In *Proceedings of The 15th Annual Network and Distributed System Security Symposium (NDSS 2008)*, San Diego, CA, February 2008.

123. Bryan D. Payne, Martim Carbone, and Wenke Lee. Secure and Flexible Monitoring of Virtual Machines. In *Proceedings of The 23rd Annual Computer Security Applications Conference (ACSAC 2007)*, Miami Beach, FL, December 2007.
124. David Dagon, Guofei Gu, Chris Lee and Wenke Lee. A Taxonomy of Botnet Structures. In *Proceedings of The 23rd Annual Computer Security Applications Conference (ACSAC 2007)*, Miami Beach, FL, December 2007.
125. Guofei Gu, Zesheng Chen, Phillip Porras and Wenke Lee. Misleading and Defeating Importance-Scanning Malware Propagation. In *Proceedings of The 3rd International Conference on Security and Privacy in Communication Networks (SecureComm'07)*, Nice, France, September 2007.
126. Takehiro Takahashi and Wenke Lee. An Assessment of VoIP Covert Channel Threats. In *Proceedings of The 3rd International Conference on Security and Privacy in Communication Networks (SecureComm'07)*, Nice, France, September 2007.
127. Monirul Sharif, Kapil Singh, Jonathon Giffin and Wenke Lee. Understanding Precision in Host Based Intrusion Detection: Formal Analysis and Practical Models. In *Proceedings of The 10th International Symposium on Recent Advances in Intrusion Detection (RAID)*, Surfers Paradise, Australia, September 2007.
128. Guofei Gu, Phillip Porras, Vinod Yegneswaran, Martin Fong, Wenke Lee. BotHunter: Detecting Malware Infection Through IDS-Driven Dialog Correlation. In *Proceedings of The 16th USENIX Security Symposium (Security'07)*, Boston, MA, August 2007.
129. David Cash, Yan Zong Ding, Yevgeniy Dodis, Wenke Lee, Richard Lipton, and Shabsi Wal-fish. Intrusion-Resilient Key Exchange in the Bounded Retrieval Model. In *Proceedings of The Fourth IACR Theory of Cryptography Conference (TCC 2007)*, Amsterdam, The Netherlands, February 2007.
130. Roberto Perdisci, Guofei Gu, and Wenke Lee. Using an Ensemble of One-Class SVM Classifiers to Harden Payload-based Anomaly Detection Systems. In *Proceedings of The 2006 IEEE International Conference on Data Mining (ICDM '06)*, Hong Kong, China, December 2006.
131. Paul Royal, Mitch Halpin, David Dagon, Robert Edmonds, and Wenke Lee. PolyUnpack: Automating the Hidden-Code Extraction of Unpack-Executing Malware. In *Proceedings of The 22th Annual Computer Security Applications Conference (ACSAC 2006)*, Miami Beach, FL, December 2006.
132. Prahlad Fogla and Wenke Lee. Evading Network Anomaly Detection Systems: Formal Reasoning and Practical Techniques. In *Proceedings of The 13th ACM Conference on Computer and Communications Security (CCS 2006)*, Alexandria, VA, October 2006.
133. Guofei Gu, Prahlad Fogla, David Dagon, Wenke Lee, and Boris Skoric. Towards an Information-Theoretic Framework for Analyzing Intrusion Detection Systems. In *Proceedings of The 11th European Symposium Research Computer Security (ESORICS 2006)*, Hamburg, Germany, September 2006.

134. Prahlad Fogla, Monirul Sharif, Roberto Perdisci, Oleg Kolesnikov, and Wenke Lee. Polymorphic Blending Attacks. In *Proceedings of The 15th USENIX Security Symposium (SECURITY '06)*, Vancouver, B.C., Canada, August 2006.
135. Collin Mulliner, Giovanni Vigna, David Dagon, and Wenke Lee. Using Labeling to Prevent Cross-Service Attacks Against Smart Phones. In *Proceedings of The 3rd Conference on Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA 2006)*, Berlin, Germany, July 2006.
136. Hongmei Deng, Roger Xu, Jason H. Li, Frank Zhang, Renato Levy, and Wenke Lee. Agent-Based Cooperative Anomaly Detection for Wireless Ad Hoc Networks. In *Proceedings of The 12th International Conference on Parallel and Distributed Systems (ICPADS 2006)*, Minneapolis, Minnesota, July 2006.
137. Guofei Gu, Prahlad Fogla, Wenke Lee, and Douglas Blough. DSO: Dependable Signing Overlay. In *Proceedings of The 4th International Conference on Applied Cryptography and Network Security (ACNS '06)*, Singapore, June 2006.
138. Roberto Perdisci, David Dagon, Wenke Lee, Prahlad Fogla, and Monirul Sharif. Misleading Worm Signature Generators Using Deliberate Noise Injection (full paper). In *Proceedings of the 2006 IEEE Symposium on Security and Privacy*, Oakland, CA, May 2006.
139. Guofei Gu, Prahlad Fogla, David Dagon, Wenke Lee, and Boris Skoric. Measuring Intrusion Detection Capability: An Information-Theoretic Approach. In *Proceedings of ACM Symposium on InformAction, Computer and Communications Security (ASIACCS '06)*, Taipei, Taiwan, March 2006.
140. David Dagon, Cliff Zou, and Wenke Lee. Modeling Botnet Propagation Using Time Zones. In *Proceedings of The 13th Annual Network and Distributed System Security Symposium (NDSS 2006)*, San Diego, CA, February 2006.
141. Yongguang Zhang, Yi-an Huang, and Wenke Lee. An Extensible Environment for Evaluating Secure MANET. In *Proceedings of The 1st International Conference on Security and Privacy for Emerging Areas in Communication Networks (SecureComm 2005)*, Athens, Greece, September 2005.
142. Jonathon T. Giffin, David Dagon, Somesh Jha, Wenke Lee, and Barton P. Miller. Environment-Sensitive Intrusion Detection. In *Proceedings of The 8th International Symposium on Recent Advances in Intrusion Detection (RAID 2005)*, Seattle, WA, September 2005.
143. David Dagon, Wenke Lee, and Richard Lipton. Protecting Secret Data from Insider Attacks. *Proceedings of The Ninth International Conference on Financial Cryptography and Data Security (FC'05)*, Roseau, Dominica, February, 2005.
144. Guofei Gu, David Dagon, Xinzhou Qin, Monirul I. Sharif, Wenke Lee, and George F. Riley. Worm Detection, Early Warning, and Response Based on Local Victim Information. *Proceedings of the 20th Annual Computer Security Applications Conference (ACSAC 2004)*, Tucson, Arizona, December 2004.

145. Xinzhou Qin and Wenke Lee. Attack Plan Recognition and Prediction Using Causal Networks. *Proceedings of the 20th Annual Computer Security Applications Conference (ACSAC 2004)*, Tucson, Arizona, December 2004.
146. Joao B.D. Cabrera, Jaykumar Gosar, Wenke Lee, and Raman K. Mehra. On the Statistical Distribution of Processing Times in Network Intrusion Detection. *Proceedings of the 43rd IEEE Conference on Decision and Control (CDC 2004)*, Bahamas, December 2004.
147. George F. Riley, Monirul I. Sharif, and Wenke Lee. Simulating Internet Worms. *Proceedings of the 12th Annual Meeting of the IEEE/ACM International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*, Volendam, The Netherlands, October 2004.
148. Yian Huang and Wenke Lee. Attack Analysis and Detection for Ad Hoc Routing Protocols. *Proceedings of the 7th International Symposium on Recent Advances in Intrusion Detection (RAID 2004)*, Sophia Antipolis, France, September 2004.
149. David Dagon, Xinzhou Qin, Guofei Gu, Wenke Lee, Julian Grizzard, John Levin, and Henry Owen. HoneyStat: Local Worm Detection Using Honeypots. *Proceedings of the 7th International Symposium on Recent Advances in Intrusion Detection (RAID 2004)*, Sophia Antipolis, France, September 2004.
150. Xinzhou Qin and Wenke Lee. Discovering Novel Attack Strategies from INFOSEC Alerts. *Proceedings of the 9th European Symposium on Research in Computer Security (ESORICS 2004)*, Sophia Antipolis, France, September 2004.
151. H. Feng, Jonathon T. Giffin, Yong Huang, Somesh Jha, Wenke Lee, and Barton P. Miller. Formalizing Sensitivity in Static Analysis for Intrusion Detection. *Proceedings of the 2004 IEEE Symposium on Security and Privacy*, Oakland, CA, May 2004.
152. Xinzhou Qin and Wenke Lee. Statistical Causality Analysis of INFOSEC Alert Data. In *Proceedings of the 6th International Symposium on Recent Advances in Intrusion Detection (RAID 2003)*, Pittsburgh, PA, September 2003.
153. H. Feng, Oleg Kolesnikov, Prahlad Fogla, Wenke Lee, and Weibo Gong. Anomaly Detection Using Call Stack Information. In *Proceedings of the 2003 IEEE Symposium on Security and Privacy*, Oakland, CA, May 2003.
154. Yi-an Huang, Wei Fan, Wenke Lee, and Philip S. Yu. Cross-Feature Analysis for Detecting Ad-Hoc Routing Anomalies. In *Proceedings of the 23rd International Conference on Distributed Computing Systems (ICDCS)*, Providence, RI, May 2003.
155. Joao B.D. Cabrera, Wenke Lee, R.K. Prasanth, Lundy Lewis, and Raman K. Mehra. Optimization and Control Problems in Real Time Intrusion Detection. *Proceedings of the 41st IEEE Conference on Decision and Control (CDC 2002)*, Las Vegas, December, 2002.
156. Wenke Lee, Joao B.D. Cabrera, Ashley Thomas, Niranjana Balwalli, Sunmeet Saluja, and Yi Zhang. Performance Adaptation in Real-Time Intrusion Detection Systems. In *Proceedings*

of the 5th International Symposium on Recent Advances in Intrusion Detection (RAID 2002), Zurich, Switzerland, October 2002.

157. Xinzhou Qin, Wenke Lee, Lundy Lewis, and Joao B.D. Cabrera. Integrating Intrusion Detection and Network Management. *Proceedings of the IEEE/IFIP Network Operations and Management Symposium (NOMS 2002)*, Florence, Italy, May 2002.
158. Wei Fan, Matt Miller, Sal Stolfo, Wenke Lee, and Phil Chan. Using Artificial Anomalies to Detect Unknown and Known Network Intrusions. *Proceedings of The First IEEE International Conference on Data Mining (ICDM)*, San Jose, CA, November 2001.
159. Wenke Lee, Sal Stolfo, Phil Chan, Eleazar Eskin, Wei Fan, Matt Miller, Shlomo Hershkop, and Junxin Zhang. Real Time Data Mining-based Intrusion Detection. *Proceedings of the 2001 DARPA Information Survivability Conference and Exposition (DISCEX II)*, Anaheim, CA, June 2001.
160. Wenke Lee and Dong Xiang. Information-Theoretic Measures for Anomaly Detection. *Proceedings of The 2001 IEEE Symposium on Security and Privacy*, Oakland, CA, May 2001.
161. Joao B.D. Cabrera, L. Lewis, X. Qin, Wenke Lee, Ravi Prasanth, B. Ravichandran, and Raman Mehra. Proactive Detection of Distributed Denial of Service Attacks Using MIB Traffic Variables - A Feasibility Study. *Proceedings of The Seventh IFIP/IEEE International Symposium on Integrated Network Management (IM 2001)*, Seattle, WA, May 2001.
162. Yongguang Zhang and Wenke Lee. Intrusion Detection in Wireless Ad-Hoc Networks. *Proceedings of The Sixth International Conference on Mobile Computing and Networking (MobiCom 2000)*, Boston, MA, August 2000.
163. Wei Fan, Wenke Lee, Sal Stolfo, and Matt Miller. A Multiple Model Cost-Sensitive Approach for Intrusion Detection. *Proceedings of The Eleventh European Conference on Machine Learning (ECML 2000)*, LNAI 1810, Barcelona, Spain, May 2000.
164. Sal Stolfo, Wei Fan, Wenke Lee, Andreas Prodromidis, and Phil Chan. Cost-based Modeling for Fraud and Intrusion Detection: Results from the JAM Project. *Proceedings of the 2000 DARPA Information Survivability Conference and Exposition (DISCEX '00)*, Hilton Head, SC, January 2000.
165. Wenke Lee, Sal Stolfo, and Kui Mok. Mining in a Data-flow Environment: Experience in Network Intrusion Detection. *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '99)*, San Diego, CA, August 1999.
166. Wenke Lee, Sal Stolfo, and Kui Mok. A Data Mining Framework for Building Intrusion Detection Models. *Proceedings of the 1999 IEEE Symposium on Security and Privacy*, Oakland, CA, May 1999.
167. Wenke Lee, Sal Stolfo, and Kui Mok. Mining Audit Data to Build Intrusion Detection Models. *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD '98)*, New York, NY, August 1998.

168. Wenke Lee and Sal Stolfo. Data Mining Approaches for Intrusion Detection. *Proceedings of the Seventh USENIX Security Symposium (SECURITY '98)*, San Antonio, TX, January 1998.
169. Sal Stolfo, Andreas Prodromidis, Shelley Tselepis, Wenke Lee, Wei Fan, and Phil Chan. JAM: Java Agents for Meta-learning over Distributed Databases. *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining (KDD '97)*, Newport Beach, CA, August 1997.
170. Naser S. Barghouti, John Mocenigo, and Wenke Lee. Grappa: A GRAPh PACKage in Java. *Proceedings of the Fifth Annual Symposium on Graph Drawing (Graph Drawing '97)*, Rome, Italy, September 1997.
171. Wenke Lee, Gail Kaiser, Paul Clayton, and Eric Sherman. OzCare: A Workflow Automation System for Care Plans. *Proceedings of the American Medical Informatics Association Annual Fall Symposium*, Washington DC, October 1996.

F. Workshop Presentations (refereed)

F.1. Workshop Presentations with Proceedings

1. Yi-an Huang and Wenke Lee. Hotspot-Based Traceback for Mobile Ad Hoc Networks. In *Proceedings of The ACM Workshop on Wireless Security (WiSe 2005)*, Cologne, Germany, September 2005.
2. Monirul Sharif, George Riley, and Wenke Lee. Comparative Study between Analytical Models and Packet-Level Worm Simulations. In *Proceedings of The 19th Workshop on Parallel and Distributed Simulation (PADS 2005)*, Monterey, CA, June 2005.
3. Chris Clark, Wenke Lee, David Schimmel, Didier Contis, Mohamed Kone, and Ashley Thomas. A Hardware Platform for Network Intrusion Detection and Prevention. *The 3rd Workshop on Network Processors and Applications (NP3)*, Madrid, Spain, February 2004.
4. Yian Huang and Wenke Lee. A Cooperative Intrusion Detection System for Ad Hoc Networks. *ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN '03)*, Fairfax VA, October 2003.
5. Mustaque Ahamad, Wenke Lee, Ling Liu, Leo Mark, Edward Omicienski, Calton Pu, and Andre dos Santos. Guarding the Next Internet Frontier: Countering Denial of Information Attacks. *Proceedings of the 2002 New Security Paradigms Workshop*, Virginia Beach, Virginia, September 2002.
6. Xinzhou Qin, Wenke Lee, Lundy Lewis, and Joao B.D. Cabrera. Using MIB III Variables for Network Anomaly Detection - A Feasibility Study. *ACM Workshop on Data Mining for Security Applications*, Philadelphia, PA, November 2001.
7. Yongguang Zhang, Harrick Vin, Lorenzo Alvisi, Wenke Lee, and Son K. Dao. Heterogeneous Networking: A New Survivability Paradigm. *Proceedings of the 2001 New Security Paradigms Workshop*, Cloudcroft, New Mexico, September 2001.

8. Wenke Lee, Wei Fan, Matt Miller, Sal Stolfo, and Erez Zadok. Toward Cost-Sensitive Modeling for Intrusion Detection and Response. *ACM Workshop on Intrusion Detection Systems*, Athens, Greece, November 2000.
9. Wenke Lee, Rahul Nimbalkar, Kam Yee, Sunil Patil, Pragnesh Desai, Thuan Tran, and Sal Stolfo. A Data Mining and CIDF Based Approach for Detecting Novel and Distributed Intrusions. *Proceedings of The Third International Workshop on Recent Advances in Intrusion Detection (RAID 2000)*, LNCS 1907, Toulouse, France, October 2000.
10. Wenke Lee, Chris Park, and Sal Stolfo. Towards Automatic Intrusion Detection using NFR. *1st USENIX Workshop on Intrusion Detection and Network Monitoring*, April 1999.
11. Wenke Lee, Sal Stolfo, and Phil Chan. Learning Patterns from Unix Process Execution Traces for Intrusion Detection. *AAAI Workshop: AI Approaches to Fraud Detection and Risk Management*, July 1997.
12. Sal Stolfo, Wei Fan, Wenke Lee, Andreas Prodromidis, and Phil Chan. Credit Card Fraud Detection Using Meta-Learning: Issues and Initial Results. *AAAI Workshop: AI Approaches to Fraud Detection and Risk Management*, July 1997.
13. Gail Kaiser and Wenke Lee. Pay No Attention to the Man Behind the Curtain. *NSF Workshop on Workflow and Process Automation*, May 1996.
14. Wenke Lee. Data Modeling and Management for Large Spatial Databases. *The Third International Workshop in Geographic Information Systems*, Beijing, China, August 1993.

G. Other

1. Matt Blaze, Sampath Kannan, Insup Lee, Oleg Sokolsky, Jonathan Smith, Angelos Keromytis, and Wenke Lee. Dynamic Trust Management. *IEEE Computer*, February 2009.
2. Martim Carbone, Diego Zamboni, and Wenke Lee. Taming Virtualization. *IEEE Security & Privacy*, vol. 6, no. 1, January/February 2008.
3. Bryan D. Payne, Reiner Sailer, Ramon Caceres, Ronald Perez, and Wenke Lee. A Layered Approach to Simplified Access Control in Virtualized Systems. *ACM SIGOPS Operating Systems Review*, 4(2), July 2007.
4. Wenke Lee. 2002. Applying Data Mining to Intrusion Detection: The Quest for Automation, Efficiency, and Credibility. *SIGKDD Explorations*, 4(2), December 2002.
5. Wenke Lee and Wei Fan. 2001. Mining System Audit Data: Opportunities and Challenges. *SIGMOD Record*, 30(4), December 2001.
6. Salvatore J. Stolfo, Wenke Lee, Philip K. Chan, Wei Fan, and Eleazar Eskin. 2001. Data Mining-Based Intrusion Detectors: An Overview of the Columbia IDS Project. *SIGMOD Record*, 30(4), December 2001.

H. Research Proposals and Grants

a. Approved and Funded

1. **Secure and Efficient Federated Learning for Distributed Systems**
Wenke Lee (PI)
Technology Innovation Institute Sole Proprietorship LLC
Funded at \$1,200,000, 12/1/2023-6/30/2026.
2. **Robust Physical Adversarial Attack and Practical Defense for Aerial Imagery**
Polo Chau (PI) and Wenke Lee (co-PI)
Technology Innovation Institute Sole Proprietorship LLC
Funded at \$1,125,000, 12/1/2023-6/30/2026.
3. **NSF AI Institute for Agent-based Cyber Threat Intelligence and Operation (ACTION)**
Giovanni Vigna (Director; UC Santa Barbara); Executive Committee Members: Christopher Kruegel (UCSB), Dawn Song (UC Berkeley), Dongyan Xu (Purdue University), and Wenke Lee (Georgia Tech)
NSF
Funded at \$1,625,000 to Georgia Tech, 6/1/2023-7/1/2027.
4. **AI Support for Cybersecurity**
David Wagner (PI; UC Berkeley); co-PIs at UC Berkeley: Michael Jordan, Vern Paxson, Jacob Steinhardt, and Nina Haghtalab; co-PI at Georgia Tech: Wenke Lee
UC Berkeley via c3.ai
Funded at \$140,000 to Georgia Tech, 7/1/2022-6/30/2023.
5. **Proof-driven Refinement of Infrastructure Software Modules (PRISM)**
Brendan Saltaformaggio (PI); co-PIs: Wenke Lee, Taesoo Kim, Alessandro Orso, Qirun Zhang, Yisroel Mirsky, Martin Osterloh, Jason Li, Michael Brown, Sukarno Mertoguno, Kevin Hamlen, and Nicholas Evancich
DARPA I2O
Funded at \$10,647,759, 7/1/2021-12/1/2025.
6. **D3: Debloating, Dialecting and Diversification for Attack Resilient Software with Real-time Constraints**
Taesoo Kim (PI); co-PI: Wenke Lee
Technology Innovation Institute Sole Proprietorship LLC
Funded at \$1,575,000, 9/1/2020-8/31/2023.
7. **HECTOR: Halting Emergent Computation Through Oracle Reinforcement**
Wenke Lee (PI); co-PIs: Sukarno Mertoguno (GTRI) and Clayton Kerce (GTRI)
DARPA I2O
Funded at \$1,000,000, 1/15/2020-7/15/2021.
8. **Interactive Editing Techniques for Subsetting and Dialecting Network Protocols**
Taesoo Kim (PI); co-PIs: Wenke Lee, Alex Orso, and Brendan Saltaformaggio

ONR

Funded at \$2,976,781, 8/1/2018-7/31/2021

9. **SaTC: CORE: Medium: Understanding and Fortifying Machine Learning Based Security Analytics**

Polo Chau (PI); co-PIs: Wenke Lee, Le Song, and Taesoo Kim

NSF

Funded at \$1,200,000, 8/1/2017-7/31/2021

10. **Comprehensive System Debloating via Path-Based Learning and Late-Stage OS Composition**

Wenke Lee (for Bill Harris) (PI); co-PIs: Bill Harris, Taesoo Kim, Alessandro Orso, and Santosh Pande

ONR

Funded at \$7,500,000, 8/1/2017-7/31/2022

11. **CyberCorps Scholarship-for-Service (SFS)**

Mustaque Ahamad (PI); co-PIs: Wenke Lee, Taesoo Kim, Sy Goodman, and Manos Antonakakis

NSF

Funded at \$5,000,000, 9/1/2016-10/31/2021.

12. **Multi-Layer Adaptive and Proactive Strategic Cyber Defense**

Radha Poovendran (PI, University of Washington); co-PIs: Wenke Lee (Georgia Tech) and Tamer Basar (UIUC)

ARO

Funded at \$1,000,000, 8/1/2016-7/31/2019.

13. **Intel Science & Technology Center for Adversary-Resilient Security Analytics (ISTC-ARSA)**

Wenke Lee (PI); co-PIs: Taesoo Kim, Polo Chau (CSE), and Le Song (CSE)

Funded as a \$1,500,000 gift.

14. **THEIA: Tagging and Tracking of Multi-Level Host Events for Transparent Computing and Information Assurance**

Wenke Lee (PI); co-PIs: Taesoo Kim, Alex Orso, and Simon Chung

DARPA

Funded at \$4,193,126, 6/26/2015-6/26/2019.

15. **ADAPT: Analytical Framework for Actionable Defense against Advanced Persistent Threats**

Radha Poovendran (PI, University of Washington); co-PIs: Wenke Lee (Georgia Tech), Shankar Sastry and Anthony Joseph (UC Berkeley), Joao Hespanha (UCSB), Tamer Basar (UIUC), and Maryam Fazel and Linda Bushnell (University of Washington)

ONR MURI

Funded at \$7,500,000, 6/1/2016-5/31/2021.

16. **Embedasploit: A “Pen-Test in a Box” for Industrial Control Systems**
Wenke Lee (PI); co-PIs: Sal Stolfo at Columbia, Brendan Dolan-Gavitt at NYU Tandon School of Engineering
Office of Naval Research
Funded at \$1,250,351, 5/1/2015-4/30/2018.
17. **BFT++: Attack Tolerance in Hard Real-Time Systems**
Taesoo Kim (PI); co-PI: Wenke Lee
Office of Naval Research
Funded at \$1,245,720, 4/1/2015-3/31/2018.
18. **TWC SBE: TTP Option: Medium: Collaborative: EPICA: Empowering People to Overcome Information Controls and Attacks**
Wenke Lee (PI); co-PIs: Nick Feamster, Hongyuan Zha, and Hans Klein
NSF
Funded at \$1,100,000, 8/1/14-7/31/17.
19. **Comprehensive Understanding of Malicious Overlay Networks**
Homeland Security Advanced Project Research Agency (HSARPA of DHS)
Wenke Lee (PI); co-PIs: David Dagon at Georgia Tech, Chris Smoak of GTRI, Roberto Perdisci (UGA), April Lorenzen of Dissect Cyber, Paul Vixie of Internet Systems Consortium, Jody Westby of Global Cyber Risk, and Matt Jonkman of Open Information Security Foundation.
Funded at \$1,873,078 for project total, with \$673,078 to CoC/GT, 10/1/12–9/30/15.
20. **EAGER: The Conceptual Landscape of Information Manipulation**
Wenke Lee (PI); co-PIs: Nick Feamster, and Hans Klein (Public Policy)
NSF
Funded at \$200,000, 8/1/12–7/31/14.
21. **Integrating Security Concepts in an Undergraduate Computer Sciences Curriculum**
Intel
Wenke Lee (PI); co-PIs: Mustaque Ahamad, Nick Feamster, and Paul Royal
Funded as unrestricted gift at \$50,000, 11/2011;
additional, unrestricted gift at \$35,000, 3/2012;
additional, unrestricted gift at \$50,000, 11/2012;
a total of \$135,000.
22. **Monitoring Free and Open Access to Information on the Internet**
Google
Nick Feamster and Wenke Lee (co-PIs)
Funded as unrestricted gift: \$1,000,000 base for 12/15/2010–12/14/2012;
\$500,000 option for 12/15/2012–12/14/2013;
additional gift at \$500,000, for 12/15/13–12/14/2014;
a total of \$2,000,000.

23. **Dimensions of Network Intelligence**
HSARPA of DHS
Wenke Lee (PI) and Paul Royal (co-PI)
Funded at \$1,022,000, 9/2012-9/2017.
24. **Dynamic Reputation for DNS**
Google
Wenke Lee (PI)
Funded as unrestricted gift at \$75,000, 7/2010.
25. **PEASOUP: Preventing Exploits Against Software Of Uncertain Provenance**
IARPA/STONESOUP Program (GrammaTech subcontract)
Wenke Lee (co-PI), with David Melski (PI) at GrammaTech, Jack Davidson and John Knight (co-PIs) at University of Virginia, and Tom Bracewell (co-PI) at Raytheon
Funded at \$13,000,000 for project total, with \$1,695,887 to Georgia Tech, 6/1/2010–5/31/2014.
26. **Research in Virtual Machine Monitoring Techniques**
Sandia National Labs/Sandia Corp.
Wenke Lee (PI)
Funded at \$50,000, 9/15/2010–9/15/2011.
27. **TC: SMALL: A Foundational and Practical Platform for Host Security Applications**
NSF CNS/Trustworthy Computing
Wenke Lee (PI)
Funded at \$429,571, 7/15/2010–7/15/2013.
28. **Botnet Attribution and Removal: From Axioms to Theory to Practice**
ONR MURI
Wenke Lee (PI); Co-PIs: Nick Feamster, and Jon Giffin at Georgia Tech; Farnam Jahanian, Mike Bailey, and Kang Shin at University of Michigan, Ann Arbor; Giovanni Vigna and Chris Kruegel at University of California, Santa Barbara; John Mitchell at Stanford University
Funded: \$7,500,000 project total, \$2,600,000 for Georgia Tech, 5/1/2009–4/30/2014.
29. **Collaborative Research: CT-L: CLEANSE: Cross-Layer Large-Scale Efficient Analysis of Network Activities to Secure the Internet**
NSF CNS/Cyber Trust
Wenke Lee (PI); Co-PIs: Nick Feamster, Jon Giffin, Mustaque Ahamad, and Xiaoming Huo (ISyE) at Georgia Tech; Mike Reiter and Fabian Monrose at University of North Carolina at Chapel Hill; Farnam Jahanian and Mike Bailey at University of Michigan; Phil Porras and Vinod Yegneswaran at SRI International; and Paul Vixie at Internet Software Consortium.
Funded: \$1,829,297 project total, \$758,297 for Georgia Tech, 9/1/2008–8/31/2012.
30. **SMITE: Scalable Monitoring in the Extreme**
DARPA
Wenke Lee and Nick Feamster (co-PIs at Georgia Tech); BBN subcontract
Funded: \$574,999 for Georgia Tech, 4/1/2008–3/31/2010

31. **Countering Botnets: Anomaly-Based Detection, Comprehensive Analysis, and Efficient Mitigation**
DHS HSARPA
Wenke Lee (PI); with Nick Feamster and Jon Giffin at Georgia Tech, David Dagon at Damballa, Rick Wesson and Adam Waters at Support Intelligence, Paul Vixie at Internet Systems Consortium, and Jody Westby at Global Cyber Risk
Funded: \$1,050,730 project total, \$300,000 for Georgia Tech, 4/1/2008–3/31/2010.
Add-on: \$305,066 project total, \$190,066 to Georgia Tech, 7/1/2009–9/30/2010.
32. **CRI-IAD: Collaborative Research: Enabling Security and Network Management Research for Future Networks**
NSF CRI
Nick Feamster (co-PI), Alex Gray, and Wenke Lee at Georgia Tech, and Z. Morley Mao (co-PI) and Farnam Jahanian at University of Michigan
Funded: \$397,426 for Georgia Tech (3/1/2008–2/28/2011)
33. **Collaborative Research: CT-T: Logic and Data Flow Extraction for Live and Informed Malware Execution**
NSF CNS/CyberTrust
Wenke Lee (co-PI); co-PIs: Paul Barford at University of Wisconsin at Madison, and Phil Porras at SRI International
Funded: \$220,000 for Georgia Tech (9/1/2007–8/31/2009)
34. **Foundational and Systems Support for Quantitative Trust Management**
Office of Naval Research MURI (University of Pennsylvania subcontract)
Sampath Kannan (PI); co-PIs: Insup Lee, Oleg Sokolsky, Matt Blazes, and Jonathan Smith at University of Pennsylvania, Wenke Lee at Georgia Tech, and Angelos Keromytis at Columbia University.
Funded: \$1,125,000 for Georgia Tech (project total: \$4,000,000) (5/1/2007–4/30/2012)
35. **Collaborative Research: CT-ISG: Modeling and Measuring Botnets**
NSF CNS/CyberTrust
Wenke Lee (PI) and Cliff C. Zou (co-PI) at University of Central Florida
Funded: \$350,000 for project total, Georgia Tech at \$175,000, (8/15/2006-8/15/2009)
36. **CT-ISG: Trusted Passage: Managing Distributed Trust to Meet the Needs of Emerging Applications**
NSF CNS/CyberTrust
Mustaque Ahamad (PI), Wenke Lee (co-PI), and Karsten Schwan (co-PI)
Funded: \$350,000, (8/15/2006-8/15/2009)
37. **Cyber Threat Analytics**
ARO via subcontract from SRI International
Funded: \$80,000, (7/15/2006-7/15/2007)
38. **ARL CTA**
ARL via subcontract from SPARTA/Telcordia
Funded: \$75,000, (7/15/2006-7/15/2007)

39. **Computer Network Defenses Technologies**
AFRL via subcontract from Scientific Systems Co. Funded: \$30,000, (9/13/2006-3/13/2007)
40. **ARO Workshop on Research in Botnets**
ARO
Funded: \$25,000, (4/1/2006-6/1/2007)
41. **Next-Generation Botnet Detection and Response**
DARPA (via ARO)
Funded: \$291,346, (12/15/2005-6/14/07)
42. **An Information-Theoretic Framework for Evaluating and Optimizing Intrusion Detection Performance**
Army Research Office
Funded: \$199,745, (4/1/2005-1/30/07)
43. **Preventing SQL Code Injection by Combining Static and Runtime Analysis, funded by Department of Homeland Security**
Department of Homeland Security
Alex Orso (PI) and Wenke Lee (co-PI)
Funded: \$200,000, (7/1/2005-12/31/2006)
44. **Anomaly and Misuse Detection in Network Traffic Streams – Checking and Machine Learning Approaches**
Office of Naval Research MURI (University of Pennsylvania subcontract)
Sampath Kannan (PI), Insup Lee, Oleg Sokolsky, and Linda Zhao at University of Pennsylvania, Wenke Lee at Georgia Tech, and Diana Spears and William Spears at University of Wyoming.
Funded: \$400,000 for Georgia Tech (\$2,000,000 project total) (7/15/2004–6/1/2006)
45. **Intrusion Detection and Security Management Technologies for Mobile Ad Hoc Networks**
Army Research Lab (Scientific Systems Company Inc. subcontract)
Funded: \$65,000 (2/1/2005-2/1/2006)
46. **A Course Module on Wireless Security**
Microsoft, Curriculum Development Grant
Funded: \$30,000 (unrestricted gift)
47. **Intrusion Detection Techniques for Mobile Ad Hoc Networks**
NSF, CISE/CCR/Trusted Computing
Funded: \$275,000 (8/15/2003-7/31/2006)
REU Funded: \$15,000 (9/10/2004-7/31/2006)
48. **High-Speed Intrusion Detection Sensors**
Army Research Lab (Scientific Systems Company Inc. subcontract)
Funded: \$65,000 (8/7/2003-8/6/2005)

49. **Distributed Intrusion Detection Feasibility Study**
U.S. Government (Columbia University subcontract)
Funded: \$45,000 (6/1/2003-6/30/2004)
50. **Agile Security for Storing Sensitive and Critical Information**
NSF, CISE/Trusted Computing
Mustaque Ahamad (PI), and co-PIs: Wenke Lee, Doug Blough, and H. Venkateswaran
Funded: \$460,000 (8/15/2002-7/31/2005)
51. **CAREER: Adaptive Intrusion Detection Systems**
NSF, CISE/CAREER
Funded: \$350,000 (8/15/2002–7/31/2007)
52. **ITR/SI: Guarding the Next Internet Frontier: Countering Denial of Information**
NSF, ITR
Mustaque Ahamad (PI), and co-PIs: Wenke Lee, Andre dos Santos, Ling Liu, Leo Mark, Ed Omiecinski, and Calton Pu
Funded: \$1,694,769 (9/1/2001–8/31/2006)
53. **Vulnerability Assessment Tools for Complex Information Networks**
Army Research Office MURI (Harvard University subcontract)
Yu-Chi Ho (PI) at Harvard University, co-PIs: Avrom Pfeffer at Harvard University, Christos G. Cassandras at Boston University, Wei-Bo Gong at University of Massachusetts, Amherst, and Wenke Lee at Georgia Tech
Funded: \$400,000 for Georgia Tech (\$2,000,000 project total) (5/1/2001–4/30/2006)
54. **A Data Mining Approach for Building Cost-sensitive and Light Intrusion Detection Models.**
DARPA, ATO
Wenke Lee (PI), and co-PIs: Salvatore J. Stolfo at Columbia University and Philip Chan at Florida Institute of Technology
Funded: \$854,000 for Georgia Tech (\$2,000,001 project total) (6/1/2000–8/31/2003)

I. Honors and Awards

1. ACSAC Test of Time Award, The 39th Annual Computer Security Applications Conference (ACSAC), for the paper “David Dagon, Guofei Gu, Christopher Lee, and Wenke Lee. A Taxonomy of Botnet Structures” published in ACSAC 2007 that have had enduring significance and impact to the security community, 2023.
2. ACSAC Test of Time Award, The 39th Annual Computer Security Applications Conference (ACSAC), for the paper “Bryan D. Payne, Martim Carbone, and Wenke Lee. Secure and Flexible Monitoring of Virtual Machines” published in ACSAC 2007 that have had enduring significance and impact to the security community, 2023.
3. ACSAC Test of Time Award, The 38th Annual Computer Security Applications Conference (ACSAC), for the paper “Paul Royal, Mitch Halpin, David Dagon, Robert Edmonds, and Wenke Lee. PolyUnpack: Automating the Hidden-Code Extraction of Unpack-Executing Malware” published in ACSAC 2006 that have had enduring significance and impact to the security community, 2022.
4. Distinguished Service Award, IEEE Technical Committee on Security and Privacy, 2022.
5. IEEE Fellow, 2021.
6. Keynote at the ACM Conference on Computer and Communications Security (CCS), 2020.
7. ACM SIGSAC (Special Interest Group on Security, Audit and Control) Outstanding Innovation Award for “contributions to network and systems security, in particular, machine-learning based approaches to security analytics, including tackling intrusion and botnet detection”, 2019.
8. CCS Test of Time Award, The 25th ACM Conference on Computer and Communications Security (CCS), for the paper “Artem Dinaburg, Paul Royal, Monirul Sharif, and Wenke Lee. Ether: Malware Analysis via Hardware Virtualization Extensions” published in CCS 2008 that have had the greatest impact on security research and practice over the past decade, 2018.
9. ACM Fellow, 2017.
10. Distinguished Practical Paper Award, for the paper “Yanick Fratantonio, Chenxiong Qian, Pak Chung, and Wenke Lee. Cloak and Dagger: From Two Permissions to Complete Control of the UI Feedback Loop” in Proceedings of The 2017 IEEE Symposium on Security and Privacy, 2017.
11. Internet Defense Prize, awarded by Facebook and USENIX, for the paper “Byoungyoung Lee, Chengyu Song, Taesoo Kim, and Wenke Lee. Type Casting Verification: Stopping an Emerging Attack Vector” in Proceedings of The 24th USENIX Security Symposium, 2015.
12. Cyber Is A Global Sport Award (awarded to David Dagon and Wenke Lee), Department of Homeland Security, Science and Technology Directorate, Cyber Security Division, 2015.

13. Outstanding Achievement in Research Program Development Award, (awarded to Mustaque Ahamad, Wenke Lee, Paul Royal, and David Dagon), Georgia Institute of Technology, 2015.
14. Outstanding Community Service Award, the IEEE Technical Committee on Security and Privacy, 2013
15. Outstanding Faculty Leadership for the Development of GRAs Award, Georgia Institute of Technology, 2012.
16. Sigma Xi Faculty Best Paper Award, Georgia Institute of Technology, 2010.
17. Best Student Paper Award, for the paper “Monirul Sharif, Andrea Lanzi, Jon Giffin, and Wenke Lee. Automatic Reverse Engineering of Malware Emulators” in Proceedings of The 2009 IEEE Symposium on Security and Privacy, 2009.
18. Outstanding Senior Faculty Research Award, College of Computing, Georgia Institute of Technology, 2009.
19. NSF CAREER Award, 2002.
20. Best Paper Award, Applied Research Category, the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '99), August 1999 (with Kui Mok and Sal Stolfo).
21. Honorable mention (runner-up) for Best Paper Award, Applied Research Category, the 4th International Conference on Knowledge Discovery and Data Mining (KDD '98), August 1998.
22. Honorable mention (runner-up) for Best Paper Award, Applied Research Category, the 3rd International Conference on Knowledge Discovery and Data Mining (KDD '97), August 1997.

III. SERVICE

A. Professional Activities

A.1 Membership and Activities in Professional Societies

1. Fellow, Association for Computing Machinery (ACM)
2. Fellow, Institute of Electrical and Electronics Engineers (IEEE)

A.2 Journal Editorial Board Activities

1. IEEE Transactions on Dependable and Secure Computing (TDSC) 2010 - 2011.
2. ACM Transactions on Information and System Security (TISSEC) 2005 - 2011.

A.3 Conference Committee Activities

1. The 32nd USENIX Security Symposium (2023).
2. The 31st USENIX Security Symposium (2022).
3. The 2022 ACM Conference on Computer and Communications Security.
4. The 2020 ACM Conference on Computer and Communications Security.
5. The 2020 IEEE Symposium on Security & Privacy.
6. The 30th USENIX Security Symposium (2021).
7. The 2019 ACM Conference on Computer and Communications Security.
8. The 2019 IEEE Symposium on Security & Privacy.
9. The 2018 ACM Conference on Computer and Communications Security.
10. The 2018 IEEE Symposium on Security & Privacy.
11. The 2018 IEEE European Symposium on Security & Privacy.
12. The 2017 ACM Conference on Computer and Communications Security.
13. The 26th USENIX Security Symposium (Security '17).
14. Co-chair, Security and Privacy Track, WWW 2017.
15. Annual Network and Distributed System Security Symposium (NDSS) 2017.
16. The 25th USENIX Security Symposium (Security '16).
17. The 23rd ACM Conference on Computer and Communications Security (CCS 2016).

18. The 2016 IEEE European Symposium on Security & Privacy.
19. Annual Network and Distributed System Security Symposium (NDSS) 2016.
20. The 22nd ACM Conference on Computer and Communications Security (CCS 2015).
21. Co-chair, Security and Privacy Track, WWW 2015.
22. Annual Network and Distributed System Security Symposium (NDSS) 2015.
23. European Symposium on Research in Computer Security (ESORICS) 2014.
24. The 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2014).
25. Security and Privacy Track, WWW 2014.
26. *PC Chair*, The 2013 IEEE Symposium on Security and Privacy.
27. The 10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 2013).
28. The 20th ACM Conference on Computer and Communications Security (CCS 2013).
29. The 16th International Symposium on Recent Advances in Intrusion Detection (RAID 2013).
30. The 19th ACM Conference on Computer and Communications Security (CCS 2012).
31. The 9th Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DMIVA 2012).
32. The 7th USENIX Workshop on Hot Topics in Security (HotSec '12).
33. *PC co-Chair*, The 2012 IEEE Symposium on Security and Privacy.
34. Financial Cryptography and Data Security (FC 2012).
35. The 19th Annual Network and Distributed System Security Symposium (NDSS 2012).
36. The 8th Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA 2011).
37. The 41st Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2011).
38. The 20th USENIX Security Symposium (Security '11).
39. The 6th USENIX Workshop on Hot Topics in Security (HotSec '11).
40. The 2011 IEEE Symposium on Security and Privacy.
41. The 13th International Symposium on Recent Advances in Intrusion Detection (RAID 2010).

42. The 15th European Symposium on Research in Computer Security (ESORICS 2010).
43. The 7th Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA 2010).
44. Security and Privacy Track, ICDCS 2010.
45. The 40th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2010).
46. Security and Privacy Track, WWW 2010.
47. *PC Chair*, The 17th Annual Network and Distributed System Security Symposium (NDSS 2010).
48. Financial Cryptography and Data Security (FC 2010).
49. The 25th Annual Computer Security Applications Conference (ACSAC 2009).
50. The 16th ACM Conference on Computer and Communications Security (CCS 2009).
51. 14th European Symposium on Research in Computer Security (ESORICS 2009).
52. The 18th USENIX Security Symposium (Security '09).
53. *PC Chair*, The Second USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET), 2009.
54. *PC Co-Chair*, The Second ACM Conference on Wireless Network Security (WiSec), 2009.
55. The 2009 IEEE Symposium on Security and Privacy.
56. The 16th Annual Network and Distributed System Security Symposium (NDSS 2009).
57. The 24th Annual Computer Security Applications Conference (ACSAC 2008).
58. Internet Measurement Conference (IMC) 2008.
59. 13th European Symposium on Research in Computer Security (ESORICS 2008).
60. The 15th ACM Conference on Computer and Communications Security (CCS 2008).
61. The 17th USENIX Security Symposium (Security '08).
62. 3rd USENIX Workshop on Hot Topics in Security (HotSec '08).
63. The 2008 IEEE Symposium on Security and Privacy.
64. The 15th Annual Network and Distributed System Security Symposium (NDSS 2008).
65. The 23rd Annual Computer Security Applications Conference (ACSAC 2007).
66. The 14th ACM Conference on Computer and Communications Security (CCS 2007).

67. The 10th International Symposium on Recent Advances in Intrusion Detection (RAID 2007).
68. The 16th USENIX Security Symposium (Security '07).
69. The 2007 IEEE Symposium on Security and Privacy.
70. The 14th Annual Network and Distributed System Security Symposium (NDSS 2007).
71. The 2nd International Conference on Security and Privacy for Emerging Areas in Communication Networks (SecureComm 2006).
72. The 15th USENIX Security Symposium (Security '06).
73. The 13th Annual Network and Distributed System Security Symposium (NDSS 2006).
74. The 1st International Conference on Security and Privacy for Emerging Areas in Communication Networks (SecureComm 2005).
75. The 2005 IEEE Symposium on Security and Privacy.
76. The 2005 ACM Workshop on Wireless Security (WiSe 2005).
77. The 2005 IEEE International Conference on Data Mining (ICDM 2005).
78. The Fourteenth International World Wide Web Conference (WWW 2005).
79. The 2005 International Conference on Distributed Computing Systems (ICDCS).
80. The 2004 ACM Workshop on Visualization and Data Mining for Computer Security (VizSEC/DMSEC 2004).
81. The 2004 ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN 2004).
82. The 2004 ACM Workshop on Wireless Security (WiSe 2004).
83. Steering Committee and the 7th International Symposium on Recent Advances in Intrusion Detection (RAID 2004).
84. The 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2004.
85. The 2004 SIAM International Conference on Data Mining, 2004.
86. The 2004 IEEE International Conference on Data Mining (ICDM 2004).
87. The 2004 IEEE Information Assurance Workshop.
88. Co-Chair of the the ICDM Workshop on Data Mining for Computer Security (DMSEC 2003).
89. The 2003 IEEE International Conference on Data Mining (ICDM 2003).
90. The 2003 ACM Workshop on Wireless Security (WiSe 2003).

91. Steering Committee and the 6th International Symposium on Recent Advances in Intrusion Detection (RAID 2003).
92. The 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2003.
93. The 2003 IEEE Information Assurance Workshop.
94. Steering Committee for the 5th International Symposium on Recent Advances in Intrusion Detection (RAID 2002).
95. The 2002 ACM New Security Paradigms Workshop.
96. The 2002 IEEE Symposium on Security and Privacy.
97. *PC Co-Chair*, The 4th International Symposium on Recent Advances in Intrusion Detection (RAID 2001).
98. Program Committee and Organizational Committee for the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2000.

B. On-campus Georgia Tech Committees

1. Chair, RPT Committee, School of Cybersecurity and Privacy, 2024.
2. Chair, Faculty Recruiting Committee, School of Cybersecurity and Privacy, 2023.
3. Chair, Faculty Recruiting Committee, School of Cybersecurity and Privacy, 2021.
4. 5-Year Review Committee for the Chair of School of Computer Science, College of Computing, 2016.
5. Faculty Recruiting Committee, School of Computer Science, 2014.
6. Faculty Recruiting Committee, School of Computer Science, 2012-13.
7. RPT Committee, College of Computing, 2010-2012.
8. College of Computing Dean Search Committee, 2009-2010, College of Computing.
9. Chair, Faculty Recruiting Committee, 2008-2009, School of Computer Science, College of Computing.
10. Chair Advisory Committee, 2008-2009, School of Computer Science, College of Computing.
11. Co-Chair, Faculty Recruiting Committee, 2007-2008, School of Computer Science, College of Computing.
12. RPT Executive Committee, 2007-2008, College of Computing.
13. Faculty Recruiting Committee, 2006-2007, College of Computing.

14. Faculty Recruiting Committee, 2005-2006, College of Computing.
15. Undergraduate Curriculum Task Force, 2004-2005, College of Computing.
16. Ph.D. Admissions Committee, 2003-2004, College of Computing.
17. Faculty Recruiting Committee, 2001-2003, College of Computing.

E. Consulting and Advisory Appointments

1. External Review Committee of the Computer Science Department at the University of California, Santa Barbara. 2022.
2. Advisory Board of the NUS-Singtel Cyber Security Research & Development Laboratory, National University of Singapore. 2019-2021.
3. Secure, Accessible & Fair Elections (SAFE) Commission, State of Georgia, 2018-2019
4. Advisory Board of the Faculty of Engineering, The Chinese University of Hong Kong. 2017-
5. Board of Trustees, Pace Academy. 2017-

IV. NATIONAL AND INTERNATIONAL PROFESSIONAL RECOGNITION

A. Patents

1. “RTCAPTCHA: A Real-Time Captcha Based Liveness Detection System”. Erkam Uzun, Pak Ho Chung, Irfan A. Essa, and Wenke Lee. U.S. Patent Number: 11,663,307, May 2023.
2. “Privacy preserving face-based authentication”. Pak Ho Chung, Wenke Lee, Erkam Uzun, and Carter Yagemann. U.S. Patent Number: 11,494,476, November 2022.
3. “Devices, systems, and methods of program identification, isolation, and profile attachment”. Ruian Duan, Ashish Bijlani, Taesoo Kim, and Wenke Lee. U.S. Patent Number: 11,036,479, June 2021.
4. “Systems and methods for using video for user and message authentication”. Pak Ho Chung, Wenke Lee, and Yeongjin Jang. U.S. Patent Number: 10,476,888, November 2019.
5. “Method and System for Detecting Malware”. Manos Antonakakis, Roberto Perdisci, Wenke Lee, and Gunter Ollmann. U.S. Patent Number: 10,257,212, April 2019.
6. “Methods and systems for detecting compromised computers”, David Dagon, Nick Feamster, Wenke Lee, Robert Edmonds, Richard Lipton, and Anirudh Ramachandran. U.S. Patent Number: 10,044,748, August 2018.
7. “Method and system for detecting malicious and/or botnet-related domain names”. Roberto Perdisci, Wenke Lee, and Gunter Ollmann. U.S. Patent Number: 10,027,688, July 2018.
8. “Method and system for network-based detecting of malware from behavioral clustering”. Roberto Perdisci and Wenke Lee. U.S. Patent Number: 9,948,671, April 2018.
9. “Method and system for detecting DGA-based malware”. Manos Antonakakis, Roberto Perdisci, Wenke Lee, and Nikolaos Vasiloglou. U.S. Patent Number: 9,922,190, March 2018.
10. “Method and system for detecting malicious domain names at an upper DNS hierarchy”. Manos Antonakakis, Roberto Perdisci, Wenke Lee, and Nikolaos Vasiloglou. U.S. Patent Number: 9,686,291, June 2017.
11. “Systems and methods of safeguarding user information while interacting with online service providers”. Wenke Lee, Sasha Boldyreva, Simon Chung, Billy Lau, and Chengyu Song. U.S. Patent Number: 9,659,189, May 2017.
12. “Method and System for Detecting Malware”. Manos Antonakakis, Roberto Perdisci, Wenke Lee, and Gunter Ollmann. U.S. Patent Number: 9,525,699, December 2016.
13. “Method and system for determining whether domain names are legitimate or malicious”, Manos Antonakakis, Roberto Perdisci, David Dagon, and Wenke Lee. U.S. Patent Number: 9,516,058, December 2016.
14. “Method and systems for detecting compromised networks and/or computers”, David Dagon, Nick Feamster, Wenke Lee, Robert Edmonds, Richard Lipton, and Anirudh Ramachandran. U.S. Patent Number: 9,306,969, April 2016.

15. "Systems and methods for secure in-VM monitoring", Monirul Sharif and Wenke Lee. U.S. Patent Number: 9,129,106, September 2015.
16. "Method and system for network-based detecting of malware from behavioral clustering". Roberto Perdisci, Wenke Lee, and Gunter Ollmann. U.S. Patent Number: 8,826,438, September 2014.
17. "Method and system for detecting malicious domain names at an upper DNS hierarchy". Manos Antonakakis, Roberto Perdisci, Wenke Lee, and Nikolaos Vasiloglou. U.S. Patent Number: 8,631,489, January 2014.
18. "Method and System for Detecting Malware". Manos Antonakakis, Roberto Perdisci, Wenke Lee, and Gunter Ollmann. U.S. Patent Number: 8,578,497, November 2013.
19. "Method and System for Detecting and Responding to Attacking Networks". David Dagon, Nick Feamster, Wenke Lee, Robert Edmonds, Richard Lipton, and Anirudh Ramachandran. U.S. Patent Number: 8,566,928, October 2013.
20. "Methods for Cost-Sensitive Modeling for Intrusion Detection and Response". Wei Fan, Wenke Lee, Matt Miller, and Sal Stolfo. U.S. Patent Number: 7,818,797, October 2010.
21. "Method and System for Using Intelligent Agents for Financial Transactions, Services, Accounting, and Advice". Dan Schutzer, Will Foster, Huanrui Hu, Wenke Lee, Sal Stolfo, and Wei Fan. U.S. Patent Number: 5,920,848, July 1999.

V. OTHER CONTRIBUTIONS

A. Seminar Presentations

1. “How To Make YOUR Computer More Secure Than The Internet”, Distinguished Lecture at the University of Wisconsin, Madison, November 2023.
2. “Machine Learning and Security: The Good, The Bad, and The Ugly”, Distinguished Lecture at Dartmouth College, April, 2021.
3. “Machine Learning and Security: The Good, The Bad, and The Ugly”, Distinguished Lecture at Ohio State University, January, 2021.
4. “Machine Learning and Security: The Good, The Bad, and The Ugly”, Keynote at the 2020 ACM Conference on Computer and Communications Security, November, 2021.
5. “Security Overlay”, Stony Brook University, March 2016.
6. “Security Overlay”, Princeton University, October 2015.
7. “Internet Monitoring via DNS Traffic Analysis”, Google-UMD Cybersecurity Seminar, November 2012.
8. “Botnet Detection and Response: Challenges and Opportunities”, Carnegie Mellon University, September 2007.
9. “Botnet Detection and Response”, Indiana University, March 2007.
10. “Botnet Detection and Response”, University of Pittsburgh, October 2006.
11. “Architecture Considerations for Anomaly Detection”, University of Texas at San Antonio, November, 2005
12. “Advanced Intrusion Detection Techniques”, National Security Research Institute, Daejeon, South Korea, February, 2005
13. “U.S. Policy and Recent Detection Techniques for Cyber Threats and Attacks”, National Cyber Security Center, Seoul South Korea, February, 2005
14. “Architecture Considerations for Anomaly Detection”, Purdue University, February, 2005
15. “Architecture Considerations for Anomaly Detection”, Graduate Center of The City University of New York, February, 2005
16. “Architecture Considerations for Anomaly Detection”, The University of Alabama, November, 2004
17. “Architecture Considerations for Anomaly Detection”, Carnegie Mellon University, September, 2004

18. "Architecture Considerations for Anomaly Detection", University of Wisconsin at Madison, August, 2004
19. "Local Worm Detection and Response: Algorithms and Analytical Models", Lawrence Livermore National Laboratory, June 2004
20. "Intrusion Detection", Carnegie Mellon University, October 2003
21. "Algorithms for Recognizing New Attack Step Relationships", University of Pennsylvania, September 2003
22. "Intrusion Detection", Emory University, October 2001
23. "Developing Data Mining Techniques for Intrusion Detection: A Progress Report", Center for Education and Research in Information Assurance and Security, Purdue University, October 2000
24. "A Data Mining Framework for Building Intrusion Detection Models", Institute for Security Technology Studies, Dartmouth College, Hanover, NH, July 2000
25. "A Data Mining Framework for Building Intrusion Detection Models", Department of Electrical and Computer Engineering, University of Massachusetts, Amherst, MA, February 2000
26. "A Data Mining Framework for Building Intrusion Detection Models", HRL Laboratories, Malibu, CA, November 1999

**APPENDIX 2: MATERIALS CONSIDERED IN THE PREPARATION OF
THIS DECLARATION**

EXHIBIT	DESCRIPTION
1001	U.S. Patent No. 11,178,104 to Coleman et al. (“ the ’104 Patent ”)
1002	Prosecution history of the ’104 Patent
1004	U.S. Patent Publication No. 2013/0318594 to Hoy et al., published on November 28, 2013 (“ Hoy ”)
1005	U.S. Patent Publication No. 2015/0096031 to Benoit et al., published on April 2, 2015 (“ Benoit ”)
1006	U.S. Patent No. 10,798,077 to Pratt et al., filed on January 23, 2015 (“ Pratt ”)
1007	U.S. Patent Publication No. 2015/0278513 to Krasin et al., published on October 1, 2015 (“ Krasin ”)
1008	U.S. Patent Publication No. 2012/0017213 to Hunt et al., published on January 19, 2012 (“ Hunt ”)
1009	Selected pages from MICROSOFT COMPUTER DICTIONARY, FIFTH EDITION (Microsoft Press, 2002)
1010	<i>Plaintiff Croga Innovations Ltd’s P.R. 4-2 Disclosure</i> , in <i>Croga Innovations Ltd. v. International Business Machines Corporation</i> , No. 2:23-cv-00634-JRG (U.S. District Court, Eastern District of Texas)

APPENDIX 3: CHALLENGED CLAIMS

[1pre] A host computer system:

[1a] a memory; and

[1b] a processor configured to:

[1c] implement a workspace, wherein the workspace is configured to enable operation of a first set of one or more applications or processes via a first memory space;

[1d] implement an isolated computing environment, the isolated computing environment using a host operating system and comprising a sandboxed computing environment that uses a second memory space to enable operation of a second set of one or more applications or processes,

[1e] wherein the isolated computing environment is configured to access an Internet-based cloud service via at least one application of the second set of one or more applications or processes;

[1f] isolate the isolated computing environment from the workspace using an internal isolation firewall;

[1g] authenticate the isolated computing environment with an authentication device; and

[1h] communicate with a proxy server to access the Internet-based cloud service to allow communication between the at least one application and the Internet-based cloud service when the isolated computing environment has been authenticated.

[2pre] The host computer system of claim 1,

[2a] wherein the processor is configured to access the Internet-based cloud service via one or more of a local area network (LAN), a wide area network (WAN), a cellular network, a Wi-Fi network, a digital subscribers line (DSL) network, a cloud computing network, a data center, or a satellite network.

[3pre] The host computer system of claim 1,

[3a] wherein the Internet-based cloud service comprises one or more of an Infrastructure as a Service (IaaS), a Software as a Service (SaaS), or a Platform as a Service (PaaS).

[8pre] The host computer system of claim 1,

[8a] wherein the internal isolation firewall is configured to segregate the first memory space from the second memory space, and

[8b] wherein the processor is configured to enforce policies for communication between the first memory space and the second memory space using the internal isolation firewall.

[10pre] A cloud computing system comprising:

[10a] a cloud computing network comprising a server,

[10b] wherein the server is configured to:

implement a workspace configured to enable operation of a first set of one or more applications or processes;

[10c] implement an isolated computing environment using a host operating system and comprising a sandboxed computing environment and being configured to enable operation of a second set of one or more applications or processes;

[10d] isolate the isolated computing environment from the workspace using an internal isolation firewall; and

[10e] authenticate the isolated computing environment with an authentication device; and

[10f] a proxy configured to control communication between the server and an Internet resource,

[10g] wherein the proxy is configured to allow communication between the second set of one or more applications or processes running in the isolated computing environment and the Internet resource when the isolated computing environment has been authenticated and

[10h] to block communication between the first set of one or more applications or processes running in the workspace and the Internet resource.

[11pre] The cloud computing system of claim 10,

[11a] wherein the server is configured to perform at least one processing task on behalf of a client device.

[17pre] The cloud computing system of claim 11,

[17a] wherein the at least one processing task performed on behalf of the client device comprises a processing task for an Infrastructure as a Service (IaaS), a Software as a Service (SaaS), or a Platform as a Service (PaaS).

[18pre] A method implemented in a host computer system, the method comprising:

[18a] implementing a workspace,

[18b] wherein the workspace is configured to enable operation of a first set of one or more applications or processes via a first memory space;

[18c] implementing an isolated computing environment,

[18d] the isolated computing environment using a host operating system and comprising a sandboxed computing environment using a second memory space to enable operation of a second set of one or more applications or processes,

[18e] wherein the isolated computing environment is configured to access an Internet-based cloud service via at least one application of the second set of one or more applications or processes;

[18f] isolating the isolated computing environment from the workspace using an internal isolation firewall;

[18g] authenticate the isolated computing environment with an authentication device; and

[18h] communicating with a proxy server to access the Internet-based cloud service to allow communication between the at least one application and the Internet-based cloud service when the isolated computing environment has been authenticated.

[19pre] The method of claim 18,

[19a] wherein the Internet-based cloud service comprises one or more of an Infrastructure as a Service (IaaS), a Software as a Service (SaaS), or a Platform as a Service (PaaS).

APPENDIX 4: UNDERSTANDING OF THE LAW

I have applied the following legal principles provided to me by counsel in arriving at the opinions set forth in this declaration.

Legal Standard for Prior Art

I am not an attorney. I have been informed by attorneys of the relevant legal principles and have applied them to arrive at the opinions set forth in this declaration.

I understand that the petitioner for *inter partes* review may request the cancelation of one or more claims of a patent based on grounds available under 35 U.S.C. §102 and 35 U.S.C. §103 using prior art that consists of patents and printed publications.

Anticipation and Prior Art

I understand that §102 specifies when a challenged claim is invalid for lacking novelty over the prior art, and that this concept is also known as “anticipation.” I understand that a prior art reference anticipates a challenged claim, and thus renders it invalid by anticipation, if all elements of the challenged claim are disclosed in the prior art reference. I understand the disclosure in the prior art reference can be either explicit or inherent, meaning it is necessarily present or implied. I understand that the prior art reference does not have to use the same words as the challenged claim, but all of the requirements of the claim

must be disclosed so that a person of ordinary skill in the art could make and use the claimed subject-matter.

In addition, I understand that §102 also defines what is available for use as a prior art reference to a challenged claim.

Under §102(a)(1), a challenged claim is anticipated if it was patented, described in a printed publication or in public use, on sale, or otherwise available to the public before the effective filing date of the claimed invention.

Under §102(a)(2), a challenged claim is anticipated if it was described in a patent issued under §151, or in an application for patent published or deemed published under §122(b), in which the patent or application, as the case may be, names another inventor and was effectively filed before the effective filing data of the claimed invention.

Under §102(b)(1)(A), a challenged claim is not anticipated if the disclosure as contemplated by §102(a) was made by the inventor or joint inventor or by another who obtained the subject matter disclosed directly or indirectly from the inventor or a joint inventor.

Under §102(b)(1)(B), a challenged claim is not anticipated if the subject matter disclosed had, before such disclosure, been publicly disclosed by the inventor or a joint inventor or another who obtained the subject matter disclosed directly or indirectly from the inventor or a joint inventor.

I understand that a challenged claim's date of invention is presumed to be the challenged patent's filing date.

I understand that the filing date of a patent is generally the filing date of the application filed in the United States that issued as the patent. However, I understand that a patent may be granted an earlier effective filing date if the patent owner properly claimed priority to an earlier patent application.

I understand that when a challenged claim covers several structures, either generically or as alternatives, the claim is deemed anticipated if any of the structures within the scope of the claim is found in the prior art reference.

I understand that when a challenged claim requires selection of an element from a list of alternatives, the prior art teaches the element if one of the alternatives is taught by the prior art.

Legal Standard for Obviousness

I understand that even if a challenged claim is not anticipated, it is still invalid if the differences between the claimed subject matter and the prior art are such that the claimed subject matter would have been obvious to a person of ordinary skill in the pertinent art at the time the alleged invention.

I understand that obviousness must be determined with respect to the challenged claim as a whole.

I understand that one cannot rely on hindsight in deciding whether a claim is obvious.

I also understand that an obviousness analysis includes the consideration of factors such as (1) the scope and content of the prior art, (2) the differences between the prior art and the challenged claim, (3) the level of ordinary skill in the pertinent art, and (4) “secondary” or “objective” evidence of non-obviousness.

Secondary or objective evidence of non-obviousness includes evidence of: (1) a long felt but unmet need in the prior art that was satisfied by the claimed invention; (2) commercial success or the lack of commercial success of the claimed invention; (3) unexpected results achieved by the claimed invention; (4) praise of the claimed invention by others skilled in the art; (5) taking of licenses under the patent by others; (6) deliberate copying of the claimed invention; and (7) contemporaneous and independent invention by others. However, I understand that there must be a relationship between any secondary evidence of non-obviousness and the claimed invention.

I understand that a challenged claim can be invalid for obviousness over a combination of prior art references if a reason existed (at the time of the alleged invention) that would have prompted a person of ordinary skill in the art to combine elements of the prior art in the manner required by the challenged claim. I understand that this requirement is also referred to as a “motivation to combine,”

“suggestion to combine,” or “reason to combine,” and that there are several rationales that meet this requirement.

I understand that the prior art references themselves may provide a motivation to combine, but other times simple common sense can link two or more prior art references. I further understand that obviousness analysis recognizes that market demand, rather than scientific literature, often drives innovation, and that a motivation to combine references may come from market forces.

I understand obviousness to include, for instance, scenarios where known techniques are simply applied to other devices, systems, or processes to improve them in an expected or known way. I also understand that practical and common-sense considerations should be applied in a proper obviousness analysis. For instance, familiar items may have obvious uses beyond their primary purposes.

I understand that the combination of familiar elements according to known methods is obvious when it yields predictable results. For instance, obviousness bars patentability of a predictable variation of a technique even if the technique originated in another field of endeavor. This is because design incentives and other market forces can prompt variations of it, and predictable variations are not the product of innovation, but rather ordinary skill and common sense.

I understand that a particular combination may be obvious if it was obvious to try the combination. For example, when there is a design need or market

pressure to solve a problem and there are a finite number of identified, predictable solutions, a person of ordinary skill has good reason to pursue the known options within his or her technical grasp. This would result in something obvious because the result is the product not of innovation but of ordinary skill and common sense. However, I understand that it may not be obvious to try a combination when it involves unpredictable technologies.

It is further my understanding that a proper obviousness analysis focuses on what was known or obvious to a person of ordinary skill in the art, not just the patentee. Accordingly, I understand that any need or problem known in the field of endeavor at the time of invention and addressed by the patent can provide a reason for combining the elements in the manner claimed.

It is my understanding that the Manual of Patent Examining Procedure §2143 sets forth the following as exemplary rationales that support a conclusion of obviousness:

- Combining prior art elements according to known methods to yield predictable results;
- Simple substitution of one known element for another to obtain predictable results;
- Use of known technique to improve similar devices (methods, or products) in the same way;

- Applying a known technique to a known device (method, or product) ready for improvement to yield predictable results;
- Choosing from a finite number of identified, predictable solutions, with a reasonable expectation of success;
- Known work in one field of endeavor may prompt variations of it for use in either the same field or a different one based on design incentives or other market forces if the variations are predictable to one of ordinary skill in the art;
- Some teaching, suggestion, or motivation in the prior art that would have led one of ordinary skill to modify the prior art reference or to combine prior art reference teachings to arrive at the claimed invention.

A person of ordinary skill in the art looking to overcome a problem will often use the teachings of multiple publications together like pieces of a puzzle, even though the prior art does not necessarily fit perfectly together. Therefore, I understand that references for obviousness need not fit perfectly together like puzzle pieces. Instead, I understand that obviousness analysis takes into account inferences, creative steps, common sense, and practical logic and applications that a person of ordinary skill in the art would employ under the circumstances.

I understand that a claim can be obvious in light of a single reference, if the elements of the challenged claim that are not explicitly or inherently disclosed in the reference can be supplied by the common sense of one of skill in the art.

I understand that obviousness also bars the patentability of applying known or obvious design choices to the prior art. One cannot patent merely substituting one prior art element for another if the substitution can be made with predictable results. Likewise, combining prior art techniques that are interoperable with respect to one another is generally obvious and not patentable.

In order for a claim to be found invalid based upon a modification or combination of the prior art, there must be reasonable expectation that a person of ordinary skill would have successfully modified or combined the prior art to arrive at the claimed arrangement. This does not mean that it must be certain that a person of ordinary skill would have been successful—the law only requires that the person of ordinary skill in the art would have perceived a reasonable expectation of success in modifying or combining the prior art to arrive at the claimed invention.

In sum, my understanding is that obviousness invalidates claims that merely recite combinations of, or obvious variations of, prior art teachings using understanding and knowledge of one of skill in the art at the time and motivated by the general problem facing the inventor at the time. Under this analysis, the prior art references themselves, or any need or problem known in the field of endeavor

at the time of the invention, can provide a reason for combining the elements of or attempting obvious variations on prior art references in the claimed manner.

Legal Standard for Claim Construction

I understand that before any invalidity analysis can be properly performed, the scope and meaning of the challenged claims must be determined by claim construction.

I understand that a patent may include two types of claims, independent claims and dependent claims. I understand that an independent claim stands alone and includes only the limitations it recites. I understand that a dependent claim depends from an independent claim or another dependent claim. I understand that a dependent claim includes all the limitations that it recites in addition to the limitations recited in the claim (or claims) from which it depends.

In comparing the challenged claims to the prior art, I have carefully considered the patent and its file history in light of the understanding of a person of skill at the time of the alleged invention.

I understand that to determine how a person of ordinary skill would have understood a claim term, one should look to sources available at the time of the alleged invention that show what a person of skill in the art would have understood disputed claim language to mean. It is my understanding that this may include what is called “intrinsic” evidence as well as “extrinsic” evidence.

I understand that, in construing a claim term, one should primarily rely on intrinsic patent evidence, which includes the words of the claims themselves, the remainder of the patent specification, and the prosecution history. I understand that extrinsic evidence, which is evidence external to the patent and the prosecution history, may also be useful in interpreting patent claims when the intrinsic evidence itself is insufficient. I understand that extrinsic evidence may include principles, concepts, terms, and other resources available to those of skill in the art at the time of the invention.

I understand that words or terms should be given their ordinary and accepted meaning unless it appears that the inventors were using them to mean something else or something more specific. I understand that to determine whether a term has special meaning, the claims, the patent specification, and the prosecution history are particularly important, and may show that the inventor gave a term a particular definition or intentionally disclaimed, disavowed, or surrendered claim scope.

I understand that the claims of a patent define the scope of the rights conferred by the patent. I understand that because the claims point out and distinctly claim the subject matter which the inventors regard as their invention, claim construction analysis must begin with and is focused on the claim language itself. I understand that the context of the term within the claim as well as other claims of the patent can inform the meaning of a claim term. For example, because

claim terms are normally used consistently throughout the patent, how a term is used in one claim can often inform the meaning of the same term in other claims. Differences among claims or claim terms can also be a useful guide in understanding the meaning of particular claim terms.

I understand that a claim term should be construed not only in the context of the particular claim in which the disputed term appears, but in the context of the entire patent, including the entire specification. I understand that because the specification is a primary basis for construing the claims, a correct construction must align with the specification.

I understand that the prosecution history of the patent as well as art incorporated by reference or otherwise cited during the prosecution history are also highly relevant in construing claim terms. For instance, art cited by or incorporated by reference may indicate how the inventor and others of skill in the art at the time of the invention understood certain terms and concepts. Additionally, the prosecution history may show that the inventors disclaimed or disavowed claim scope, or further explained the meaning of a claim term.

With regard to extrinsic evidence, I understand that all evidence external to the patent and prosecution history, including expert and inventor testimony, dictionaries, and learned treatises, can also be considered. For example, technical dictionaries may indicate how one of skill in the art used or understood the claim

terms. However, I understand that extrinsic evidence is considered to be less reliable than intrinsic evidence, and for that reason is generally given less weight than intrinsic evidence.

I understand that in general, a term or phrase found in the introductory words or preamble of the claim, should be construed as a limitation if it recites essential structure or steps, or is necessary to give meaning to the claim. For instance, I understand preamble language may limit claim scope: (i) if dependence on a preamble phrase for antecedent basis indicates a reliance on both the preamble and claim body to define the claimed invention; (ii) if reference to the preamble is necessary to understand limitations or terms in the claim body; or (iii) if the preamble recites additional structure or steps that the specification identifies as important.

On the other hand, I understand that a preamble term or phrase is not limiting where a challenged claim defines a structurally complete invention in the claim body and uses the preamble only to state a purpose or intended use for the invention. I understand that to make this determination, one should review the entire patent to gain an understanding of what the inventors claim they invented and intended to encompass in the claims.

I understand that 35 U.S.C. §112 (f) created an exception to the general rule of claim construction called a “means plus function” limitation. These types of

terms and limitations should be interpreted to cover only the corresponding structure described in the specification, and equivalents thereof. I also understand that a limitation is presumed to be a means plus function limitation if (a) the claim limitation uses the phrase “means for”; (b) the “means for” is modified by functional language; and (c) the phrase “means for” is not modified by sufficient structure for achieving the specified function.

I understand that a structure is considered structurally equivalent to the corresponding structure identified in the specification only if the differences between them are insubstantial. For instance, if the structure performs the same function in substantially the same way to achieve substantially the same result. I further understand that a structural equivalent must have been available at the time of the issuance of the claim.