UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

FORTINET, INC.,
Petitioner

v.

SOPHOS INC.,
Patent Owner

Inter Partes Review No.: IPR2015-00618
U.S. Patent No.: 8,261,344 B2

DECLARATION OF Dr. Frederick B. Cohen IN SUPPORT OF PATENT OWNER'S RESPONSE PURSUANT TO 37 C.F.R. § 42.120

Background and introduction

On or about 2015-09-30, I received a copy of documentation regarding this matter, including without limit, the Declaration of Dr. Seth Nielson, Ph.D. (Ex. 1007v2), heretofore [**SN**]. I have been asked by Patent Owner's counsel to analyze U.S. Patent No. 8,261,344 B2 (Ex. 1001v2), heretofore '344, in light of my relevant knowledge, skills, training, education, and experience; and to submit this Declaration in Support of the Patent Owner's Response in the instant proceeding, and in rebuttal to [**SN**]. In particular, this declaration sets forth my opinion on the following grounds on which trial was instituted for the '344 patent in this inter partes review:

Ground	Description
1	Claims 1 and 2 as being anticipated by Bodorin (US Patent 7,913,305)
2	Claims 10 and 13 as obvious over Bodorin (US Patent 7,913,305)
3	Claims 16 and 17 as obvious over Bodorin (US Patent 7,913,305) in view of Kissel (US Patent 7,373,664 B2) and Apap (US Patent 7,448,084)

The claims at issue in the grounds noted above appear in the detailed section of this Declaration.

My background with respect to the matters at hand

My name is Fred Cohen and I have been asked to write a Declaration discussing some of the issues involved in this matter. Specifically, within the limits of available time and information, I have been asked to provide opinions related to the patents and claims at issue with regard to issues in this case. This includes, without limit, technical matters involving (1) design, implementation, and operation of computers, operating systems, networks, hardware, firmware, applications, software, and security and the specific areas identified throughout this Declaration; (2) the patents at issue, other patents, methods, apparatus, and related material associated with the technologies and areas of art related to those patents; (3) the state of the relevant art at the dates at issue in this matter; and (4) expectations of what those normally skilled in the relevant arts would reasonably understand and develop at the dates at issue.

My experience with computers started in the early 1960s when I built a small mechanical computer from a kit, and continues through today, where I work as an expert and sometimes testify as an expert witness for legal matters, research, develop, patent new technologies, write peer reviewed papers, peer review papers from others, speak at and attend professional conferences, act as interim director of the Webster University CyberLaboratories, and do consulting through my company Management Analytics; and develop, support, manage, and provide repeatable scalable protection assessment processes through Fearless Security, LLC and Fearless Ridge, LLC, companies in which I am a member. Over that time I have and continue to design, build, operate, configure, manage, and program computers and computer networks including work with hardware, drivers, operating systems, libraries, externally and internally accessible services, and a wide range of related components, subsystems, systems, and technologies.

The focus of my career has been largely on information protection and related issues. This includes design and simulation of protocols for securing networks in the early 1970s; seminal work on computer viruses and defenses against them in the 1980s including creating the first cryptographic checksum methods used for detecting corruption in computer systems and forming the foundational understandings used for much whitelisting and blacklisting technology in use

today; work in the early 1990s on critical infrastructure protection and integrity issues associated with the Internet at that time; the development of forensic methods for attribution and related matters in the late 1990s, work at Sandia National Laboratories in the 1990s and 2000s related to information protection including creating and managing the College Cyber Defenders program, work as an industry analyst for Burton Group in the early 2000s, teaching courses at the University of New Haven and other universities, and a wide range of other related activities.

I earned and received a B.S. in Electrical Engineering from Carnegie-Mellon University in 1977, an M.S. in Information Science from the University of Pittsburgh in 1981, and a Ph.D. in Electrical Engineering from the University of Southern California in 1986. My dissertation was titled "Computer Viruses" and my graduate work in electrical engineering was largely oriented toward issues related to information protection and the design, analysis, and operation of information technology, including parallel processing architectures similar to those at issue in this case. I was also granted an honorary doctorate (Honoris causa) of Computer Science at the University of Pretoria, South Africa in 2011 for my work in information protection and digital forensics.

I have taught and continue to teach courses at the graduate level in the area of digital forensics. information protection, and other related areas, and peer review articles and other publications written by authors in these fields. I have worked as a research professor creating and teaching graduate level courses in related areas for the University of New Haven, collaborated in the creation of new curriculum for doctorate level graduate degrees in digital forensics for California Sciences Institute, taught as a guest instructor in digital forensics for the Federal Law Enforcement Training Center, acted as a California POST certified law enforcement instructor in digital forensics, participated in the New York Electronic Crimes Task Force before there were regional task forces, and currently participate in the San Francisco Electronic Crimes Task Force, both of which are run by the United States Secret Service. I have taught courses and students from government agencies including, without limit, the United States Secret Service, the National Security Agency, branches of the US Department of Defense, State police from many states, agencies that are now members of the Department of Homeland Security, and other intelligence agencies.

I have performed and continue to perform research in the areas of information protection and digital forensics. I led a research and development team at Sandia National Laboratories that developed digital forensic methods and mechanisms, participated in the development of national guidelines for digital forensic evidence, authored chapters in books and two full books on this subject, performed and continue to perform a wide variety of other activities related to this field, and have given invited keynote speeches with associated papers at several international conferences over the last several years.

I have designed, implemented, and reviewed computer hardware and software throughout my career and continue to do so to this day. This includes without limit, writing and reviewing programs for microcode, BIOS and other processor instruction and bootstrap sequences, internal device programs, device drivers for use within operating systems, TSR programs, operating system components, library programs, application programs, network infrastructure, server, and client programs. I have written and reviewed software in a wide variety of computer languages and for a wide range of different sorts of computing devices, including without limit in and for all of the computer languages, operating systems, and hardware architectures applicable in this matter.

I have published more than 200 articles and other papers in the information protection and digital forensics areas, I have written several books on these subjects, I am a member of editorial boards of professional publications in these areas, and was honored in 2011 by the International Information Systems Security Certification Consortium $(ISC)^2$ being named a Fellow of the $(ISC)^2$. I have also been given various awards and honors over the years, details of which are provided in

my included curriculum vitae. I regularly attend and speak at conferences on related matters and this area has been the focus of my career since the 1970s.

I have included additional background information and my curriculum vitae in Appendix A of this Declaration, which includes a listing of my peer reviewed publications. I have also identified specific information regarding specific areas of expertise throughout this Declaration where appropriate to the specific issues at hand.

Summary of my opinions in this matter regarding Grounds 1, 2, & 3

My conclusions with regard to the issues in this case are presented here in summary form. They are based on the more detailed information provided later in this Declaration and my understanding of the issues in this matter. To the extent there is any difference between this summary and the detailed basis provided later in the Declaration, the detailed basis is the more authoritative source and expression of those conclusions.

I conclude as follows:

With regard to Ground 1:

- Claims 1 and 2 of '344 are not shown to be anticipated by Bodorin by Fortinet's expert.

With regard to Ground 2:

- Claims 10 and 13 of '344 are not shown to be obvious over Bodorin by Fortinet's expert.

With regard to Ground 3:

- Claims 16 and 17 of '344 are not shown to be obvious over Bodorin in view of Kissel and Apap by Fortinet's expert.

My opinions in this matter and their detailed basis

My opinions and conclusions in this matter and their detailed basis are included below. Without limit, the basis for my opinions include the section earlier on my background, my knowledge, skills, training, education, and experience in the relevant fields at issue, the content of the body of this Declaration, the appendices hereto, the material I have referenced and considered herein, and all other material considered in this matter.

Table of Contents

Legal Understandings

Counsel has informed me and I understand the following in regard to this matter:

Counsel has informed me and it is my understanding that, in an inter partes review, claim terms in an unexpired patent are interpreted according to their broadest reasonable construction in light of the specification of the patent in which they appear.

Counsel has informed me and it is also my understanding that under a broadest reasonable interpretation, words of the claim must be given their plain meaning, unless such meaning is inconsistent with the specification, such as where the inventor acted as his or her own lexicographer, used terms without an established plain and ordinary meaning in the art, or redefined a well-known term of art. It is the use of the words in the context of the written description and customarily by those skilled in the relevant art that accurately reflects both the 'ordinary' and the 'customary' meaning of the terms in the claims. I understand that the plain meaning to one of skill in the art is considered at the time of the invention (i.e., the date the earliest supporting priority application was filed,

Counsel has informed me and I further understand that, if there is no plain and ordinary meaning of a claim term, then the construction of the claim term should be derived from the specification. I also understand that the specification plays a crucial role in claim construction, and that the claims must be read in view of the specification of which they are a part. I also understand that the specification may reveal a special definition given to a claim term by the patentee that differs from the meaning it would otherwise possess, and that in such cases the lexicography governs. I further understand that the prosecution history of the patent should also be considered, and that it provides evidence of how the U.S. Patent and Trademark Office and the inventor understood the patent.

Counsel has informed me and I further understand that the claim language is to be viewed from the perspective of a person of ordinary skill in the art at the time of invention. When analyzing the '347 patent, the disputed claims, and the prior art, I apply this standard set forth above to reach my conclusions, and any reference to a "person of ordinary skill in the art" below refers to a person of ordinary skill in the relevant art of the '344 patent at the time of the invention.

I have been informed that, in the context of the inter partes review, the party asserting invalidity of a patent must prove invalidity by a preponderance of the evidence. I have been informed that a "preponderance of the evidence" is evidence sufficient to show that a fact is more likely than not.

Patent Owner's counsel has informed me that a patent claim is invalid for lack of novelty, or as "anticipated," under 35 U.S.C. § 102, if, among other things, (a) the alleged invention was known or used by others in this country, or patented or described in a printed publication in the United States or a foreign country, before the alleged invention thereof by the patent's applicant(s), or (b) the alleged invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of the application for patent in the United States, or (e) the alleged invention was described in an application for patent by another person in the United States before the alleged invention by the applicant thereof.

Patent Owner's counsel has also informed me that references or items that fall into one or more of these categories are called "prior art," and that in order to anticipate a patent claim pursuant to 35 U.S.C. § 102, a reference must contain all of the elements and limitations described in the claim either expressly or inherently. As such, counsel for Patent Owner has informed me that in deciding whether or not a single item of prior art anticipates a patent claim, one should consider what is expressly stated or present in the item of prior art, and what is inherently present.

I understand that, to establish inherency, the missing characteristic must be necessarily present in the single reference. I also understand that the missing descriptive material cannot merely be probably or possibly present. It is my understanding that one of ordinary skill in the art may not have recognized the inherent characteristics or functioning of the prior art at the time. I further understand that obviousness is not inherent anticipation, and that it is insufficient that a missing limitation is so similar to a limitation actually disclosed in the reference that one of ordinary skill in the art would see the substitution as obvious. I also understand that, if it is necessary to reach

beyond the boundaries of a single reference to provide missing disclosure of the claimed invention, the proper ground is not anticipation, but obviousness. I also understand that all of the claim elements must be present in the reference arranged as in the claim in order to established inherency.

I understand that invalidity based on anticipation requires that the reference enable the subject matter of the reference and therefore the patented invention without undue experimentation. I also understand that the proper test of a publication as prior art is whether one skilled in the art to which the invention pertains could take the description in the printed publication and combine it with his own knowledge of the particular art and from this combination be put in possession of the invention on which a patent is sought.

I understand that a patent claim is invalid because it lacks novelty or is "obvious" under 35 U.S.C. § 103 if the claimed subject matter would have been obvious to a person of ordinary skill in the art at the time the application for patent was filed, based upon one or more prior art references. I understand that an obviousness analysis should take into account (1) the scope and content of the prior art; (2) the differences between the claims and the prior art; (3) the level of ordinary skill in the pertinent art; and (4) secondary considerations, if any, of obviousness (such as unexpected results, commercial success, long-felt but unsolved needs, failure of others, copy by others, licensing, and skepticism of experts).

I understand that a conclusion of obviousness may be based upon a combination of prior art references. However, I also understand that a patent composed of several elements may not be proved obvious merely by demonstrating that each of its elements was independently known in the art. I further understand that there must be an appropriate articulation of a reason to combine the elements from the prior art in the manner claimed, and obviousness cannot be based on a hindsight combination of components selected from the prior art using the patent claims as a roadmap.

I understand that the following exemplary rationales may lead to a conclusion of obviousness: the combination of prior art elements according to known methods to yield predictable results; the substitution of one known element for another to obtain predictable results; the use of known techniques to improve similar devices in the same way; and some teaching, suggestion, or motivation in the prior art that would have led one of ordinary skill to modify the prior art reference teachings to arrive at the claimed invention. However, a claim is not obvious if the improvement is more than the predictable use of prior art elements according to their established functions. Similarly, a claim is not obvious if the application of a known technique is beyond the level of ordinary skill in the art. Furthermore, when the prior art teaches away from combining certain known elements, discovery of a successful means of combining them is not obvious.

I further understand that, to determine obviousness, the courts look to the interrelated teachings of multiple patents, the effects of demands known to the design community or present in the marketplace, and the background knowledge possessed by a person having ordinary skill in the art.

I understand that a prior art reference must be considered in its entirety, i.e., as a whole, including portions that would lead away from the claimed invention. I also understand that if the proposed modification or combination of the prior art would change the principle of operation of the prior art invention being modified, then the teachings of the references are not sufficient to render a claim prima facie obvious. I further understand that, when considering a disclosure or reference, it is proper to take into account not only specific teachings of the reference but also the inferences which one skilled in the art would reasonable be expected to draw therefrom.

I understand that, to establish prima facie obviousness of a claimed invention, all the claim limitations must be taught or suggested by the prior art.

Proposed claim constructions

I understand that the claim language is to be viewed from the perspective of a person of ordinary skill in the art at the time of invention. When analyzing the '344 patent, the disputed claims, and the prior art, I apply this standard set forth above to reach my conclusions, and any reference to a "person of ordinary skill in the art" below refers to a person of ordinary skill in the relevant art of the '344 patent at the time of the invention.

The patent claims at issue

The patent claims at issue are identified here along with my annotations **[in brackets]** for referring to them herein.

[**Claim 1**] 1. A method for classifying software, said method comprising;

[**1A**] providing a library of gene information including a number of classifications based on groupings of genes;

[**1B**] identifying at least one functional block and at least one property of the software;

[**1C**] identifying one or more genes each describing one or more of the at least one functional block and the at least one property of the software as a sequence of APIs and strings;

[**1D**] matching the one or more genes against one or more of the number of classifications using a processor;

[**1E**] classifying the software based on the matching to provide a classification for the software; and

[**1F**] notifying a user of the classification of the software.

[**Claim 2**] 2. The method of claim 1, wherein

[**2A**] the classification for the software includes malware.

[**Claim 10**] 10. The method of claim 1, further comprising

[**10A**] removing the software

[**10B**] when the software is classified as malware or unwanted software.

[**Claim 13**] 13. The method of claims 1, further comprising

[**13A**] blocking access to the software

[**13B**] when the software is classified as malware or unwanted software.

[**Claim 16**] 16. A method for generating software classifications for use in classifying software, said method comprising:

[**16A**] providing a library of gene information including a number of classifications based on groupings of genes;

[**16B**] identifying one or more genes each describing a functionality or a property of the software as a sequence of APIs and strings;

[**16C**] combining a plurality of genes that describe the software, thereby providing a set of genes;

[**16D**] testing the set of genes for false-positives against one or more reference files using a processor;

[**16E**] defining the software classification based on the set of genes; and

[**16F**] storing the set of genes and the software classification in the library.

[**Claim 17**] 17. The method of claim 16, wherein

[**17A**] the software classification includes malware.

Background on the prior art and the general fields at issue

The priority date of '344 is June 30, 2006, and for the purposes of understanding the fields at issue, the general time frame I will address is that of mid-2006.

[**SN** para 18] asserts (**emphasis** added):

> *(18.) I have carefully reviewed the '344 Patent as well as the file history of the '344 Patent. Based on my review of this material, I believe that the relevant field for the purposes of the '344 Patent is generally computer security. In particular, the '344 Patent is* ***related to*** *anti-virus computer software that* ***can dynamically detect*** *viruses and other malware by examining certain characteristics of the software being examined. I have been informed that the relevant timeframe is on or before June 30, 2006.*

To here, [**SN**] has not identified any specific basis for asserting "**dynamically**", and '344 does not use that term or express such a mechanism as far as I have identified. It references prior art identifying the term "dynamic translation" in two cases and "Dynamic heuristic" in another case, all citations to prior art.

Starting in the early 1980s, I studies methods of testing and analysis of hardware and software, including taking graduate courses in these areas, working as a research assistant on funded research projects in testing, and a wide range of activities between then and the early 2000s involving writing, implementing, and working in software testing, verification, analysis, and related areas, generally as part of my work in information protection, including virus and other "malware" detection, but also including proof of program properties and correctness, reviewing programs for errors and intentional subversion, and in related areas. This also included writing software and working with those who proved that software met specifications.

Generally speaking, in the area of detection of computer viruses and malware, as in the more general field of software analysis for security and other properties, there are distinct and different approaches often called "static" and "dynamic" analysis:

- **Static analysis** is oriented toward examining a stored form of a "program". That is, its form and content as it exists when stored and unchanging (thus static).
- **Dynamic analysis** is oriented toward examining the execution of a program in the operating environment. That is, the behavior of a "program" as it is being "executed" (thus dynamic).

Static detection involves a different set of techniques than dynamic detection for several reasons and each has advantages over the other in particular ways, including without limit:

- **Static detection** has potentially unlimited amounts of time to examine content. It can apply large amounts of computing resource and use a large store of prior information in a wide range of ways. It need not be done in real-time and thus as new analysis methods become available, it can be incrementally enhanced so that previously performed analysis need not be reproduced in order to add additional results. It can potentially examine many different paths through a program, including paths rarely exercised in operation and that can only occur in rare and hard to reproduce circumstances. It typically has access to the stored form of a program containing the code, and thus can potentially alter the stored form to make it not access the portions of code performing undesired operations without causing operational side effects.
- **Dynamic detection** has access to the current and changing state of the operating environment over time. It can typically operate in real time, and thus actively protect an operating environment as it executes. It may observe complex interactions between multiple seemingly unrelated phenomena in the operating environment not available to static analysis, and this includes seeing how portions of multiple programs interact, and the effect of programs that dynamically change or create new instructions to be executed. Dynamic detection normally does not have access to the stored forms available to static analysis, and thus cannot do things like removing or blocking access to portions of programs found to be problematic or performing undesired operations except by stopping their use in real-time during the execution path they are analyzing. Further, removal from an executing program is very complex and may disable other "normal" operations of the program and/or cause side effects on the operating environment that could themselves be harmful.

There are also approaches that mix the static and dynamic approaches to gain advantages of each at the cost of gaining some of the disadvantages of each.

['344 C2L13-19] explicitly identifies dynamic detection per "*Other behavior based technologies rely on running malware and attempting to stop execution if malicious behavior is observer to happen. By allowing malware to execute, the malware may already have caused damage before it is blocked. Additionally, behavior-based technology often requires extensive user interaction to authorize false positives.*" as background before describing the invention. It uses ['344 Fig. 1] to exemplify its methods which examines a "file" (i.e., a stored program) and shows no indication of executing the program at any step. The same is true of the other Figures of '344.

As an example embodiment, ['344 C3L56-8] discloses "*emulating all or part of a file's code to try and detect malware, such as polymorphic viruses, which may reveal themselves during execution,*" as one of many methods that might be employed as part of its list of detection methods. Thus the actual method of '344, while "***related to** anti-virus computer software that **can dynamically detect***" is not "*software that" "**dynamically detect***"s "*viruses and other malware*". Similarly, "*by examining certain characteristics of the software being examined*", even ignoring the potential circularity of the statement, does not adequately or accurately characterize '344 in terms of what it claims.

A Person of Ordinary Skill in the Art (POS)

I will use the abbreviation **POS** for a Person of Ordinary Skill in the Art from here forward.

I understand that Petitioner's expert proposed a **POS** per [**SN** para 21] as:

> *In my opinion, the POSITA relevant to the '344 Patent, at the time the '344 Patent was effectively filed, would have at least a Master's degree in electrical engineering, computer engineering or computer science; or a Bachelor's degree in electrical engineering, computer engineering or computer science with at least two years experience working with hardware and software for computer and network security.*

I will adopt the *POSITA* of [**SN**] as the **POS** for the purposes of this declaration only.

I will outline here the education and relevant understandings of a person with a "*Master's degree in electrical engineering, computer engineering or computer science*" and no other experience to exemplify the things they would likely know about the subject matter of '344 prior to reading the relevant disclosures to this matter.

To get a sense of this, I downloaded the Brigham Young University (Fortinet's expert's University) 2005-2006 curriculum year electrical engineering details on graduation requirements.[1] This is the time frame relevant to the matter at hand per [**SN**]. Not one of the courses I found there identified anything in their names regarding information protection, computer security, or related specialty areas. I also downloaded the BYU current listing of graduate courses from recent years.[2] This lists all courses offered for the last several years, including as far back as 2002, and none of the listed courses indicate anything in their names regarding information protection, computer security, or related specialty areas.

I also downloaded the BYU computer science graduate program information on courses[3] from modern times and found only one non-required course on computer security: "665. Advanced Computer Security. (3) Prerequisite: CS 465 or instructor's consent.". I looked further[4] to identify the specifics of these courses, which led me to the course description for undergraduate non-required prerequisite course[5] which only had historical information back to 2014, and to the description for the graduate course[6], also with historical information back to 2014. The graduate course has no specifics about coverage and appears to be a project-oriented course, while the undergraduate class indicates the following topics covered:

> *This course will cover fundamental principles of computer security. The course consists of two parts:*
>
> > *Part 1: Applied Cryptography - We will study and experiment with basic cryptographic primitives (symmetric encryption, asymmetric encryption, MAC, and cryptographic hash functions). We will learn how these primitives are used to achieve certain security properties. We will then study real-world protocols like HTTPS and secure email to see how these primitives are used in real systems.*
> >
> > *Part 2: Software Security - We will learn about some of the most common errors that software developers make that attackers then exploit. We will learn how to avoid or prevent these mistakes.*

1 Downloaded from https://registrar.byu.edu/advisement/pdf/05/393550.pdf and included herein by reference, attached as 2005-06-BYU-EE-393550.pdf
2 Downloaded from http://www.ee.byu.edu/grad/courses and included herein by reference, attached as 2015-10-10-BYU-ECE-GraduateCourseOfferings.pdf
3 Downloaded from https://cs.byu.edu/graduate-policy-handbook-appendix included herein bu reference, attached as 2015-10-10-BYU-CS-GraduatePolicyHandbookAppendix.pdf
4 See https://cs.byu.edu/courses
5 Downloaded from http://wiki.cs.byu.edu/cs-465/course-information stored as 2015-10-10-BYU-CS465.pdf
6 Downloaded from http://wiki.cs.byu.edu/cs-665/course-information?do= stored as 2015-10-10-BYU-CS-665.pdf

> *… Learning outcomes*
>
> *1) Build a system: Implement a cryptographic algorithm from a standards specification.*
>
> *2) Break and fix a system: Demonstrate how attackers compromise real-world systems, and then show how to prevent these attacks.*

While the second part of this course covers some issues related to programming errors and exploitation, it appears to focus on "*how to avoid or prevent these mistakes*" and indicates nothing regarding detection or mitigation, but rather only prevention. Detection and mitigation is the subject matter of '344. This course has, as a prerequisite, "CS 360", titled "Internet Programming" which[7] has a course description self-indicating it is not a syllabus and only indicates "Use security to protect their web application" as a learning outcome and "security" as one of 5 elements of the course all sharing a total of 6 hours of course time, or in other words, approximately 1.2 hours of coverage. Again nothing specific is indicated regarding detection and response or computer viruses or malware.

Thus from the time frame at issue, in the university of the expert of [**SN**], a *POSITA* of [**SN**] could have no exposure to computer security or related topics whatsoever (i.e., "*a Master's degree in electrical engineering*"), and even for the highest educational level taking all of the related courses, none of which are required to graduate with the degree (i.e., "*a Master's degree in ... computer science*" taking all of the computer security courses offered) , would have only general exposure to a limited subset of areas.

I have identified below the elements for the relevant claims of the '344 patent would likely be well understood by a *POSITA* of [**SN**]. By way of comparison, I have annotated with *underlined italics* below, those elements I would expect, in the best case, a *POSITA* of [**SN**] per the programs offered at BYU today to understand based on the description provided.

- Understand the context of information protection, specifically regarding computer viruses and malware and the manner in which they were implemented, operated, detected, and defeated.
- Understand *code-based attacks* and defenses and *the general field of information protection regarding such attacks* and defenses involving detection and response.
- Specifically understand the terms of art in the relevant fields, including without limit with regard to claims at issue (Claims 1, 2, 10, 13, 16, and 17):
  - Claim 1:
    - Classification of digital information based on content and/or behavior
    - Genetic programming, the use of genes in computer virus generation and detection, the biological analogy between computer and biological viruses, and related areas
    - *The identification of functional blocks within programs.*
    - *The identification of properties associated with functional blocks of programs.*
    - *Searching for application program interfaces (APIs) and strings in software*

7 Downloaded from https://cs.byu.edu/course/cs-360/fall-2015 and included herein and stored in 2015-10-11-BYU-CS360.pdf

- Matching genes in programs to libraries or databases
- *Notification of users*

- Claim 2:
  - Everything identified for Claim 1 plus
    - Classification of malware
- Claim 10:
  - Everything identified for Claim 1 plus
    - *Removal of* malicious portions of *software*
- Claim 13:
  - Everything identified for Claim 1 plus
    - *Blocking access to* malicious portions of *software*
- Claim 16:
  - Classification of digital information based on content and/or behavior
  - Genetic programming, the use of genes in computer virus generation and detection, the biological analogy between computer and biological viruses, and related areas
  - The identification of functionalities or properties associated with programs as a way to describe genes within programs.
  - Combining genes in such a way as to form sets of genes
  - The use of reference files (or other references) for testing purposes
  - Testing genes within programs, specifically for false positives, and specifically against a reference file.
  - The process of defining classifications based on genes or other similar things
  - Storing genes and related classification information in libraries.
- Claim 17:
  - Everything identified for Claim 16 plus
    - Classifying software as “malware”

The assertions of [SN]

[**SN**] asserts and I respond as follows:

As engineering background underlying '344

[**SN** para 42] asserts (**emphasis** added):

> *The ‘344 Patent claims methods for detecting malware by identifying “genes” from the suspected malware and matching the identified genes against a library containing*

> *classifications based on groupings of genes. Ex. 1001 at claim 1.* ***"Genes" is not a term of art ordinarily used in computer science. Rather, it generally is a term used in the biological sciences****. Although the patentee coined a new term in computer science buy repurposing a biological term, the concepts that relate to the term "genes," as explained in the '344 Patent, are not new. As I explained above and in detail below, virus or malware detection by dynamically identifying features of the software and comparing the identified features to a library of known malware features was well-known in the prior art, and one of ordinary skill in the art would have combined any of the well-known design choices/features of these systems to arrive at the claimed invention. Calling these identified features "genes" does not render the '344 Patent novel. My opinion is consistent with* ***the construction I was asked to apply for this Declaration, which construes "gene" generically as "a piece of functionality or property of a program."*** *The '344 Patent offers nothing new, novel, or inventive that was not already known and widely practiced at the time the patent was originally filed.*

Contrary to the assertion of [**SN**], the use of the term "genes" and other related terms of "genetic programming" and virus detection related to such biological models existed for some time before the dates at issue in this matter. '344 has a priority date of Jun 30, 2006, and the term "gene" was, in the relevant time frames and before, a term of art in areas of computer science.

Thus [**SN**] does not represent the requisite knowledge of at least this key aspect of '344 necessary to its understanding and has effectively indicated a lack of relevant understanding of the history of the relevant field of art necessary for that understanding.

Grounds 1 and 2

[**SN** para 53] asserts (**emphasis** added):

> *U.S. Patent No. 7,913,305 to Bodorin et al. ("Bodorin") was filed on January 30, 2004. Bodorin discloses a malware detection system that determines whether an executable code module is malware according to behaviors exhibited by the code module while executing. Ex. 1003 at Abstract. A block diagram of the malware detection system:*

The diagram identifies (208) a "malware behavior signature store" pointing to a "behavior signature comparison module" which is also fed from the external (212) "code module (potential malware)" flowing to a (202) "management module" flowing to a (204) "dynamic behavior evaluation module" flowing to a (210) "behavior signature". In other words, the diagram discloses a sequence of components that act to produce a behavioral evaluation leading to a behavioral signature that is compared to pre-existing malware behavioral signatures to determine the result.

[**SN** para 54] asserts (**emphasis** added):

> *(54.) The management module 202 first evaluates the code module 212 to determine the appropriate type of dynamic behavior evaluation module needed. Ex. 1003 at 4:26-29. Once determined, the code module 212 is run inside the dynamic behavior evaluation module 204 while the dynamic behavior evaluation module 204 records behaviors, including interesting behaviors that are potentially associated with malware, exhibited by the code module 212. Ex. 1003 at 4:39-58. Interesting behaviors include:*

[**SN** para 55] continues (**emphasis** added):

> *(55.) As shown in the table, the interesting behaviors include API calls such as RegisterServiceProcess() and strings such as application_name. Bodorin collects recorded interesting behaviors into behavior signatures and they are illustrated in Figs. 5A and 5B:*

[**SN** para 56] continues (**emphasis** added):

> *(56.) Once the behavior signatures have been generated, the behavior signature comparison module 206 takes the behavior signature and compares it against behavior signatures of known malware that are stored in the malware behavior signature store 208. Ex. 1003 at 5:27-31. The malware behavior signature store 208 is a repository of behavior signatures of known malware. Ex. 1003 at 5:31-32. If the behavior signature comparison module 206 is able to completely match the behavior signature 210 against a known malware behavior signature in the malware behavior signature store 208, the malware detection system 200 will report that the code module is malware. Ex. 1003 at 5:44-49.*

[**SN** para 57] continues (**emphasis** added):

> *(57.) In connection with the above, I have reviewed, had input into, and endorse as if set forth fully herein the claim charts in the accompanying petition showing that each element of claims 1 and 2 are disclosed by Bodorin.*

I continue with a claim chart addressing the specific element of '344 at issue with regard to the identified prior art.

Claims of '344	Per [SN] regarding '305	My response
[Claim 1] 1. A method for classifying software, said method comprising;	[SN] identifies that '305 has a date prior to '344. [**SN** para 58] continues: *(58.) Specifically first, like the '344 Patent, Bodorin discloses a method for classifying software. Bodorin discloses malware detection system that determines whether an executable code module is malware according to behaviors exhibited while executing. Ex. 1003 at Abstract. The detection of malware is classifying software.*	'344 has a priority date of Jun 30, 2006.
[**1A**] providing a library of gene information including a number of classifications based on groupings of genes;	[**SN** para 59-61] continues (**emphasis** added): *(59.) Next, Bodorin **discloses providing a library of gene information including a number of classifications based on groupings of genes**. Bodorin discloses extracting "interesting behaviors" from software, examples of which are listed in Table A.* *…* *(60.) As shown in the table, the **interesting behaviors** are **based on functionality in the form of API calls and strings** such as URL names. These "interesting behaviors" map to "genes," construed as "a piece of functionality or property of a program." For reference, the specification of the '344 Patent gives the following as examples of APIs used in gene definitions: GetSystemDirectoryA, GetModuleFileNameA, RegSetValueExA, CopyFileA, and CreateProcessA. Ex. 1001 at 6:11-45. These API calls are standard Windows API calls and both Bodorin and the '344 Patent use*	[SN para 59] asserts but does not show "***providing a library of gene information including a number of classifications based on groupings of genes***". "***interesting behaviors** are **based on functionality in the form of API calls and strings*** " of [**SN**] are not shown to be the same as "classifications based on groupings of genes". Even if "genes" is construed as "*a piece of functionality or property of a program*" "groupings of genes" of [**1A**] would then be read as "groupings of *piece[s] of functionality or propert[ies] of a program*". But the identified APIs, asserted by [**SN**] to be "*used in gene definitions*", even if used "*in the same way*", which is not shown in [**SN**], are not "groupings of" "*functionality or property*". Also, [**SN** para 59-61] fails to show that "behaviors" of programs when executed (the method of '305) are the same as "*a piece of functionality or property of a program*". "Functionality" is not shown to be the same as behavior, and neither is "behavior" shown to be a "property of a program". Indeed a program only "behaves"

Claims of '344	Per [SN] regarding '305	My response
	*them in the same way. It should be noted that **a single Bodorin behavior may combine both an API call and one or more strings that represent parameters**. Ex. 1003 at 4:58-61.* *(61.) These behaviors are compiled into a "behavior signatures," as shown in Figs. 5A and 5B of Bodorin, which map to the "groupings of genes" as claimed. Bodorin discloses a library called the malware behavior signature store that is a repository of behavior signatures, which maps to the "library" claimed in the '344 Patent. Ex. 1003 at 5:27-43. The malware behavior signature store classifies the behavior signatures according to whether they are based on known malware. Id.*	when interpreted in the context of a specific environment, and the environment is not a property or functionality of a program, nor is the behavior of a program within an environment inherent to the program. The same program may demonstrate different behaviors in different environments, and thus the properties and functionality of a program in an environment is not that of the program, but rather that of the program in the environment, and this is not what '305 claims at [**1A**]. [**SN** para 60] cites to "*Table A includes an exemplary, representative list of "interesting" behaviors, including parameters that are also considered interesting, that are recorded by a dynamic behavior evaluation module 204.*" of [305 as disclosing "*a single Bodorin behavior may combine both an API call and one or more strings that represent parameters*" but the reference does not disclose what [**SN**] asserts. [**SN** para 61] asserts that the set of behavioral signatures produced by the analysis of code (per Figs A and 5B) "*map to the "groupings of genes"*", but this is not what [305] discloses. ['305 C7L16-22] state "FIGS. 5A and 5B are block diagrams illustrating exemplary behavior signatures of hypothetical code modules, such as the behavior signature 210 described above. Each row of items in the exemplary behavior signatures, including behavior signature 500 and behavior signature 512, represents a behavior that was exhibited by a code module and recorded by the dynamic behavior evaluation module 204." Thus, '205 discloses that the set of "behavior signatures" are produced by the analysis of code and the elements

Claims of '344	Per [SN] regarding '305	My response
		of this set are then compared to the signature store, not indicating explicitly or inherently that the signature store contains "groupings of genes" required by [1A]. [**SN**] does not show that '305 discloses "a library of gene information including a number of classifications based on **groupings of** genes;". Rather, even accepting all of the constructions of [**SN**] and non-demonstrated linkages between the elements of '305 and '344, it only shows "a library of gene information including a number of classifications based on **individual** genes;" [**SN**] does not show that the store (asserted as "library") of '305 contains classifications based on "**groupings of** genes;" This [**VN**] fails to show that '305 anticipated [**1A**] of '344.
[1B] identifying at least one functional block and at least one property of the software;	[**SN** para 62] continues (**emphasis** added): *(62.) Next, Bodorin discloses identifying at least one functional block and at least one property of the software. As construed, "functional block" means a segment of the program that illustrates the function and execution flow of the program. As explained above, Bodorin discloses examining the suspected program for behaviors. Ex. 1003 at 4:51-54. These behaviors include API calls that illustrate the function and execution flow of the program. A subset of these behaviors are extracted as being "interesting."*	[**SN** para 62] asserts (**emphasis** added) "*As construed, "functional block" means a **segment** of the program that illustrates the **function and execution flow** of the program.*" [**SN** Depo P34L3-P36L11] states, in pertinent parts[8]: *9 Q. Okay. So a functional block, then,* *10 illustrates a function and execution for the* *11 program, correct? ...* *18 Q. Right. So yeah, okay, this is a* *19 segment of the program that illustrates the* *20 function and execution flow of the program* *21 according to this construction?* *22 A. Yes.* *23 Q. So that's how one of ordinary skill in* *24 the art would understand the term?* *25 A. I think that is a very reasonable way* *00035* *1 S. NIELSON* *2 for one of ordinary skill in the art to* *3 understand that term.* *4 Q. Okay. So in this context, what would* *5 a person of ordinary skill in the art understand* *6 the term "program" to mean?* *7 A. Program?* *8 Q. Program, yeah.*

8 See VIDEOTAPED DEPOSITION OF SETH NIELSON, Ph.D. Baltimore, Maryland Wednesday, October 21, 2015.

Claims of '344	Per [SN] regarding '305	My response
		9 A. I would say any piece of software. *10 Q. Okay. And what about the function of* *11 the program?* *12 A. The way it behaves, what it does.* *13 Q. And what about execution flow of the* *14 program?* *15 A. Well, so functional block is actually* *16 a term that I have seen used in the art, and it's* *17 completely in line with what we're describing* *18 here, the execution flow and function of the* *19 program.* *20 But let's say that you've got like* *21 something that's not binary, like a script. Even* *22 with a binary it's true, though, that a function* *23 block could be a chunk of code that has* *24 recognizable boundaries, a loop for example, ora* *25 call to a -- even a binary has the equivalent of* *00036* *1 S. NIELSON* *2 function calls. And so you could identify those* *3 chunks as a functional block, and that would* *4 definitely illustrate the flow of the program.* *5 Q. Okay. So could you elaborate, how* *6 would a chunk like that illustrate the flow of* *7 the program?* *8 A. Well, for example, if you've got a* *9 loop, a loop changes the program flow. A branch* *10 instruction in assembly, the binary, would change* *11 the flow. Function call changes the flow.* Thus [**SN**] and [**SN** Depo] agree that a single instruction does not constitute a functional block, and I also agree. For example, a loop that performs a function essentially always has more than one instruction, and the term "flow" regarding a program inherently involves more than one instruction, because to be a flow, there has to be a place the flow comes from and goes to (i.e., from instruction to instruction). [**SN** para 62] asserts (**emphasis** added) "*As construed, "**Bodorin** discloses **examining** the suspected program for behaviors. Ex. 1003 at 4:51-54. These **behaviors** include **API calls** that **illustrate** the **function and execution flow** of the program. **A subset** of these behaviors are **extracted** as being "interesting."*" Thus [**SN**] does not assert that the "behaviors" of Bodorin constitute allow one to perform "identifying at least one functional block", but rather only that they allow one to "***illustrate** the **function and execution flow** of the*

Claims of '344	Per [SN] regarding '305	My response
		program". Illustrating is not the same as identifying. Illustration starts with the thing to be illustrated (i.e., the cause, in this case the functional block) and produces the illustration (i.e., the effect, in this case a sequence of system calls and/or instructions) as a way to show the thing being illustrated. Identifying in this context (from the behavior) is the opposite of illustration in that it requires we start with the effect (i.e., the behavior) and produce the cause (i.e., the functional block). But causality does not work backwards in this way because many causes may produce the same effects. For example, two completely different program sources (e.g., lisp programs) with different functional blocks, may produce identical or different byte code sequences, and those byte code sequences may be interpreted using the same sequences of machine instructions, and may make exactly the same API calls in the same sequence. The original sources and intermediate byte codes being executed thus have different functional blocks, and they cannot be differentiated based solely on behaviors because they have identical behaviors. Further, only an "**interesting**" "subset" of behaviors are identified by Bodorin. [SN Depo P40L6-P48L9] states, in pertinent parts: *6 Q. Could you point out where in Bodorin* *7 it talks about that?* *8 A. Yes. Column 4, starting in line 39,* *9 it says: "Each dynamic behavior evaluation* *10 module, such as dynamic behavior evaluation* *11 module 204, represents a virtual environment,* *12 sometimes called a sandbox, in which the code* *13 module 212 may be 'executed.'...* *23 Q. All right. So how does the -- Excuse* *24 me.* *25 How does the -- How are the dynamic* *00041* *1 S. NIELSON* *2 behaviors evaluated and recorded?* *3 A. So as you continue reading into the*

Claims of '344	Per [SN] regarding '305	My response
		next paragraph, it says: "As the code module 212 executes within the dynamic behavior evaluation module 204, the dynamic behavior evaluation module records 'interesting' behaviors exhibited by the code module. Interesting behaviors are those which a user or implementer of the malware detection system 200 has identified as interesting..." *And so if you look over at Table A, it lists the sorts of behaviors, as an example only it says, that may be interesting. These would be examples of system calls.* *Q. All right. So this table is a table of behaviors that are observed; is that correct?* *A. Well, so -- again, these are -- these are examples of system calls. You'll notice there are some even that we saw in the '344 patent, like GetModuleFileNameA. These are a standard call for Windows, for example.* *And so what you've got is you have a virtual environment where it's executing, and because you control the virtual environment, you* *00042* *S. NIELSON* *see every system call. So as system calls are going by, when you see one that is interesting, you pull it out and record it.* *Q. All right. You mentioned that this is one way that Bodorin teaches observing the interesting behaviors.* *A. Yes.* *Q. Does Bodorin disclose observing interesting behaviors in any other way, or is this the only embodiment?* *MR. LIU: Objection; vague. Assumes facts not in evidence.* *THE WITNESS: So Bodorin talks about a dynamic behavior evaluation module. And so I'm not aware of any way other than dynamic behavior evaluation. But this particular example given here -- I'm sorry. This particular example given here would actually cover -- it's a -- a sandbox is a broad term. Let me show you what I'm talking about.* *If you look in Column 4 at line 30, it points out that there are different types of code modules that can be evaluated. And so* *00043* *S. NIELSON* *it mentions as an example a Microsoft Windows executable. And what it's showing in Table A would be the sort of things that you would find in a Microsoft Windows executable. But then it lists a Microsoft.net executable and a Java applet or application. Those wouldn't have the same type of API calls. Java programs use what's known as a Java virtual machine, and their access to the system looks different.* *So in any of these examples you can have a virtual machine -- I'm sorry, a sandbox, a virtual environment in which you can observe the behaviors dynamically because the software has to interact with the system somewhere. And as long as you can see how it's interacting with the system, then you can see its execution.* *BY MR. PANNO:*

Claims of '344	Per [SN] regarding '305	My response
		21 Q. Okay. So that's an example there. So *22 you have Windows executable which has one type of 23 command- one command for one action. You have* *24 a Java applet or application that has a different* *25 command for the same action; is that correct?* *00044* *1 S. NIELSON* *2 MR. LIU: Objection; compound.* *3 Mischaracterizes the testimony.* *4 THE WITNESS: So the API calls that* *5 you would see on Windows would even be* *6 different on maybe a different operating* *7 system, but definitely different to a Java* *8 application. But you would be able to -- in* *9 all cases you would have something like open* *10 a file, write to a file, execute a file;* *11 they might just have different names and* *12 formats.* *13 BY MR. PANNO:* *14 Q. Okay. So would Bodorin be able to* *15 detect open a file for different languages having* *16 different commands for open a file?* *17 MR. LIU: Objection; assumes facts not* *18 in evidence. Vague.* *19 THE WITNESS: Well, I would assume* *20 that one of skill in the art, reading the* *21 Bodorin patent, would be able to take the* *22 example described for the Windows executable* *23 and be able to do that exact same thing for* *24 a Java applet.* *25 BY MR. PANNO:* *00045* *1 S. NIELSON* *2 Q. Okay. And so if the user considered* *3 open file to be an interesting behavior --* *4 A. Yes.* *5 Q. -- the user would be able to configure* *6 a system based on the teachings of Bodorin to* *7 identify open a file in a Java program, a Java* *8 applet?* *9 A. I believe so, yes.* *10 Q. And does Bodorin explain what the* *11 behavior signature for Java or anything other* *12 than the example of Table A would look like?* *13 MR. LIU: Objection; vague.* *14 THE WITNESS: So Table A isn't* *15 actually the behavior signature. Table A* *16 gives examples of APIs that may be* *17 interesting. It does give an example of* *18 what a behavior signature might look like in* *19 Tables 5A and 5B.* *20 BY MR. PANNO:* *21 Q. Okay. So the signature, for example,* *22 of Figure 5A --* *23 A. Yes.* *24 Q. -- could you point out what part of* *25 the table illustrates the interesting behavior in* *00046* *1 S. NIELSON* *2 Figure 5A?* *3 A. Yes. So what you have is -- You see* *4 the columns there?* *5 Q. Yes.* *6 A. So on the left-handmost column, it* *7 refers to it as a behavior token but it maps to* *8 one of these APIs. And so you can see there* *9 Internet read file or Internet open URL and write* *10 file. Those correspond to the interesting*

Claims of '344	Per [SN] regarding '305	My response
		11 behaviors. *12 Q. Okay. So if Bodorin is used with a* *13 Java program that has a command that causes a* *14 file to be written, would the behavior signature* *15 look like the third behavior signature in the* *16 third line in Figure 5A?* *17 MR. LIU: Objection; incomplete* *18 hypothetical.* *19 THE WITNESS: I think so. The text in* *20 Bodorin, if we can turn back to the text for* *21 a minute, it talks about these figures in* *22 Column 7. So -- Excuse me.* *23 So what it says here is it says that* *24 these figures starting in line 16, "are* *25 block diagrams illustrating exemplary* *00047* *1 S. NIELSON* *2 behavior signatures of hypothetical code* *3 modules, such as the behavior signature 210* *4 described above."* *5 Let me skip down to 23. "As* *6 illustrated in both behavior signature 500* *7 and behavior signature 512, each behavior* *8 includes three elements; a behavior token,* *9 such as behavior token 502..." And let me* *10 just stop there.* *11 So the behavior token was what we were* *12 looking at before, the InternetOpenUrl,* *13 Internet read file and write file.* *14 And here's the interesting part. If* *15 we skip down to 32, it says: "A behavior* *16 token, such as behavior token 502, is used* *17 to identify the particular 'interesting'* *18 behavior recorded by the dynamic behavior* *19 evaluation module..."* *20 So what this says is that your dynamic* *21 evaluation module would find the stuff like* *22 what's described in Table A and it would say* *23 oh, look, I have a Windows write file API* *24 command. And then when it goes into the* *25 Figure 5A, it gets replaced by a more* *00048* *1 S. NIELSON* *2 generic behavior token called write file.* *3 So in answer to your original* *4 question, yes, I would imagine a Java* *5 signature could conceivably look just like* *6 this because whatever Java's call for write* *7 file might look like, it could be replaced* *8 with the write file behavior token that you* *9 see here.* [**SN** Depo] agrees that only an interesting subset of behaviors are identified in Bodorin, and that different sequences of instructions executed in different languages may produce the same “behavior” for Bodorin. Thus [SN Depo] agrees that observing system call behaviors do not on their own identify functional blocks and that effect does not imply cause in this situation and thus the functional block of '344 is not what is anticipated by

Claims of '344	Per [SN] regarding '305	My response
		Bodorin. Effect does not imply cause and thus behaviors do not imply functional blocks.
[1C] identifying one or more genes each describing one or more of the at least one functional block and the at least one property of the software as a sequence of APIs and strings;	[**SN** para 63] continues (**emphasis** added): *(63.) Next, Bodorin discloses identifying one or more genes each describing one or more of the at least one functional block and the at least one property of the software as a sequence of APIs and strings. As explained above, Bodorin discloses the extraction of "interesting behaviors" that are based on APIs and strings. These "interesting behaviors" are "genes," construed as "a piece of functionality or property of a program," that describe one or more of the at least one functional block and the at least one property of the software as a sequence of APIs and strings.*	The use of the term "sequence" and the plurals in [**1C**] are definitive in that "a sequence of APIs and strings;" is a plurality and not a single API or a single string and no construction has been made that refutes or addresses this limitation as anything else. [**SN** para 63] asserts "*Bodorin discloses the extraction of "interesting behaviors" that are based on APIs and strings.*" and as detailed for [1B], this is not the same as "identifying" "genes" "describing" "at least one functional block" as required by the limitation of [1C], in this case because "describing" a "functional block" is not the same as what Bodorin discloses. Rather, Bodorin discloses "*examining the suspected program for behaviors.*" per [**SN** para 62] when actually referencing '305. As detailed above, behaviors do not imply functional blocks because effect does not imply cause in this context, as agreed in [**SN** Depo].
[1D] matching the one or more genes against one or more of the number of classifications using a processor;	[**SN** para 64] continues (**emphasis** added): *(64.) Next, Bodorin discloses matching the one or more genes against one or more of the number of classifications using a processor and classifying the software based on the matching to provide a classification for the software. Ex. 1003 at 5:27-43. If the extracted behavior signature is matched in the store, then the software is classified as malware. Ex. 1003 at 5:44-49.* The references state:	[**SN** para 64] asserts but fails to show "matching the one or more genes against one or more of the number of classifications using a processor". This is not what is disclosed in ['305 C5L27-43]. In particular, and without limit, it fails to disclose "classifications based on groupings of genes" of [**1A**] for the same reasons that this is not shown in [**1A**]. In more detail (**emphasis** added): "*the behavior signature comparison module 206 takes **the behavior signature** and compares it against **behavior signatures of known malware** that are stored in the*

Claims of '344	Per [SN] regarding '305	My response
	"Once the behavior signature 210 has been generated, the behavior signature comparison module 206 takes the behavior signature and compares it against behavior signatures of known malware that are stored in the malware behavior signature store 208. The malware behavior signature store 208 is a repository of behavior signatures of known malware. However, unlike signature files of antivirus software 104, behavior signatures stored in the malware behavior signature store 208 are based on exhibited behaviors of malware. As such, even though all variable names and routine names within source code for malware are modified by a malicious party, the exhibited behaviors of the malware remain the same, and will be detected by the malware detection system 200 when the behavior signature comparison module 206 matches a code module's behavior signature 210 to a known malware's behavior signature in the malware behavior signature store 208." *"According to aspects of the present invention, if the behavior signature comparison module 206 is able to completely match the behavior signature 210 against a known malware behavior signature in the malware behavior signature store 208, the malware detection system 200 will report that the code module is malware."*	*malware behavior signature store 208*" shows that the result of the code analysis of the running program is a single signature, but that the known malware comparison compares to multiple "signatures", thus not disclosing that there is a single signature in the store for each identified known malware item. While the second paragraph discloses (**emphasis** added) "*According to aspects of the present invention, **if** the behavior signature comparison module 206 **is able to completely match** the behavior signature 210 **against a known malware behavior signature** in the malware behavior signature store 208, the malware detection system 200 will report that the code module is malware*.", this, in my understanding of patent language does not restrict the match via "***a known malware behavior signature***" to a single such signature,. Nor does the disclosure indicate that this is the only method of matching. Thus a "behavior signature" consisting of only one gene rather than "a grouping of genes" of [**1A**] matching a "***known malware behavior signature***" does not demonstrate that the store contains classifications based on "a grouping of genes", nor has it been shown to be inherent. Indeed it is not inherent because the signature does not require "groupings of genes" be the basis for the classification. This [**SN**] fails to show that '305 anticipated [**1D**] of '344.
[1E] classifying the software based on the matching to provide a classification for the software; and	*... and classifying the software based on the matching to provide a classification for the software. Ex. 1003 at 5:27-43. If the extracted behavior signature is matched in the store, then the software is classified as malware. Ex. 1003 at 5:44-49.*	

Claims of '344	Per [SN] regarding '305	My response
[1F] notifying a user of the classification of the software.	[**SN** para 65] states "*(65.) Finally, Bodorin discloses notifying a user of the classification of the software. Ex. 1003 at 5:44-49.*" This reference states: *"According to aspects of the present invention, if the behavior signature comparison module 206 is able to completely match the behavior signature 210 against a known malware behavior signature in the malware behavior signature store 208, the malware detection system 200 will report that the code module is malware."*	[SN] However, the actual wording of the cited part of '305 does not disclose "notifying a user" of anything. There is a report generated, but this report is not shown to notify "the user" as referenced and this is not inherent. Reporting might, for example, be made to the manufacturer of the mechanism, a systems administrator, an automated response system, or elsewhere without the user being informed at all. In fact, the method of '305 does not require that there be any user. It could simply be a mechanism used within a data center, for example at a firewall. [**SN**] does not tell us anything different. This [**SN**] fails to show that '305 anticipated [**1F**] of '344.
	and then "**Accordingly, Claims 1 and 2 are anticipated by Bodorin.**"	Per my response, [**SN**] fails to show that '305 anticipated Claim 1 of '344
Claim 2		
[Claim 2] 2. The method of claim 1, wherein		Per my response to Claim 1, Claim 2 is not anticipated because it requires that Claim 1 be anticipated, and [**SN**] fails to show that.
[2A] the classification for the software includes malware.		
Claim 10		
[Claim 10] 10. The method of claim 1, further comprising		Per my response to Claim 1, Claim 10 is not anticipated because it requires that Claim 1 be anticipated, and [**SN**] fails to show that. In addition, [**SN**] fails to claim or show obviousness for Claim 1 of '344, and thus cannot show obviousness for Claim 10.
[10A] removing	[**SN** para 66] states:	It is unclear to me is how '305, which,

Claims of '344	Per [SN] regarding '305	My response
the software [10B] when the software is classified as malware or unwanted software.	*(66.) Claims 10 and 13 of the '344 Patent recite limitations that relate to removing and blocking access to identified malware. Although the malware detection system disclosed in Bodorin does not include functionality that removes and blocks access to identified malware, it would have been obvious to a person of ordinary skill in the art at the time of the '344 priority date.* *(67.) Since the creation of signature scanners more than a decade and a half before the '344 Patent, the stated advantage of a scanner over a behavior blocker or other defensive mechanism was that the scanner could proactively prevent damage. Consider this discussion of scanners from 1995:* *More generic anti-virus tools [than scanners]... can detect a virus... only after the virus has run an infected other programs. In many cases, the risk of allowing a virus to run on your computer is just not affordable. Using a heuristic scanner... allows detection of most new viruses... before an infected program is copied to your system and executed. Ex. 1019 at 227 (emphasis in original).* *(68.) Those of skill in the art would recognize that there would be little advantage to Bodorin if it did not proactively remove and/or block access to identified malware.* *(69.) Unsurprisingly then, there is support in view of Bodorin itself to add such functionality. This is because there is explicit teaching, suggestion, and motivation in Bodorin to combine. Bodorin explains that existing anti-virus software known in the art includes*	per [**SN**] "**does not include functionality that removes and blocks access to identified malware**" and per [**SN**] discloses only prior art that was well known prior to the submission of '344, can render obvious what it does not add anything new to. If it was obvious with '305 it was obvious without '305 and obviousness without '305 was not instituted. '305 contributes nothing to '344 relevant to Claim 10 that was not known without '305. [**SN**] asserts the basis for obviousness as "(i) determine the scope and content of the prior art; (ii) ascertain the differences between the prior art and the claims at issue; (iii) resolve the level of ordinary skill in the pertinent art; and (iv) consider objective evidence of non-obviousness." By this very standard, there is nothing in [**SN**] that discloses anything new or different regarding "removing the software". The assertions of [**SN**] fail to recognize that '305 does not discuss the nature of the source of the "code module". The source of any "code module" from the perspective of '305 is unavailable to the method of '305, and thus there is no way in which '305 could be expected to be revealing with regard to the removal of any such "code module". Indeed '305 has no capacity to remove any such thing as it is not indicated to have access to such a thing except as provided to it by an external mechanism not part of '305. [**SN**] asserts: *Ex. 1003 at 4:2-4. Bodorin further explains that "when the present invention is used in combination with current anti-virus software, it may be beneficial to use*

Claims of '344	Per [SN] regarding '305	My response
	functionality to remove or block access to 26 known malware. Ex. 1003 at 1:44-61. Bodorin then explicitly states that it would be advantageous to combine the dynamic malware detection system it discloses with prior art anti-virus software. Ex. 1003 at 3:58-4:12. Specifically, Bodorin states "while the malware detection system may be used as a stand-alone product, it may also be used in conjunction with current anti-virus software." Ex. 1003 at 4:2-4. Bodorin further explains that "when the present invention is used in combination with current anti-virus software, it may be beneficial to use the anti-virus software's signature matching techniques as a first step in securing a computer from malware, before turning to the malware detection system described herein." Ex. 1003 at 4:8-12. Accordingly, one of ordinary skill reading Bodorin would be motivated to combine its malware detection system with prior art anti-virus software that removes and blocks access to identified malware. *(70.) In connection with the above, I have reviewed, had input into, and endorse as if set forth fully herein the claim charts in the accompanying petition showing that each element of claims 10 and 13 are obvious in light of Bodorin itself to a person of ordinary skill in the field. Accordingly, Claims 10 and 13 are obvious in light of Bodorin itself.*	*the anti- virus software's signature matching techniques as a first step in securing a computer from malware, before turning to the malware detection system described herein." Ex. 1003 at 4:8-12. Accordingly, one of ordinary skill reading Bodorin would be motivated to combine its malware detection system with prior art anti-virus software that removes and blocks access to identified malware.* Also, it seems clear that in '305, this example actually teaches away from [**10A**] because, in essence, it states that the other methods where you might, for example, remove software known to be malware, should be used before the methods of '305, and that '305 should be used after such methods. Presumably, if there were other methods that were able to remove such malware, detecting the items as malware prior to the use of such methods would allow those methods to perform such removal. But because '305 asserts it should be used after such methods have failed to identify malware, and because it operates during execution and not prior to such execution, it is presumably not suitable to such removal. Also, is such other software used prior to the method of '305 were to remove the code module, then the method of '305 would not have access to the code module because it would have been removed. Further, a *POSITA* of [**SN**] would not have the requisite knowledge to render removal of software as part of detection of functional blocks of software containing genes indicative of malware by understanding that examination of execution of portions of software containing behaviors indicative of malware obvious even in

Claims of '344	Per [SN] regarding '305	My response
		light of the fact that other methods prior to the method of '305 had the capacity to remove malicious software. Obviousness of Claim 10 requires more than recognizing Claim 10 follows from Claim 1 in light of prior art, which has not been shown. It also requires recognizing Claim 1 from the prior art, and this has not been shown. I conclude that [**SN**] does not provide support for obviousness of [Claim 10] in light of '305 and that '305 further teaches away from the method of Claim 10.
Claim 13		
13. The method of claims 1, further comprising		Per my response to Claim 1, Claim 13 is not anticipated because it requires that Claim 1 be anticipated, and [**SN**] fails to show that. In addition, [**SN**] fails to claim or show obviousness for Claim 1 of '344, and thus cannot show obviousness for Claim 13.
[13A] blocking access to the software [13B] when the software is classified as malware or unwanted software.	[**SN** para 66] states: *(66.) Claims 10 and 13 of the '344 Patent recite limitations that relate to removing and blocking access to identified malware. Although the malware detection system disclosed in Bodorin does not include functionality that removes and blocks access to identified malware, it would have been obvious to a person of ordinary skill in the art at the time of the '344 priority date.* *(67.) Since the creation of signature scanners more than a decade and a half before the '344 Patent, the stated advantage of a scanner over a behavior blocker or other defensive mechanism was that the scanner could proactively prevent damage. Consider this*	It is unclear to me is how '305, which, per [**SN**] "**does not include functionality that removes and blocks access to identified malware**" and per [**SN**] discloses only prior art that was well known prior to the submission of '344, can render obvious what it does not add anything new to. If it was obvious with '305 it was obvious without '305 and obviousness without '305 was not instituted. '305 contributes nothing to '344 relevant to Claim 13 that was not known without '305. [**SN**] asserts the basis for obviousness as "(i) determine the scope and content of the prior art; (ii) ascertain the differences between the prior art and the claims at issue; (iii) resolve the level of ordinary skill in the

Claims of '344	Per [SN] regarding '305	My response
	discussion of scanners from 1995: *More generic anti-virus tools [than scanners]... can detect a virus... only after the virus has run an infected other programs. In many cases, the risk of allowing a virus to run on your computer is just not affordable. Using a heuristic scanner... allows detection of most new viruses... before an infected program is copied to your system and executed. Ex. 1019 at 227 (emphasis in original).* *(68.) Those of skill in the art would recognize that there would be little advantage to Bodorin if it did not proactively remove and/or block access to identified malware.* *(69.) Unsurprisingly then, there is support in view of Bodorin itself to add such functionality. This is because there is explicit teaching, suggestion, and motivation in Bodorin to combine. Bodorin explains that existing anti-virus software known in the art includes functionality to remove or block access to 26 known malware. Ex. 1003 at 1:44-61. Bodorin then explicitly states that it would be advantageous to combine the dynamic malware detection system it discloses with prior art anti-virus software. Ex. 1003 at 3:58-4:12. Specifically, Bodorin states "while the malware detection system may be used as a stand-alone product, it may also be used in conjunction with current anti-virus software." Ex. 1003 at 4:2-4. Bodorin further explains that "when the present invention is used in combination with current anti-virus software, it may be beneficial to use the anti-virus software's signature matching techniques as a first step in securing a computer from malware,*	pertinent art; and (iv) consider objective evidence of non-obviousness." By this very standard, there is nothing in [**SN**] that discloses something new or different regarding "blocking access to the software" Also, the assertions of [**SN**] fail to recognize that '305 does not discuss the nature of the source of the "code module". The source of any "code module" from the perspective of '305 is unavailable to the method of '305, and thus there is no way in which '305 could be expected to be revealing with regard to the blocking access to any such "code module". It further seems clear that in '305, this example actually teaches away from [**10A**] because, in essence, it states that the other methods where you might, for example, block access to software known to be malware, should be used before the methods of '305, and that '305 should be used after such methods. Presumably, if there were other methods that were able to block access to such malware, detecting the items as malware prior to the use of such methods would allow those methods to perform such blocking of access. But because '305 asserts it should be used after such methods have failed to identify malware, and because it operates during execution and not prior to such execution, it is presumably not suitable to such blocking of access. Also, is such other software used prior to the method of '305 were to block access to the code module, then the method of '305 would not have access to the code module and not be able to operate. Further, a *POSITA* of [**SN**] would not have the requisite knowledge to render blocking access to software as

Claims of '344	Per [SN] regarding '305	My response
	before turning to the malware detection system described herein." Ex. 1003 at 4:8-12. Accordingly, one of ordinary skill reading Bodorin would be motivated to combine its malware detection system with prior art anti-virus software that removes and blocks access to identified malware. *(70.) In connection with the above, I have reviewed, had input into, and endorse as if set forth fully herein the claim charts in the accompanying petition showing that each element of claims 10 and 13 are obvious in light of Bodorin itself to a person of ordinary skill in the field. Accordingly, Claims 10 and 13 are obvious in light of Bodorin itself.*	part of detection of functional blocks of software containing genes indicative of malware by understanding that examination of execution of portions of software containing behaviors indicative of malware obvious even in light of the fact that other methods prior to the method of '305 had the capacity to block access to malicious software. Obviousness of Claim 13 requires more than recognizing Claim 13 follows from Claim 1 in light of prior art, which has not been shown. It also requires recognizing Claim 1 from the prior art, and this has not been shown. I conclude that [**VM**] does not provide support for obviousness of [Claim 13] in light of '305 and that '305 further teaches away from the method of Claim 13.

Claims 1 and 2 of '344 are not shown to be anticipated by Bodorin by Fortinet's expert.

Claims 10 and 13 of '344 are not shown to be obvious over Bodorin by Fortinet's expert.

In addition, [**SN**-Depo P14L8-P19L4] states [P18L14-P18L25] "I can certainly imagine there would be some person with a master's degree in electrical engineering out there somewhere who could not do this.". Thus Foritnet's expert agrees that Claims 10 and 13 are not obvious.

Ground 3

[**SN**] asserts and the board has instituted proceedings based on the assertion that '344 Claims 16 and 17 are rendered obvious to a *POSITA* of [**SN**] based on the combination of patents 7,373,664 ('664) and '305 and 7,448,084 ('084). As I understand this, in order for this to be true, whatever is not obvious from '305 must be then rendered obvious by '664 or '084. In this case, the board did not institute on the grounds that "Claims 16 and 17 are obvious in light of Kissel in combination with Apap" even though that was identified in [**SN**]. For that reason, something must be added to '305 combined with '664 by '084 to render '344 obvious to a *POSITA* of [**SN**].

'084 is titled "System and methods for detecting intrusions in a computer system by monitoring operating system registry accesses" and is specifically, per its abstract, about

> *A method for detecting intrusions in the operation of a computer system is disclosed which comprises gathering features from records of normal processes that access the files system of the computer, such as the Windows registry, and generating a probabilistic model of normal computer system usage based on occurrences of said features. The features of a record of a process that accesses the Windows registry are analyzed to determine whether said access to the Windows registry is an anomaly. A system is disclosed, comprising a registry auditing module configured to gather records regarding processes that access the Windows registry; a model generator configured to*

> *generate a probabilistic model of normal computer system usage based on records of a plurality of processes that access the Windows registry and that are indicative of normal computer system usage; and a model comparator configured to determine whether the access of the Windows registry is an anomaly.*

It is unclear to me how this reference and subject matter relates to '344 from the perspective of a *POSITA* of [**SA**].

In this case, [**SN**] asserts that

I continue from [**SN** para 91] (**emphasis** added):

> *(91.) As explained above, Bodorin discloses a malware detection system that generates behavior signature from programs, which corresponds to the "genes" in the claims of the '344 Patent, and compares it against behavior signatures of known malware in the malware behavior signature store, which corresponds to the "library" in the claims of the '344 Patent.* ***Bodorin does not disclose updating the malware behavior signature store by testing the generated behavior signature for false positives and storing the tested generated behavior signature, which is claimed in claim 16.*** *However, as explained in detail above, Kissel in combination with Apap disclose every limitation of claim 16.*

While I disagree with much of the premise of [**SN** para 91], I do agree that "*Bodorin does not disclose updating the malware behavior signature store by testing the generated behavior signature for false positives and storing the tested generated behavior signature, which is claimed in claim 16.*" The assertion that "*Kissel in combination with Apap disclose every limitation of claim 16.*" is contrary to the conclusion of the Board which did not institute on this basis.

I continue from [**SN** para 92] (**emphasis** added):

> *(92.) It would have been obvious to a person of ordinary skill in the art to combine Bodorin with Kissel and Apap. There was explicit teaching, suggestion, and motivation in the prior art references themselves to combine. Claim 16 discloses a method for testing a generated set of genes for false-positives and storing the set of genes.* ***Kissel discloses this method as a way of updating the library****. Ex. 1004 at 6:7-8:62. Although not explicitly disclosed in Kissel,* ***Apap discloses testing for false-positives using one or more reference files****. Ex. 1006 at 15:44-51. Although Bodorin does not explicitly disclose a method for updating its malware behavior signature store,* ***Bodorin nevertheless explicitly states that it is desirable to update the store.*** *Specifically, Bodorin states "[i]t is anticipated that the malware behavior signature store 208 will be periodically updated with behavior signatures of known malware. The malware behavior signature store 208 should be especially updated as the more rare, 'new' malware is discovered." Ex. 1003 at 6:1-5. Accordingly,* ***one of ordinary skill in the art reading this passage in Bodorin would be motivated to combine the disclosures of Bodorin with a method for updating the malware behavior signature store, which is exactly what is disclosed in Kissel and Apap.***

The notion that a *POSITA* of [**SN**] seeking new methods for improving on dynamic code analysis for malware (i.e., '305) would decide to examine, in detail, methods for "Proactive protection against e-mail worms and spam" (i.e., '664) and would also decide to examine "monitoring operating system registry accesses" (i.e., '084), and after examining those would see as obvious Claim 16 of '334 (i.e., analyzing groupings of genes found in programs, combining these genes into sets of genes, testing them for false positives, defining classifications based on these outcomes, and storing those results in a classification library) seems ridiculous on its face.

Recall that a *POSITA* of [**SN**] need have no background in computer security or malware detection, and would have to somehow relate monitoring Windows registry access (about which they may know nothing and not even be aware of), to worm and spam detection, which are largely about network-based events rather than within-processor events, (and about which they may know nothing and not even be aware of), and relate this to dynamic code analysis, which has nothing obvious to do with Windows registry access or detection of network-related spam and worm attacks (and about which they may know nothing and not even be aware of) in order to even consider such a combination. Then, they would have to dig into the specifications of these patents to extract the referenced parts, which [**SN**] does not asserts are disclosed in the claims, recognize that these were somehow related to each other, and come up with a completely different application of those combined parts. [**SN**] asserts this as "obvious", and I conclude it is not obvious at all.

By my understanding, as detailed above:

- a patent composed of several elements may not be proved obvious merely by demonstrating that each of its elements was independently known in the art
- there must be an appropriate articulation of a reason to combine the elements from the prior art in the manner claimed
- obviousness cannot be based on a hindsight combination of components selected from the prior art using the patent claims as a roadmap

The claimed combination of [**SN**] is a hindsight analysis used to combine components of selected prior art using the patent claims as a roadmap. Specifically, it appears that the analysis starts with the list of claim elements for Claim 16 and seeks combinations of prior art references that fulfill this roadmap. For example, and without limit:

- [**SN**] does not assert that '664 identifies the sequence of steps involved in '344 only leaving out or slightly missing one or two steps. In fact, the method of '664 identifies (per Claim 1) the use of "a plurality of emails" for detection, where the plurality of emails are compared to each other to find common features and using a weighting scheme to determine whether a threshold of likelihood it met. This is not the method of '344 claim 16 missing or slightly differing in a few steps, and this is not even the basis for combination cited. The basis cited is ['644 C6L7-C8L62], an extensive detailing of setting thresholds and scoring methods associated with what is in essence similarity detection where similarity is relative to some built-in or learned mechanisms associated with malware.

  The other stated basis is from ['084 C15L44-51] which states

  - *The first exemplary anomaly detection algorithm discussed above in equations (1)-(3) were trained over the normal data. Each record in the testing set was evaluated against this training data. The results were evaluated by computing two statistics: the detection rate and the false positive rate. The performance of the system was evaluated by measuring detection performance over processes labeled as either normal or malicious.*

  In context, it is clear to see that the extracted part of '084 relates to detection of anomalies and false positives associated with anomaly detection, and not false positives associated with detecting malware. Anomalies can be detected caused by all manner of things that have nothing to do with malware, and detection of differences from normal is a completely different thing than detection of similarity to known undesired.

Thus [**SN**] asserts that there are specific unfulfilled niches of the method of '344 Claim 16 in '305 from '305 that appear in other prior art and that could only reasonably be identified by a *POSITA* of [**SN**] once the elements forming the roadmap of '344 Claim 16 are known, if they could be identified at all. But this is not a permissible basis for the assertion of obviousness.

- [**SN**] does not assert that '084 identifies the sequence of steps involved in '344 only leaving out or slightly missing one or two steps. In fact, the method of '084 identifies (per Claim 1) a method of "detecting intrusions", not identifying or categorizing malware, "generating a probabilistic model of normal computer system usage", which is not a method of characterizing malware, "determining the likelihood of observing an event that was not observed during the gathering of features from the records of normal processes", which is oriented toward detecting anomalies in the operating environment and not "malicious programs", and "analyzing features from a record of a process that accesses the operating system registry to detect deviations from normal computer system usage to determine whether the access to the operating system registry is an anomaly", which is detection of anomalies from records and not malware from behaviors. This is not the method of '344 claim 16 missing or slightly differing in a few steps, and the claim is not even the basis for combination cited. The basis cited is ['084 C15L44-51] which states:
    - *The first exemplary anomaly detection algorithm discussed above in equations (1)-(3) were trained over the normal data. Each record in the testing set was evaluated against this training data. The results were evaluated by computing two statistics: the detection rate and the false positive rate. The performance of the system was evaluated by measuring detection performance over processes labeled as either normal or malicious.*

  In context, it is clear to see that the extracted part of '084 relates to detection of anomalies and false positives associated with anomaly detection, and not false positives associated with detecting malware. Anomalies can be detected caused by all manner of things that have nothing to do with malware, and detection of differences from normal is a completely different thing than detection of similarity to known undesired.

  Thus [**SN**] asserts that there are specific unfulfilled niches of the method of '344 Claim 16 in '305 from '305 that appear in other prior art and that could only reasonably be identified by a *POSITA* of [**SN**] once the elements forming the roadmap of '344 Claim 16 are known, if they could be identified at all. But this is not a permissible basis for the assertion of obviousness. Indeed, in this case, the unfulfilled elements asserted as fulfilled are not even the same sorts of things identified in '344.

[**SN**] does assert that '305 identifies the sequence of steps involved in '344 only leaving out or slightly missing a few steps.

[**SN** para 92] asserts motivation to combine as (**emphasis** added in pertinent parts):

> *(92.) It would have been obvious to a person of ordinary skill in the art to combine Bodorin with Kissel and Apap. There was explicit teaching, suggestion, and motivation in the prior art references themselves to combine. Claim 16 discloses a method for testing a generated set of genes for false-positives and storing the set of genes.* ***Kissel discloses this method as a way of updating the library. Ex. 1004 at 6:7-8:62.*** *Although not explicitly disclosed in Kissel,* ***Apap discloses testing for false-positives using one or more reference files. Ex. 1006 at 15:44-51****. Although Bodorin does not explicitly disclose a method for updating its malware behavior signature store, Bodorin nevertheless explicitly states that it is desirable to update the store. Specifically,* ***Bodorin***

states "[i]t is anticipated that the malware behavior signature store 208 will be periodically updated with behavior signatures of known malware. The malware behavior signature store 208 should be especially updated as the more rare, 'new' malware is discovered." *Ex. 1003 at 6:1-5. Accordingly, one of ordinary skill in the art reading this passage in Bodorin would be* ***motivated to combine the disclosures of Bodorin with a method for updating the malware behavior signature store, which is exactly what is disclosed in Kissel and Apap.***

The facts of these references are as follows:

Ex. 1004 at 6:7-8:62 (i.e., ['644 C6L7-C8L62]) asserted as "a way of updating the library" (equations approximated, my comments indented and in not *italicized*) states:

> *If none of the feature instances extracted in 35 correspond to an entry in library 6, step 38 is entered, during which gateway 2 updates feature bin table 7 with one entry for each feature that has been extracted. As stated previously, an entry consists of a message ID and a time index. At step 39, gateway 2 checks the timestamps of all of the entries within feature bin table 7. If any entry has a timestamp outside a preselected window of time Q (which is the sample time used in calculating score S), that entry is removed from feature bin table 7 and placed into recently removed entry storage area 9, which stores those entries that have been removed from table 7 since the last time the method steps of FIG. 3 were executed.*
>
> > Feature bin table 7 contains a list of features, and is not the "blocked feature instance library" of block 6.
>
> *At step 40, all scores S in score table 8 are updated. At step 41, gateway 2 determines whether any of the recently updated scores exceed a preselected malicious threshold for that score, indicative of malicious code. If this malicious threshold has been crossed, library 6 at step 42 is updated with an entry corresponding to the offending feature instance. Then, at step 43, all e-mails in queue 5 containing this feature instance are deleted, and the method reverts to step 37.*
>
> > The "*library 6 at step 42*" is updated regrading an "*the offending feature instance*" - note that this is a single feature instance, whereas the method of '344 Claim 16 identifies "storing the set of genes and the software classification in the library." and this is "combining a plurality of genes that describe the software, thereby providing a set of genes;" - which is to say, the update identified in '644 of this library is distinctly different from that of '344 Claim 16.
> >
> > Also, as detailed from here forward, the method of '644 regards the identification of a threshold for detection, and deals with the determination and adaptation of the system to different threshold values and not the adaptation by adding of sets of genes to the library. This is unrelated to the classification identified in '344.
>
> *In one embodiment, a second threshold for each score S is also pre-established: a threshold representing not malicious code but a suspicion of malicious code. In this embodiment, at step 41, gateway 2 further asks whether this suspicion threshold has been crossed. For example, assume that the score S for a particular feature was 4 in the previous iteration of the method of FIG. 3, the present score S is 6, and the suspicion threshold is 5. Thus, the suspicion*

threshold has been crossed, and crossed in a positive (upwards) direction. Then, at optional step 44, X is adjusted, also in a positive direction because the suspicion threshold has been crossed in a positive direction. This increases the time that e-mails, including e-mails presently in queue 5 and e-mails to be placed into queue 5 later, are made to languish in queue 5. The rationale behind optional step 44 is that, while the score S is not deemed to be sufficiently high to make a declaration of malicious code, it is suspiciously moving in that direction, and therefore this warrants that time X be increased a bit. Every time the suspicion threshold has been crossed upwardly, X can be increased a bit more, up to a preselected maximum. Similarly, if the suspicion threshold is crossed in a negative direction, X can be decreased, but it should not be decreased below Q. When X is adjusted in step 44, it can be adjusted for just those e-mails having the feature instance whose score S crossed the suspicion threshold; or it can be adjusted for all e-mails.

Normally, the malicious threshold and the suspicion threshold are the same for each S, regardless of feature instance.

The method steps of the present invention can be carried out by hardware, firmware, and/or software modules 61 through 64 as illustrated in FIG. 6. Modules 61 through 64 may be contained within gateway 2, or, alternatively, may be contained within a separate computer coupled to gateway 2. First calculating means 61, which may comprise a central processing unit, registers, memory cells, and other conventional calculating means, performs step 35 for each incoming or outgoing e-mail that arrives at gateway 2: calculating feature vector 80. Second calculating means 62, which may comprise a CPU, registers, memory cells, and other conventional calculating means, performs steps 38 and 39 for each incoming or outgoing e-mail arriving at gateway 2: calculating at least one score S based upon said feature vector 80, each S being representative of a frequency of occurrence of an instance of one of the preselected features.

First calculating means 61 and second calculating means 62 may consist of the same module or modules. Alternatively, module 61 is coupled to module 62. Module 62 is coupled to module 63, and module 63 is coupled to module 64.

Determining means 63, which comprises conventional comparison logic, performs step 41: determining whether a score S exceeds a preselected malicious threshold representative of malicious code. Blocking means 64, comprising conventional computer logic, performs step 37: blocking the e-mail when a score S exceeds a preselected malicious threshold.

In one embodiment, score S consists of the following three components:

1. A number T of entries in table 7 in the current time window Q.

2. A decaying weighted value R representing recently removed entries from table 7.

3. A "memory" decaying weighted value P of all previously calculated score(s) S. Consequently, the total score S is calculated as follows:

$$S=T+w_R \cdot R+w_P \cdot P$$

wR and wP are weighting factors to parameterize the weighting of R and P.

To obtain T, module 62 simply counts the number of entries remaining in feature bin table 7 after removing all entries that fall outside of the evaluation time window Q.

In order to obtain R and P, the concept of a decay function d is introduced. This function is used as a weighting factor to weigh entries depending on their difference in time from another entry. This function has the following properties:

1. For time 0, its value is 1 (as its weight should be 1 for no time difference).

2. For a definable time difference h, its value is 0.5 (this allows for the definition of a "half decay point" h in time).

3. As a time difference t approaches infinity, its value approaches 0.

The following decay function dh(t) has these properties (with t being a time difference and h being the definable "half decay point" in time):

$$d_h(t) = h^t + h$$

To obtain R, module 62 uses the following equation:

$$R = \sum_i d_h(c - r_i) = \sum_i h^R(c - r_i) + h^R$$

Here, the ri are the times of all entries removed from the feature bin table 7, c is the current time, and i is the index of all removed entries since the last time S was updated. c−ri must be greater than Q, or else the entry wouldn't get into this expression.

To obtain P, module 62 uses the following equation:

$$P = d_h(c - c_{prev}) \cdot S_{prev} = S_{prev} \cdot h^P(c - c_{prev}) + h^P$$

Here, c is the current time, cprev is the time stamp of the previous (last) entry in feature bin table 7 (i.e., the time of the last score calculation for the bin), and Sprev is the previous (last) total score calculated for this feature instance. Initially, this third term P should be zero, because there is no previous score S. So, to accomplish this, we set Sprev=0 initially. This third term P takes into account just one entry deletion, the most recent deletion. We can have different h's for the second and third terms (hR and hP, respectively). It doesn't matter whether hR is greater than hP, hP is greater than hR, or they are the same.

Consequently, the total score S is calculated by module 62 as follows:

$$S = T + w^R \cdot \sum_{i=1}^{n} h^R(c - r_i) + h^R + w^P \cdot S_{prev} \cdot h^P(c - c_{prev}) + h^P$$

It should be noted that for the edge condition of multiple timestamps arriving at gateway 2 at exactly the same point in time, the following simplified expression results for the calculation of score S:

$$S = T + w^P \cdot S_{prev}$$

The effectiveness of this technique for calculating the frequency score S depends on appropriate values for wR, wP, hR, and hP. These values should be determined empirically, so that S can be tuned to provide meaningful results on real-world data. To determine these values empirically, one can set up a test network representing a (scaled-down) model of an enterprise 11 computer network. Then, a set of known e-mail worms (with their malicious payload removed) are introduced into this model network while the algorithm for

> *calculating S described above is running. The algorithm is modified to merely log the feature vectors 80 of all incoming and outgoing messages passing through the model e-mail gateway 2, along with their timestamps. The spreading characteristics of the e-mail worms are then captured in these feature vector/timestamp logs.*
>
> *Then, an offline analysis is performed on these logs to investigate an optimum combination of values for wR, wP, hR, and hP, as well as the time window size Q. Since this is a multi-parameter optimization problem that is further complicated by different spreading characteristics of different (known and unknown) e-mail worms, a multi-parameter non-linear best fit (in the least mean squares sense) optimization algorithm (such as a neural network) can advantageously be used. However, visual investigation of the worm spreading characteristics, along with manual "tweaking" of the scoring parameters, may result in a satisfactory set of parameters. This manual setting of the scoring parameters also gives system administrators the option to customize the behavior of the invention, in particular, the false positive threshold appropriate for their computing environment.*

As can be clearly seen from the details provided, '644 as cited does not discloses "a way of updating the library" of '344. It shows a way of updating a different kind of library with different contents in a different way, and most of it is irrelevant in every way to the specifics of '344.

Ex. 1006 at 15:44-51 (i.e., ['084 C15L44-51]) asserted as "testing for false-positives using one or more reference files" states (**emphasis** added):

> *The first exemplary anomaly detection algorithm discussed above in equations (1)-(3) were trained over **the normal data**. Each **record in the testing set was evaluated against this training data.** The **results were evaluated by computing** two statistics: the detection rate and **the false positive rate**. The performance of the system was evaluated by measuring detection performance over processes labeled as either normal or malicious.*

But this is not the "testing for false-positives using one or more reference files" of '344 Claim 16 or anything like it. In particular and without limit:

- '084 "false positive" is for anomaly detection and not for detection of malware – they are addressing different issues and merely using the same terms.
- "processes" of '084 are not "software" of '344.
- "the normal data" of '084 is not "reference files" of '344.
- "record in the testing" is not "gene information" or anything like it of '344.

Quote from '305

> *"[i]t is anticipated that the malware behavior signature store 208 will be periodically updated with behavior signatures of known malware. The malware behavior signature store 208 should be especially updated as the more rare, 'new' malware is discovered."*

I fail to see how a *POSITA* of [**SN**] would be motivated to combine different methods applying different sources of information to different ends in different contexts to produce Claim 16 of '344. I find this basis for motivation of combination to be inadequate to the standard as I understand it, to wit:

- all the claim limitations must be taught or suggested by the prior art
    - This is not shown by [**SN**].
- the use of known techniques to improve similar devices in the same way
    - This is not shown by [**SN**]. Different devices are improved in different ways using only the prior art identified.
- some teaching, suggestion, or motivation in the prior art would have led one of ordinary skill to modify the prior art reference teachings to arrive at the claimed invention.
    - A *POSITA* of [**SN**] would not be led to modify the prior art teachings to arrive at the invention without the hindsight of where '344 was going.
- The improvement required is more than the predictable use of prior art elements according to their established functions. Specifically, and without limit,
    - With regard to '084, a referenced required art for the combination is art regarding false positives related to anomaly detection. As such, any application not to anomaly detection is not according to its established functions. The art of '344 is not for anomaly detection, but rather for detection of false positives associated with detected sets of genes present and previously known. These are not the same function and thus obviousness is refuted.
    - With regard to '644, a referenced required art for the combination is art regarding adding single elements to a feature table and performing statistical analysis to determine what to place in or remove from the feature table. As such, any application not to that method is not according to its established functions. The method of '344 is not for this function, but rather for the function of “defining the software classification based on the set of genes; and storing the set of genes and the software classification in the library”. These are not the same function (i.e., defining a classification is not the same function as determining what to place in a table based on statistics) and thus obviousness is refuted.
- Similarly, a claim is not obvious if the application of a known technique is beyond the level of ordinary skill in the art.
    - While the *POSITA* of [**SN**] has a great deal of skill in many possibly relevant arts, the application of these techniques is not something that such a *POSITA* could be expected to do without substantial experimentation, particularly in the context of the remaining parts of '344 Claim 16.
- Furthermore, when the prior art teaches away from combining certain known elements, discovery of a successful means of combining them is not obvious.
    - As indicated earlier, the prior art teaches away from the method of '344, at least in that the prior art of '084 teaches that detecting anomalies detected statistically and adapting those things considered anomalies based on statistics avoids the problems of the area of art by stating ['084 Background] (**emphasis** added):
        - *Two conventional approaches to respond to malicious software include virus scanners, which **attempt to detect the malicious software**, and security patches that are created to repair the security “hole” in the operating system that the malicious software has been found to exploit. Both of **these methods for protecting hosts against malicious software suffer from drawbacks**. **While***

> ***they may be effective against known attacks, they are unable to detect and prevent new and previously unseen types of malicious software.*** *…*
>
> *Many virus scanners are signature-based, which generally means that* ***they use byte sequences or embedded strings in software to identify certain programs as malicious.*** *If a virus scanner's signature database does not contain a signature for a malicious program, the virus scanner is unable to detect or protect against that malicious program. In general, virus scanners require frequent updating of signature databases, otherwise the scanners become useless to detect new attacks. Similarly, security patches protect systems only when they have been written, distributed and applied to host systems in response to known attacks. Until then, systems remain vulnerable and attacks are potentially able to spread widely.*
>
> ***Frequent updates of virus scanner signature databases*** *and security patches are necessary to protect computer systems using these approaches to defend against attacks. If these updates do not occur on a timely basis, these systems remain vulnerable to very damaging attacks caused by malicious software. Even in environments where updates are frequent and timely, the systems are inherently vulnerable from the time new malicious software is created until the software is discovered, new signatures and patches are created, and ultimately distributed to the vulnerable systems. Since malicious software may be propagated through email, the malicious software may reach the vulnerable systems long before the updates are in place.*
>
> *Another approach is the use of* ***intrusion detection systems (IDS)****. Host-based IDS systems* ***monitor a host system and attempt to detect an intrusion****. In an ideal case,* ***an IDS can detect the effects or behavior of malicious software rather than distinct signatures of that software****. In practice, many of the commercial IDS systems that are in widespread use are signature-based algorithms, having the drawbacks discussed above.* ***Typically, these algorithms match host activity to a database of signatures which correspond to known attacks. This approach, like virus detection algorithms, requires previous knowledge of an attack and is rarely effective on new attacks.*** *However, recently there has been growing interest in the use of data mining techniques, such as anomaly detection, in IDS systems.* ***Anomaly detection algorithms may build models of normal behavior in order to detect behavior that deviates from normal behavior and which may correspond to an attack. One important advantage of anomaly detection is that it may detect new attacks, and consequently may be an effective defense against new malicious software.***

Thus '084 explicitly teaches that detecting and preventing "***new and previously unseen types of malicious software***" is not able to be done by the "***attempt to detect the malicious software***". However, '344 specifically addresses attempting to detect malicious software for cases where other prior art methods of virus detection fail. Throughout the background, '084 consistently teaches against the use of the methods of '344.

Thus discovery of a successful means of combining them is not obvious.

[**SN** para 93] asserts (**emphasis** added):

> *(93.) Furthermore, t**here was explicit teaching, suggestion, and motivation generally in the art to combine. New malware appear regularly and frequently. A major challenge for makers of anti-virus software is to keep its anti-virus software up to date so that it can recognize new malware before the new malware can infect users' computers. Thus, one of ordinary skill in the art,** in view of a system disclosed in Bodorin that generates behavior signature from programs and compares it against behavior signatures of known malware in the malware behavior signature store, would be motivated to keep the stored updated so that the system can detect the latest malware. Accordingly, one of ordinary skill in the art would be motivated to combine Bodorin with Kissel and Apap.*

[**SN**] essentially asserts that general knowledge of "***makers of anti-virus software***" constitutes motivation to combine by a *POSTIA* of [**SN**]. But a *POSITA* of [**SN**] is not asserted or shown to be or have the skills or motivations of "***makers of anti-virus software***". Thus [**SN**] applies the wrong standard.

[**SN** para 94] asserts (**emphasis** added):

> *(94.) In connection with the above, **I have reviewed, had input into, and endorse as if set forth fully herein the claim charts in the accompanying petition** showing that each element of claims 16 and 17 are disclosed by Bodorin and Apap, **and that**, as set forth above, **a person of ordinary skill in the field would have combined the teachings of the references as claimed**. Accordingly, Claims 16 and 17 are obvious in light of Bodorin in combination with Kissel and Apap.*

The relevant claim chart entries form the accompanying petition are addressed here.

Claims of '344	**Per [SN] citation to Petition**	**My response**
Claim 16		
16. A method for generating software classifications for use in classifying software, said method comprising:	Bodorin discloses that the malware behavior signature store should be updated. Ex. 1003 at 6:1-5. *Kissel discloses a method to update its blocked feature instance library for use the classifying malicious code. See Ex. 1004 at 6:7-8:62.* This refers to: *See also Ex. 1007 at ¶ 83.* This refers to: *(83.) As shown in steps 36-42 in Figs. 3A and 3B of Kissel and explained in column 6, lines 7 through 38, if the extracted feature vector is not in library, it is stored and a score associated with the extracted feature vector is generated. If the score exceeds a certain threshold, it indicates that*	The claim chart does not address "A method for generating software classifications for use in classifying software" "*Ex. 1004 at 6:7-8:62.*" refers to the details I refuted relative top [SN para 92] above and those refutations apply here. The cited reference does not show or address "A method for generating software classifications for use in classifying software" "*Ex. 1007 at ¶ 83.*" does not show or address "A method for generating software classifications for use in classifying software" I conclude that [**SN**] does not provide support for obviousness of the method of Claim 16 in its preamble.

Claims of '344	Per [SN] citation to Petition	My response
	this unknown extracted feature vector is associated with malware and the library is updated. Ex. 1004 at 6:7-38. Kissel further discloses calibrating the threshold to test whether these new extracted feature vectors are false-positives, i.e. not associated with malware. Ex. 1004 at 8:47-62.	
[**16A**] providing a library of gene information including [**16A1**] a number of classifications based on groupings of genes;	*Ex. 1003 at 5:27-43.* *See also Ex. 1007 at ¶¶ 59-61.*	The claim chart adds nothing to the prior anticipation argument for [**SN**] regarding anticipation under '305. Further, neither of the references identifies any claim of obviousness or any basis for obviousness. This [**VN**] fails to show that [**16A** and **16B**] are obvious. Further, as I identified above, the basis for obviousness is refuted.
[**16B**] identifying one or more genes each describing a functionality or a property of the software as a sequence of APIs and strings;	*Ex. 1003 at 5:27-43.* *See also Ex. 1007 at ¶¶ 59-61.*	The claim chart adds nothing to the prior anticipation argument for [**SN**] regarding anticipation under '305. Further, neither of the references identifies any claim of obviousness or any basis for obviousness. This [**VN**] fails to show that [**16A** and **16B**] are obvious. Further, as I identified above, the basis for obviousness is refuted.
[**16C**] combining a plurality of genes that describe the software, thereby providing a set of genes;	*Ex. 1003 at 5:27-43.* *See also Ex. 1007 at ¶¶ 59-61.*	The claim chart adds nothing to the prior anticipation argument for [**SN**] regarding anticipation under '305. Further, neither of the references identifies any claim of obviousness or any basis for obviousness. This [**VN**] fails to show that [**16C**] is obvious. Further, as I identified above, the basis for obviousness is refuted.
[**16D**] testing the	*Ex. 1004 at 6:7-28: If none of the*	Reference 1004 refers to '644. As

Claims of '344	Per [SN] citation to Petition	My response
set of genes for false-positives against one or more reference files using a processor;	*feature instances extracted in 35 correspond to an entry in library 6, step 38 is entered, during which gateway 2 updates feature bin table 7 with one entry for each feature that has been extracted. As stated previously, an entry consists of a message ID and a time index. At step 39, gateway 2 checks the timestamps of all of the entries within feature bin table 7. If any entry has a timestamp outside a preselected window of time Q (which is the sample time used in calculating score S), that entry is removed from feature bin table 7 and placed into recently removed entry storage area 9, which stores those entries that have been removed from table 7 since the last time the method steps of FIG. 3 were executed. At step 40, all scores S in score table 8 are updated. At step 41, gateway 2 determines whether any of the recently updated scores exceed a preselected malicious threshold for that score, indicative of malicious code. If this malicious threshold has been crossed, library 6 at step 42 is updated with an entry corresponding to the offending feature instance. Then, at step 43, all e-mails in queue 5 containing this feature instance are deleted, and the method reverts to step 37.* *Ex. 1004 at 8:47-62 (**emphasis** added): Then, an offline analysis is performed on these logs to investigate an optimum combination of values for wR, wP, hR, and hP, as well as the time window size Q. Since this is a multi-parameter optimization problem that is further complicated by different spreading characteristics of different (known and unknown) e-mail worms, a multi-parameter non-linear best fit*	discussed above: Feature bin table 7 contains a list of features, and is not the "blocked feature instance library" of block 6. *Ex. 1004 at 6:7-28* ('644) identifies "library 6 at step 42" is updated regrading an "the offending feature instance" - note that this is a single feature instance, whereas the method of '344 Claim 16 identifies "storing the set of genes and the software classification in the library." and this is "combining a plurality of genes that describe the software, thereby providing a set of genes;" - which is to say, the update identified in '644 of this library is distinctly different from that of '344 Claim 16. Also, the referenced method of '644 regards the identification of a threshold for detection, and deals with the determination and adaptation of the system to different threshold values and not the adaptation by adding of sets of genes to the library. This is unrelated to the classification identified in ['344 **16E**]. As can be clearly seen from the details provided her and above, '644 as cited does not discloses "a way of updating the library" of '344 Claim 6. It shows a way of updating a different kind of library with different contents in a different way, and most of it is irrelevant in every way to the specifics of '344. *Ex. 1004 at 8:47-62* ('644) discusses manual process for systems administrators setting the false positive rate for spreading characteristics of worms, time window sizes, and so forth. But this does not disclose or relate to [**16D**] "testing the set of genes for false-positives against one or more reference files using a processor;". Specifically, and without

Claims of '344	Per [SN] citation to Petition	My response
	(in the least mean squares sense) optimization algorithm (such as a neural network) can advantageously be used. ***However, visual investigation of the worm spreading characteristics, along with manual "tweaking" of the scoring parameters, may result in a satisfactory set of parameters. This manual setting of the scoring parameters also gives system administrators the option to customize the behavior of the invention, in particular, the false positive threshold appropriate for their computing environment.*** *Ex. 1006 at 15:44-51 (emphasis added):* ***The first exemplary anomaly detection algorithm discussed above in equations (1)-(3) were trained over the normal data.*** *Each record in the testing set was evaluated against this training data.* ***The results were evaluated by computing two statistics: the detection rate and the false positive rate.*** *The performance of the system was evaluated by measuring detection performance over processes labeled as either normal or malicious.* *See also Ex. 1007 at ¶¶ 83-89.*	limit, it fails to disclose "reference files" at all and "using a processor" is refuted by "***This manual setting of the scoring parameters also gives system administrators the option to customize the behavior of the invention, in particular, the false positive threshold appropriate for their computing environment.***" *Ex. 1006 at 15:44-51* ['084] relates to the '084 patent. As I discussed above, this is not the "testing for false-positives using one or more reference files" of '344 Claim 16 or anything like it. In particular and without limit: '084 "false positive" is for anomaly detection and not for detection of malware – they are addressing different issues and merely using the same terms. "processes" of '084 are not "software" of '344. "the normal data" of '084 is not "reference files" of '344. "record in the testing" is not "gene information" or anything like it of '344. *See also Ex. 1007 at ¶¶ 83-89.* [**SN**] references a combination not instituted in this case. This references Ex. 1004 at 6:7-38 adding threshold-based details to the already referenced details. However, the update identified in '644 of this library is distinctly different from that of '344 Claim 16. The referenced method of '644 regards the identification of a threshold for detection, and deals with the determination and adaptation of the system to different threshold values and not the adaptation by adding of sets of genes to the library. This is unrelated to the classification identified in ['344 **16E**]. The referenced portion of '644 does not discloses "a way of updating the

Claims of '344	Per [SN] citation to Petition	My response
		library" of '344 Claim 6. It shows a way of updating a different kind of library with different contents in a different way, and most of it is irrelevant in every way to the specifics of '344. *Ex. 1007 at* ¶¶ *84* [**SN**] discloses in pertinent parts, *An algorithm for detecting anomalous behavior is then employed, and the algorithm is trained and tested by comparing the detected anomalous behavior against normal behavior to generate statistics related to detection and false positives.* Again, this addresses false positives in anomaly detection of false detections of abnormal behavior, not false detection of malicious behavior of '344 and it does not disclose or make obvious "against one or more reference files" *Ex. 1007 at* ¶¶ *85* [**SN**] discloses in pertinent parts *when the extracted features do not match those in the library, then additional processing is done at steps 38-41 to determine if the feature vector under examination crosses a certain threshold. Ex. 1004 at 6:7-8:62. If it exceeds the threshold, then the email is deemed to be malicious and the library is updated at step 42. Id. The threshold levels are calibrated by testing for false-positives.* Again, this addresses false positives in anomaly detection of false detections of abnormal behavior, not false detection of malicious behavior of '344 and it does not disclose or make obvious "against one or more reference files" It then asserts, without further basis: "*To the extent Kissel does not disclose testing for false-positives against one or more reference files, Apap discloses testing its algorithm*

Claims of '344	Per [SN] citation to Petition	My response
		for detecting anomalous behavior by referencing the detected anomalies against recorded normal behavior." But despite the assertion, "recorded normal behavior" does not render obvious "testing for false-positives against one or more reference files" because, among other things, the files identified for '344 are not shown as files indicative of "normal behavior". *Ex. 1007 at ¶¶ 86* [**SN**] states, in pertinent parts, "*(86.) It would have been obvious to a person of ordinary skill in the art to combine Kissel and Apap so that suspected malware is tested for false-positives against one or more reference files. Testing, for example, suspected malware, for false-positives by using a reference that indicates whether the suspected malware is indeed malware was a well understood method known in the art to test for false- positives.*" But this is not the standard for obviousness in such combination claims. The reasons for this include, without limit: It is not shown to be the use of known techniques to improve similar devices in the same way; when the prior art teaches away from combining certain known elements, discovery of a successful means of combining them is not obvious, and the cited reference ('084) teaches away as identified above; and the asserted motive of [**SN** para 94] refutes the requirement for a *POSITA*. *Ex. 1007 at ¶¶ 87* [**SN**] makes general historical claims regarding obviousness and not claims specific to the identified prior art and an assertion that does not state anything about "testing the set of genes for false-positives against one or more reference files using a processor;" but merely has the term "false positives".

Claims of '344	Per [SN] citation to Petition	My response
		Ex. 1007 at ¶¶ *88* [**SN**] asserts yet another patent 8,713,686 not part of the instituted obviousness combination and thus not usable for this purpose. *Ex. 1007 at* ¶¶ *89* [**SN**] then conflates all of these demonstrable untrue assertions asserting "*(89.) Combining Kissel and Apap is also a combination that unites old elements with no change in their respective functions, would not present technical challenges, and would not negatively impact the functionality of Kissel's malware detection system. Kissel already discloses checking suspected malware for false- positives. Kissel does not explicitly disclose using one or more reference files to performing the operation of testing for false-positives, but because using reference files was a commonly understood way to perform such testing, a combination of Kissel and Apap would unite old elements with no change in their respective functions.*" However, as shown above, the individual parts are not true, so the conclusion is also not true. This [**VN**] fails to show that [**16D**] is obvious. Further, as I identified above, the basis for obviousness is refuted.
[**16E**] defining the software classification based on the set of genes; and	Ex. 1004 at 6:20-28 (emphasis added): At step 40, all scores S in score table 8 are updated. At step 41, **gateway 2 determines whether any of the recently updated scores exceed a preselected malicious threshold for that score, indicative of malicious code.** If this malicious threshold has been crossed, library 6 at step 42 is updated with an entry corresponding to the offending feature instance. Then, at step 43,	*Ex. 1004 at 6:20-28* ('664) fails to show that or how "defining the software classification based on the set of genes" is obvious. The setting of scores is not relevant to the "defining the software classification" as it doesn't assert to or actually define anything. Further, it refers to "emails" which are not shown to be "software". This is merely about deleting emails if they are found to exceed a threshold under statistical analysis and this does not render

Claims of '344	Per [SN] citation to Petition	My response
	all e-mails in queue 5 containing this feature instance are deleted, and the method reverts to step 37. See also Ex. 1007 at ¶ 83.	"defining the software classification based on the set of genes" obvious. Ex. 1007 at ¶ 83 [**SN**] refers to "(83.) As shown in steps 36-42 in Figs. 3A and 3B of Kissel and explained in column 6, lines 7 through 38, if the extracted feature vector is not in library, it is stored and a score associated with the extracted feature vector is generated. If the score exceeds a certain threshold, it indicates that this unknown extracted feature vector is associated with malware and the library is updated. Ex. 1004 At 6:7-38. Kissel further discloses calibrating the threshold to test whether these new extracted feature vectors are false-positives, i.e. not associated with malware. Ex. 1004 at 8:47-62." (Ex. 1004 is '664) None of this discloses, refers to, or is shown to be related to "defining the software classification based on the set of genes" either as "obvious" or at all. This [**SN**] fails to show that [**16E**] is obvious. Further, as I identified above, the basis for obviousness is refuted.
[**16F**] storing the set of genes and the software classification in the library.	Ex. 1004 at 6:20-28 (emphasis added): At step 40, all scores S in score table 8 are updated. At step 41, **gateway 2 determines whether any of the recently updated scores exceed a preselected malicious threshold for that score, indicative of malicious code.** If this malicious threshold has been crossed, library 6 at step 42 is updated with an entry corresponding to the offending feature instance. Then, at step 43, all e-mails in queue 5 containing this feature instance are deleted, and the method reverts to step 37.	This is the identical entry to the entry for [**16E**]. As discussed above, the "library 6 at step 42" is not the library disclosed in Claim 16. The mere fact of using the same word does not show obviousness. It performs a deferent function and contains different things. Ex. 1007 at ¶ 83 [**SN**] refers to "(83.) As shown in steps 36-42 in Figs. 3A and 3B of Kissel and explained in column 6, lines 7 through 38, if the extracted feature vector is not in library, it is stored and a score associated with the extracted feature vector is generated. If the score exceeds a certain threshold, it

Claims of '344	Per [SN] citation to Petition	My response
	See also Ex. 1007 at ¶ 83.	indicates that this unknown extracted feature vector is associated with malware and the library is updated. Ex. 1004 At 6:7-38. Kissel further discloses calibrating the threshold to test whether these new extracted feature vectors are false-positives, i.e. not associated with malware. Ex. 1004 at 8:47-62." (Ex. 1004 is '664) None of this discloses, refers to, or is shown to be related to "storing the set of genes and the software classification in the library." either as "obvious" or at all. This [**SN**] fails to show that [**16F**] is obvious. Further, as I identified above, the basis for obviousness is refuted.
		I conclude that Claim 16 of '344 is not rendered obvious over Bodorin in view of Kissel and Apap.
Claim 17		
17. The method of claim 16, wherein		The addition of the references of [**17A**] to the specification of Claim 16 does not render Claim 17 obvious over Bodorin in view of Kissel and Apap and still does not render Claim 16 obvious over Bodorin in view of Kissel and Apap.
[**17A**] the software classification includes malware.	Ex. 1003 at 5:44-49. See also Ex. 1007 at ¶ 64.	

Claims 16 and 17 of '344 are not shown to be obvious over Bodorin in view of Kissel and Apap by Fortinet's expert.

In addition, [**SN**-Depo P14L8-P19L4] states [P18L14-P18L25] "I can certainly imagine there would be some person with a master's degree in electrical engineering out there somewhere who could not do this.". Thus Foritnet's expert agrees that Claims 16 and 17 are not obvious.

Items referenced and considered

Items cited, included by reference, and considered are in the material provided herewith. This, plus files disclosed by counsel as part of this Declaration, include all of the documents referenced or considered in the writing of this Declaration. The Appendices hereto (Appendices A and C) are also part of this Declaration. Files external to this Declaration are listed in Appendix C. Out of an abundance of caution, I have included files that I received in this matter but that may not have been taken into consideration.

Within the limits of available time and information, I have also considered and applied my knowledge, training, education, skills, and experience as part of the basis for the statements made in this Declaration.

Compensation for this case

My compensation for this case is through my company Management Analytics, which is currently paid at the rate of $600 per hour for my time, less a discount of $100/h for timely payment made within 5 days of invoice.

Other cases in the last 10 years

The following is a listing of all other cases in which I have testified as an expert at trial or by deposition within the preceding ten years.

In court:

In the Matter of Certain Computer Forensic Devices And Products Containing The Same, Investigation No. 337-TA-799, United States International Trade Commission, Washington, D.C., Before the Honorable Charles E. Bullock Chief Administrative Law Judge, 2012-08-08,09.

Rose v. Albritton, Superior Court of the City and County of San Francisco, Case No.: FDV-09-806677, July 14, 2009 (qualified as an expert)

United States v. Bayly, et. al., United States District Court, Southern District of Texas, case no. Cr. No. H-03-363. 2004-10-25 (qualified as an expert)

Witness Statement:
In the Matter of Certain Computer Forensic Devices And Products Containing The Same, Investigation No. 337-TA-799, United States International Trade Commission, Washington, D.C., Before the Honorable Charles E. Bullock Chief Administrative Law Judge, 2012-07-07.

Depositions:

FORTINET, INC., a corporation, Plaintiff, vs. SOPHOS, INC., a corporation, Defendants, CASE NO. 3:13-cv-05831-EMC, 2015-08-25

In the Matter of Certain Computer Forensic Devices And Products Containing The Same, Investigation No. 337-TA-799, United States International Trade Commission, Washington, D.C., Before the Honorable Charles E. Bullock Chief Administrative Law Judge, 2012-07-06.

Finjan, Inc. v. McAfee, Inc. et. al. The United States District Court For The District Of Delaware C.A. #10-593 (GMS) 2012-05-24

Scienton Technologies, Inc., NI Group, Inc. and Secure-It, Inc., v. Computer Associates International, Inc., The United States District Court Eastern District Of New York, Case No. 04-CV-2652 (JS) (ETB)

Beyond Systems, Inc., v. Kraft Foods, Inc., et al., Case No. 8:08-CV-00409, United States District Court, District of Maryland

ASIS Internet Services, v. Optin Global, Inc., et. al., - United States District Court, Northern District Of California Case No. C-05-5124 JCS, 2008-01-07

Paul Cozza v. McAffee, Inc. United States District Court, District of Massachusetts, case no. 02-11135RGS, 2006-07-13

Other disclosable matters:

Susan Polgar v. US Chess Federation et. al. In the US District Court, Northern district of Texas, Lubbock Division, C.A. NO. 5-08CV0169-C, Sep 15, 2009.

Ongoing analysis

Analysis and examination is ongoing and will be supplemented in accordance with applicable Court Orders and rules.

Signature

I declare under penalty of perjury under the laws of the United States of America 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.

Executed on this 28th day of October, 2015, in Pebble Beach, California.

Dr. Frederick B. Cohen

Appendix A – My CV and Related materials

Education:

Ph.D. Electrical Engineering - University of Southern California, 1986
M.S. Information Science - University of Pittsburgh, 1980
B.S. Electrical Engineering - Carnegie-Mellon University, 1977

Positions (current):*

* CEO, Management Analytics, 2012-07 - present
* CEO, Fred Cohen & Associates, 1986-03 – present
* Acting Director, Webster University Cyber Lab, 2013-11 – present
* Member. Fearless Security, LLC 2014-08 – present
* Member, Fearless Ridge, LLC, 2015-05 – present
President, California Sciences Institute, 2007-11 - 2012-12
Community Faculty Member, Metropolitan State University, 04/08-01/09
Adjunct Professor, University of San Francisco, 08/07-08/08
Research Professor, University of New Haven, 09/02-08/08
Chairman, Security Posture, 10/02-09/06
Principal Analyst, Burton Group, 04/03-04/06
Principal Member of Technical Staff, Sandia National Laboratories, 10/97-11/02
Chief Computer Forensic Scientist/Investigator, TAL Global, 09/01-11/02
Senior Member Technical Staff, Sandia National Laboratories, 7/96-9/97
Practitioner in Residence, University of New Haven, 06/98-09/02
Senior Scientist, SAIC, Inc., 7/93-6/95
Chairman, AllThings Incorporated, 1/95-1/96
Board Member, AllThings Incorporated, 6/94-1/96
Professor, Queensland University of Technology (visiting), 7/92-12/92
President, The Radon Project, Inc. 3/87-10/89
Asst. Professor, University of Cincinnati, 9/87-12/88
Asst. Professor, Lehigh University, 1/86-8/87
Lecturer, Lehigh University, 1/85-12/85

Editorial Boards, Program Committees:

2013-present, International Conference on Digital Forensics & Cyber Crime
2011-present, Journal of Digital Forensics, Security, and Law (Topic editor: Science)
2010-present, Board of Referees for the journal Digital Investigation
2009-present, IFIP WG 11.10 Program Committee (reviewer)
2008-present, IFIP WG 11.9 Program Committee (reviewer)
2007-present, EDP Audit, Control, and Security (EDPACS)
2005-present, Journal of Computer Virology
1987-present, IFIP TC-11 journal Computers and Security
2007-2011, MiniMetriCon Program Committee (and founding chair)
2010-2011, IEEE Security & Privacy, Beyond the Horizon (co-editor)
1995-2002, Network Security Magazine
1990-1993, ACM/SigSAC Annual Student Paper Review Board
1989-1991, DPMA, IEEE, and ACM Computer Virus and Security Conference
1989-1991, Computer Virus Bulletin

Honors and Awards:

2011: Honorary Doctorate (Honoris causa) Computer Science, University of Pretoria, South Africa
2011: International Information Systems Security Certification Consortium – Fellow of the $(ISC)^2$
2004: Burton Group Best Award
2002: Sandia Certificate of Appreciation (CCD Program)
2002: Techno-Security Industry Professional of the Year
2001: Sandia Employee Recognition Award (College Cyber Defenders)
2000: Sandia Meritorious Achievement Award (Security Immersion Program)
2000: Sandia Award for Excellence (Cyber Security / Surety)
1999: Sandia Exceptional Service Award (Security Awareness Work)
1998: Sandia Exceptional Service Award (Red Team Work)
1989: UK IT Auditors: Information Technology Award

Professional Societies

International Information Systems Security Certification Consortium – Fellow of the $(ISC)^2$
IEEE - Senior member

Books and Book Chapters not otherwise listed:

F. Cohen, “Protection and Engineering Design Issues in Critical Infrastructures”, pp67-153 in Thomas A. Johnson Ed. “CyberSecurity – Protecting Critical Infrastructures from Cyber Attack and Cyber Warfare”, 2015, CRC Press, ISBN 978-1-4822-3922-5.
F. Cohen, “Challenges to Digital Forensic Evidence in the Cloud”, in “Cybercrime and Cloud Forensics: Applications for Investigation Processes”, K. Ruan, Ed. 978-1-4666-2662-1, 2013.
F. Cohen, ”Digital Forensic Evidence Examination”, ASP Press, 2nd ed., 2010, 3rd ed.. 2011, 4th ed, 2012
F. Cohen, “Fundamentals of Digital Forensic Evidence”, (chapter in Handbook of Information and Communications Security, Springer, 2010, pp 789-808, Mark Stamp and Peter Stavroulakis, Ed.).
F. Cohen, ”Digital Forensic Evidence Examination”, ASP Press, 2009
F. Cohen, ”Enterprise Information Protection Architecture”, ASP Press, 2008
F. Cohen, “Challenges to Digital Forensic Evidence”, ASP Press, 2008
F. Cohen, “A Framework for Deception”, (chapter in National Security Issues in Science, Law, and Technology), Thomas A. Johnson, Ed. Taylor & Francis, 2007.
F. Cohen, “Critical Infrastructure Protection: Issues and Answers”, (one chapter in National Security Issues in Science, Law, and Technology), Thomas A. Johnson, Ed. Taylor & Francis, 2007. (in press).
F. Cohen, “Information Warfare, Netwar, and Cyber Intelligence.”, (chapter in National Security Issues in Science, Law, and Technology), Thomas A. Johnson, Ed. Taylor & Francis, 2007.
F. Cohen, "Challenges to Digital Forensic Evidence", (chapter in "Forensic Computer Crime Investigation", Thomas A. Johnson, Ed. Taylor & Francis, 2006
F. Cohen, “Security Decisions”, ASP Press, 2006
F. Cohen, “IT Security Governance Guidebook with Security Program Metrics on CD-ROM.” , Taylor & Francis/CRC Press, 2006
F. Cohen, “The Use of Deception Techniques: Honeypots and Decoys”, (chapter in Handbook of Information Security), V3, p646. Wiley and Sons, 2006.
World War 3 ... Information Warfare Basics , ASP Press, 2006
Information Security Awareness Basics , ASP Press, 2006
Frauds, Spies, and Lies - and how to defeat them , ASP Press, 2005

The CISO ToolKit: Security Checklists - Governance , ASP Press, 2005
The CISO ToolKit: Security Metrics , ASP Press, 2005
The CISO ToolKit: Governance Guidebook , ASP Press, 2005
Protection and Security on the Information Superhighway , John Wiley and Sons (1995)
F. Cohen, “It's Alive!!!”, John Wiley and Sons (1994)
F. Cohen, “A Short Course in Computer Viruses (2nd edition)”, John Wiley and Sons (1994)
F. Cohen, “A Short Course on Systems Administration and Security Under Unix”, ASP Press, (1992)
F. Cohen, “Payback - Automated Bill Collection System”, ASP Press (1992)
F. Cohen, “A Short Course in Computer Viruses”, ASP Press (1991)
F. Cohen, “The ASP Integrity Toolkit”, ASP Press (1990)
F. Cohen, “Introductory Information Protection”, ASP Press (1987)
F. Cohen, “Computer Viruses”, ASP Press, (1985)

Patents:

F. Cohen, “Detecting Inconsistencies in Data”, (pending) 61/531,609
F. Cohen, “Detecting Inconsistencies Consistent With Subversion of Normal Operations in an Operating Environment ” (pending) 61/531,609
F. Cohen, “Depiction of Digital Data for Forensic Purposes”, (pending)
F. Cohen, “Method and/or System for Providing and/or Analyzing and/or Presenting Decision Strategies“, 60/957,455 (pending)
F. Cohen, “Method and/or System for Providing and/or Analyzing Influence Strategies“, US Pat. 8,095,492.
F. Cohen, “Method and Apparatus for Invisible Network Responder” 20040148521 (pending) 20040162994 (pending)
F. Cohen, “Method and Apparatus for Specifying Communication Indication Matching and/or Responses” 20040153574 (pending)
F. Cohen, D Koike, V. Nagaeu, “Method and Apparatus Providing Deception and/or Altered Operation in an Information System Operating System”, US Pat. 7,437,766 10/679,186
F. Cohen, “Method and Apparatus for Configurable Communication Network Defenses”
F. Cohen, “Method and Apparatus for Network Deception/Emulation”, US Pat. 7,107,347.
F. Cohen, “Method and Apparatus for Providing Deception and/or Altered Execution of Logic in an Information System US Pat. 7,296,274.

Refereed journal articles:

F. Cohen, “Digital diplomatics and forensics: Going forward on a global basis”, Records Management Journal, 2015-03
Cohen F, Cohen D, “Time and space interval record schedule consistency analysis for atomic items without interactions in open spaces with stationary locations”, Computers & Security (2014), http://dx.doi.org/10.1016/ j.cose.2014.03.002
F. Cohen, "Identifying and Attributing Similar Traces with Greatest Common Factor Analysis", J. of Digital Forensics, Security, and Law, V7#2, 2012.
F. Cohen, "A Case Study in Forensic Analysis of Control", Journal of Digital Forensics, Security, and Law., V6#1, 2011.
F. Cohen, "A Method for Forensic Analysis of Control", IFIP TC11, Computers & Security, V29#8, pp 891-902, Nov., 2010, doi: 10.1016/j.cose.2010.05.003
F. Cohen, et. al. "Leading Attackers Through Attack Graphs with Deceptions", IFIP-TC11, `Computers and Security', V22#5, July 2003, pp. 402-411(10)
F. Cohen, et. al. "A Mathematical Structure of Simple Defensive Network Deceptions", IFIP-TC11,

`Computers and Security', Volume 19, Number 6, 1 October 2000, pp. 520-528(9)
Cohen F.; Phillips C.; Swiler L.P.; Gaylor T.; Leary P.; Rupley F.; Isler R., "A Cause and Effect Model of Attacks on Information Systems. Some Analysis Based on That Model, and The Application of that Model for CyberWarfare in CID", IFIP-TC11 Computers and Security, V17#3, 1998 pp. 211-221 (11)
F. Cohen, "Providing for Responsibility in a Global Information Infrastructure ", IFIP-TC11, `Computers and Security', 1997.
F. Cohen, "Simulating Cyber Attacks, Defenses, and Consequences", IFIP-TC11, `Computers and Security', 1999, vol. 18, no. 6, pp. 479-518(40)
F. Cohen, et. al. "Intrusion Detection and Response", IFIP-TC11, `Computers and Security', V16,#6, 1997, pp. 516-516(1) (also appearing below under National Info-Sec Technical Baseline studies from Dec, 1996)
F. Cohen, "A Note on the Role of Deception in Information Protection", IFIP-TC11, Computers and Security, 1998, vol. 17, no. 6, pp. 483-506(24)
F. Cohen, "Information System Defences: A Preliminary Classification Scheme", IFIP TC-11, Computers and Security, V16,#2, 1997, pp. 94-114(21)
F. Cohen, "A Note on Distributed Coordinated Attacks", IFIP-TC11, `Computers and Security', Volume 15, Number 2, 1996, pp. 103-121(19), also appearing as an invited paper in 4th Computer Misuse and Anomaly Detection Workshop, Monterey, 1996 (referenced below).
F. Cohen, "A Secure World-Wide-Web daemon", IFIP-TC11, `Computers and Security', V15#8, 1996, pp. 707-724(18)
F. Cohen, "Operating Systems Protection Through Program Evolution", IFIP-TC11 `Computers and Security' (1993) V12#6 (Oct. 1993) pp.565 - 584
J. Voas, J. Payne, F. Cohen "A Model for Detecting the Existence of Software Corruption in Real-Time", IFIP-TC11 "Computers and Security", V12#3 May, 1993 pp. 275-283.
F. Cohen, "A Formal Definition of Computer Worms and Some Related Results", IFIP-TC11 "Computers and Security" V11#7, November, 1992, pp. 641-652.
F. Cohen, "Defense-In-Depth Against Computer Viruses", IFIP-TC11 "Computers and Security", V11#6, 1992 pp. 563-579.
F. Cohen, "A DOS Based POset Implementation", IFIP-TC11 "Computers and Security", V10#6, October 1991.
F. Cohen, "A Note On High Integrity PC Bootstrapping", IFIP-TC11 "Computers and Security", V10#6, October 1991.
F. Cohen, "A Cost Analysis of Typical Computer Viruses and Defenses", IFIP-TC11 "Computers and Security", V10#3, May, 1991 (also appearing in 4th DPMA, IEEE, ACM Computer Virus and Security Conference, 1991)
F. Cohen, "Automated Integrity Maintenance for Viral Defense", IFIP-TC11 "Computers and Security", 1990
F. Cohen, "Computational Aspects of Computer Viruses", IFIP-TC11, "Computers and Security", V8 pp325-344, 1989.
Y. J. Huang and F. Cohen, "Some Weak Points of One Fast Cryptographic Checksum Algorithm and its Improvement", IFIP-TC11 "Computers and Security", V8#1, February, 1989
B. Cohen and F. Cohen, "Error Prevention at a Radon Measurement Service Laboratory", Radiation Protection Management, V6#1, pp43-47, Jan. 1989
F. Cohen, "Models of Practical Defenses Against Computer Viruses", IFIP-TC11, "Computers and Security", V8#2, April, 1989 pp149-160.
F. Cohen, "Two Secure Network File Servers", IFIP-TC11, "Computers and Security", V7#4, August, 1988.
F. Cohen, "Designing Provably Correct Information Networks with Digital Diodes", IFIP-TC11, "Computers and Security", V7#3, June, 1988.
F. Cohen, "On the Implications of Computer Viruses and Methods of Defense", Invited Paper,

IFIP-TC11, "Computers and Security", V7#2, April, 1988,
F. Cohen, "Maintaining a Poor Person's Information Integrity", IFIP-TC11, "Computers and Security", V7#1, Feb 1988.
F. Cohen, "A Cryptographic Checksum for Integrity Protection", IFIP-TC11 "Computers and Security", V6#6 (Dec. 1987), pp 505-810.
F. Cohen, "Design and Protection of an Information Network Under a Partial Ordering: A Case Study", IFIP-TC11, "Computers and Security", V6#4 (Aug. 1987) pp 332-338.
F. Cohen, "Design and Administration of Distributed and Hierarchical Information Networks Under Partial Orderings", IFIP-TC11, "Computers and Security", V6#3 (June 1987), pp 219-228.
F. Cohen, "Protection and Administration of Information Networks with Partial Orderings", IFIP-TC11, "Computers and Security", V6#2 (April 1987) pp 118-128.
F. Cohen, "Computer Viruses - Theory and Experiments", DOD/NBS 7th Conference on Computer Security, originally appearing in IFIP-sec 84, also appearing as invited paper in IFIP-TC11, "Computers and Security", V6#1 (Jan. 1987), pp 22-35 and other publications in several languages.
F. Cohen, "A Secure Computer Network Design", IFIP-TC11, "Computers and Security", V4#3, (Sept. 1985), pp 189-205, also appearing in AFCEA Symp. and Expo. on Physical and Electronic Security, Aug. 1985.
F. Cohen, "Algorithmic Authentication of Identification", Information Age, V7#1 (Jan. 1985), pp 35-41.

Invited Papers and Keynotes with Published Papers:

F. Cohen, “The Future of Digital Forensics” - 2012-09-21. 1st Chinese Conference on Digital Forensics, Keynote Address and Paper.
F. Cohen, "What Makes Critical Infrastructures Critical?", International Journal of Critical Infrastructure Protection(2010), doi: 10.1016/j.ijcip.2010.06.002.
F. Cohen, "The Smarter Grid", IEEE Security and Privacy - On the Horizon, 2010, V8#1, January/February, 2010.
F. Cohen, "Toward a Science of Digital Forensic Evidence Examination", IFIP TC11.8 International Conference on Digital Forensics", Hong Kong, China, Jan 4, 2010 (Keynote), Published in Advances in Digital Forensics VI, pp17-36, Springer, ISBN#3-642-15505-7, 2010.
F. Cohen, "The past and future history of computer viruses", EICAR annual conference, Berlin, Germany, May 11-12, 2009.
F. Cohen, "30 Lies About Secure Electronic Commerce: The Truth Exposed", Institute for International Research Electronic Commerce Conference, Feb. 1999.
F. Cohen, Cynthia Phillips, Laura Painton Swiler, Timothy Gaylor, Patricia Leary, Fran Rupley, Richard Isler, and Eli Dart "A Preliminary Classification Scheme for Information System Threats, Attacks, and Defenses; A Cause and Effect Model; and Some Analysis Based on That Model", Encyclopedia of Computer Science, 1999.
F. Cohen, "National Info-Sec Technical Baseline - Intrusion Detection and Response" National InfoSec Research Council, Dec, 1996. (also appearing above in Computers and Security, 1997).
F. Cohen, "A Note on Distributed Coordinated Attacks", 4th Computer Misuse and Anomaly Detection Workshop, Monterey, 1996 (also appearing above in Computers and Security, 1996).
F. Cohen, "The Internet, ..., and Information Security", Computer Society of South Africa, 7th Annual Conference, August, 1995, South Africa.
F. Cohen, "The Internet, Corporate Networks, and Firewalls", Computer Society of South Africa, 7th Annual Conference, August, 1995, South Africa.
F. Cohen, "Information Assurance", IFIP TC-11 World Congress, May, 1995, Cape Town, South Africa.
F. Cohen, "`Information Warfare Considerations", Norwegian Academy of Sciences, September,

1993.
F. Cohen, "Computer Viruses", (one chapter in "The Computer Security Reference Book", Butterworth/Heinemann, Oxford, England, 2004.
F. Cohen, "Fault Tolerant Software for Computer Virus Defense", November, 1991.
F. Cohen, "Some Applications of Benevolent Viruses in Networked Computing Environments", `DPMA, IEEE, ACM Computer Virus and Security Conference', March 1993
F. Cohen and S. Mishra, "Some Initial Results From the QUT Virus Research Network", `The Virus Bulletin Conference', Edinburgh, Scotland, July, 1992 (keynote).
F. Cohen, "Computer Viruses", (one chapter in "The Computer Security Reference Book" Butterworth/Heinemann (1992), Oxford, England
F. Cohen, "Current Trends in Computer Viruses", Invited Paper, International Symposium on Information Security, Oct. 17-18, 1991, Tokyo, Japan
F. Cohen, "Current Best Practice Against Computer Viruses", Invited Paper, 1991, 25th IEEE International Carnahan Conference on Security Technology, Oct. 1-3, 1991, Taiwan ROC.
F. Cohen, "Exploiting Defense-In-Depth Against Computer Viruses", Invited Paper, The Oxbridge Sessions, Sept. 3-5, 1991, The Netherlands.
F. Cohen, "Computers Under Attack" (one paper), ACM/Addison Wesley (1990)
F. Cohen, "A Summary of Results on Computer Viruses and Defenses", Invited Paper, 1990 NIST/DOD Conference on Computer Security.
F. Cohen, "Integrity Maintenance in Untrusted Computing Environments", Invited Paper, IBC Computer Virus Conference, London, 1990.
F. Cohen, "Information Systems as a Competitive Weapon", Keynote Address, Utah State School of Business, Annual Computer Conference, March 1-2, 1990.
F. Cohen, "Recent Advances in Integrity Maintenance in Untrusted Systems", Invited Paper, The Netherlands Computer Security Seminar, April 10-12, 1990.
F. Cohen, "A Note on the use of Pattern Matching in Computer Virus Detection", Invited Paper, Computer Security Conference, London, England, Oct 11-13, 1989, also appearing in DPMA, IEEE, ACM Computer Virus Clinic, 1990.
F. Cohen, "Computer Viruses - Attacks and Defensive Measures", London Corporate Computer Security Conference - Keynote Address, London, England, Feb. 14, 1989.
F. Cohen, "Current Trends in Computer Virus Research", 2nd Annual Invited Symp. on Computer Viruses - Keynote Address, Oct. 10, 1988. New York, NY
F. Cohen, "Recovery Techniques in Computer Virus Attack", Invited Paper, Invitational Conference on Computer Viruses, 1988.

Peer Reviewed Conference and Other Papers:

F. Cohen, “A Tale of Two Traces - Archives, Diplomatics, and Digital Forensic”, IFIP Tenth annual IFIP WG 11.9 International Conference on Digital Forensics, 2015-01-26-8, also published as a chapter in “Advances in Digital Forensics X”.
F. Cohen, “As the consequences rise, where is the risk management?”, EDPACS: The EDP Audit, Control, and Security Newsletter, V41# 4, (2013), DOI:10.1080/07366981.2013.775779, reprint from analyst report.
F. Cohen, “The Bottom Ten List—Information Security Worst Practices”, EDPACS: The EDP Audit, Control, and Security Newsletter, V41#1, (2012), DOI:10.1080/07366981003634460, reprint from analyst report.
F. Cohen, “F. Cohen, “The Physics of Digital Information – Part 2”, JDFSL v7#1”, July, 2012 (editorial column)
F. Cohen, “Forensic Methods for Detecting Insider Turning Behaviors”, IEEE Workshop on Research for Insider Threat, 2012-05-25, with the IEEE Oakland Conference, IEEE Computer Society Press.

F. Cohen, “Update on the State of the Science of Digital Evidence Examination”, Conference on Digital Forensics, Security, and Law, May 29-31, 2012 – pending publication.
F. Cohen, “Consistency under deception implies integrity”, ICSJWG Quarterly Newsletter. Republished from 2011-09 Short Analyst Reports.
F. Cohen, “F. Cohen, “The Physics of Digital Information”, Journal of Digital Forensics, Security, and Law, V6#3, October, 2011 (editorial column)
F. Cohen, “Progress and evolution of critical infrastructure protection over the last 10 years?”, The CIP Report, Center for Infrastructure Protection and Homeland Security, V10#3, 2011-09, Republished from 2011-08 Short Analyst Reports.
F. Cohen, “Putting the Science in Digital Forensics” Journal of Digital Forensics, Security, and Law, V6#1, July, 2011 (editorial column)
F. Cohen, “How Do We Measure Security?”, Insight, V14#2, 2011-07, pp 30-32, International Council on Systems Engineering.
F. Cohen, "Change your passwords how often?”, EDPACS 44(4), April, 2010. (Reprinted from Analyst Newsletter)
F. Cohen, J. Lowrie, C. Preston, “The State of the Science of Digital Evidence Examination”, IFIP Seventh annual IFIP WG 11.9 International Conference on Digital Forensics, 2011/01/30, also published as a chapter in “Advances in Digital Forensics VII”.
F. Cohen, “Digital Forensic Evidence Examination - The State of the Science - and Where to Go From Here”, NeFX Workshop, Sep 14, 2010.
F. Cohen, "Fonts for Forensics", IEEE SADFE (in conjunction with the IEEE Oakland Conference), 2010-05-19, Oakland, CA.
F. Cohen, "The Bottom Ten List - Information Security Worst Practices”, EDPACS 41(1), January, 2010. (Reprinted from Analyst Newsletter)
F. Cohen, "Attribution of messages to sources in digital forensics cases", HICSS-43, Jan 7, 2010.
F. Cohen, "Analysis of redundant traces for consistency", IEEE International Workshop on Computer Forensics in Software Engineering (CFSE 09), Seattle, Washington, USA, July 20-24, 2009
F. Cohen, "Two models of digital forensic examination", IEEE SADFE (in conjunction with the IEEE Oakland Conference), 2009-05-21, Oakland, CA
F. Cohen, "Issues and a case study in bulk email forensics", Fifth annual IFIP WG 11.9 International Conference on Digital Forensics, 2009/01/27, published as "Bulk Email Forensics" in the conference publication.
F. Cohen and T. Johnson, "A Ph.D. Curriculum for Digital Forensics", HICSS-42, Jan 7, 2009.
F. Cohen, "Information Assurance Architectural Models", HICSS-42 Product and Process Assurance Symposium Position Paper, Jan 5, 2009,
F. Cohen, "Social tension and separation of duties”, EDPACS 38(5) (2009). (Reprinted from Analyst Newsletter).
F. Cohen, “Control Requirements for Control Systems... Matching Surety to Risk”, EDPACS, 2008
F. Cohen, “Making Compliance Simple – Not”, EDPACS, March 2008, V37#3. (Reprinted from Analyst Newsletter)
F. Cohen and C. Preston, “A Method for Recovering Data From Failing Floppy Disks with a Practical Example”, Fourth annual IFIP WG 11.9 International Conference on Digital Forensics, 2008/01/27 to 30. Published in “Advances in Digital Forensics IV”, Springer, ISBN 978-0-387-84926-3, pp29-42, 2008.
F. Cohen, “Fault Modeling and Root Cause Analysis for Information Security Governance”, Computers and Security, 2007.
Cohen, “A Note on Detecting Tampering with Audit Trails”', 1995. Available at http://all.net/books/audit/audmod.html
F. Cohen and D. Koike, “Misleading attackers with deception”, Information Assurance Workshop,

2004. Proceedings from the Fifth Annual IEEE SMC Publication Date: 10-11 June 2004\l : pp. 30- 37
F. Cohen, "Airbag Inflator Inspection System", LumenX Corporation, November, 1994.
F. Cohen, R. Knecht, C. Preston, et. al., "Planning Considerations for Defensive Information Warfare - Information Assurance", Contract DCA 100-90-C-0058 T.O. 90-SAIC-019, November, 1993.
F. Cohen, "Threats and Defenses for WCCS", US Air Force, Wing Command and Control System, August, 1993.
F. Cohen, "Information Warfare Considerations", National Academy of Sciences - National Research Council, September, 1993.
F. Cohen, "A Case for Benevolent Viruses" DPMA, IEEE, ACM Computer Virus and Security Conference, March 1992
F. Cohen, "Current Best Practice Against Computer Viruses with Examples from the DOS Operating System", DPMA, IEEE, ACM Computer Virus and Security Conference, March 1992
F. Cohen, "A Roving Emulator", Conference on Modeling and Simulation, April, 1987.
F. Cohen, "Information Protection", Curriculum Module for the graduate degree in Software Engineering, The Software Engineering Institute, June, 1986, also appearing in ACM SIGSAC in abbreviated form.
F. Cohen, "A Complexity Based Integrity Maintenance Mechanism", Conference on Information Sciences and Systems, Princeton University, March 1986.
F. Cohen, "Recent Results in Computer Viruses", Conference on Information Sciences and Systems, Johns Hopkins University, March 1985.
F. Cohen, "The HAD Cryptosystem", IACR Crypto84 rump session, Aug. 1984.
F. Cohen, "Computer Security Methods and Systems", Conference on Information Sciences and Systems, Princeton University, March 1984.
F. Cohen, "Learning Networks for Database Access", Yale Conference on Adaptive Systems Theory, New Haven, CT, June, 1983.
M.A. Breuer, F. Cohen, and A.A. Ismaeel, "Roving Emulation", Built-In Self-Test Conference, March 1983.
F. Cohen, "The Delta-Net Model of Computation", Conference on Information Sciences and Systems, Johns Hopkins University, March 1983.
F. Cohen, "The U.S.C. Roving Emulator", U.S.C. DISC Report #82-8, Dept of Electrical Engineering, University Park, LA, Ca. 90089-0781, Dec. 1982.
M.A. Breuer, F. Cohen, A.A. Ismaeel, "Roving Emulation as Applied to a (255,223) RS-encoder System", U.S.C. DISC Report #82-6, Dec. 1982.

Burton Group Reports:

F. Cohen, "Outsourcing, Offshoring, and Security: What's the Difference?" 28 Jun 2006
F. Cohen, "IT Risk Management and COSO" 24 May 2006
F. Cohen, "Defending Against the Evil Insider" 16 Nov 2005
F. Cohen, "Raising the Bar: Solving Medium-Risk Problems with Medium-Surety Solutions" 27 Sep 2005
P. Schacter and F. Cohen, "Enterprise Strategies for Defending Against Spyware" 23 Aug 2005
F. Cohen, "Security Metrics: Horses for Courses" 24 Jun 2005
F. Cohen, "Security Governance for the Enterprise" 31 Mar 2005
F. Cohen, "Business Continuity Planning for IT" 24 Mar 2005
D. Blum and F. Cohen, "A Systematic, Comprehensive Approach to Information Security" 24 Feb 2005
F. Cohen, "Security Awareness, Training, and Education Programs for the Enterprise" 17 Jan 2005

F. Cohen, "Change Management for the Enterprise" 17 Jan 2005
D. Blum and F. Cohen, "Concepts and Definitions" 17 Jan 2005
F. Cohen, "Database Security: Protecting the Critical Content of the Enterprise" 28 Oct 2004
F. Cohen, "Building Secure Applications: How secure do you want to be today?" 09 Sep 2004
F. Cohen, "Risk Aggregation: The Unintended Consequence", 27 Apr 2004
F. Cohen, "Auditing and Audit Trails", 18 Mar 2004
F. Cohen, "Patch Management", 2003
F. Cohen, "The Evolving Role of Firewalls", 2003
F. Cohen, "Linux Security Features", 2003 (unpublished)
F. Cohen, "Intrusion Detection and Response Systems", October, 2003
F. Cohen, "Analysis of Information-Related Threats to Enterprises", 18 Sep 2003
F. Cohen, "Risk Management: Concepts and Frameworks", 18 July, 2003
F. Cohen, "Policy-Based Security and Enterprise Policy Management", 9 Dec 2003

Special Cyber Terrorism Studies:

The Provisional Irish Republican Army - 2000
The Animal Liberation Front (ALF) - 2000
Revolutionary Armed Forces of Colombia (FARC) - 2000
National Liberation Army (ELN)--Colombia - 2000
Issues in Cyber-Terrorism – 2000

Short Analyst Reports and Other Substantial Collections:

2015-10: Send me the Snowden docs
2015-09: Why can't we make any of the systems we use secure from remote attack?
2015-08: Who's to blame?
2015-07: The greatest strategic blunder of all time?
2015-06-B: Incident at All.Net - 2015
2015-06: Reducing the Effects of Malicious Insiders Non-technologically
2015-06: Reducing the effects of malicious insiders non-technologically
2015-05: Iran(t) on merchantability for software
2015-04: Fishing and phishing
2015-03 B: Error-induced misoperation - rowhammer
2015-03 Temporal microzones and end-user workstations
2015-02 Input checking
2015-01 The year of the Trojans (and their unintended side effects)
2014-12 Stupid security getting even stupider
2014-11-B Eat your own dog food deal about big data loss (actually theft)?
2014-10 Cyber (whatever that is) insurance yet again?
2014-09 2-factor this into your thinking
2014-08-B A touch of the Ebola
2014-08 Aurora and why it doesn't really matter
2014-07 Encrypt it all!!!
2014-06 Is it secure?
2014-05 May Day - attack mechanisms revisited - were you surprised by the NSA's activities?
2014-04 The RSA: Science Fiction and Humor
2014-03-B The Snowden virus - disrupting the secret world by exploiting their policies
2014-03 The four tactical situations of cyber conflict
2014-02 Countering hardware storage device Trojans
2014-01-B After the Red Team

2014-01 Why we need better reporters to solve our security problems
2013-12 Return of the telnet return
2013-11-B Transparency - a different protection objective
2013-11 Demystifying control architecture
2013-10-B The "big deal" approach to risk management
2013-10 Trust and worthiness
2013-09 The surveillance society: pros, cons, alternatives, and my view.
2013-08 50 Ways to respond to "Computer Repair..."
2013-08 Three words you should never use in security and risk management
2013-07-B How to justify (security) metrics and what to measure
2013-07 Mobility and industrial control systems
2013-06 Separation of Duties and RFPs
2013-05-B The harder problems
2013-05 Write lock the past, access control the present, anticipate the future
2013-04-B Actionable metrics (Guest Editor)
2013-04 Managing Oops
2013-03-C Limiting Insider Effects Through Micro-Zoning
2013-03-B Welcome to the Information Age - a 1-page primer
2013-03 Security Heroes
2013-02-B Stupid Security Winner for 2012
2013-02 Thinking more clearly
2013-01 Raising all boats - by improving the average
2012-12 Enterprise Security Architecture Options and Basis
2012-12 Ten Bad Assumptions
2012-11 The Design Basis Threat
2012-10 Changing the leverage
2012-10 Industrial Control System Security Decisions and Architecture
2012-09 Eventually, you are going to make a mistake
2012-08 As the consequences rise, where is the risk management?
2012-07-01B The Facebook debacle and what it says about the other providers
2012-07-01 - Open CyberWar - Early Release
2012-06-01 - Question everything
2012-05-01 - The threat reduction approach - Point – Counterpoint
2012-04-01 - The insider turned bad
2012-03-01 - Three emerging technologies
2012-02-01 - Ethics in security research
2012-01-31 - Influence Operations
2012-01-01 - The security squeeze
2011-12-01 - Can we attribute authorship or human characteristics by automated inspection?
2011-11-03 - Saving SMBs from data leakage
2011-11-01 - Webification and Authentication Insanity
2011-10-01 - Consistency Under Deception Implies Integrity - ICSJWG version
2011-10-01 - Security vs. Convenience - The Cloud - Mobile Devices - and Synchronization
2011-09-11 - CIP version of "Progress and evolution of critical infrastructure protection over the last 10 years?"
2011-09 - Consistency under deception implies integrity
2011-08 - Progress and evolution of critical infrastructure protection over the last 10 years?
2011-07 - The structure of risk and reward
2011-06 - Security Metrics - A Matter of Type
2011-05 - The "R" word
2011-04 - Change your passwords how often?

2011-03 - Any is not All
2011-02 - Why are we so concerned about governments getting our data?
2011-01b - The Bottom Ten List - Information Security Worst Practices - Getting Even Worse
2011-01 - Risk aggregation - again and again and again...
2010-12b - Code book cryptography may be nearly dead
2010-12 - Changes to the Federal Rules of Evidence - Rule 26
2010-11 - How do we measure "security"?
2010-10 - Moving target defenses with and without cover deception
2010-09 - User Platform Selection Revisited
2010-08 – The DMCA Still Restricts Forensics
2010-07 - Mediated Investigative Electronic Discovery
2010-06 - The difference between responsibility and control
2010-05 - The Virtualization Solution
2010-04 - Attacks on information systems - a bedtime story
2010-03 - The attacker only has to be right once - another information protection fallacy
2010-02b - Another ridiculous cyber warfare game to scare deciders into action
2010-02 - Developing the science of information protection
2010-01 - The Bottom Ten List - Information Security Worst Practices
2009-12-B - COFEE and the state of digital forensics
2009-12 - Using the right words
2009-11 - Passwords again - why we can't leave well enough alone
2009-10 - Partitioning and virtualization - a strategic approach
2009-09 - Forensics: The limits of my tools, my techniques, and myself
2009-08 - Virtualization and the cloud - Risks and Rewards
2009-07 - The speed of light, it's easy to forge, email is always fast, and more
2009-06 - Security Decisions: Deception - When and where to use it
2009-05-B - Culture clash: Cloud computing and digital forensics
2009-05 - Protection testing: What protection testing should we do?
2009-04-B - Proposed Cyber-Security Law: What's the problem?
2009-04 - Risk management: There are no black swans
2009-03 - How spam vigilantes are wrecking email and encourage violations of law
2009-02-B Digital forensics must come of age
2009-02 - A structure for addressing digital forensics
2009-01 - Change management: How should I handle it?
2008-12-B - Short Note: Twittering away your privacy
2008-12 - Digital Forensic Evidence: A Wave Starting to Break
2008-11 - Security Decision: Zoning your network
2008-10 - Social tension and separation of duties
2008-09 - Default deny is best practice? Not anymore!
2008-08 - Control architecture: Access controls
2008-07 - Fault modeling, the scientific method, and thinking out of the box
2008-06 - Inventory Revisited - How to reduce security losses by 70%?
2008-05 - Control Requirements for Control Systems... Matching Surety to Risk
2008-04 - The Botnets have come - The Botnets have come...
2008-03 - Enterprise Information Protection - It's About the Business
2008-02 - The Digital Forensics World
2008-02 - Who Should Do Your Digital Forensics?
2008-01 - Unintended Consequences
2008-01 - Accidental Security
2007-12 - Security, justice, and the future
2007-12 - Security End-of-year

2007-11 - Security by Psychology
2007-11 - Covert Awareness
2007-10 - Making compliance simple - not
2007-10 - Measuring Compliance
2007-09 - Identity Assurance and Risk Aggregation
2007-09 - Identity Assurance
2007-08 - The ethical challenge
2007-08 - Conflicts of Interest
2007-07 - Security Decision Support
2007-07 - Making Better Security Decisions
2007-06 - User platform selection
2007-06 - Which User Platform
2007-05 - Risk Management
2007-05 - Managing Risks
2007-04 - Security Ethics and the Professional Societies
2007-04 - Information Content Inventory
2007-03 - Emerging Risk Management Space
2007-03 - Sensible Security - You Wouldn't?
2007-02 - Emerging Market Presence
2007-02 - Measuring Security
2007-01 - Market Maturity and Adoption Analysis Summary
2007-01 - Closing the Gap
2007-00 - Analysis Framework
2006-12 - The Security Schedule
2006-11 - The Holidays Bring the Fraudsters
2006-10 - Physical/Logical Convergence??
2006-09 - How can I Show I am Me in Email?
2006-08 - Service Oriented Architecture Security Elements
2006-07 - The Life Expectancy of Defenses
2006-07 - BONUS ISSUE: The End of the World as we Know it
2006-06 - Why the CISO should work for the CEO - Three Case Studies

Professional magazine articles:

F. Cohen, "Managing Network Security" Nov. 2002, Breaking In... to test security?
F. Cohen, "Managing Network Security" Oct., 2002 - Reworking Your Firewalls
F. Cohen, "Managing Network Security" Sept, 2002 - Deception Rising
F. Cohen, "Managing Network Security" Aug, 2002 - You're in a Bind!
F. Cohen, "Managing Network Security" July, 2002 - Smashed Again by Stupid Security (appears in Computer Frauds and Security Bulletin)
F. Cohen, "Managing Network Security" July, 2002 - Is Open Source More or Less Secure?
F. Cohen, "Managing Network Security" June, 2002 - Academia's Vital Role in Information Protection
F. Cohen, "Managing Network Security" May, 2002 - Terrorism and Cyberspace
F. Cohen, "Managing Network Security" April, 2002 - Misimpressions We Need to Extinguish
F. Cohen, "Managing Network Security" March, 2002 - Embedded Security
F. Cohen, "Managing Network Security" February, 2002 - How to Get Around Your ISP
F. Cohen, "Managing Network Security" January, 2002 - The End of the Internet as we Know it
F. Cohen, " 2001: Red Teaming Experiments with Deception Technologies"
F. Cohen, " 2001: A Framework for Deception"
F. Cohen, "Managing Network Security - The World Doesn't Want to be Fixed", Network Security,

Dec., 2001.
F. Cohen, "Managing Network Security: The Deception Defense", Network Security, Nov., 2001.
F. Cohen, "Managing Network Security: The DMCA ", Network Security, Oct., 2001.
F. Cohen, "Managing Network Security: The Best Security Book Ever Written ", Network Security, Sep., 2001.
F. Cohen, "Managing Network Security: Special Issue - The Balancing Act ", Network Security, Sep., 2001.
F. Cohen, "Managing Network Security: Bootable CDs", Network Security, Aug., 2001.
F. Cohen, "Managing Network Security: A Matter of Power", Network Security, Jul., 2001.
F. Cohen, "Managing Network Security: The Wireless Revolution", Network Security, Jun., 2001.
F. Cohen, "Managing Network Security: The New Cyber Gang - A Real Threat Profile ", Network Security, May., 2001.
F. Cohen, "Managing Network Security: To Prosecute or Not to Prosecute", Network Security, Apr., 2001.
F. Cohen, "Managing Network Security: Corporate Security Intelligence", Network Security, Mar., 2001.
F. Cohen, "Managing Network Security: Testing Your Security by Breaking In - NOT ", Network Security, Feb., 2001.
F. Cohen, "Managing Network Security: Marketing Hyperbole at its Finest", Network Security, Jan., 2001.
F. Cohen, "Managing Network Security: The Millennium Article - Yet Again! - The Bots are Coming!!! The Bots are Coming!!!", Network Security, Dec., 2000.
F. Cohen, "Managing Network Security: Why Everything Keeps Failing", Network Security, Nov., 2000.
F. Cohen, "Managing Network Security: The Threat", Network Security, Oct., 2000.
F. Cohen, "Managing Network Security: Chipping ", Network Security, Sep., 2000.
F. Cohen, "Managing Network Security: Understanding Viruses Bio-logically", Network Security, Aug., 2000.
F. Cohen, "Managing Network Security: What does it do behind your back?", Network Security, Jul., 2000.
F. Cohen, "Managing Network Security: Why Can't We Do DNS Right?", Network Security, June, 2000.
F. Cohen, "Managing Network Security: Eliminating IP Address Forgery - 5 Years Old and Going Strong ", Network Security, May, 2000.
F. Cohen, "Managing Network Security: Countering DCAs", Network Security, Apr., 2000.
F. Cohen, "Managing Network Security: Collaborative Defense", Network Security, Mar., 2000.
F. Cohen, "Managing Network Security: Worker Monitoring ", Network Security, Feb., 2000.
F. Cohen, "Managing Network Security: Digital Forensics ", Network Security, Jan., 2000.
F. Cohen, "Managing Network Security: Why it was done that way", Network Security, Dec., 1999.
F. Cohen, " So Much Evidence... So Little Time", Information Security Magazine, November. 1999. Special issue with articles by "The 20 Most Influential Figures in Information Security Today."
F. Cohen, "50 Ways to Defeat Your PKI and Other Cryptosystems", The 50 Ways Series at all.net. (/journal/50/index.html)
F. Cohen, "50 Ways to Defeat Your Firewall", The 50 Ways Series at all.net. (/journal/50/index.html)
F. Cohen, "Managing Network Security: The Limits of Cryptography", Network Security, Nov., 1999.
F. Cohen, "Managing Network Security: Security Education in the Information Age", Network Security, Oct., 1999.

F. Cohen, "Managing Network Security: In Your Face Information Warfare", Network Security, Sep., 1999.
F. Cohen, "Managing Network Security: What's Happening Out There", Network Security, Aug., 1999.
F. Cohen, "Managing Network Security: Attack and Defense Strategies", Network Security, July, 1999.
F. Cohen, "Managing Network Security: The Limits of Awareness", Network Security, June, 1999.
F. Cohen, "Managing Network Security: Watching the World ", Network Security, May, 1999.
F. Cohen, "Managing Network Security: Simulating Network Security ", Network Security, Apr., 1999.
F. Cohen, "Managing Network Security: The Millisecond Fantasy", Network Security, Mar., 1999.
F. Cohen, Eli Dart "DARE: Distributed Analysis and REsponse", SANS conference, San Diego, 1999.
F. Cohen, "Managing Network Security: Returning Fire", Network Security, Feb., 1999.
F. Cohen, "Managing Network Security: Anatomy of a Successful Sophisticated Attack", Network Security, Jan., 1999.
F. Cohen, "Managing Network Security: Balancing Risk", Network Security, Dec., 1998.
F. Cohen, "Managing Network Security: The Real Y2K Issue?", Network Security, Nov., 1998.
F. Cohen, "Managing Network Security: Time-Based Security?", Network Security, Oct., 1998.
F. Cohen, "Managing Network Security: What Should I Report to Whom?", Network Security, Sep., 1998.
F. Cohen, "Managing Network Security: Third Anniversary Article - The Seedy Side of Security", Network Security, Aug., 1998.
F. Cohen, "Managing Network Security: How Does a Typical IT Audit Work?", Network Security, Jul., 1998.
F. Cohen, "Managing Network Security: Technical Protection for the Joint Venture", Network Security, Jun., 1998.
F. Cohen, "Managing Network Security: Risk Staging", Network Security, May., 1998.
F. Cohen, "Managing Network Security: The Unpredictability Defense", Network Security, Apr., 1998.
F. Cohen, "Managing Network Security: Red Teaming", Network Security, Mar., 1998.
F. Cohen, "Managing Network Security: The Management of Fear", Network Security, Feb., 1998.
F. Cohen, "Managing Network Security: Y2K - Alternative Solutions", Network Security, Jan., 1998.
F. Cohen, "Managing Network Security: 50 Ways to Defeat Your Intrusion Detection System", Network Security, Dec., 1997.
F. Cohen, "Managing Network Security: To Outsource or Not to Outsource - That is the Question.", Network Security, Nov., 1997.
F. Cohen, "Managing Network Security: The Network Security Game", Network Security, Oct., 1997.
F. Cohen, "Managing Network Security: Change Your Password - Doe See Doe", Network Security, Sep., 1997.
F. Cohen, "Managing Network Security: Penetration Testing?", Network Security, Aug., 1997.
F. Cohen, "Managing Network Security: Relativistic Risk Analysis", Network Security, Jun., 1997.
F. Cohen, "Managing Network Security: Prevent, Detect, and React", Network Security, May., 1997.
F. Cohen, "Managing Network Security: Would you like to play a game?", Network Security, Apr., 1997.
F. Cohen, "Protection Issues in ASCII Red Based on a Limited Unclassified Briefing" (C).
F. Cohen, "Managing Network Security: Risk Management or Risk Analysis?", Network Security, Mar., 1997.

F. Cohen, "Managing Network Security: Network Security as a Control Issue?", Network Security, Feb., 1997.
F. Cohen, "Managing Network Security: Integrity First - Usually", Network Security, Jan., 1997.
F. Cohen, S. Cooper, et. al. "Intrusion Detection and Response", National InfoSec Technical Baseline, October, 1996. (Also appearing in SecureNet 97, March, 1997 and Computers and Security as cited above)
F. Cohen, "Managing Network Security: Where Should We Concentrate Protection?", Network Security, Dec., 1996.
F. Cohen, "Managing Network Security: How Good Do You Have to Be?", Network Security, Nov., 1996.
F. Cohen, "Managing Network Security: Why Bother?", Network Security, Oct., 1996.
F. Cohen, "Internet Holes - The SYN Flood", Network Security, Sep., 1996.
F. Cohen, "Internet Holes - Internet Incident Response", Network Security, Aug., 1996.
F. Cohen, "Internet Holes - Internet Lightning Rods", Network Security, July, 1996.
F. Cohen, "Internet Holes - UDP Viruses", Network Security, June, 1996.
F. Cohen, "Internet Holes - Eliminating IP Address Forgery", Network Security, May, 1996.
F. Cohen, "Internet Holes - Spam", Network Security,April , 1996.
F. Cohen, "Internet Holes - The Human Element", Network Security, March, 1996.
F. Cohen, "Internet Holes - Automated Attack and Defense", Network Security, February, 1996.
F. Cohen, "Internet Holes - 50 Ways to Attack Your World Wide Web Systems", Network Security, December, 1995 - January, 1996.
F. Cohen, "Internet Holes - Network News Transfer Protocol ", Network Security, November, 1995.
F. Cohen, "Internet Holes - Sendmail Attacks", Network Security, October, 1995.
F. Cohen, "Internet Holes - Packet Fragmentation Attacks", Network Security, September, 1995.
F. Cohen, "Internet Holes - Internet Control Message Protocol", Network Security, August, 1995.

Software products developed:

G. Gee, C. Preston, and F. Cohen – Webster CyberLabs, 2015
G. Gee and F. Cohen – White Glove 2015
F. Cohen – Web-based JDM, 2015
F. Cohen - Forensic Fonts, 2009
F. Cohen - Decider, 2007
F. Cohen - JDM, 2006
F. Cohen - Security Decisions, 2006
F. Cohen - Surveyor, 2006
F. Cohen - Influence, 2006
F. Cohen and Garrett Gee, 2002 - White Glove Bootable Linux CD
F. Cohen, 2002 - Responder - large-scale network deception system
F. Cohen and Garrett Gee, 2002 - White Glove developer platform
F. Cohen, Darrian Hale, et. al., 2002 - Resilience - Resilient network infrastructure
F. Cohen, Anthony Carathemus, et. al. 2001 - Secure DNS Server
F. Cohen, Anthony Carathemus, et. al. 2001 - Partition Dump
F. Cohen, Eric Thomas, and Anthony Carathemus, 2001 - Invisible Router
F. Cohen, 2000 - D-Wall - Large-scale high fidelity deception system
F. Cohen, 1999 – ForensiX – Digital Forensics ToolKit for Linux and Unix
F. Cohen, 1999 - Network Security Simulator
F. Cohen and E. Dart, 1998 - DARE - Distributed Analysis and Response
F. Cohen, 1998 - Deception Toolkit
F. Cohen, 1998 - The Security Maze

F. Cohen, 1997 - The Cracking Game
F. Cohen, 1997 - Automated Threat, Attack, and Defense Analysis Tool
F. Cohen, 1996 - CID Database Analysis Tool
F. Cohen, 1995 - Auditor - Internal Audit Tool
F. Cohen, 1995 - Analyzer - Network Audit Tool
F. Cohen, 1995 - Trivial HTTP Daemon - provably secure web server
F. Cohen, 1995 - Secure Gopher Server - provably secure gopher server
F. Cohen, 1993 - Calendar Supplement
F. Cohen, 1992 - PayBack Automated Bill Collection Software
F. Cohen, 1989 - Integrity ToolKit - Integrity Shell and Access Control System
F. Cohen, 1988 - Advanced Software Protection Scanner - Virus Scanner
F. Cohen, 1987 - Advanced Software Protection - Crypto-Checksum Integrity Checker
F. Cohen, 1987 - TRP - Small business office software
F. Cohen, 1986 - VCE - Viral Computing Environment
F. Cohen, 1985 - Legal Assistant - Law Office Software

Appendix C – Material Considered

The Brigham Young University (Fortinet's expert's University) 2005-2006 curriculum year electrical engineering details on graduation requirements - Downloaded from https://registrar.byu.edu/advisement/pdf/05/393550.pdf and included herein by reference, attached as 2005-06-BYU-EE-393550.pdf

The BYU current listing of graduate courses from recent years. Downloaded from http://www.ee.byu.edu/grad/courses and included herein by reference, attached as 2015-10-10-BYU-ECE-GraduateCourseOfferings.pdf

The BYU computer science graduate program information on courses - Downloaded from https://cs.byu.edu/graduate-policy-handbook-appendix included herein bu reference, attached as 2015-10-10-BYU-CS-GraduatePolicyHandbookAppendix.pdf

BYU course description - See https://cs.byu.edu/courses

Course description for undergraduate non-required prerequisite course - Downloaded from http://wiki.cs.byu.edu/cs-465/course-information stored as 2015-10-10-BYU-CS465.pdf

The description for the graduate course in computer security at BYU - Downloaded from http://wiki.cs.byu.edu/cs-665/course-information?do= stored as 2015-10-10-BYU-CS-665.pdf

BYU “CS 360”, titled “Internet Programming” - Downloaded from https://cs.byu.edu/course/cs-360/fall-2015 and included herein and stored in 2015-10-11-BYU-CS360.pdf

VIDEOTAPED DEPOSITION OF SETH NIELSON, Ph.D. Baltimore, Maryland Wednesday, October 21, 2015.

Declaration of Dr. Seth Nielson, Ph.D. (Ex. 1007v2),

U.S. Patent No. 8,261,344 B2 (Ex. 1001V2),

DECISION Institution of Inter Partes Review 37 C.F.R. § 42.108

US 7,913,305 B2

US 7,373,664 B2

US 5,591,698

US 7,448,084 B1

PETITION FOR INTER PARTES REVIEW OF U.S. PATENT NO. 8,261,344 UNDER 35 U.S.C §§ 311-319 AND 37 C.F.R. §§ 42.1-.80 & 42.100-.123

Exhibits 1001 – 1019

Exhibit 3001 – NDCA No. C-13-5831 EMC CLAIM CONSTRUCTION ORDER