

1 **INOCULATOR AND ANTIBODY FOR COMPUTER SECURITY**

2
3 **Inventors: Michael Gregory Hoglund**

4 **Shawn Michael Bracken**

5
6
7 CROSS-REFERENCE TO RELATED APPLICATION

8 This application claims a priority to and claims the
9 benefit of U.S. Provisional Application No. 61/456,192.

10 U.S. Provisional Application No. 61/456,192 is incorporated
11 herein by reference in its entirety.

12
13 BACKGROUND

14 Undesired programs, such as malware, are computer
15 programs or software that perform actions on a computer (or
16 computer system) without the consent of the computer user.

17 Malware can include, for example, viruses, worms, Trojan
18 horses, backdoors, spyware, adware, botnets, or other
19 malware or other programs that can have a malicious effect
20 {or perform other undesired functions) on a computer. A
21 malicious effect on a computer is, for example, when the
22 computer (or computer resources or objects) becomes
23 damaged, disrupted, corrupted, undesirably burdened, or
24 otherwise affected in an undesirable manner. As known to

1 those skilled in the art, what is undesirable to a user may
2 actually be desirable to another user or may even be a
3 viable software product of a vendor or software tool in the
4 malware testing procedures.

5 As some specific examples, botnets are software agents
6 used to create and send spam, malware, or viruses, or flood
7 a network with messages such as a denial of service attack.
8 Spyware is a type of malware that can be installed on
9 computers and collects information about users without
10 their knowledge. Viruses are programs that can damage or
11 disrupt programs or computers, and can reproduce itself and
12 infect other programs or computers.

13 Current approaches to cleaning {e.g., removing,
14 deleting, or quarantining) of undesired programs or
15 preventing infections of undesired programs require, for
16 example, the use of software agents, active running code
17 (processes), or behavioral- blocking software. For example,
18 firewalls are used to block known malware, but uses
19 computer resources to function. Conventional malware
20 protection solutions may often be expensive and difficult
21 for large businesses (or organizations) to maintain.

22 Some specific non-malicious computer programs may also
23 be undesired by a business. For example, a business {e.g.,
24 a large corporation) may not want peer-to-peer filesharing

1 programs (or other types of computer programs) in the
2 network of the business. Additionally, these non-malicious
3 undesired programs can consume computer resources such as,
4 for example, hardware resources (e.g., memory space and/or
5 processor resource) and/or software resources (e.g.,
6 operating system tasks). Therefore, a computer user may
7 desire to prevent the installment of particular computer
8 programs that are not necessarily malicious code.

9 When a user discovers the presence of an undesired
10 computer program in a computer (or the presence of an
11 undesired program in devices in a network, or the presence
12 of an undesired program in a portable computing device such
13 as, for example, a personal digital assistant, smart phone,
14 iPad, or iPhone), the user can usually remove the undesired
15 program by use of a currently available solution. For
16 example, current software programs are available to remove
17 malware or viruses in a computer. However, the computer
18 can often be re-infected by the same undesired program. If
19 the re-infection rate is high (e.g., approximately 50% or
20 greater), then the cost of maintaining the computer will
21 become expensive for a business, particularly if multiple
22 computers become re-infected with the same undesired
23 program.

1 This increased cost is likely incurred if the business
2 uses an "enterprise system" which is a large-scale,
3 organization-wide, integrated application-software package.
4 For example, if a business is re-imaging a computer for use
5 with or as part of the enterprise system, and after
6 analyzing the re-imaged computer, the re-imaged computer is
7 found to be infected or re-infected by an undesired program
8 {e.g., malware or virus) that has previously infiltrated
9 the computers of the business, then the procedure of having
10 to remove the re-infection will lead to increased costs for
11 the business.

12 The current technology does not provide a less
13 complicated and less expensive approach to preventing the
14 re-infection of computers or networked devices. Therefore,
15 the current technology is limited in its capabilities and
16 suffers from at least the above constraints and
17 deficiencies.

1 BRIEF DESCRIPTION OF THE DRAWINGS

2 Non-limiting and non-exhaustive embodiments of the
3 present invention are described with reference to the
4 following figures, wherein like reference numerals refer to
5 like parts throughout the various views unless otherwise
6 specified.

7 Figure 1 is a block diagram of an apparatus (system)
8 in accordance with an embodiment of the invention.

9 Figure 2 is a block diagram showing various features
10 in the apparatus of Figure 1, in accordance with various
11 embodiments of the invention.

12 Figure 3 is a block diagram of a method in accordance
13 with an embodiment of the invention.

1 DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS

2 In the description herein, numerous specific details
3 are provided, such as examples of components, modules,
4 software, programmed code, and/or methods, to provide a
5 thorough understanding of embodiments of the invention.
6 One skilled in the relevant art will recognize, however,
7 that an embodiment of the invention can be practiced
8 without one or more of the specific details, or with other
9 apparatus, systems, methods, modules, software, programmed
10 code, components, materials, parts, and/or the like. In
11 other instances, well-known structures, materials, modules,
12 software, programmed code, or operations are not shown or
13 described in detail to avoid obscuring aspects of
14 embodiments of the invention. Additionally, the figures
15 are representative in nature and their shapes are not
16 intended to illustrate the precise shape or precise size of
17 any element and are not intended to limit the scope of the
18 invention.

19

20 Figure 1 is a block diagram of an apparatus (system}
21 100 in accordance with an embodiment of the invention, or
22 is a block diagram an apparatus (system) that can be used
23 in an embodiment of the invention. An example computing
24 device 105 (or computer 105) is connectable to and can

1 communicate along a communication network 110. Therefore,
2 the computing device 105 can communicate with and download
3 data objects or programs from a remote server 112 via the
4 network 110. An inoculator engine 114 in accordance with
5 an embodiment of the invention will operate on the platform
6 of the server 114, and the functionality of the inoculator
7 engine 114 will be discussed below.

8 The computing device 105 can be, for example and
9 without limitations, a computer (e.g., a personal
10 computer), server, workstation, mainframe computer,
11 minicomputer, microcomputer, notebook or laptop computer,
12 palmtop, a smart device (e.g., a smart printer, smart
13 camera, smart scanner, or other smart devices), or a
14 portable computing device (e.g., cell phone, smart phone,
15 personal digital assistant, iPad, iPhone, or other portable
16 computing devices), or another type of computer.

17 The communication network 110 is, for example and
18 without limitations, a public network such as a wide area
19 network (e.g., Internet), a local area network (LAN), or a
20 different type of private network, semi-private, or public
21 network. The link(s) forming the network 110 can be wired,
22 wireless, or a combination of both.

23 The computing device 105 includes standard elements
24 115 that are used in computing operations or data

1 transmissions. Some of the elements 115 shown in Figure 1
2 are hardware elements or objects (e.g., disk drivers). Of
3 course, the components in the elements 115 may vary
4 depending on the type of computing device. For example,
5 some hardware elements in a personal computer can vary in
6 comparison to the hardware elements in a PDA or smart
7 phone. As an example, the elements 115 include a processor
8 120, one or more memory devices 125, storage devices 130
9 such as disks, ports 140 (which could alternatively be
10 components in a network interface 142}, a disk driver 145,
11 a network driver 150 (which could alternatively be a
12 component in the network interface 142), and/or other known
13 elements that are used in computing devices.

14 The computing device 105 also includes software
15 elements 152 such as, for example, an operating system (OS)
16 155 that performs management functions and other functions
17 that are known to those skilled in the art. The OS 155 can
18 be one of various types of known OS 155 such as, for
19 example, the Microsoft WINDOWS® based operating system, MAC
20 OS®, MAC OS X®, iOS (mobile operating system), UNIX or
21 UNIX-like OS, BSD and its descendant operating systems,
22 LINUX AND GNU, Google CROME OS®, OS/2, or another operating
23 system. It is noted that at least some of the above names
24 for operating systems are registered trademarks. As will

1 be discussed below, the inoculator 114 is configured to
2 transmit a block object 160 (i.e., block 160 or wedge 160)
3 onto the computing device 105 to prevent a known specific
4 software program from being installed or executing in the
5 computing device 105. The block object 160 is a data
6 object that is an inert object {or a null object or a data
7 object that does not perform a function and does not
8 execute a function) . This block object 160 is also
9 alternatively referred herein as a "block 160" or a "wedge
10 160". A particular block object 160 will block the
11 installation or execution of a particular computer program
12 on the computing device 105, as will be discussed below in
13 additional details. The computer program that will be
14 blocked can be a malicious computer program or a non-
15 malicious computer program.

16 The computing device 105 can also include a user
17 interface 165 or/and other peripheral components. Of
18 course, the user interface 165 may vary, depending on the
19 type of computing device 105. For example, a user
20 interface for a notebook computer can vary from a user
21 interface for a PDA or a user interface for a smart phone.
22 The interface 165 can be integrated with or can be coupled
23 to and function with the computing device 105. As an
24 example, the interface 165 for a personal computer can

1 include at least one of the following: mouse {or control
2 pad) 170, keyboard 176, and/or screen (or touchscreen) 180.
3 A bus 153 can permit the above discussed components for the
4 computing device 105 to communicate with and function
5 together. Other standard hardware, software, or firmware
6 components that can be used in the device 105 are not shown
7 in Figure 1 for purposes of clarity in the drawings.

8 The inoculator engine 114 is formed by software code
9 based on a standard programming language (e.g., C, C++, or
10 other suitable languages)

11

12 Figure 2 is a block diagram that shows additional
13 details of the apparatus 100 of Figure 1, in accordance
14 with an embodiment of the invention. Assume that a known
15 computer program 205 (i.e., unwanted or undesired computer
16 program 205, or unwanted or undesired software program 205)
17 is, for example, a known malware or a known non-malicious
18 program. A malware could be, for example, viruses, worms,
19 Trojan horses, backdoors, spyware, adware, botnets, or
20 other malware or other programs that can have a malicious
21 effect (or perform other undesired functions) on a
22 computer. A non-malicious program could be, for example,
23 peer-to-peer filesharing programs (or other types of
24 computer programs that are non-malicious) . Other non-

1 malicious programs can be, for example, computer programs
2 that execute computer games, videos, or other programs that
3 a business would like to prevent from being installed (or
4 re-installed) in the computing device 105 so that employees
5 are not distracted and can remain productive in the
6 workplace. The known computer program 205 could also be a
7 recently discovered or recently analyzed malware program or
8 non-malicious program that has previously infected the
9 computing device 105 (or infected any other devices that
10 are connected to the network 110 or connected to the
11 computing device 105), has been previously deleted from the
12 computing device 105, and then previously analyzed in its
13 attributes 210. The attributes 210 of a computer program
14 205 can be stored in, for example, a separate remote
15 database 212 or in a database in the remote server 112, or
16 in another device (not shown in Figure 2) that can be
17 accessed by the remote server 112. Various methods for
18 analyzing attributes of a computer program are known to
19 those skilled in the art.

20 In one example, the computer program 205 has been
21 previously installed in the computing device 105 (or
22 another computing device that can communicate with the
23 computing device 105 or can communicate on network 110) .

1 Assume that computer program 205 has infected the
2 computing device 105 (or another device that can
3 communicate with the computing device 105 or can
4 communicate on network 110) . As one example, in order to
5 remove the computer program 205 from the computing device
6 105, the Windows Management Interface (WMI) can be used to
7 connect to the computing device 105. As known to those
8 skilled in the art, WMI allows management and control of
9 computing devices with a particular operating system such
10 as, for example, Windows OS. As an example, the computing
11 devices to be managed via WMI can be in a network such as,
12 for example, an enterprise network. Network administrators
13 can use WMI to query and set information on computing
14 devices, applications, and networks. Use of WMI allows the
15 administrator to use Microsoft-supported API (application
16 program interface} calls to access the registry 230 (in the
17 file system 220), and determine if a particular computer
18 program 205 (malicious computer program or non-malicious
19 computer program) has been installed in the computing
20 device 105.

21 The administrator can use API calls to write data to a
22 specific registry key 222. As an example, this specific
23 registry key 222 is the registry key that is used from
24 Microsoft service pack installations or updates, and this

1 registry key will contain instructions about the particular
2 files and registry keys to remove after a re-boot of the
3 computing device 105. To remove a particular computer
4 program 205 from the computing device 105, the instructions
5 for removing the particular computer program 205 is written
6 into the specific registry key 222. When the computing
7 device 105 is re-booted, the operating system 115 will read
8 the specific registry key 222 and will remove the
9 particular computer program 205 (and also remove the
10 registry key 206 for the computer program 205) from the
11 computing device 105. The OS 155 performs this removal of
12 the program 205 and registry key 206 based on the
13 instructions in the registry key 222 for removing the
14 particular program 205 and registry key 206. Therefore,
15 features in the WMI can be advantageously used for cleaning
16 (e.g., inoculating, removing or deleting or quarantining) a
17 computer program 205 from the computing device 105. Based
18 on the discussion herein, it is understood that other
19 methods for removing a computer program 205 from the
20 computing device 105 may also be advantageously used.

21

22 Assume that the attributes 210 of the known computer
23 program 205 are then analyzed. Those skilled in the art
24 can analyze a computer program to determine the following

1 characteristics or details of the attributes 210. As an
2 example, the attributes 210 indicate that the known program
3 205 has a software name "UNWANTED- 1" and file extension
4 "ext", Assume further that attributes 210 indicate that
5 the known computer program 205 will be stored by the files
6 system 220 into the location 215. As known to those
7 skilled in the art, a file system is one of the subsystems
8 in any operating system type (platform) . Typically, the
9 location 215 will be an area in the memory 125 (Figure 1}
10 of the computing device 105.

11 The attributes 210 may also indicate that the known
12 program 205 will create the registry key 206 into the
13 registry 230 (which is managed by the file system 220 and
14 is in a memory area 235 of memory 125 (Figure 1)), if the
15 known program 205 is attempting to install into a WINDOWS-
16 based operating system or is attempting to install in
17 another type of operating system that uses a registry-type
18 system for installed software.

19 As known to those skilled in the art, a registry is a
20 database used by the Windows operating system to store
21 configuration information about the computer program. Most
22 computer programs would write data to the registry, at
23 least during installation.

1 Those skilled in the art will realize that variations
2 or alternatives in the configurations discussed herein can
3 be made. For example and as discussed herein, some objects
4 (or other storage components) that are stored in memory
5 (e.g., Random Access Memory) may not be resident in the
6 memory at all times, and stored on disk and loaded into
7 memory when these objects are accessed. As a more specific
8 example, some objects (or other storage components) such as
9 the disk driver 145 (Figure 1), the registry 230, and/or
10 the file system 220 are typically stored on disk and may
11 not be resident in RAM at all times, and loaded into RAM
12 only when these objects (or storage components) are
13 accessed.

14

15 Alternatively or additionally, the attributes 210 may
16 indicate that the known program 205 will create a
17 configuration information file 240 (i.e., settings file
18 240) if the known program 205 is attempting to install into
19 a non-WINDOWS based operating system. The configurations
20 file 240 effectively registers the known program 205 when
21 the program 205 installs into the computing device 105,
22 For example, in the MAC® OS® operating system, the
23 configurations file 240 is a preference file 240. A
24 computer program 205 would rebuild its preference file 240

1 if the preference file is deleted. It is within the scope
2 of embodiments of the invention for the configuration file
3 240 to encompass any suitable configuration file or
4 metadata file, or other data file that would be associated
5 with a computer program 205 and that would be used by any
6 suitable type of operating system that is not necessarily
7 limited to the MAC OS operating system.

8
9 In an embodiment of the invention, an inoculator
10 engine 114 can be stored in a memory 250 and can be
11 executed by a processor 255 in the remote computing device
12 112 (which can be, for example and without limitations, a
13 server or another type of computer) . If the operating
14 system 155 is a Windows operating system, then the
15 administrator can permit the processor 255 to execute the
16 inoculator engine 114. The inoculator engine 114 will read
17 the attributes 210, including registry key 206, that has
18 been stored or gathered in the database 212 and that are
19 attributes or metadata associated with the known computer
20 program 205. The attributes 210 would typically indicate
21 the memory space or memory address space (in target device
22 105) that the computer program 205 will reside after
23 installation or the file system location (in target device
24 105) in which the computer program 205 will reside after

1 installation. The attributes 210 may also indicate the
2 registry location 226 that will receive a registry key 206
3 of the undesired computer program 205.

4 In one embodiment of the invention, the inoculator
5 engine 114 will provide and transmit 260 across network 110
6 a block object 160 (i.e., block 160 or wedge 160) onto the
7 computing device 105 to prevent a known specific software
8 program (e.g., known program 205) from being installed or
9 executing in the target computing device 105. The block
10 object 160 is a data object that is an inert object (or a
11 null object or a data object that does not perform a
12 function and does not execute a function} . This block
13 object 160 is also alternatively referred herein as a
14 "block 160" or a "wedge 160" or a "dummy file 160" which
15 does not execute but only occupies memory space. This
16 block object 160 can be, for example, a given data of a
17 particular size that does not perform any function and that
18 occupies the memory space (or memory address space) of the
19 known computer program 205 that is to be prevented from re-
20 installing into the computing device 105.

21 A file system 220 of the OS 155 will install the block
22 data 160 into a memory area 215. For a Windows type
23 operating system, the file system 220 will also install a
24 fake registry key 225 (null registry key 225) in a registry

1 area 226 in which an unwanted computer program 205 would
2 otherwise generate and install a registry key 206 when the
3 unwanted computer program 205 will attempt to install.
4 This null registry key 225 is also given data of a
5 particular size that does not perform any function (i.e.,
6 does not execute) and that occupies the registry area 226
7 of the registry key 206 of the undesired computer program
8 205.

9 In another type of operating system, the file system
10 220 will install a fake configuration information 241 (null
11 configuration information 241) into the configuration file
12 240, where this fake configuration information 241 is for
13 the block object 160 (null object 160, data object 160, or
14 dummy object 160) . For MAC OS operating systems, this
15 configuration file is commonly known as a preference file.
16 The fake configuration information 241 is also an inert
17 object or non-functioning data object. In other words,
18 this null configuration information 241 is also given data
19 of a particular size that does not perform any function
20 (i.e., does not execute) and that occupies the memory area
21 of the configuration information of an undesired computer
22 program 205 to be installed in a non-Windows system.
23

1 The null registry key 225 (for Windows-based systems
2 or other OS-types the use registry keys) is also referred
3 herein and in the below claims as a null configuration
4 object 225. The null configuration information 241 (for
5 non-Windows-based systems) is also referred herein and in
6 the below claims as a null configuration object 241. A
7 null configuration object 225 (or 241) is associated with
8 the null object 160 (data object 160) .

9
10 It is noted that all types of operating systems (e.g.,
11 operating systems in mobile devices such as the iPad or
12 iPhone or smart phones) have file systems. These file
13 systems can then be used to install an embodiment of the
14 block object 160 and/or an embodiment of the fake
15 configuration information 241, in order to prevent the
16 installation of an unwanted computer program 205 into a
17 target device 105.

18 A particular block object 160 (data object 160) will
19 block the installation (re-installation) or execution of a
20 particular computer program 205 on the computing device
21 105. The computer program 205 that will be blocked can be
22 a malicious computer program or a non-malicious computer
23 program. When the program 205 attempts to re-install into
24 the device 105, the block object 160 will occupy the memory

1 area 215 that is the destination memory space of the
2 program 205. As a result, the block object 160 prevents
3 the program from re-installing into the device 105 and is
4 an antibody (or antibody in functionality) that prevents
5 the program 205 from installing into or re-infecting the
6 device 105. Since the block object 160 is in the
7 destination memory space 215 of the program 205, the
8 program 205 will not be able to install and will be subject
9 to an error occurrence.

10 As known to those skilled in the art, for a computer
11 program that installs into a Windows operating system, that
12 computer program will also register in the registry. In an
13 embodiment of the invention, the inoculator engine 114
14 would also provide and transmit 260 a fake registry key 225
15 (or dummy registry key 225 or substitute registry key 225)
16 into the registry 230. This fake registry key 225 is
17 associated with the block object 160. The operating system
18 155 will install the fake registry key 225 in the registry
19 230 as a substitute for the actual registry key 206 of the
20 computer program 205. This fake registry key 225 is
21 installed in the area 226 of the registry 230 where the
22 computer program 205 (e.g., a malware 205 or non-malicious
23 program 205) would create a registry key 206 during the
24 installation of the computer program 205 into the target

1 device 105. Since the fake registry key 225 is already
2 installed in the registry 230 when the computer program 205
3 is attempting to install into the target device 105, the
4 computer program 205 will encounter an error because the
5 computer program 205 will be unable to delete the fake
6 registry key 225.

7 Therefore, the block data 160 and/or fake registry key
8 225 will block the target device 105 areas in which the
9 computer program 205 will need to "live" or install.

10 In one embodiment of the invention, the computer
11 program 205 will be unable to delete the fake registry key
12 225 because the security permission settings 265 (which is
13 managed by the security subsystem 266) will prevent the
14 computer program 205 from modifying the value of the fake
15 registry key 225 into a proper value associated with the
16 undesired program 205. As a result, the computer program
17 205 will not successfully install in the target device 105
18 and the target device 105 is protected from re-infection by
19 the computer program 205. A creator of the computer
20 program 205 would have to, for example, change the name of
21 the computer program 205 or use different registry keys
22 other than the registry key 206 for the program 205, in
23 order to potentially allow successful installation of the
24 computer program 205 into the device 105.

1

2 In another embodiment of the invention, the security
3 permission 265 in the OS 155 can be set for the block
4 object 160 so that the unwanted computer program 205 is
5 unable to modify or replace the installed block object 160.
6 For example, the security permission 265 may be set so that
7 only a network administrator can modify the installed block
8 object 160. The security permissions 265 for data objects
9 can be set by use of the security subsystem 266 and can be
10 remotely set or configured, via network 110, by a network
11 administrator. The security permission 265 settings
12 provide a security or a guard against an unwanted software
13 program 205 from successfully over-writing, deleting,
14 replacing, or modifying the block object 160. The security
15 subsystem 266 sets the security permissions 265 for the
16 block object 160 and/or for the null registry key 225.

17

18 Assume that the name of the unwanted computer program
19 205 is "UNWANTED- 1". In another embodiment of the
20 invention, to prevent the unwanted software program 205
21 from successfully over-writing, deleting, replacing, or
22 modifying the block object 160 which blocks future access
23 of the computer program 205 into memory space 215, the
24 network administrator or user can change, via file system

1 220 and security subsystem 266, the type of the block
2 object 160 but keep the name of the block object to be the
3 same. For example, the network administrator can change
4 the block object 160 from a file object 160 into a
5 directory 160A which is a different type of object from a
6 file object. The name of the directory 160A will be the
7 same as the previous file object 160 which is "UNWANTED- 1".
8 When the unwanted computer program 205 (with the name
9 "UNWANTED- 1") tries to install into the memory area 215 in
10 the future, the unwanted computer program 205 will not be
11 able to install because of a name collision with the
12 directory 160A which has the same name "UNWANTED- 1". This
13 name collision will cause the OS 155 to trigger an error
14 code, which can alert the network administrator or user of
15 target device 105 of an attempted installation into the
16 target device 105 by an unwanted computer program 205.

17.

18 In another embodiment of the invention, to prevent the
19 unwanted software program 205 from successfully over-
20 writing, deleting, replacing, or modifying the directory
21 160A (which, as mentioned in the above example, is named
22 "UNWANTED-1") , the network administrator or user can, via
23 file system 220 and security subsystem 266, place another
24 block object 160B in the directory 160A which is in memory

1 area 215. As with the block object 160 in the above
2 examples, the block object 160b is also a non-functioning
3 or non-executing piece of data object. The block object
4 160B can have a name (e.g., "UNWANTED- 2") which is
5 different from the name of the block object 160 or can have
6 a same name (e.g., "UNWANTED- 1") as the name of the block
7 object 160. Since the block object 160B is in the
8 directory 160A, the unwanted computer program 205 will not
9 be able to delete or modify the directory 160A when the
10 computer program 205 attempts to install into the target
11 device 105 in the future.

12

13 In another embodiment of the invention, the OS 155 can
14 be programmed to generate a security event 272, if the fake
15 registry key 225 is attempted to be accessed, modified, or
16 deleted by an intruding computer program 205 that is
17 attempting to install or re-install into the target device
18 105, or if the block object 160 is attempted to be
19 accessed, modified, or deleted by an intruding computer
20 program 205 that is attempting to install or re-install
21 into the target device 105. The OS 155 (e.g., via security
22 subsystem 266) can trigger this security event 155 if a
23 computer program 205 attempts to access (e.g., attempted
24 reading, attempted writing, attempted modification, or

1 attempted deletion) a data object in a particular memory
2 area 215 and/or particular registry area 226. This
3 security event 272 is then transmitted to the auditing
4 subsystem 274, and the subsystem 274 can record this
5 security event 275 into a security event log 276. Various
6 commercially available software and open source software
7 can be used to read and analyze the security event log 276.
8 Effectively, the security event 272 is a "trip wire" or
9 "burglar alarm" that alerts the network administrator or user
10 of target device 105 when a computer program 205 is trying
11 to install into the target device 105 and accesses or
12 attempts to modify or delete the fake registry 225 and/or
13 the block object 160. In another embodiment of the
14 invention, the above-discussed security event 272 feature
15 is omitted.

16

17 In an example operation that is a non-limiting
18 example, assume that a piece of software 205 (which can be
19 malicious or non-malicious) registers as a service in a
20 computer system with, for example, a Microsoft Windows
21 operating system or other type of operating systems. In
22 the Windows OS, assume that the software 205 registers
23 under the registry key 206 named "current control
24 set/services" and the service name is "bad-software". When

1 this software 205 installs or infects the target device
2 105, the software 205 creates that registry key 206 into
3 the registry 230. The software 205 can be cleaned (e.g.,
4 deleted or removed) from the target system 205 by use of
5 methods as, for example discussed above.

6 When the attributes 210 of this software 205 has been
7 analyzed and recorded (e.g., in database 212), the
8 inoculator engine 114 can place an antibody into the target
9 system 105 to prevent the software 205 from installing into
10 the target device 105 in the future. As discussed above,
11 this antibody can be a block object 160 (wedge 160) ,
12 directory 160A which would have the same name as the
13 software 205, directory 160A with a block object 160B, fake
14 registry key 225 that is associated with the block object
15 160, and/or fake configuration information 241 that is
16 associated with the block object 160. The above components
17 in an antibody can differ based on, for example, the type
18 of operating system and/or the amount of security that the
19 network administrator prefers to place on a block object 160,
20 directory 160A, registry key 225, and/or fake configuration
21 information 241, as discussed above.

22 The inoculator engine 114 would open the registry 235,
23 via remote procedure calls (RPCs) along network 110. RPCs
24 are built-in capabilities in the Microsoft based operating

1 systems. As known to those skilled in the art, RPC is a
2 type of protocol that allows a program on one computer to
3 execute a program in another computer. The program in one
4 computer sends a message to the other computer with
5 appropriate arguments and the other computer returns a
6 message containing the results of the program executed.

7 The inoculator engine 114 would create a registry key
8 named as "bad-software" underneath the "current control
9 set/services" registry path. Therefore, the inoculator
10 engine 114 creates a registry key of the same name that
11 would have been created by the program 205 attempted to
12 install into the target device 105.

13 As similarly discussed above, the inoculator engine
14 114 can be used to set the security permission 265 on the
15 registry key 225. For example, the security permission 265
16 would prevent read access and write access to the registry
17 key 225 for anyone except the owner of the registry key
18 225. The owner of the registry key 225 is typically the
19 network administrator.

20 In this same example, assume that later, the user of
21 device 105 is browsing the Internet or another networks,
22 and the same program 205 (named "bad-software") attempts to
23 exploit the target device 105. The "bad-software" program
24 205 will create its registry key 206 (named "current

1 control set/services/bad-software") and attempt to install
2 its registry key 206 into the registry 230, as the program
3 205 is attempting to install into the target device 105.
4 Since the registry key 225 was previously installed in the
5 registry 230, an access denied error would be generated in
6 a code line in the program 205. This error code is
7 generated because the registry key 225 is already installed
8 in the registry 230 and indicates that access to the
9 registry area occupied by registry key 225 is denied for
10 the program 205. Since the program 205 will typically not
11 have the proper security permission credentials to over-
12 write or modify the registry key 225, the error code is
13 generated. The program 205 will have no programming
14 response for dealing with this error code, the program 205
15 will be unable to successfully complete installation into
16 the target device 105 and when the target device 105 is re-
17 booted, the program 205 is deleted by the OS 155 from the
18 target device 105 because the proper registry key and file
19 path is not installed for the program 205 to survive during
20 re-boot of the target device 105.

21 In another embodiment of the invention, the antibodies
22 discussed above (e.g., block object 160, directory 160A,
23 registry key 225, and/or fake configuration information
24 241) can be placed in a target device 105 that has not been

1 previously infected by the computer program 205. This
2 solution permits active inoculation of target devices 105
3 against exploits by known unwanted software 205.

4

5 Therefore, in an embodiment of the invention, an
6 apparatus and/or method is provided with an inoculator and
7 antibody for computer security. A block object (i.e.,
8 block or wedge) is installed onto a target computing device
9 105 (i.e., target computer 105 or computing system 105 in
10 Figure 1), where the block object prevents a specific known
11 software program from being installed and executing in the
12 target computing device. As an example, this block object
13 can be installed from a remote device, via a network, to
14 the target device 105 by use of Microsoft Windows RPC
15 calls. The solution provided by an embodiment of the
16 invention advantageously does not require the installation
17 of a software agent on the target device. Instead, the
18 solution provides data objects, files, and/or registry keys
19 to prevent installation of an unwanted computer software
20 program 205. Thus, an embodiment of the invention
21 advantageously avoids the prior solutions' use of agents,
22 active running code, behavioral block software, or other
23 prior solutions that can be expensive or can consume system
24 resources. The target system in the target device can be,

1 for example, the Microsoft Windows OS, although other types
2 of operating systems can be protected by an embodiment of
3 the invention. Once the block object is installed
4 {"wedged"}) in the target system, the target system will be
5 protected or immune against the specific unwanted software
6 program that tries to install or execute in the target
7 system. As discussed above, fake registry keys, along with
8 the block object, may also be used to protect the target
9 system.

10 The solution provided by an embodiment of the
11 invention can make a target system immune against specific
12 malicious software program (typically called "malware") .
13 However, the solution provided by an embodiment of the
14 invention can also make a target system immune against non-
15 malicious software as selected by a network administrator
16 or other users.

17

18 Figure 3 is a block diagram of a method 300 in
19 accordance with an embodiment of the invention. An area
20 where an undesired computer program will reside is first
21 determined (block 305) . A data object is then placed in the
22 area, so that the data object is an antibody that provides
23 security to the computer and immunity against the undesired
24 computer program {block 310) . As an optional step in block

1 315, actions can be performed to prevent the access (e.g.,
2 deletion and/or modification) of the data object. As
3 discussed above, preventing the access, deletion and/or
4 modification of the data object can include setting
5 security permissions for the data object, modifying the
6 type of the data object, or placing a second data object
7 into the data object.

8 In another embodiment of the invention, the access
9 (e.g., deletion and/or modification) to the null registry
10 key 225 (or null configuration information 241, if
11 applicable) can also be prevented by, for example, setting
12 security permissions for the null registry key 225 or null
13 configuration information 241, by use of the security
14 subsystem 266.

15 Prior to determining the area in block 305, the
16 undesired computer program can be cleaned (inoculated) from
17 the computer or another device in a network.

18

19 It is also within the scope of the present invention
20 to implement a program or code that can be stored in a
21 machine-readable or computer-readable medium to permit a
22 computer to perform any of the inventive techniques
23 described above, or a program or code that can be stored in
24 an article of manufacture that includes a computer readable

1 medium on which computer-readable instructions for carrying
2 out embodiments of the inventive techniques are stored,
3 including embedded devices such as Field Programmable Gate
4 Arrays (FPGAs) or other specialized equipment. Other
5 variations and modifications of the above-described
6 embodiments and methods are possible in light of the
7 teaching discussed herein.

8 The above description of illustrated embodiments of
9 the invention, including what is described in the Abstract,
10 is not intended to be exhaustive or to limit the invention
11 to the precise forms disclosed. While specific embodiments
12 of, and examples for, the invention are described herein
13 for illustrative purposes, various equivalent modifications
14 are possible within the scope of the invention, as those
15 skilled in the relevant art will recognize.

16 These modifications can be made to the invention in
17 light of the above detailed description. The terms used in
18 the following claims should not be construed to limit the
19 invention to the specific embodiments disclosed in the
20 specification and the claims. Rather, the scope of the
21 invention is to be determined entirely by the following
22 claims, which are to be construed in accordance with
23 established doctrines of claim interpretation.

What is claimed is:

1. A method performed by a first computing device, the method comprising:
determining, in a second computing device that is distinct and separate from the first computing device, a location where an undesired computer program is configured to reside upon installation on the second computing device; and
providing, to the second computing device, a data object to occupy the determined installation location of the undesired computer program in the second computing device to prevent the undesired computer program from being installed in the determined installation location in the second computing device.
2. The method of claim 1, further comprising:
providing a null configuration object in a second location in the second computing device, wherein the null configuration object is associated with the data object.
3. The method of claim 1, further comprising:
preventing a deletion or modification of the data object.
4. The method of claim 3, wherein preventing the deletion or modification of the data object comprises:
setting a security permission for the data object.
5. The method of claim 3, wherein preventing the deletion or modification of the data object comprises:
setting the data object to have an object type that differs from the undesired program.
6. The method of claim 5, wherein preventing the deletion or modification of the data object comprises:
placing a second data object in the data object.
7. The method of claim 1, wherein the location comprises a memory space and/or a file system location.
8. The method of claim 1, wherein the data object has the same name as the undesired computer program.

9. A first computing device, comprising:

one or more processors; and

memory storing one or more programs for execution by the one or more processors, the one or more programs including instructions for:

determining, in a second computing device that is distinct and separate from the first computing device, a location where an undesired computer program is configured to reside upon installation on the second computing device; and

providing, to the second computing device, a data object to occupy the determined installation location of the undesired computer program in the second computing device to prevent the undesired computer program from being installed in the determined installation location on the second computing device.

10. The first computing device of claim 9, wherein the one or more programs include instructions for: providing a null configuration object in a second location in the second computing device, wherein the null configuration object is associated with the data object.

11. The first computing device of claim 9, wherein the location comprises a memory space and/or a file system location.

12. The first computing device of claim 9, wherein the data object has the same name as the undesired computer program.

13. A non-transitory computer readable storage medium storing one or more programs for execution by one or more processors of a first computing device, the one or more programs including instructions for: determining, in a second computing device that is distinct and separate from the first computing device, a location where an undesired computer program is configured to reside upon installation on the second computing device; and providing, to the second computing device, a data object to occupy the determined installation location of the undesired computer program in the second computing device to prevent the undesired computer program from being installed in the determined installation location on the second computing device.

14. The computer readable storage medium of claim 13, wherein the one or more programs include instructions for:
providing a null configuration object in a second location in the second computing device, wherein the null configuration object is associated with the data object.
15. The computer readable storage medium of claim 13, wherein the one or more programs include instructions for:
preventing a deletion or modification of the data object.
16. The computer readable storage medium of claim 15, wherein preventing the deletion or modification of the data object comprises:
setting a security permission for the data object.
17. The computer readable storage medium of claim 15, wherein preventing the deletion or modification of the data object comprises:
setting the data object to have an object type that differs from the undesired program.
18. The computer readable storage medium of claim 17, wherein preventing the deletion or modification of the data object comprises:
placing a second data object in the data object.
19. The computer readable storage medium of claim 13, wherein the location comprises a memory space and/or a file system location.
20. The computer readable storage medium of claim 13, wherein the data object has the same name as the undesired computer program.