



(51) International Patent Classification:
G06F 21/00 (2006.01)

(21) International Application Number:
PCT/DK2010/050049

(22) International Filing Date:
25 February 2010 (25.02.2010)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
PA 2009 00265 25 February 2009 (25.02.2009) DK
61/155,387 25 February 2009 (25.02.2009) US

(71) Applicants (for all designated States except US):
AARHUS UNIVERSITET [DK/DK]; Nordre Ringgade
1, DK-8000 Århus C (DK). **MJØLNER INFORMATICS A/S** [DK/DK]; Finlandsgade 10, DK-8210 Århus V (DK).

(72) Inventors; and

(75) Inventors/Applicants (for US only): **GAMMELGAARD, Christian, Horsdal** [DK/DK]; Sandøgade 4, DK-8200 Århus N (DK). **SKYUM, Sven** [DK/DK];

Skoleparken 125, DK-8330 Beder (DK). **DALSGAARD, Søren, Pingel** [DK/DK]; Bakkehæddet 5, DK-8250 Lystrup (DK). **JØRGENSEN, Aksel, Byskov** [DK/DK]; Niels Bohrs Vej 1, DK-8260 Viby J (DK).

(74) Agent: **PLOUGMANN & VINGTOFT A/S**; Sundkrogs-
gade 9, P.O. Box 831, DK-2100 Copenhagen Ø (DK).

(81) Designated States (unless otherwise indicated, for every kind of national protection available): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States (unless otherwise indicated, for every kind of regional protection available): ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV,

[Continued on next page]

(54) Title: CONTROLLED COMPUTER ENVIRONMENT

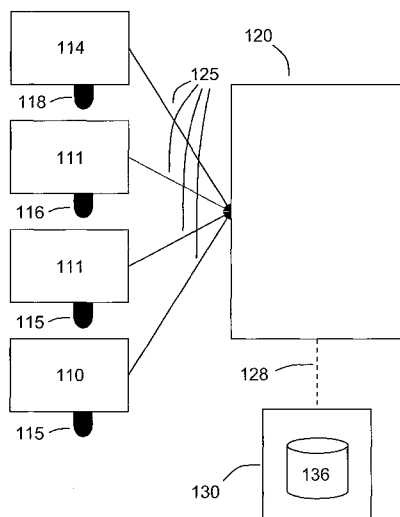


FIG. 1

(57) Abstract: The present invention relates to an authorized operating system environment and ensuring that a computer is running the authorized operating system environment. The invention provides a method that improves the security of virtual machines. A computer operating in the authorized operating system environment is adapted to send notification signals to a verification server. The signal comprises information identifying the computer. In case the verification server fails to receive a valid notification signal from the computer within a specified time, the verification server will provide a fault indication. The invention also provides an associated operating system environment and other aspects.



MC, MK, MT, NL, NO, PL, PT, RO, SE, SI, SK, SM, **Published:**

TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW,

ML, MR, NE, SN, TD, TG).

— *without international search report and to be republished
upon receipt of that report (Rule 48.2(g))*

Controlled computer environment

The present invention relates to an authorized operating system environment and ensuring that a computer is running in the authorized operating system

5 environment.

Background of the invention

It is desirable in many cases to provide a controlled computer environment that allows a system administrator to control which actions are allowed and which hardware can be used on a given computer. For instance, a corporation may need
10 in-house consulting from a software programmer to gain support on how to use an advanced software program. If the corporation is extremely sensitive about its information technology (IT) systems, including the data stored in them, it may require that the consultant uses a computer provided by the corporation's own IT personnel, rather than a computer, say a laptop, that the consultant brings along
15 with him. The IT personnel can set up the computer with only the programs and enabled hardware that the consultant needs in order to provide his service. The computer may therefore have the advanced software program installed on it, as well as other programs that might be needed, perhaps a word processing program for taking notes or documenting a process.

20

A general purpose computer can be tightly controlled using a virtual environment such as for instance VMWare ACE. An entire operating system can be generated and loaded onto the computer along with the specialized software and the word processing program. Different hardware devices on the computer can be
25 activated, or not, according to the needs. The computer may for instance have an Infrared Port device, but if the functionality is not needed, there is no reason to provide it. Accordingly, the virtual machine is specified to not load the Infrared Port device. Other communication devices such as a Bluetooth radio device may also be present in the computer hardware but not be necessary. Accordingly, the
30 Bluetooth radio is also not activated.

These environments are helpful tools for a system administrator to control what a non-privileged individual can do on the system administrator's network. They are

also advantageously used simply to provide standardized, controlled computer environments that are simple to maintain widely in a corporation and easy to control, for instance with the aim of conforming them to company-wide IT policies.

5

These environments alleviate some of the concerns that system administrators today may have with respect to controlling computer environments.

Summary of the invention

It may be that the system administrator can provide a controlled computer
10 environment, since he can control which features the operating system has, including access to software and hardware.

However, the inventors of the present invention have realized that such systems nevertheless lack an important feature.

15

The system administrator can provide a standardized and controlled environment in accordance with his or hers specifications. The thinking and practice of a skilled system administrator is that he will build an environment, test it, and then release it once it has displayed the degree of robustness that the system administrator
20 demands. It is therefore obvious to trust such a system.

However, just as the system administrator can provide a standardized and controlled environment in accordance with his set of specifications, so can a determined individual with ill intentions. He can provide an alternative
25 environment that fulfils entirely different specifications. Such an environment can in fact be stored on a USB (Universal Serial Bus) stick, and by booting the general purpose computer with the operating system environment stored on the USB stick, the configuration of the computer can be changed drastically, and the controlled environment is no longer intact. A person with bad intentions and skills
30 in exposing computer content (for instance using computer worms, viruses, software backdoors and the like) may be able to quickly infect and compromise a computer and/or computer network.

The inventors have realized a way of avoiding this problem.

A first aspect of the invention provides a method for determining whether a first computer is running an authorized operating system environment. The method comprises:

5

- the first computer sending a notification signal to a verification server in accordance with a notification protocol;
- the verification server providing a fault indication in case a valid notification signal is not received at the verification server.

10

The notification signal is sent via an operable data connection between the first computer and the verification server. This could be a wired ethernet connection; a wireless network connection; a High Speed Packet Access (HSPA) protocol; a Bluetooth connection; or an infrared connection. Examples of standards

15 specifications include, but are not necessarily limited to, the IEEE 802.11 specification set; the IEEE 802.3 specification set, and the IEEE 802.15 specifications.

In the context of the present invention, the word protocol is used in a broad sense
20 to denote any given set of rules which specify a means of communication between two or more machines or computers across one or more networks. A protocol may be part of a protocol stack, as exemplified, but not limited, by the OSI reference model. A protocol can specify one, more or all layers in the protocol stack it is part of.

25

In some cases, however, it is desirable in accordance with the present invention to limit a user's access to some of these data connection types. As described previously, a system administrator might be interested in preventing certain types of communication, such as via Bluetooth.

30

The authorized operating system environment can be an operating system, such as a Microsoft Windows operating system, a Linux operating system, a UNIX operating system, or yet another operating system. Alternatively, and advantageously, the authorized operating system environment is a virtual
35 machine having an operating system, such as one of those just mentioned,

installed. This is also referred to as "a virtual machine running an operating system", even if the virtual machine and the operating system are not switched on.

5 A virtual machine (VM) may, in the context of the present invention, be defined as any software implementation of a machine or computer that is capable of executing – known as hosting - other software programs as if it were a physical machine. A virtual machine may emulate a concrete type of physical machine or it may define and implement its own model of a machine. A virtual machine may
10 allow the hosted software or parts of the hosted software to run directly on the underlying physical hardware or it may emulate the hardware completely. A virtual machine can execute directly on hardware, be hosted in another virtual machine or run on top an operating system. Virtual machines are also known as e.g. hypervisors, system virtual machines and virtualization platforms.

15

It should be noted that the authorized operating system environment may, in the context of the present invention, be a plurality of operating systems and/or virtual machines, a so-called "stack" of operating systems. Examples include:

- 1) A general purpose operating system hosting a virtual machine which in turn
20 hosts a second operating system.
- 2) A dedicated operating system hosting a virtual machine which in turn hosts a second operating system.
- 3) A virtual machine running directly on hardware hosting an operating system. Furthermore the above examples can be combined in various ways by
25 stacking them on top of each other.

Additionally, the operating system (OS), or one of them, may particularly be a dedicated operating system, tailored to the specific task of implementing the present invention, as opposed to a general-purpose operating system. In a special
30 example, two operating systems may be running, a dedicated operating system according to the present invention, and a general purpose operating system providing services to a user of the first computer.

Alternatively, the authorized operating system, or part of the authorized operating
35 system, or part of the "stack" of operating systems mentioned above can be

downloaded from a server during the initial run of the authorized operating system environment.

Thus, one embodiment of the present invention may be implemented in so-called terminal-server solution, where the fundamental operating system in the stack of operating systems may be loaded from a portable storage medium, e.g. memory stick like a USB stick. Instead of running the virtual machine the next operating system in the stack may be loaded.

10 In another contemplated embodiment, the authorized operating system environment may be implemented on the first computer by executing an application on the first computer's pre-installed operating system, the said application initializing and configuring the authorized operating system environment on the first computer. Thereby, the authorized operating system
15 environment, including any notification signals, may be implemented on the first computer.

That the operating system is "authorized" could mean that the operating system has been approved by a system administrator. However, the term "authorized" in
20 relation to "authorized operating system environment" shall not be construed as limiting the scope. The term is used as a reference to the particular operating system environment that the first computer is supposed to be running.

To assist in describing motivations of a system administrator, the term "non-trusted user" will be used to describe a person to whom the system administrator has granted access to a computer that the system administrator administers. At the same time, a "non-trusted user" is a person that the system administrator is concerned may, whether deliberately or not, compromise the system administrator's computer network.

30

The first computer is running an authorized operating system. The operating system is capable of causing the first computer to send the notification signal to the verification server. Either, this function is built into the operating system as such. A simple solution is to implement the function via a software program

installed in the operating system. This software program will be referred to as a "security program".

The method according to the first aspect allows a system administrator to ensure
5 that the operating system running the security program is operable. In case the operating system is shut down, and potentially replaced by another operating system, the verification server will no longer receive a notification signal.

The notification protocol is a protocol according to which notification signals are
10 sent, or attempted sent, by the first computer to the verification server. They may be specifiable, or they may be fixed and not subject to alteration.

The notification protocol may ensure that two notification signals are sent within an acceptable amount of time. "Acceptable amount of time" is subjective, and it is
15 clear to a person skilled in the art that this amount of time can be implemented in an infinite number of ways.

Examples include: A "beacon" protocol by which a signal is sent (or attempted sent) to the server each 10 seconds or 1 minute or other amount of time; a
20 protocol in which there is a degree of variation, for instance via a random (or pseudorandom) number generator generating a number representing the amount of time that is to pass before a next notification signal is attempted to be sent. Examples for the protocol may include various kinds of periodic patterns for the notification signals, the periodic pattern comprising one of more frequencies of
25 reoccurrence for the said signal.

Alternatively, a notification signal may be attempted to be sent upon a specific event taking place within the operating system, such as accessing a shutdown command or attempting to investigate system files or log files on the computer. In
30 case such actions are considered unnecessary, then it might be appropriate for the first computer to send a notification signal to the verification server. This can be considered an early warning that the non-trusted user are about to perform unwanted actions.

In cases where the system administrator wishes to receive an "early warning" as exemplified above, the notification signal might comprise information indicative of an action having been performed on the computer. The information indicative of the action ("action information") can comprise the name of the command, and
5 might also include a file path for the file.

In one embodiment, the action information may comprise a command capable of disabling a functionality associated with the first computer. Particularly, the first computer may, in response to receiving the command capable of disabling
10 functionality, perform one, or more, of:

- preventing one or more commands from being executed correctly in the authorized operating system environment; and
- closing a program which is running on the authorized operating system environment.

15

Alternatively or additionally, the said action information may comprise a command capable of enabling a functionality associated with the first computer. Particularly, the first computer may, in response to receiving the command capable of enabling a functionality, perform one, or more, of:

- 20 • opening a window on a graphical user interface connected to the first computer, the window comprising an information indicating that an unwanted action has been performed on the first computer;
- initiating recording of a logging file, e.g. a key grabber or similar, related to actions performed on the first computer, the logging file, or part thereof,
25 being accessible from the verification server, and/or being transferred to the verification server, and
- executing a program, preferably a dedicated analyzing program, related to the event that an unwanted action has been performed on the first computer.

30

The notification signal might alternatively, or additionally, comprise information indicating a degree of severeness (degree of non-acceptability) of the action or indicating a certain type of action. For instance, certain system commands might, when executed, be classified according to a translation table or other type of
35 classification. The viewing of a system event log, for instance, can reveal

operating system weaknesses, user names, and other information. Such information is typically irrelevant to a user if the operating system is operating correctly in the respects in which the user actually requires it to operate correctly. However, a non-trusted user might find this information interesting and access it
5 for the purpose of compromising the system. The security software might translate the action of viewing a system event log to a "1" or "true", whereas opening a word processing program might translate into a "0" or "false", where a "1" (or "true"). A "1" (or "true") might then be indicative of an unwanted action and "0" (or "false") indicative of an acceptable action. The computer sends this
10 information ("1" or "0", for instance, as appropriate) to the verification server, which is then accordingly adapted to provide the fault signal if the notification signal comprises a "1" (or "true").

In the context of the present invention, it is to be understood that the meaning of
15 the concept of "not receiving a valid notification signal" may be interpreted in, at least, two non-limiting ways, i.e.:

- 1) **not** receiving the notification signal, and
- 2) receiving an **invalid** notification signal.

20 It should be mentioned that the present invention may be implemented in various protocols with respect to the status of the first computer and the verification server. In some embodiments, the server is passive and the first computer is active, i.e. the first computer is initiating the communication, and the server is responding. In other embodiments, both entities may initiate the communication
25 according to the specific protocol. In alternative embodiments, the server is active and the first computer is passive, i.e. the verification server is initiating the communication, and the first computer is responding. The latter embodiments are particularly advantageous in that this enables mass broadcasting from the verification server to a plurality of computers that are monitored or under
30 surveillance according to the teaching of the present invention.

In a convenient embodiment, the first computer receives at least part of the implementation of the notification protocol from the verification server, when connected. This has the advantage that the operating system does not require
35 pre-configuration with respect to the behaviour relating to notifications, and

accordingly the protocol can be changed with no, or few, preceding steps or preparations. This information, or at least part of it, can be transferred, e.g. downloaded, from the verification server, which means that the notification protocol can be adapted by the system administrator according to the specific
5 needs that he or she is facing in a given situation.

The notification protocol can, in some embodiments, be dependent on a credential provided at the first computer. The notification protocol may, in some cases, be provided by the verification server.

10

In some embodiments of the invention, the security program implements a default protocol. This ensures that the security program is operable without receiving a notification protocol or parts of a notification protocol from the verification server. A default protocol can also define some boundaries. For
15 instance, they can ensure a first notification signal and an immediately subsequent notification are separated by an amount of time that is at most 1 minute, no matter what a notification protocol from the verification server provides for. Furthermore, the default protocol can define information that must be transmitted to the verification server. A person skilled in the art will easily
20 recognize that a default protocol and a protocol from the verification server can be varied in an infinite number of ways that still fall within the scope as defined by the wording in the claims and the general teaching of the invention.

In some embodiments, the notification signal can comprise very specific
25 information, such as information about central processing unit (CPU) processes or process threads, or system event log information. Such information could also be comprised in a separate signal – a signal that is not a notification signal in accordance with the notification protocol.

30 For some purposes, it can be advantageous that the notification information comprises a file. In this way, files that are being manipulated in the operating system can be backed up via the verification server (on the verification server or a separate storage facility).

35

In some embodiments, the authorized operating system environment running on the first computer may be stored on a server from which the first computer, when the server is connected to the first computer, can load and run the authorized operating system environment. This may be implemented as a so-called
5 networking booting sequence.

In some other embodiments, the authorized operating system environment may comprise a plurality of operating systems, wherein at least one of the operating systems is a non-authorized operating system. This may be implemented as so-
10 called "citrix solution". The initially non-authorized operating system may be authorized or approved by the verification server and/or a system administrator, for example the non-authorized operating system may be installed by a user of first computer, and later approved by the verification server.

15 In some embodiments, the authorized operating system environment running on the first computer may be stored on a portion of a readable, and optionally writeable, memory connected with the first computer, the said memory being a main storage memory of the first computer. This may for instance be a memory such as hard disk (HD), or alternatively other kinds of memory entities, e.g. solid
20 state disks (SSD), etc. A partition of a HD may be used, the partition having restricted access and/or limited possibility for modification by a user of the first computer. It is then to be understood that the stored, authorized operating system environment is pre-installed or installed from verification server or portable storage medium as explained above and below.

25

In some embodiments, the authorized operating system environment running on the first computer may be stored on a portable storage medium from which the first computer, when the portable storage medium is connected to the first computer, can load and run the authorized operating system environment. The
30 authorized operating system environment stored on the portable storage medium may be encrypted, the authorized operating system environment being decrypted as a part of an initial running of the authorized operating system environment.

In some embodiments of the present invention, the notification protocol may comprise sending notification signals at one or more periodic intervals. Thus, the notification protocol may comprise sending notification signals at one of more pre-defined events performed by, or at, the first computer.

5

Particularly, the said pre-defined events may comprise one, or more, of:

- attempting to access, alter and/or close the authorized operating system environment,
- attempting to access and/or alter a system file or a logging file on the first
10 computer,
- Attempting to access, alter and/or close any other part of the operating system environment other than a single, authorized operating system.
- Attempting to access and/or alter any system settings in the authorized operating system environment

15 Additionally or alternatively, the said pre-defined events may beneficially comprise initializing and/or finalizing of the authorized operating system environment.

In some embodiments of the present invention, the notification signal itself may comprise logging information so as to document actions performed on the first
20 computer.

In some embodiments of the present invention, the said notification protocol may be implemented by means of tcp/ip, http, ssh, SQL connection protocol, telnet, ftp, icmp, xml-over-html, RCP, or any combination thereof. An explanatory, and
25 non-exhaustive, list of some selected protocols are given below:

Low level/data link layer: ethernet (IEEE820.*), USB, GSM, Edge, IEEE1394

Medium level/network or transport layer: IPv4/6

High level/session layer: tcp, udp, ssl, telnet

30 Even higher level/presentation or application layer: ATOM, RSS, SOAP, OData, GData, http(s), ssh.

In an alternative embodiment, the notification server(s) may regularly request ATOM or RSS feeds from the first computer, and the other computers. The

computer(s) may then publish their feed at regular intervals, e.g. after 1, 10 or 100 seconds. The server(s) may monitor the activity one or more feeds.

In some embodiments of the invention, there is a handshake protocol between
5 the verification server and the first computer to reduce the risk of for instance
man-in-the-middle-type interference. The security software may generate a first
token that it sends to the verification server. The verification server checks that
the first token is valid. The verification server then provides a second token that it
returns to the first computer, which then checks that the second is valid. By
10 having the tokens be generated in accordance with a non-disclosed or a disclosed
protocol, it will be very difficult, if not impossible, for a skilled person to provide a
false notification signal to cheat the verification server into behaving as if the first
computer is still operating in an authorized manner. The non-disclosed protocol
could for instance be a simple algorithm. As another example, the security
15 program and the verification server might implement encryption certificates. In an
embodiment, the verification server provides an authentication token. The
verification server sends the authentication token to the computer, which then
signs or encrypts the authentication token using an authorized private key. The
signed or encrypted authentication token is then returned to the verification
20 server in a notification signal or a separate signal. The verification server can then
check that the authentication token reached the first computer and that the
returned notification signal (or separate signal) comes from the first computer
(that is aware of the authorized private key). This is done by checking the
signature of the returned authentication token or decrypting the returned
25 authentication token and comparing the decrypted token to the token that the
verification server sent to the first computer.

In some embodiments of the invention, the verification server will, in case it does
not receive a valid notification signal, return an interference signal to the first
30 computer. In such embodiments, the security software is adapted to perform
some interference action in response. For instance, the interference signal may
cause the security program to lock the operating system, requiring a login by an
authorized person, such as the system administrator. Alternatively, the security
program may cause the keyboard and/or mouse to stop functioning correctly, or it
35 can bring up a notification window that containing information to the effect that

an unwanted action has been performed. The notification window may include information about the unwanted action, such as the name of an offending command or offending action. A person skilled in the art will recognize that the possibilities for providing an interference action are endless. The skilled person
5 should implement that interference action or actions that suit his or her purposes.

The notification signal may further comprise logging information to further document actions performed on the first computer. Such logging information could for instance comprise information about commands executed on the first
10 computer; content of system log files on the first computer; screen shots from the first computer; information about processes running on the first computer.

A second aspect of the invention provides an operating system environment adapted to enable appropriate computer hardware, when the operating system
15 environment is executed thereon, to operate as the first computer referred to in the first aspect of the invention.

A third aspect of the invention provides a computer program product comprising the operating system environment referred to as the second aspect of the
20 invention. A fourth aspect of the invention provides data storage medium comprising such an operating system environment.

A fifth aspect of the invention provides computer hardware adapted to operate as the first computer described in relation to the first aspect of the invention.
25

A sixth aspect of the invention provides the security program referred to above, for execution on the first computer.

A seventh aspect of the invention provides computer hardware adapted to operate
30 as the verification server described in relation to the first aspect of the invention.

Other aspects of the invention provide: a data storage medium comprising the security program of the sixth aspect of the invention; a verification server computer program adapted to enable appropriate computer hardware, when
35 executing the verification server computer program, to operate as the verification

server of the seventh aspect of the invention; a data storage medium comprising the verification server computer program.

Brief description of the figures

Figure 1 illustrates a system comprising computers running an authorized
5 operating system environment in accordance with the invention, connected to a verification server in accordance with the invention.

Figure 2 illustrates schematically a computer architecture. Computer 200
comprises a Central Processing Unit (CPU) 202, memory 203, a harddisk storage
10 204, other types of media 205, network card 206, keyboard 207, audio hardware 208 and graphics hardware 209. These are all in some way connected to system bus 201. These are aspects that are well known to persons skilled in computers. Graphics 209 can be thought of either as a graphics hardware card or a graphics hardware card having a display connected to it, just as keyboard 207 is not
15 necessarily a part of the computer, but may be. Also, a CPU is often connected in a bridged fashion to a PCI system bus. This sort of detail is not crucial in the present context and will therefore not be discussed further.

Figure 3 illustrates a method in accordance with the invention by which it can be
20 determined whether a computer is running an authorized operating system.

Figure 4 illustrates a method in accordance with the invention, where the computer running the authorized operating system environment can receive and execute commands.

25 Detailed description of selected embodiments

Fig. 1 illustrates computers 110, 111, 111, 114 connected to a verification server 120 via connections 125. The computers 110, 111, 111, 114 may be different computers having different hardware and operating system. The connections 125 can be any type of operable data connection, such as ethernet connections or
30 IEEE 802.11 wireless connections. The connections 125 need not all be of same type.

The computers 110, 111, 114 run operating systems stored on storage devices 115, 116, 116 and 118. They can be different operating systems, implementing different software, hardware etc, and the storage devices can be of different types, and might be USB devices, DVD devices, network cards or the like.

5

The operating systems running on computers 110, 111, 114 from storage device 115, 116 and 118 are authorized operating systems that all have a functionality that allow them to send notification signals to the verification server 120.

Notification signals are sent by the computers to the verification server in order to
10 check whether the authorized operating systems are operable. In case computer 114, which is running an authorized operating system loaded from device 118, is switched off, the computer will stop sending notification signals to the verification server 120.

15 The verification server 120 is capable of connecting to computers 110, 111, 111, 114. When the computers 110, 111, 114 run authorized operating systems, the verification server will listen for notification signals from these computers. In case valid notification signals are no longer received from computer 114, the status of computer 114 becomes unknown to the verification server 120, as the verification
20 server is no longer – in this example – in operable connection with computer 114. From the point of view of a system administrator who has sensitive and critical information stored on his or her computers and network devices, this state may be unacceptable.

25 Fig. 2 illustrates, schematically and greatly simplified, a computer architecture that computers 110, 111, and 114 might have. The computers may be different and implement different and other architectures. The verification server 120 may have a similar or different architecture.

30 Fig. 3 illustrates a method in accordance with the present invention. The figure illustrates a login procedure for registering a computer with the verification server, as well as the protocol for determining if a computer, such as computer 114 in Fig. 1 is running an authorized operating system.

The login procedure can be used for different purposes. One purpose is to uniquely associate a user of the computer with data stored either on the verification server or on another facility, such as storage 136 on computer 130 in Fig. 1, which might be connected to the verification server as illustrated by connection 128 in Fig. 1.

Alternatively, it may be a way by which a system administrator can allow the computer (such as computer 114) to connect to the verification server. Credentials, secret or not, can also be a way for the system administrator to indicate to the verification server an identity of a user of the computer 114. In Fig. 3 it is used to cause the verification server to begin listening to notification signals from a computer. Step 302 illustrates the login action at the computer. The login may for instance comprise a username (USER) and a password (PASS). If the verification server accepts the credentials, in step 304, the verification server may send an indication AUTH OK, in step 306, to the computer to inform it that authenticated credentials have been used.

If the credentials are not accepted, some path of action "A" is taken, step 315 in Fig. 3.

At the same time the verification server sends AUTH OK to the computer, the verification server starts listening for notification signals from the computer. When the computer receives AUTH OK, this is a signal to the computer that the verification server is listening for notification signals, and that it should begin sending notification signals in accordance with the notification protocol that it is aware of. This is illustrated by step 308.

The computer sends a notification signal *NS* to the verification server in step 310. In Depending on the selected implementation, it may or may not be necessary to include credentials in order for the verification server to be able to identify the computer 114 from which the notification signal is sent.

Next, in step 312, the verification server determines whether a valid notification signal has been received. If the computer 114 is switched off, the verification server will not receive a notification signal from computer 114 at all. The

verification server has therefore not received a valid notification signal. The verification server determines whether a notification signal is valid by comparing with the notification protocol that the computer implements. In this way, the verification server can determine if the computer 114 has fulfilled the obligation
5 set forth by the protocol. For instance, if the protocol prescribes that the computer sends a notification signal each 30 seconds, then the verification server can determine that if one notification signal from the computer is not followed by another notification signal within 30 seconds (plus a margin for network transit time, say 2 seconds, for example), then the verification server will give off a fault
10 indication in step 314. Using this signal as a trigger, a message, e.g. an e-mail an SMS or MMS or similar messages, could then be sent to a device which is accessible by a system administrator as indication to him. or her, that the computer 114 is in an unknown state. The system administrator may also use a centralised system, e.g. a server or a computer in a network, for monitoring or
15 administering.

As illustrated in Fig. 3, the verification server may include a token, *T1*, in a message to the computer. *T1* might for instance be a pseudorandom number. This might be part of a handshake procedure by which the verification server and
20 computer ensure that the connection 125 (see Fig. 1) is intact and that there is no interference, such as a man-in-the-middle-attack that attempts to trick the verification server into incorrectly determining that the computer is running the authorized operating system, even if it is not. The token could be signed by the computer using a private key. The result is sent as token *T2* (see Fig. 3). The
25 verification server can then check that the signature on the token *T2* is valid and that the token *T2* corresponds to the token *T1*.

It is further illustrated in Fig. 3 that the verification server can respond with a server response signal, *SRS*. A token *T3* might also be included. The token *T3*
30 could be a new pseudorandom number that the computer must sign and return as part of the handshake procedure for ensuring the integrity of the connection.

The server response signal may also comprise instructions to the security program running on the computer. In case the verification server finds that notification
35 signal is has received is invalid, the verification server can return a response

containing an instruction that causes the security program on the computer to disable the keyboard or bring up a window on a display connected to the computer, informing a user that an unwanted action has been performed.

Depending on the requirements and needs, various actions can be taken. This is a
5 matter of design.

The notification signal could also comprise file data. For instance, for someone working on a project on the computer, the notification signal can carry a copy of the project file (a data file), which the verification server can then store, either in
10 a store of its own, or send to another store, such as store 136 on computer 130 via connection 128.

Fig. 3 also illustrates how an SMS (short message service) message can be sent (for instance to the system administrator), in step 324. It could also be an email,
15 as in step 326. Or it could be an alarm sounded at the computer (per instruction from the verification server), in step 322.

Fig. 4 illustrates additional steps by which the computer 114 can for instance be disabled. In Fig. 4, the computer 114 investigates, in step 402, whether the *SRS*
20 comprises a command. If not, the computer returns to following the notification protocol, in step 310. If the *SRS* comprises a command, the computer executes, in step 404, the command (or sets of commands, if desired). This can either be a benign command that merely causes an extraordinary event to take place at the computer, namely the event causes by the performing the command. In that
25 case, the computer can return to step 310 and continue following the notification protocol.

In case the command is disabling, the computer may no longer return to step 310 and continue following the notification protocol. The computer will be in a disabled
30 state 414 and can no longer be operated normally.

The invention is applicable in scenarios in which a system administrator wants to increased awareness about a computer's operational state.

A situation in which the invention is particularly suitable relates to computer-implemented testing in for instance schools and universities. In some cases, students are allowed to use all available information. Even so, it can be a problem to simply let student use their computers freely. Two students at a testing location
5 may enable for instance Bluetooth radios or wireless network connections and exchange data that way. Unless this is allowed, the possibility of doing so should be eliminated. By using a standardized operating system, all students use the same operating system, and hardware on the computers can be controlled tightly. Bluetooth radios can be disabled (or rather: not enabled), thereby eliminating the
10 possibility of students exchanging data over a Bluetooth connection. Network interfaces can be controlled to prevent computer-to-computer network communication. In some types of tests, students may not use any information other than that which they remember. Thus, hard disks should be disabled in this case.

15 Thus, in particular the present invention may be applied for not enabling certain functionalities on a computer, e.g. wireless connections as above, but the present invention may alternatively, or additionally, provide a logging of certain events or actions that may be possible to perform as such on the computer but the said
20 events or actions may nevertheless be forbidden (or not allowed) to be performed in a certain situations, e.g. a testing or exam situation, or other events where restrictions are desirable.

In a specific application, the present invention may be utilised to limit, restrict
25 and/or control the access to certain websites, both in an exam/testing situation and in more general teaching situation, the use of these web based services may be considered counter-productive to the overall testing or teaching purpose, e.g. cheating. Thus, for example in a translation exam or teaching situation, the present invention may be effectively close access to a translation web-service,
30 e.g. Google™ translate, or other similar services, but allow access to a web-based dictionary.

The present invention allows this kind of control and at the same time ensures that the authorized operating system is not disabled whereby other information
35 can potentially be accessed.

At the start of a test, student receives a USB stick with an authorized operating system. By logging in (using for instance a student ID and password), the system can determine that a given student is using a computer that is running the
5 authorized operating system. If notification signals are no longer received from this computer, the student could be attempting to circumvent some of the limitations imposed by the authorized operating system. The verification server of the present invention will determine if this is the case, and if so, will give of a fault indication. The issue is that the student might be in the process of rebooting his or
10 her computer to be able to access information stored on their hard disk, or be able to communicate via for instance Bluetooth radios.

In this scenario, students are often working on text and/or graphics documents. It will be very advantageous if the notification signal or a separate signal comprises
15 a copy of the document or documents. By storing these on the verification server (or on the separate computer 130), students have a backup of their work. In case their computer fails, they can borrow a spare computer and retrieve a backup copy of their work from the verification server (or separate computer).

20 The verification server responds with a second signal. The second signal is also provided in a way that ensures prevention of a non-authorized provision. The second signal allows the first computer to determine that the verification server is a trusted verification server. This allows the first computer to determine whether its verification is performed correctly. In case the alleged verification server is
25 actually a non-authorized server, the first computer can determine this because the verification signal is incorrect. As a consequence, the first computer can take action to prevent correct operation. For instance, the operating system can block operation of the keyboard and permanently display a message to the effect that it cannot proceed because of an irregularity.

30

This embodiment is further advantageous because communication of information to the server, such as transmission of a document, is not performed unless the first computer has determined that the verification server is authorized.

The operating system can for convenience be placed on a USB storage medium, such as a USB flash drive. The first computer can boot the operating system from the USB drive. This allows a system administrator to provide a very well-controlled computing environment on a computer. The communication with the verification server is a safe indication that the computer is in fact running the well-controlled environment.

The operating system is advantageously provided in such a way that it provides only the required functionality. Communication protocols such as Bluetooth, that is available of most present-day laptops, for instance, might be used to transfer data from the computer to a mobile storage medium. The system administrator can prevent this from happening by configuring the operating system to not recognize Bluetooth devices. Only network functionality needs to be activated so that the computer can communicate with the verification server. In case the operating system is switched off, the verification server will provide a fault indication to notify for instance the system administrator that the authorized operating system is not active. During this fault scenario, a person might be operating the computer outside the authorized boundaries. For instance, the person might bring a non-controlled operating system that allows the Bluetooth radio module to be operated, whereby information can be transmitted from the computer to a receiving Bluetooth device.

The invention can be implemented by means of hardware, software, firmware or any combination of these. The invention or some of the features thereof can also be implemented as software running on one or more data processors and/or digital signal processors.

The individual elements of an embodiment of the invention may be physically, functionally and logically implemented in any suitable way such as in a single unit, in a plurality of units or as part of separate functional units. The invention may be implemented in a single unit, or be both physically and functionally distributed between different units and processors.

Although the present invention has been described in connection with the specified embodiments, it should not be construed as being in any way limited to

the presented examples. The scope of the present invention is to be interpreted in the light of the accompanying claim set. In the context of the claims, the terms "comprising" or "comprises" do not exclude other possible elements or steps. Also, the mentioning of references such as "a" or "an" etc. should not be construed as
5 excluding a plurality. The use of reference signs in the claims with respect to elements indicated in the figures shall also not be construed as limiting the scope of the invention. Furthermore, individual features mentioned in different claims, may possibly be advantageously combined, and the mentioning of these features in different claims does not exclude that a combination of features is not possible
10 and advantageous.

Claims

1. A method for determining whether a first computer (110, 111, 114) is running an authorized operating system environment, comprising:
 - 5 • the first computer sending (310) a notification signal to a verification server (120) in accordance with a notification protocol; and
 - the verification server providing (314) a fault indication in case a valid notification signal is not received (312) at the verification server.
- 10 2. A method in accordance with claim 1, wherein the authorized operating system environment is a virtual machine running an operating system.
3. A method in accordance with claim 1, wherein a part of the authorized operating system environment is downloaded from a server during the initial run
15 of the operating system environment.
4. A method in accordance with claim 1, 2 or 3, wherein at least a part of the implementation of the notification protocol is transferred from the verification server to the first computer.
- 20 5. A method in accordance with claim 1, wherein the notification protocol is dependent on a credential provided at the first computer.
6. A method in accordance with any of the preceding claims, comprising:
 - 25 • providing (302) a credential information at the first computer and sending (302) the credential information to the verification server;
 - the verification server providing (304) an indication of whether the credential information is a valid credential information;
 - the first computer sending (310) a notification signal to a verification
30 server (120) in accordance with a notification protocol; and
 - the verification server providing (314) a fault indication in case a valid notification signal is not received (312) at the verification server.

7. A method in accordance with any of the preceding claims, wherein the verification server sends a server response signal (SRS) to the first computer in case the verification server has not received a valid notification signal.

5 8. A method in accordance with claim 7, wherein

- the server response signal (SRS) comprises an action information comprising a command; and
- the first computer performs (404), in response to receiving the server response signal, the command comprised in the action information.

10

9. A method in accordance with claim 8, wherein the action information comprises a command capable of disabling a functionality associated with the first computer.

10. A method in accordance with claim 9, wherein the first computer, in response
15 to receiving the command capable of disabling a functionality, performs (404) one, or more, of:

- preventing one or more commands from being executed correctly in the authorized operating system environment; and
- closing a program which is running on the authorized operating system
20 environment.

11. A method in accordance with claim 8, wherein the action information comprises a command capable of enabling a functionality associated with the first computer.

25

12. A method in accordance with claim 11, wherein the first computer, in response to receiving the command capable of enabling a functionality, performs (404) one, or more, of:

- opening a window on a graphical user interface connected to the first
30 computer, the window comprising an information indicating that an unwanted action has been performed on the first computer;
- initiating recording of a logging file related to actions performed on the first computer, the logging file, or part thereof, being accessible from the verification server, and/or being transferred to the verification server, and

- executing a program, preferably a dedicated analyzing program, related to the event that an unwanted action has been performed on the first computer.

- 5 13. A method in accordance with any of the preceding claims, wherein the authorized operating system environment running on the first computer is stored on a server from which the first computer, when the server is connected to the first computer, can load and run the authorized operating system environment.
- 10 14. A method in accordance with any of the preceding claims, wherein the authorized operating system environment comprises a plurality of operating systems, wherein at least one of the operating systems is a non-authorised operating system.
- 15 15. A method to accordance with any of the preceding claims, wherein the authorized operating system environment running on the first computer is stored on a portion of a readable, and optionally writeable, memory connected with the first computer, the said memory being a main storage memory of the first computer.
- 20 16. A method in accordance with any of the preceding claims, wherein the authorized operating system environment running on the first computer is stored on a portable storage medium (115, 116, 118) from which the first computer, when the portable storage medium is connected to the first computer, can load
- 25 and run the authorized operating system environment.
17. A method according to claim 16, wherein the authorized operating system environment stored on the portable storage medium (115, 116, 118) is encrypted, the authorized operating system environment being decrypted as a part of an
- 30 initial running of the authorized operating system environment.
18. A method according to claim 1, wherein the notification protocol comprises sending notification signals at one or more periodic intervals.

19. A method according to claim 1, wherein the notification protocol comprises sending notification signals at one of more pre-defined events performed by, or at, the first computer.

5 20. A method according to claim 19, wherein the said pre-defined events comprises one, or more, of:

- attempting to access, alter and/or close the authorized operating system environment,
- attempting to access and/or alter a system file or a logging file on the first
10 computer,
- attempting to access, alter and/or close any other part of the operating system environment other than a single, authorized operating system, and
- attempting to access and/or alter any system settings in the authorized operating system environment.

15

21. A method according to claim 19, wherein the said pre-defined events comprises initializing and/or finalizing of the authorized operating system environment.

20 22. A method according to claim 1, wherein the notification signal comprises logging information so as to document actions performed on the first computer.

23. A method according to claim 1, wherein said notification protocol is implemented by means of tcp/ip, http, ssh, SQL connection protocol, telnet, ftp,
25 icmp, xml-over-html, RCP, or any combination thereof.

24. A method in accordance with any of the preceding claims, wherein the providing of a fault indication comprises sending (324) a message to a dedicated device, the device being suitable for surveillance.

30

25. A method in accordance with any of the preceding claims, wherein the first computer is in use for testing a student.

26. A method in accordance with any of the preceding claims, wherein the first
35 computer sends a file to the verification server, or another server.

27. An operating system environment adapted to enable appropriate computer hardware, when the operating system environment is executed thereon, to operate as the first computer of one, or more, of claims 1-26.

5

28. A computer program product comprising the operating system environment of claim 27.

29. A data storage medium comprising the operating system environment of claim
10 27.

30. Computer hardware adapted to operate as the first computer of one of claims 1-26.

15 31. A security computer program which, when executed on appropriate computer hardware, enables the computer hardware to operate as the first computer of one of claims 1-26.

32. A security computer program product comprising the security computer
20 program of claim 31.

33. A data storage medium comprising the security computer program of claim 31.

25 34. Computer hardware adapted to operate as the verification server in one of claims 1-26.

1/4

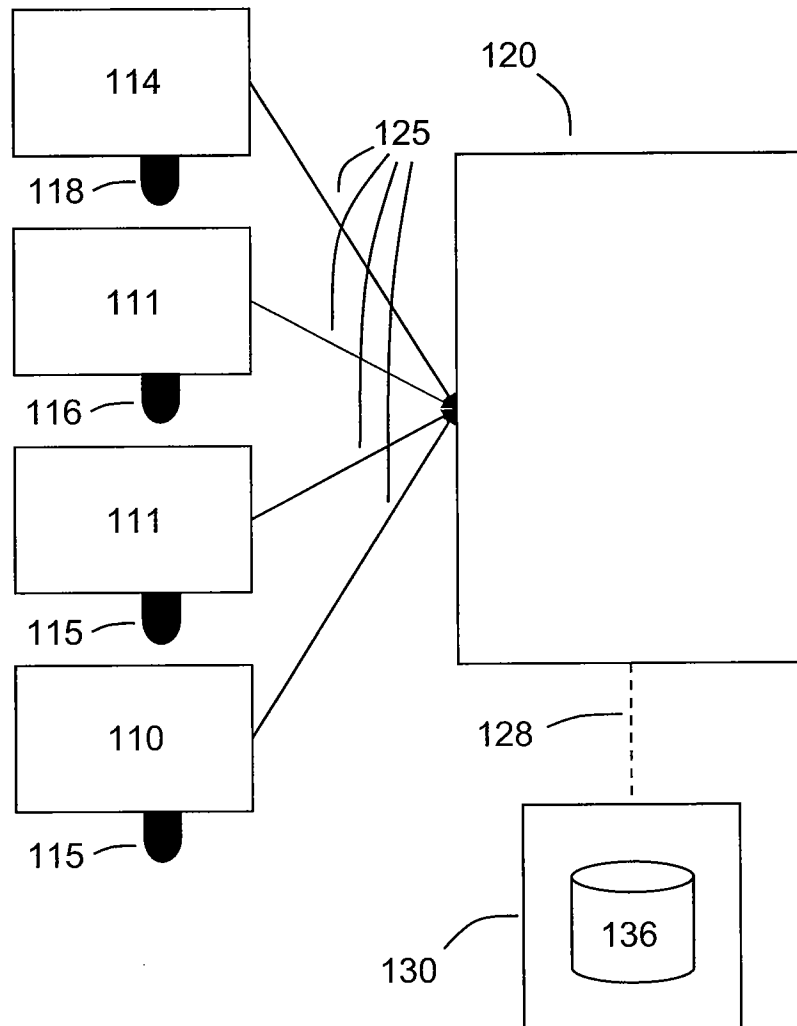


FIG. 1

2/4

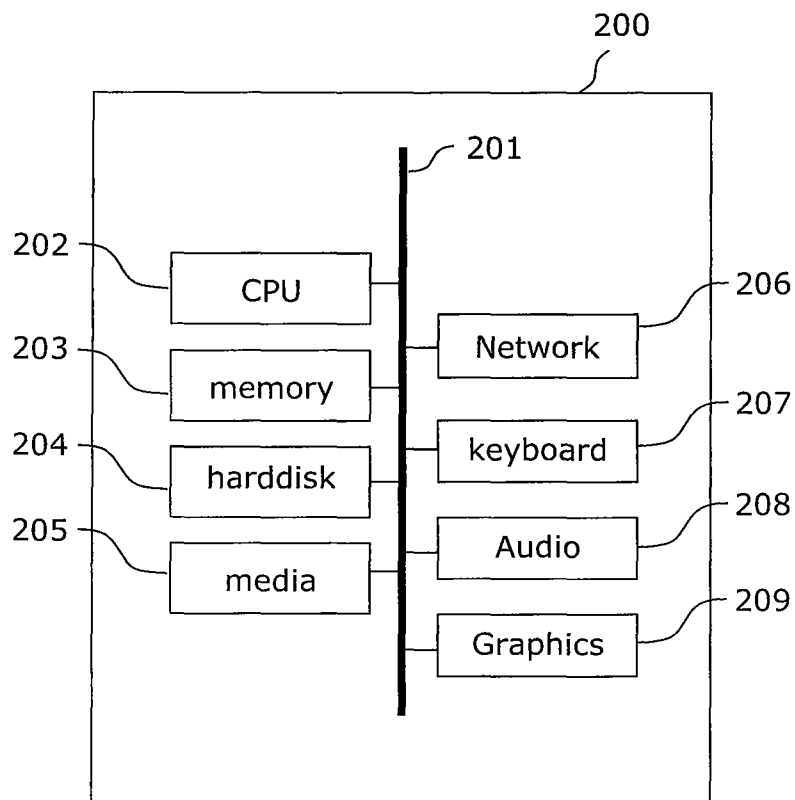


FIG. 2

3/4

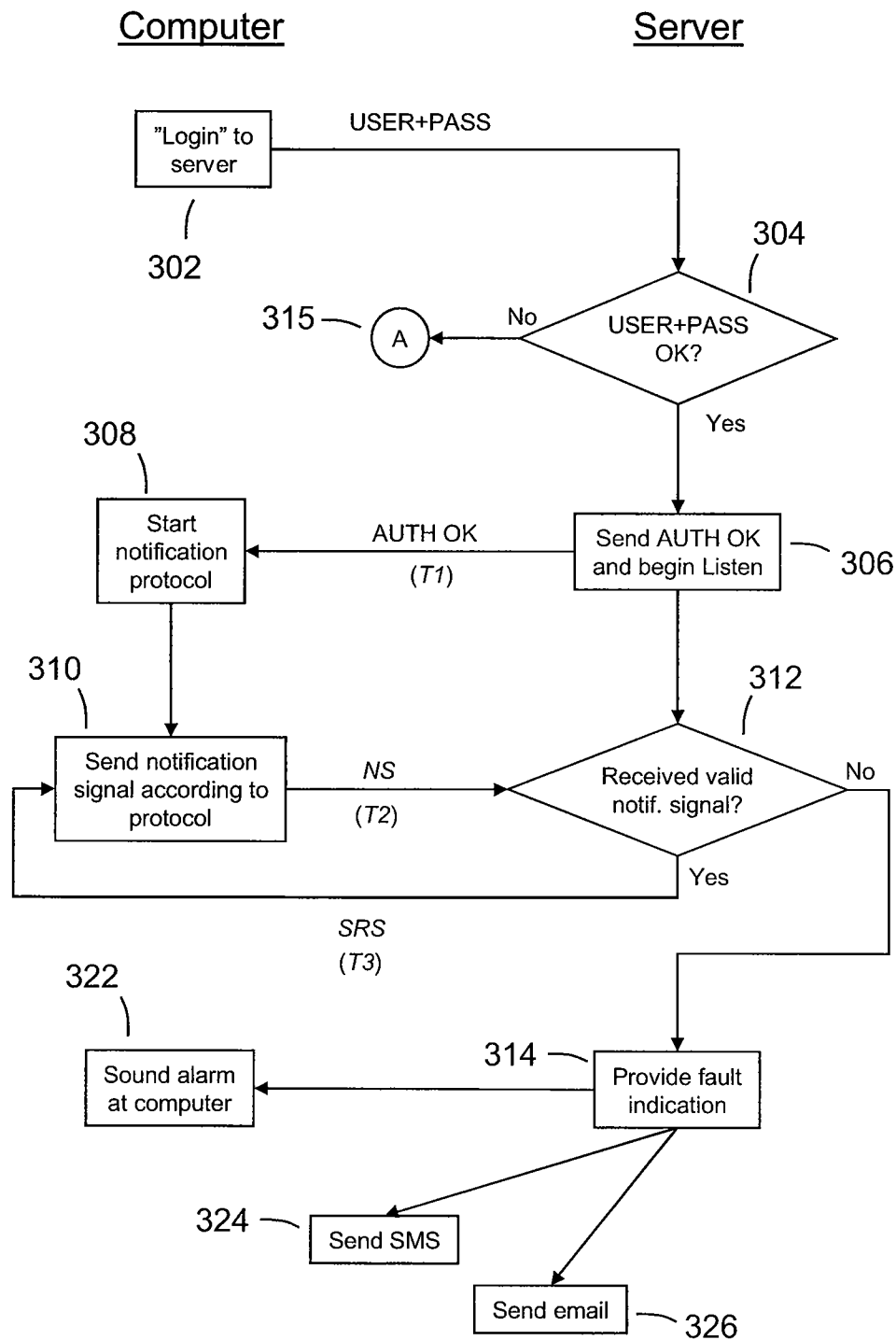


FIG. 3

4/4

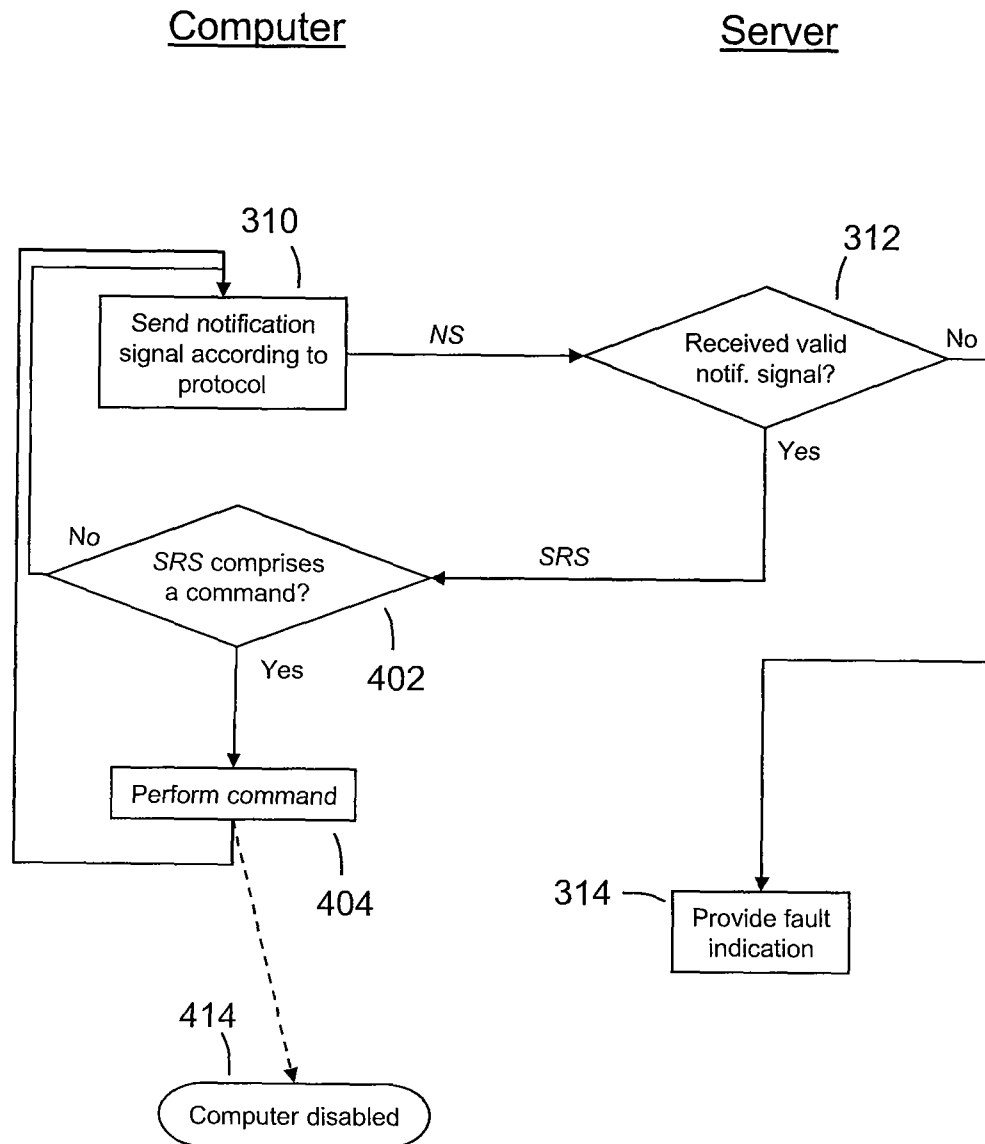


FIG. 4