without departing from the scope of the invention, and that no particular order is implied by the arrangement of blocks shown and described herein.

The following description of Figures 3A-B is intended to provide an overview of computer hardware and other operating components suitable for performing the methods of the invention described above, but is not intended to limit the applicable environments. One of skill in the art will immediately appreciate that the invention can be practiced with other computer system configurations, including hand-held devices, multiprocessor systems, microprocessor-based or programmable consumer electronics, network PCs, minicomputers, mainframe computers, and the like. The invention can also be practiced in distributed computing environments where tasks are performed by remote processing devices that are linked through a communications network.

Figure 3A shows several computer systems that are coupled together through a network 3, such as the Internet. The term "Internet" as used herein refers to a network of networks which uses certain protocols, such as the TCP/IP protocol, and possibly other protocols such as the hypertext transfer protocol (HTTP) for hypertext markup language (HTML) documents that make up the World Wide Web (web). The physical connections of the Internet and the protocols and communication procedures of the Internet are well known to those of skill in the art. Access to the Internet 3 is typically provided by Internet service providers (ISP), such as the ISPs 5 and 7. Users on client systems, such as client computer systems 21, 25, 35, and 37 obtain access to the Internet through the Internet service providers, such as ISPs 5 and 7. Access to the Internet allows users of the client computer systems to exchange information, receive and send e-mails, and view documents, such as documents which have been prepared in the HTML format. These documents are often provided by web servers, such as web server 9 which is considered to be "on" the Internet. Often these web servers are provided by the ISPs, such as ISP 5, although a computer system can be set up and connected to the Internet without that system being also an ISP as is well known in the art.

The web server 9 is typically at least one computer system which operates as a server computer system and is configured to operate with the protocols of the World Wide Web and is coupled to the Internet. Optionally, the web server 9 can be part of an ISP which provides access to the Internet for client systems. The web server 9 is shown



ForcePoint LLC
EX-1009
Page345

coupled to the server computer system 11 which itself is coupled to web content 10, which can be considered a form of a media database. It will be appreciated that while two computer systems 9 and 11 are shown in Figure 3A, the web server system 9 and the server computer system 11 can be one computer system having different software components providing the web server functionality and the server functionality provided by the server computer system 11 which will be described further below.

Client computer systems 21, 25, 35, and 37 can each, with the appropriate web browsing software, view HTML pages provided by the web server 9. The ISP 5 provides Internet connectivity to the client computer system 21 through the modem interface 23 which can be considered part of the client computer system 21. The client computer system can be a personal computer system, a network computer, a Web TV system, or other such computer system. Similarly, the ISP 7 provides Internet connectivity for client systems 25, 35, and 37, although as shown in Figure 3A, the connections are not the same for these three computer systems. Client computer system 25 is coupled through a modem interface 27 while client computer systems 35 and 37 are part of a LAN. While Figure 3A shows the interfaces 23 and 27 as generically as a "modem," it will be appreciated that each of these interfaces can be an analog modem, ISDN modem, cable modem, satellite transmission interface (e.g. "Direct PC"), or other interfaces for coupling a computer system to other computer systems. Client computer systems 35 and 37 are coupled to a LAN 33 through network interfaces 39 and 41, which can be Ethernet network or other network interfaces. The LAN 33 is also coupled to a gateway computer system 31 which can provide firewall and other Internet related services for the local area network. This gateway computer system 31 is coupled to the ISP 7 to provide Internet connectivity to the client computer systems 35 and 37. The gateway computer system 31 can be a conventional server computer system. Also, the web server system 9 can be a conventional server computer system.

Alternatively, as well-known, a server computer system 43 can be directly coupled to the LAN 33 through a network interface 45 to provide files 47 and other services to the clients 35, 37, without the need to connect to the Internet through the gateway system 31.



ForcePoint LLC
EX-1009
Page346

Figure 3B shows one example of a conventional computer system that can be used as a client computer system or a server computer system or as a web server system. It will also be appreciated that such a computer system can be used to perform many of the functions of an Internet service provider, such as ISP 5. The computer system 51 interfaces to external systems through the modem or network interface 53. It will be appreciated that the modem or network interface 53 can be considered to be part of the computer system 51. This interface 53 can be an analog modem, ISDN modem, cable modem, token ring interface, satellite transmission interface (e.g. "Direct PC"), or other interfaces for coupling a computer system to other computer systems. The computer system 51 includes a processing unit 55, which can be a conventional microprocessor such as an Intel Pentium microprocessor or Motorola Power PC microprocessor. Memory 59 is coupled to the processor 55 by a bus 57. Memory 59 can be dynamic random access memory (DRAM) and can also include static RAM (SRAM). The bus 57 couples the processor 55 to the memory 59 and also to non-volatile storage 65 and to display controller 61 and to the input/output (I/O) controller 67. The display controller 61 controls in the conventional manner a display on a display device 63 which can be a cathode ray tube (CRT) or liquid crystal display. The input/output devices 69 can include a keyboard, disk drives, printers, a scanner, and other input and output devices, including a mouse or other pointing device. The display controller 61 and the I/O controller 67 can be implemented with conventional well known technology. A digital image input device 71 can be a digital camera which is coupled to the I/O controller 67 in order to allow images from the digital camera to be input into the computer system 51. The non-volatile storage 65 is often a magnetic hard disk, an optical disk, or another form of storage for large amounts of data. Some of this data is often written, by a direct memory access process, into memory 59 during execution of software in the computer system 51. One of skill in the art will immediately recognize that the term "computer-readable medium" includes any type of storage device that is accessible by the processor 55 and also encompasses a carrier wave that encodes a data signal.

It will be appreciated that the computer system 51 is one example of many possible computer systems which have different architectures. For example, personal computers based on an Intel microprocessor often have multiple buses, one of which can



ForcePoint LLC
EX-1009
Page347

be an input/output (I/O) bus for the peripherals and one that directly connects the processor 55 and the memory 59 (often referred to as a memory bus). The buses are connected together through bridge components that perform any necessary translation due to differing bus protocols.

Network computers are another type of computer system that can be used with the present invention. Network computers do not usually include a hard disk or other mass storage, and the executable programs are loaded from a network connection into the memory 59 for execution by the processor 55. A Web TV system, which is known in the art, is also considered to be a computer system according to the present invention, but it may lack some of the features shown in Figure 3B, such as certain input or output devices. A typical computer system will usually include at least a processor, memory, and a bus coupling the memory to the processor.

It will also be appreciated that the computer system 51 is controlled by operating system software which includes a file management system, such as a disk operating system, which is part of the operating system software. One example of an operating system software with its associated file management system software is the family of operating systems known as Windows® from Microsoft Corporation of Redmond, Washington, and their associated file management systems. The file management system is typically stored in the non-volatile storage 65 and causes the processor 55 to execute the various acts required by the operating system to input and output data and to store data in memory, including storing files on the non-volatile storage 65.

A security policy distribution system that is managed by a master security policy server has been described. Although specific embodiments have been illustrated and described herein, it will be appreciated by those of ordinary skill in the art that any arrangement which is calculated to achieve the same purpose may be substituted for the specific embodiments shown. This application is intended to cover any adaptations or variations of the present invention.

The terminology used in this application with respect to network communications is meant to include all communication media and environments, including local and wide area networks, public and private communications environments, and wired and wireless



ForcePoint LLC
EX-1009
Page348

communications media. Therefore, it is manifestly intended that this invention be limited only by the following claims and equivalents thereof.



ForcePoint LLC
EX-1009
Page349

## CLAIMS

What is claimed is:

1. A computerized method of distributing security policies comprising:
maintaining a security policy at a master policy server; and
periodically synchronizing the master policy server and a local policy server to replicate the security policy at the local policy server.

2. The computerized method of claim 1, wherein the synchronizing further comprises obtaining security statistics from the local policy server by the master policy server.

3. The computerized method of claim 2 further comprising:
deriving a global status from the statistics.

4. The computerized method of claim 1, further comprising:
obtaining security statistics by the master policy server upon request to the local policy server.

5. The computerized method of claim 1 further comprising:
replicating the security policy to the local policy server upon request to the master policy server.

6. The computerized method of claim 1 further comprising:
creating the security policy at the local policy server; and
transferring the security policy to the master policy server.

7. The computerized method of claim 1 further comprising:
creating the security policy at the master policy server.



ForcePoint LLC
EX-1009
Page350

8. The computerized method of claim 1, wherein the synchronizing is performed securely across a communications medium coupling the master policy server and the local policy server.

9. The computerized method of claim 1, further comprising:

managing security for a plurality of client platforms by the local policy server, the security policy comprising security parameters particular to each client platform.

10. The computerized method of claim 9, further comprising:

deriving the security policy parameters particular to each client platform from global security parameters, the security policy at the master policy comprising the global security parameters.

11. A computer-readable medium having executable instructions to cause a computer to perform a method comprising:

maintaining a security policy at a master policy server; and

periodically synchronizing the master policy server and a local policy server to replicate the security policy at the local policy server.

12. The computer-readable medium of claim 11, wherein the synchronizing further comprises obtaining security statistics from the local policy server by the master policy server.

13. The computer-readable medium of claim 12, wherein the method further comprises:

deriving a global status from the statistics.

14. The computer-readable medium of claim 11, wherein the method further comprises:

obtaining security statistics by the master policy server upon request to the local policy server.



ForcePoint LLC
EX-1009
Page351

15. The computer-readable medium of claim 11, wherein the method further comprises:

replicating the security policy to the local policy server upon request to the master policy server.

16. The computer-readable medium of claim 11, wherein the method further comprises:

creating the security policy at the local policy server; and

transferring the security policy to the master policy server.

17. The computer-readable medium of claim 11, wherein the method further comprises:

creating the security policy at the master policy server.

18. The computer-readable medium of claim 11, wherein the synchronizing is performed securely across a communications medium coupling the master policy server and the local policy server.

19. The computer-readable medium of claim 11, wherein the method further comprises:

managing security for a plurality of client platforms by the local policy server, the security policy comprising security parameters particular to each client platform.

20. The computer-readable medium of claim 19, wherein the method further comprises:

deriving the security policy parameters particular to each client platform from global security parameters, the security policy at the master policy comprising the global security parameters.

21. A computer system comprising:

a processor coupled to a memory through a bus;

a network interface coupled to the processor through the bus; and



ForcePoint LLC
EX-1009
Page352

a master server process executed from the memory by the processor to cause the processor to maintain a security policy and to periodically synchronize with a local policy server through the network interface to replicate the security policy at the local policy server.

22. The computer system of claim 21, wherein the master server process further causes the processor to obtain security statistics from the local policy server through the network interface during synchronization.

23. The computer system of claim 22, wherein the master server process further causes the processor to derive a global status from the statistics.

24. The computer system of claim 21, wherein the master server process further causes the processor to request security statistics from the local policy server through the network interface.

25. The computer system of claim 21, wherein the master server process further causes the processor to receive a request from the local policy server and to replicate the security policy to the local policy server in response.

26. The computer system of claim 21, wherein the master server process further causes the processor to create the security policy.

27. The computer system of claim 21, wherein the master server process further causes the processor to couple the network interface to a secure communications medium for synchronization.

28. A computer system comprising:

a processor coupled to a memory through a bus;

a network interface coupled to the processor through the bus; and



ForcePoint LLC
EX-1009
Page353

a local server process executed from the memory by the processor to cause the processor to periodically synchronize with a master policy server through the network interface to receive a security policy from a master policy server.

29. The computer system of claim 28, wherein the local server process further causes the processor to transfer security statistics to the master policy server through the network interface during synchronization.

30. The computer system of claim 28, wherein the local server process further causes the processor to receive a request for security statistics from the master policy server and to transfer the security statistics to the master policy server through the network interface in response.

31. The computer system of claim 28, wherein the local server process further causes the processor to request a security policy from the master policy server through the network interface and to receive the security policy in response.

32. The computer system of claim 28, wherein the local server process further causes the processor to create the security policy and to transfer the security policy to the master policy server through the network interface.

33. The computer system of claim 28, wherein the local server process further causes the processor to manage security for a plurality of client platforms, the security policy comprising security parameters particular to each client platform.

34. The computer system of claim 33, wherein the local server process further causes the processor to derive the security policy parameters particular to each client platform from global security parameters, the security policy at the master policy server comprising the global security parameters.

35. An apparatus comprising:

network means for interfacing to a network; and



ForcePoint LLC
EX-1009
Page354

master policy means for maintaining a security policy and for periodically synchronizing with a local policy means through the network interface to replicate the security policy at the local policy means.

36. The apparatus of claim 35, wherein the master policy means is further operable for obtaining security statistics from the local policy means through the network means during synchronization.

37. The apparatus of claim 36, wherein the master policy means is further operable for deriving a global status from the statistics.

38. The apparatus of claim 35, wherein the master policy means is further operable for requesting security statistics from the local policy means through the network means.

39. The apparatus of claim 35, wherein the master policy means is further operable for receiving a request from the local policy means through the network means and for replicating the security policy to the local policy means through the network means in response.

40. The apparatus of claim 35, wherein the master policy means is further operable for creating the security policy.

41. The apparatus of claim 35, wherein the network means is further operable for coupling to a secure communications medium for synchronization between the master policy means and the local policy means.

42. An apparatus comprising:

network means for interfacing to a network; and

local policy means for periodically synchronizing to a master policy means through the network means to receive a security policy form the master policy means.



ForcePoint LLC
EX-1009
Page355

43. The apparatus of claim 42, wherein the local policy means is further operable for transferring security statistics to the master policy means through the network means during synchronization.

44. The apparatus of claim 42, wherein the local policy means is further operable for receiving a request for security statistics from the master policy means through the network means and for transferring the security statistics to the master policy means through the network means in response.

45. The apparatus of claim 42, wherein the local policy means is further operable for requesting a security policy from the master policy means through the network means and for receiving the security policy from the master policy means through the network means in response.

46. The apparatus of claim 42, wherein the local policy means is further operable for creating the security policy and for transferring the security policy to the master policy means through the network means.

47. The computer system of claim 42, wherein the local security means is further operable for managing security for a plurality of client platforms, the security policy comprising security parameters particular to each client platform.

48. The computer system of claim 47, wherein the local security means is further operable for deriving the security policy parameters particular to each client platform from global security parameters, the security policy at the master policy means comprising the global security parameters.



ForcePoint LLC
EX-1009
Page356

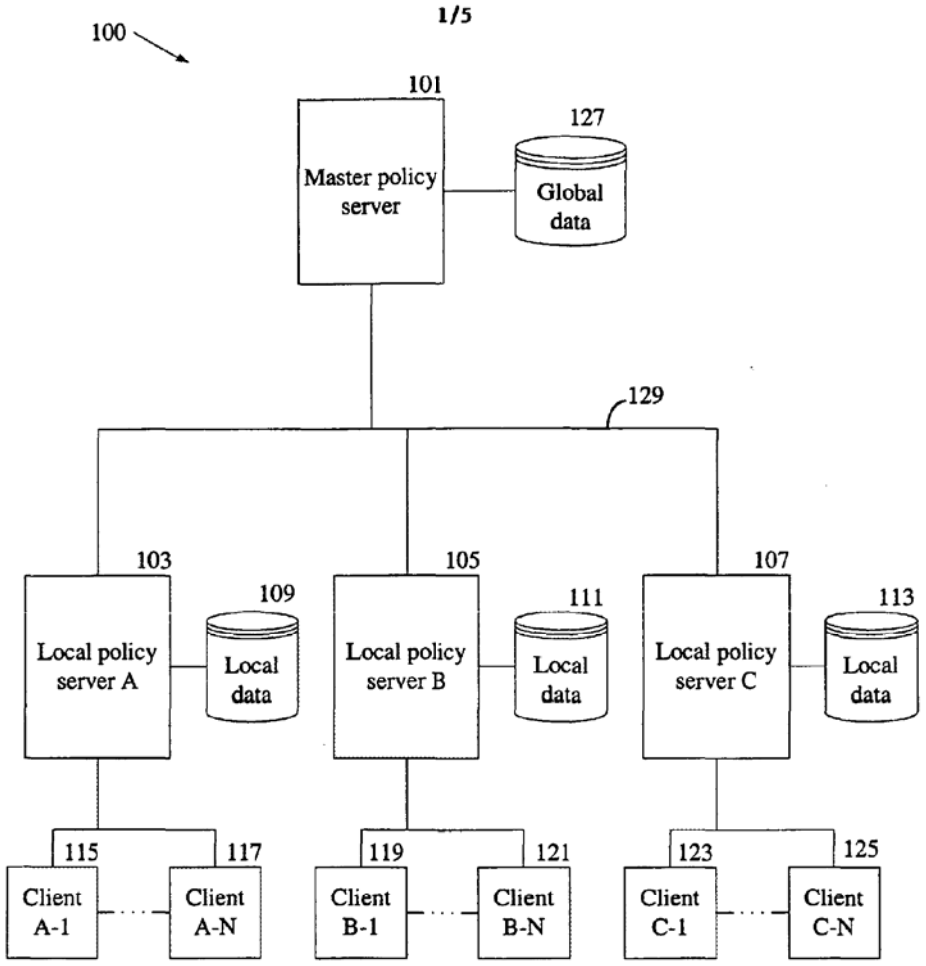

100
101
Master policy server
127
Global data
129
103
Local policy server A
109
Local data
105
Local policy server B
111
Local data
107
Local policy server C
113
Local data
115
Client A-1
117
Client A-N
119
Client B-1
121
Client B-N
123
Client C-1
125
Client C-N

Figure 1

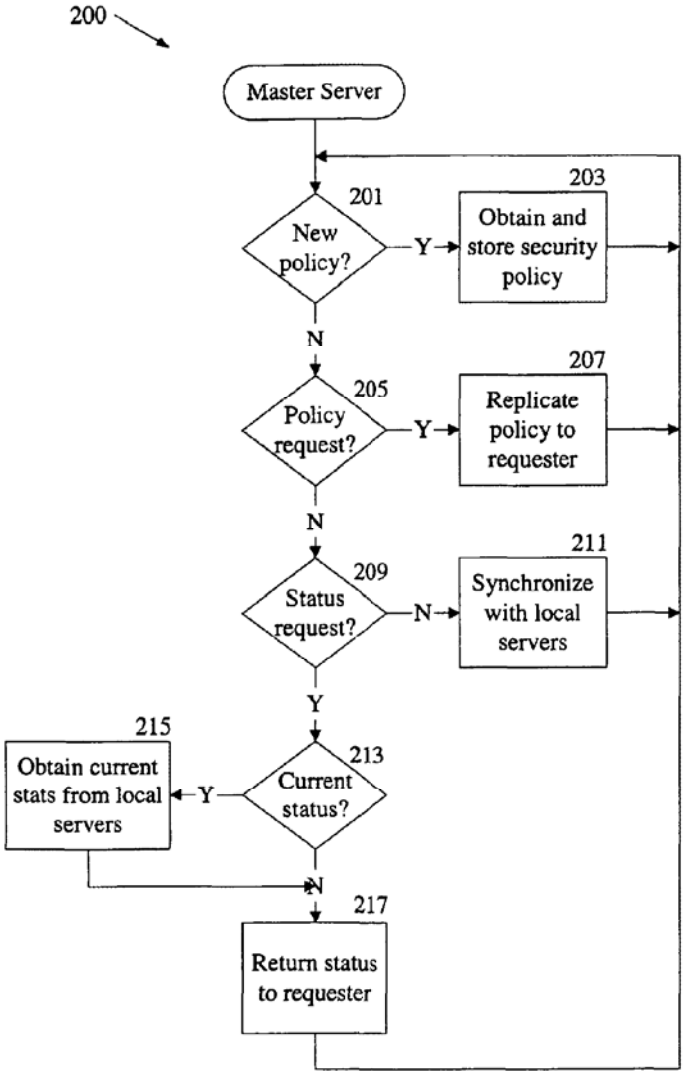

200
Master Server
201
New policy?
Y
203
Obtain and store security policy
N
205
Policy request?
Y
207
Replicate policy to requester
N
209
Status request?
N
211
Synchronize with local servers
Y
213
Current status?
Y
215
Obtain current stats from local servers
N
217
Return status to requester

Figure 2A

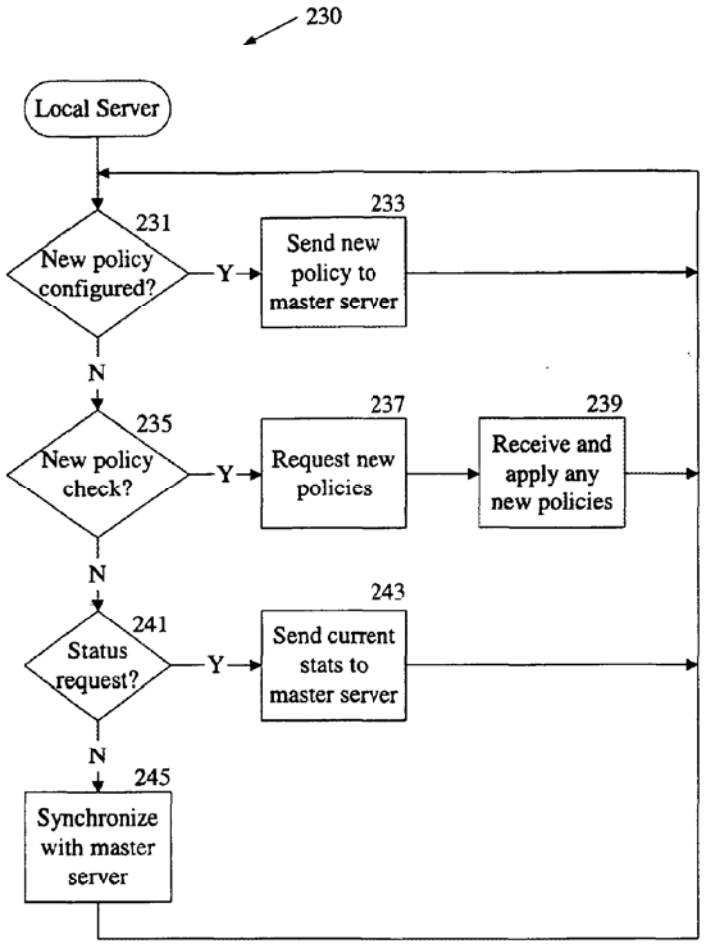

230
Local Server
231
New policy configured?
Y
233
Send new policy to master server
N
235
New policy check?
Y
237
Request new policies
239
Receive and apply any new policies
N
241
Status request?
Y
243
Send current stats to master server
N
245
Synchronize with master server

Figure 2B

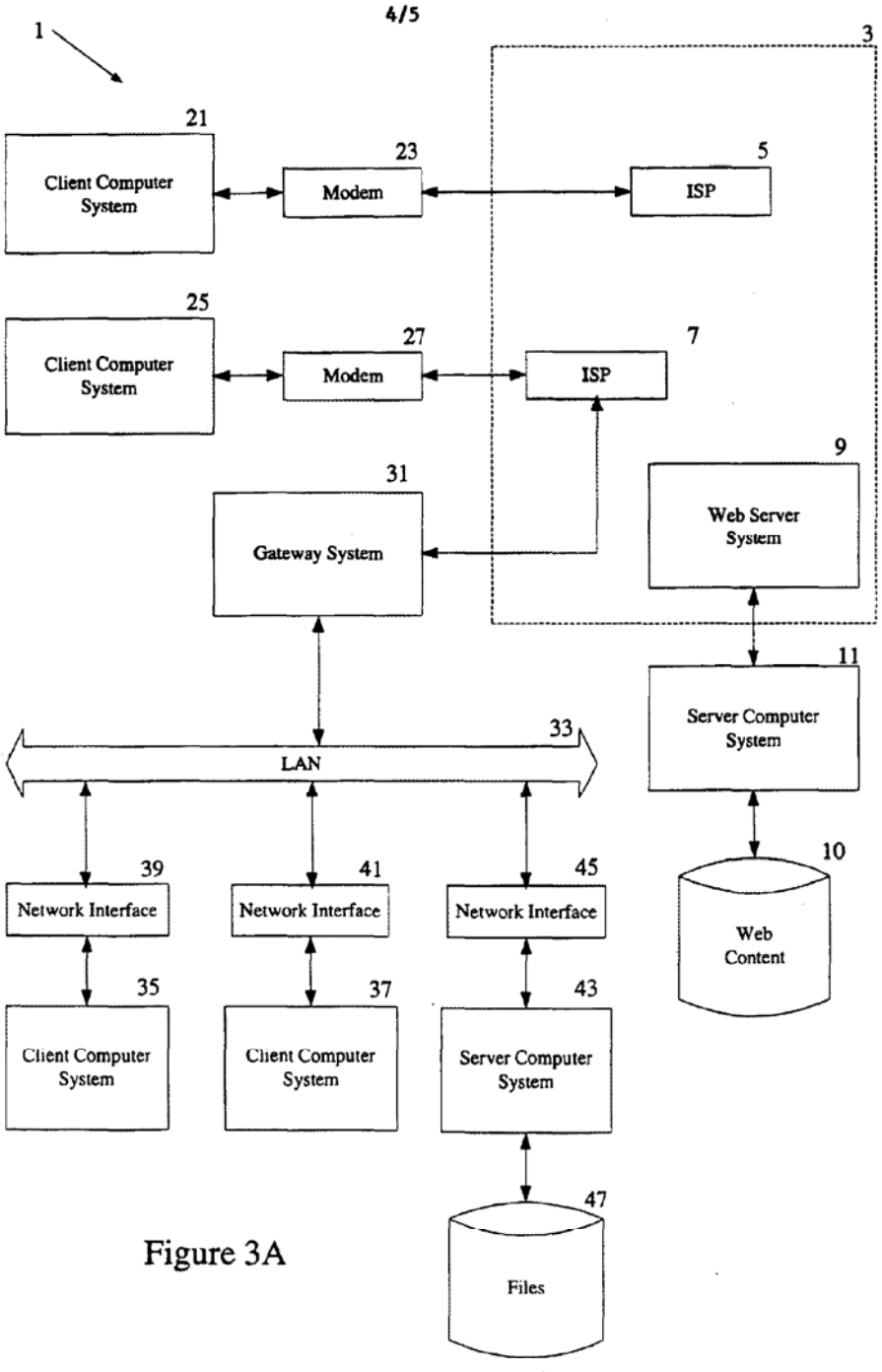



Figure 3A

ForcePoint LLC
EX-1009
Page360

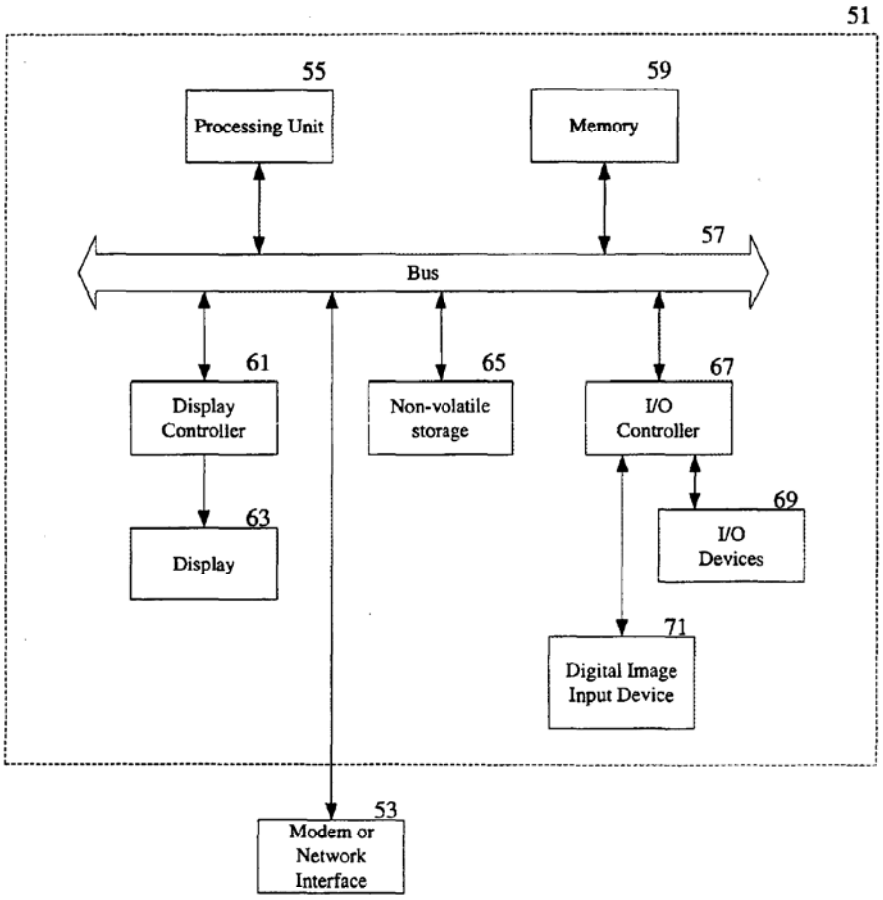



Figure 3B

ForcePoint LLC
EX-1009
Page361

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
7 October 2004 (07.10.2004)

PCT

(10) International Publication Number
WO 2004/086168 A2

(51) International Patent Classification[7]: G06F

(21) International Application Number:
PCT/US2004/006328

(22) International Filing Date: 2 March 2004 (02.03.2004)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
10/394,447 20 March 2003 (20.03.2003) US

(71) Applicant *(for all designated States except US)*: **PATCHLINK CORPORATION** [US/US]; 3370 North Hayden Road, Suite 123PMB175, Scottsdale, AZ 85251 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **MOSHIR, Sean** [US/US]; 3370 North Hayden Road, Suite 123PMB175, Scottsdale, AZ 85251 (US). **ANDREW, Christopher, A., H.** [US/US]; 3290 West Mulberry Drive, Chandler, AZ 85248 (US). **GORDON, Jonathan, M.** [US/US]; 5001 East Hilton Ave., Mesa, AZ 85206 (US). **BACON, Michael** [US/US]; 6932 West Briles Road, Peoria, AZ 85383 (US).

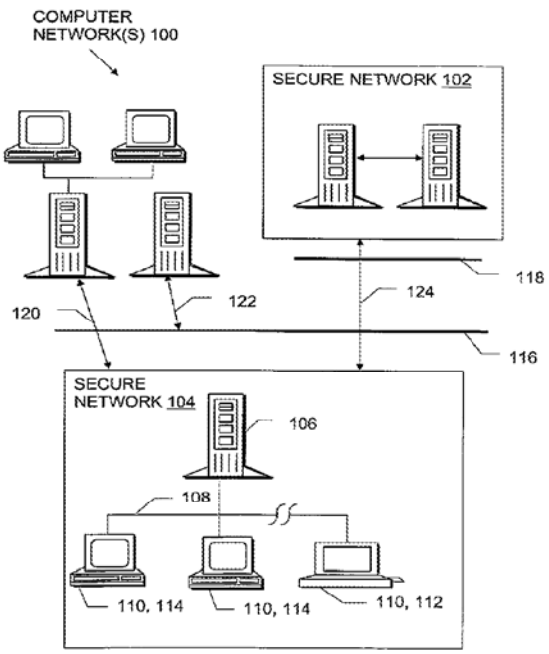
(74) Agents: **SHAHPAR, Shahpar** et al.; Snell & Wilmer LLP, One Arizona Center, 400 East Van Buren, Phoenix, AZ 85004-2202 (US).

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

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZM, ZW),

*[Continued on next page]*

(54) Title: NON-INVASIVE AUTOMATIC OFFSITE PATCH FINGERPRINTING AND UPDATING SYSTEM AND METHOD




(57) Abstract: Methods, systems, and configured storage media are provided for discovering software updates (232), discovering if a given computer (202) can use the software update, and then updating the computers with the software as needed automatically across a network (200). Furthermore, when a failure is detected (316), the rollout is stopped (318) and the software can be automatically removed (322) from those computers that already were updated. The Software update can be stored originally at an address that is inaccessible through the network firewall (214) by intermediately uploading the software update to an update computer (220) which is not a part of the network but has access to a package computer (230) through the firewall.

WO 2004/086168 A2

ForcePoint LLC
EX-1009
Page362

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, HU, IE, IT, LU, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

**Published:**

— *without international search report and to be republished upon receipt of that report*

*For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.*

# NON-INVASIVE AUTOMATIC OFFSITE PATCH FINGERPRINTING AND UPDATING SYSTEM AND METHOD

## COPYRIGHT NOTICE

A portion of the disclosure of this patent document contains material which is subject to copyright protection. The copyright owner has no objection to the facsimile reproduction by anyone of the patent document or the patent disclosure, as it appears in the Patent and Trademark Office patent file or records, but otherwise reserves all copyright rights whatsoever. The copyright owner does not hereby waive any of its rights to have this patent document maintained in secrecy, including without limitation its rights pursuant to 37 C.F.R. § 1.14.

## FIELD OF THE INVENTION

The present invention relates to systems and methods which update existing software across a remote network. The invention relates more specifically to checking for the need for updating and then updating the software as required across a client-server system without the need for human oversight, and without requiring that a target network administrative machine keep copies of software patches.

## TECHNICAL BACKGROUND OF THE INVENTION

The 'state of the art' in the computer software industry remains such that software is often delivered with various anomalies in its desired behavior. These anomalous behaviors have come to be called "bugs".

The original computer bug is in the operations log of the Harvard University Mark II Aiken Relay Calculator, now preserved in the Smithsonian. The operators removed a moth that had become trapped between relay switches in the computer, and wrote the entry "First actual case of bug being found." Problems with computer hardware and software have since been called "bugs", with the process of removing problems called "debugging".

Each time software is "debugged", a change to that piece of software is created – this change sometimes results in an additional piece of software called a "patch" or



ForcePoint LLC
EX-1009
Page364

"fix". The industry's software vendors often call these patches by the more formal names "Service Packs" or "Support Packs."

This process has become so prevalent in the industry that software vendors use various naming and numbering schemes to keep track of their available "Support Packs". The difficulty of keeping these "Support Packs" straight is increased when vendors fail to agree on a standard scheme of naming and numbering.

Microsoft, for instance, for its Windows NT family of operating system software products has no less than six major "Service Packs" available to be applied to solve problems its customers may experience. More generally, the total number of patches, fixes, solutions, and/or service packs for any given operating system may be enormous.

When an application is installed, it may contain one or more of these operating systems file patches along with the standard computer files. The patches are generally included because the application vendor discovered some anomalous behavior in one or more of the operating system files, and so sent a "fix" in the form of a different version of one of these troublesome files. This would cause relatively little difficulty if only one application vendor performed this service, or if the file modified by the application vendor is used only by that vendor's application. However, this is often not the case.

When another application is installed, that application may include a more recent version of a shared piece of code. One subset of these shared operating system files are called DLL's (dynamically linked libraries), though they also go by other names. These shared operating system files are often executable, and they expect a fixed number of parameters, certain kinds of parameters, and so on. If the nature of the shared file has changed (e.g., the parameter set is different, the name is different, the function is different), the calling application may no longer behave correctly. Many common computer functions such as "print" are referenced in this fashion.

Many software vendors try to provide the "latest" version of the operating system file. However, when a different application is loaded onto a computer, it may overwrite and subtly (or not-so-subtly) change an operating system file that the original application needed to function as planned.



ForcePoint LLC
EX-1009
Page365

Assume an administrator for an organization is charged with keeping one hundred servers up and running while supporting three thousand users connecting to these servers. The administrator is also responsible for installing user requested or management dictated applications – either shrink wrapped purchases or internally developed applications. The administrator also has responsibility for the timely distribution, locally or remotely, of time sensitive documents.

Now imagine that six service packs must be installed on the network and distributed to all of the clients. Applying the six service packs could easily result in seven visits to each and every one of the hundred servers for a total of seven hundred visits. This number assumes one extra visit per machine because the application of one of the service packs may cause more problems than it fixed, so it had to be undone.

If the three thousand clients were all running the same workstation operating system, that could mean another twenty-one thousand visits to apply the patches. Remember this all has to be accomplished while installing and patching the internally developed applications and the shrink-wrapped products. Distribution of software patches and files and their subsequent application becomes the first indication of what might be called "administrator agony".

While all the installation is occurring, the individual servers must still be monitored. When a server needs attention the administrator is often contacted by another person, who may frantically report that their server is down and must be fixed. If the administrator had some method to monitor these devices, he or she could become more responsive and further reduce the impact of problems. Monitoring needs are a second indication of "administrator agony". There is often high turnover in the administrator's job, and the users of these systems may experience lower productivity.

Traditionally, the administrator had been helped by being given extra staff. Of course this remedy is not without problems – the addition of personnel increases the number of communication channels between them. The people involved in installation and updates need a tracking device or system so they don't perform or attempt to perform the same unit of work. This lack of coordination between team members is a third indication of "administrator agony".



ForcePoint LLC
EX-1009
Page366

Proposed solutions are currently available in varying forms, implementations, and coverage or completeness. Typically these proposed solutions are available as shrink-wrap products that are installable (e.g. patchable) locally in the administrator's environment. Some emerging products are helpful, but many conventional solutions are invasive in that they require massive modification of the administrator's environment. The shrink-wrap solution requires additional invasive full product installations in the administrator's network, thereby adding to the problem, and lacks a central "command center" to coordinate the support or distribution plan. Emerging solutions may provide a somewhat lesser degree of invasion, but nonetheless require a special connection between the administrator and the solution, and they often do not provide a center for coordinated efforts.

Furthermore, it is not always obvious exactly what patches, if any, a given piece of software has received. Updates don't always clearly announce their presence. So, it is not always clear whether a specific computer has previously received a specific patch. Accordingly, there is a need for improved tools and techniques for updating computers across a network. Such tools and techniques are described and claimed herein.

## BRIEF SUMMARY OF THE INVENTION

The present invention relates to methods, articles, signals, and systems for determining if software needs updating, and if so, then updating the software across a network with reduced demands on a human administrator. If the update fails, the computer(s) upon which the update software was installed may be restored to a non-updated state. The invention is defined by the appended claims, which take precedence over this summary.

In various embodiments, the invention facilitates software deployment, software installation, software updating, and file distribution based on software and patch finger printing across multiple operating systems and devices, across a network. Any computer with a network connection and with an update agent running on it may connect to an update server, and then process whatever tasks the administrator has designated for that agent.



ForcePoint LLC
EX-1009
Page367

Figures 2 shows an overview of one such system. A network 200, shown with only two target computers and an update computer for simplicity of illustration, is protected from the internet by a firewall 214. The software that is needed to update network target computers 202, 208 resides on package computers 230, 234 that are located inside or outside the firewall and barred by the firewall 214 from direct communications with the target computers 202, 208. However, an update server 220 does have access 216 to the network 200, potentially through internal firewalls – as well as access through the firewall 214. The system is designed to work both as an onsite purchased solution as well as a fully offsite hosted solution, and can operate through firewalls and proxy circuits at any level within the Intranet/Extranet infrastructure.

Patch fingerprints 902 give a recipe to allow a repository component to determine if a given software package (associated with the patch fingerprint) , patch, driver, etc. should be loaded onto a computer in the system. These fingerprints are stored in a patch component database location 900 that may be inside or outside the firewall 214. It may be at a separate location or it may be installed on the update server 528. The repository component also includes an inventory library database 918 that contains basic hardware and software information about each of the network target computers 202, 208. Using the information in the patch fingerprint, the inventory library, and specific information gleaned from each network target computer, the system is able to intelligently recommend which patches and drivers are required for a given computer.

As shown in Figure 5, the preferred embodiment of the invention employs an additional agent known as the discovery agent 548 installed on the target computer 500, which routinely discovers the hardware and software on that machine. This inventory information is then reported back to an inventory library 918 located somewhere else in the repository component. In addition to the computer inventory, the discovery agents also return scan results for patch fingerprints, which indicate whether it is appropriate to install a specific patch associated with each patch fingerprint.

The Inventory Database thus collects a complete inventory of the software, hardware and current patch fingerprints that are installed on any particular target



ForcePoint LLC
EX-1009
Page368

computer within the network. With this information, the update server 528 can present the user with detailed reports of the current patch status for all computers within the network. This illustrates the number of computers needing the patch as well as the computers already installed with the patch.

In addition, Finger Print definitions 906 are also normally associated with an update package suitable for deployment by the system. Once the need for a particular patch has been established by scanning for its signature(s) on all or any computers within the network it can then be quickly deployed by the administrator by merely selecting the date and time.

In some embodiments, fingerprint definitions 906 may be combined with one or more of the following to form a portable patch definition file: vendor bulletin(s) discussing the patch(es), report(s) prepared by embodiments of the invention for administrators, target computer 500 signature(s), deployment package(s). This patch definition file provides information that can be used to update other networks. The patch definition file (a.k.a. "patch metafile") provides a portable uniform data representation which can be employed by embodiments of the invention to move or replicate patches among update servers 528 of different networks. Suitable networks 100 include without limitation networks that are not connected to the Internet and/or to each other, such as military networks that are isolated to provide greater security. This movement/replication can be done by email, tape write/read, and/or other conventional data transfer means. The patch metafile may also aid the interchange and interoperability of patches between inventive embodiments supplied by different vendors.

The patches that need to be loaded onto specific target computers are listed on the update server 220 in update lists 222 associated with update agents 204, 210; in the illustration, list 224 is associated with Target1 202, and list 226 is associated with Target2 208. The update lists specify at least one location (through means such as a universal resource locator, or URL) where the patch can be found, and optionally include a date which is the earliest date that the software can be installed.

In operation, the update agent 204 of Target1 202 checks its update list 224 at the onsite or offsite update server 220 to see if a new package should be installed. If one is there, the update agent 204 checks to see if the package is already in memory on



ForcePoint LLC
EX-1009
Page369

the update server 220. If so, the update agent 204 attempts to install the software patch directly from the update server 220. If not, the update agent 204 attempts to install the software patch directly from the package computer location 232. In some instances, this is successful, in which case the update list 224 is updated.

In other cases, a download 218 will be obstructed by the firewall 214. If this happens, the update agent 210 informs the update server 220 and then the update server 220 itself will attempt to retrieve the package and place it in memory 228. From that memory on the update server, the software is installed directly to the target machine.

A monitor checks to see that the software installs properly on the target 202, 208, and then continues checking (or can be notified) to ensure that the updated software runs correctly and that the target computer itself doesn't experience any problems in what appear to be unrelated areas. Should the package fail to install properly, or create problems for the software program that was patched, or create other problems on the target computer, the package can be automatically removed and the computer restored to its preinstalled state or another acceptable state in which the update has been removed or disabled, and the target computer is in a workable state. If the package has been installed on more than one computer, they all can be removed. If the error occurs in the middle of a rollout to many computers, the rollout can be halted and the software removed or disabled. The monitor may be located on the update server 220, on a repository site 600, at least partially in the update agent 204, 210, and/or in a combination of such locations.

When there is a problem with an installation, or when an installation is successful, an administrator can be notified by email, by pager, or by some other notification means.

The update agent 204, 210 can also be used to survey its own target computer, and this information can be stored in a database offsite or at another location. This information can then be used to determine what updates a given target computer needs in order to have the most appropriate configuration. When a new software patch becomes available, the stored information can be used to determine if a particular target computer needs the patch.



ForcePoint LLC
EX-1009
Page370

It should be noted that target computer can include any type of server or workstation, regardless of operating system or installed software. Moreover, the scope of the invention applies to many other devices including wireless devices (mobile phone, personal digital assistant, pocket computer, etc.), intelligent switch devices, hubs, routers, and any other type of Internet-attachable device.

Other aspects and advantages of the present invention will become more fully apparent through the following description.

## BRIEF DESCRIPTION OF THE DRAWINGS

To illustrate the manner in which the advantages and features of the invention are obtained, a more particular description of the invention will be given with reference to the attached drawings. These drawings only illustrate selected aspects of the invention and thus do not limit the invention's scope. In the drawings:

Figure 1 is a diagram illustrating one of the many distributed computing systems suitable for use according to the present invention.

Figure 2 is a diagram illustrating systems according to the present invention.

Figure 3 is a diagram illustrating methods according to the present invention.

Figure 4 is a diagram further illustrating methods according to the present invention.

Figure 5 is a diagram further illustrating systems according to the present invention.

Figure 6 is a diagram further illustrating systems according to the present invention.

Figure 7 is a diagram further illustrating systems according to the present invention.

Figure 8 is a diagram further illustrating methods according to the present invention.

Figure 9 is a diagram further illustrating systems according to the present invention.



ForcePoint LLC
EX-1009
Page371

## DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

The present invention provides systems, methods, articles, and signals which help update existing software across a remote network. The invention relates more specifically to updating software across a client-server system with little or no need for human oversight, and without requiring copies of the software patches on an administrative machine on the network whose clients are being updated. The update is automatic, and it can detect errors within a specific update and automatically rollback a faulty update to leave the network in a usable state.

Separate figures discussed herein illustrate various embodiments of the present invention, but the discussion of a given figure is not necessarily limited to a particular type of embodiment. For example, those of skill will appreciate that the inventive methods may also be used in configured storage media and/or computer system implementations according to the invention. To prevent unnecessary repetition, the discussion of methods thus applies to articles and systems, and vice versa, unless indicated otherwise. It will also be appreciated that method steps or system components may be renamed, regrouped, repeated, or omitted, and that method steps may be ordered differently and/or performed in overlapping execution, unless the claims as properly understood call for particular steps or components and/or require a particular order of execution.

For the reader's convenience, some pertinent information on related technologies such as networks and firewalls is provided below. The invention goes beyond previously known technologies, but it may partially comprise or rely on earlier advances in computing and networking and/or be used together with such earlier advances.

### Systems Generally

As shown in Figure 1, computer networks 100 such as secure computer networks 102, 104, may be configured according to the invention. Suitable computer networks 100, 102, 104 include, by way of example, local networks, wide area networks, and/or portions of the internet. "Internet" as used herein includes variations such as a private internet, a secure internet, a value-added network, a virtual private network, or an intranet. Secure networks may be secured with a security perimeter



ForcePoint LLC
EX-1009
Page372

which is defined by firewall software 116, 118 routing limitations, encryption, virtual private networking, and/or other means. The network 100, 102, 104 may also include or consist of a secure intranet, which is a secure network such as a local area network that employs TCP/IP and/or HTTP protocols internally. The computers 110 connected by the network for operation according to the invention may be workstations 114, laptop computers 112, disconnectable mobile computers (such as PDAs or other wireless devices), servers, computing clusters, mainframes, or a combination thereof. The computer hardware may be general-purpose, special purpose, stand-alone, and/or embedded. The network 100 may include other networks, such as one or more LANs, wide-area networks, wireless networks (including infrared networks), internet servers and clients, intranet servers and clients, or a combination thereof, which may be protected by their own firewalls.

A given network 100 may include Novell Netware® network operating system software (NETWARE is a registered trademark of Novell, Inc.), NetWare Connect Services, VINES, Windows NT, Windows 95, Windows 98, Windows 2000, Windows ME, Windows XP, Windows 2K3, LAN Manager, or LANtastic network operating system software, UNIX, TCP/IP, AppleTalk and NFS-based systems, Distributed Computing Environment software, and/or SAA software, for instance (VINES is a trademark of Banyan Systems; NT, WINDOWS 95, WINDOWS 98, WINDOWS 2000, WINDOWS ME, WINDOWS XP and LAN MANAGER are trademarks of Microsoft Corporation; LANTASTIC is a trademark of Artisoft; SAA is a mark of IBM). The network may include a local area network which is connectable to other networks through a gateway or similar mechanism.

One system according to the invention includes one or more servers 106 that are connected by network signal lines 108 to one or more network clients 110. The servers and network clients may be configured by those of skill in the art in a wide variety of ways to operate according to the present invention. The servers may be configured as internet servers, as intranet servers, as directory service providers or name servers, as software component servers, as file servers, or as a combination of these and other functions. The servers may be uniprocessor or multiprocessor machines. The servers 106 and clients 110 each include an addressable storage medium such as random access memory and/or a non-volatile storage medium such as



ForcePoint LLC
EX-1009
Page373

a magnetic or optical disk. The signal lines 108 may include twisted pair, coaxial, or optical fiber cables, telephone lines, satellites, microwave relays, modulated AC power lines, and other data transmission "wires" known to those of skill in the art, including wireless connections. Signals according to the invention may be embodied in such "wires" and/or in the addressable storage media.

In addition to the network client computers, a printer, an array of disks and other peripherals may be attached to a particular system. A given computer may function both as a client 110 and a server 106; this may occur, for instance, on computers running Microsoft Windows NT software. Although particular individual and network computer systems and components are mentioned, those of skill in the art will appreciate that the present invention also works with a variety of other networks and computers.

Suitable software and/or hardware implementations of the invention are readily provided by those of skill in the art using the teachings presented here and programming languages and tools such as Java, Pascal, C++, C, Perl, shell scripts, assembly, firmware, microcode, logic arrays, PALs, ASICs, PROMS, and/or other languages, circuits, or tools.

**Configured Media Generally**

The servers 106 and the network clients 110 and individual computers 110, 114 are capable of using floppy drives, tape drives, optical drives or other means to read a storage medium. A suitable storage medium includes a magnetic, optical, or other computer-readable storage device. Suitable storage devices include floppy disks, hard disks, tape, CD-ROMs, PROMs, RAM and other computer system storage devices. The substrate configuration represents data and instructions which cause the computer system to operate in a specific and predefined manner as described herein. Thus, the medium tangibly embodies a program, functions, and/or instructions that are executable by the servers and/or network client computers and/or individual computers to perform updating, monitoring, administrative and/or other steps of the present invention substantially as described herein.



ForcePoint LLC
EX-1009
Page374

**Firewalls**

Network administrators generally do not allow just any information into their system. Rather, they use a firewall 116, 118 to protect the network. A firewall is hardware and/or software device that screens incoming messages (often based on content, origin, or nature of request) and only allows to pass those that are deemed safe. Three main types of firewalls are screening routers (also called packet filters), proxy server circuit-level gateways, and proxy server application-level gateways. Screening routers can base decisions on external information about a network packet such as its domain name and IP address, so messages that come from acceptable domain names and IP addresses can be allowed through 120, 124 while refusing messages from other locations 122. Proxy server circuit-level gateways disguise information about an internal system when passing the information to an external system. The IP addresses of the internal computers are typically replaced with the IP address of the proxy. At this level, authentication can be required. Proxy server application-level gateways provide all of the features of screening routers and circuit level gateways while also allowing the contents of the packets themselves to be evaluated. Messages can be rejected for content as well as for security violations.

**Software Installation**

System administrators often need to change the software on a specific machine. A new piece of software must be installed for the first time, as when a new application is added to a machine. An already-installed piece of software can be updated, as when a new version of an existing piece of software will be installed on a specific machine; this is also referred to as "replacing" the software. A data file of an existing piece of software can be updated without otherwise changing the software configuration, as when tax tables are updated on an accounting program, or when anti-virus software files are updated. If a problem is discovered in an existing piece of software, then a fix or patch can be installed. Any or all of these changes to the state of a specific machine or machines are referred to in this patent as "installation". Similarly, the words "package," "patch," and "update" should be given the broadest possible meaning. For example, package could refer to an entire program including all the necessary files, to one or more data files, to a software patch to an existing file, to



ForcePoint LLC
EX-1009
Page375

a change to a configuration file, to a *.dll file, a driver file for a specific piece of hardware attached to a computer and/or a computer network, and so on. "Update" refers to at least attempting to install a package on a computer.

**Methods Generally**

With reference to Figures 3, 4, and 5, one embodiment of a method operating according to the present invention includes a target computer 500 in a pre-update state. The target computer 500 is the computer that the invention will at least attempt to update; not every embodiment of the invention requires that the update be successful. An update server 528 is connected across a network 524 to the target computer. In some implementations the target computer has a network connection, such as a connection through a winsock layer. Typically, the target computer is protected by a firewall 526, as explained above, but the update server can drill through the firewall to access the target computer.

Many existing enterprise software management tools use agents. Examples include Microsoft SMS software, Microsoft Active Directory software, IBM Tivoli products, Symantec anti-virus software, McAfee anti-virus software, and Novell ZENworks software (marks of their respective owners). In large networks, agents can wake up and report in parallel to a server when they have information to report. By contrast, tools that lack agents rely on remote API calls, which are polled continuously by the server, making them linearly scaleable in performance rather than parallel processing as seen in the preferred embodiment.

Agents in embodiments of the present invention can receive compressed files to conserve network bandwidth. Compression also enhances security, because decompression errors may indicate that a patch has been tampered with.

An inventive agent can also resume a download when a mobile target bearing the agent is disconnected and then reconnected to the network at another location, unlike patch management tools that lack agents and therefore download entire service packs or files after being interrupted. Tools lacking agents may also generate uncontrolled spikes in bandwidth utilization as patches are deployed, whereas some embodiments of the present invention permit an update server to be controlled by an



ForcePoint LLC
EX-1009
Page376

administrator so that the server uses only a specified amount of bandwidth per agent connection (bandwidth throttling).

Conventional patch tools that rely on a permanent LAN/domain connection and lack client agents may rely on a remote registry service, which provides registry information to a remote computer. The remote registry service is not available on the Windows 95, Windows 98, or Windows ME platforms. Such a service can be a security risk in organizations whose client computers are on the Internet, because they allow remote computers to read the registry of a client, thereby providing information that can be used to guide infiltration or other attacks on the client's security. Embodiments of the present invention preferably avoid using a remote registry service, due to the security risk.

An update agent 508 is located at each computer that is to be updated. The update agent is a software component (usually not very large) that may be installed initially, either in the traditional manner or by using the invention, on the network target machines such as workstation(s) and/or server(s). The update agent is capable of operating in the place of the human administrator, at the direction of the human administrator, to perform work in a manner similar to what could be performed if the human administrator was actually present at the machine. The update agent knows how to perform four basic tasks: 1) how to contact the update server 528 to retrieve a list of tasks, 2) how to start the tasks in the task list received, 3) how to process policy information for hours of operation and so forth, and 4) how to register with the Update Server.

The update agent is capable of updating, configuring, or replacing itself without the need for manual intervention after the initial install. Typically a small boot-strap agent will be installed initially, but will grow in abilities as the administrator dictates or as required to fulfill administrator requests. The update agents of different sorts of target computers 500, such as onsite administrator computers, server computers, and client machines, can all start out as the same version of a single agent. Machines in a given network can all have a the same agent installed, or machines can have unique agents installed. When there is more than one client, each can have a different update agent initially, or a mix of agents can be installed on different client machines, as is chosen by the administrator, or as is set up as a default.



ForcePoint LLC
EX-1009
Page377

Similarly, multiple servers and administrators can also have a mix of different agents initially installed. As the agents for the individual target computers change, they can all change in concert or they can diverge. In some embodiments, the agents may all start out different and then converge in functionality.

The offsite location of the update server 528 is a location distinct from the target computer. The location may be offsite at a completely different vendor, or offsite at a different physical location from the target computer 500 but at a location managed by the same entity, at the same physical location. It may be at a different-appearing location from the target computer 500, such as at a subcontractor location, or at some other distinct-appearing location. The important point is that as far as an individual target computer operating system is concerned, the work appears to be off-site. One embodiment locates the update server on the target computer 500 but in a fashion (such as in a different partition) that appears offsite.

The actual update material that is to be installed on the target computer is often stored at a separate location (known as the package computer) apart from the update server and the target computer. The software update itself can be any of a wide variety of software that can be updated across a network, such as an incremental software patch, a new software program never before installed on the target computer, an update to an old program, software scripts, data files, or even an update of the update agent.

If a known condition is met, during a placing step 300, a task id is placed on an update task list 222. The known condition could be that the patch is not currently on the computer, that the administrator has given assent, the owner of the target computer 500 has permission from the owner of the package, the fact that no one has specifically denied placing privileges, or some other known or inventive condition. The update task list located on the update server is associated with a specific target computer 500, and specifies at least one download address where the software update can be found. The download address can be in any format that is understandable to the computers. The invention does not depend upon any specific addressing convention. Two common addressing formats in use currently are the "Universal Resource Locator" and "fully qualified domain name" formats. Other formats are PURLs (Persistent Uniform Resource Locators) and URN's (Uniform Resource Names), and



ForcePoint LLC
EX-1009
Page378

other naming schemes may be known in the future. Other information that may be included in the task identifier, such as a date the download will first be attempted. Multiple download addresses, each of which specifies a location where the software update can be found, may all be associated with a specific software update.

During a starting task step 304, the software update is at least attempted to be uploaded from the package computer 567 to the update server 528. During an optional choose package computer step 306, if more than one download location is placed on task id list 226, the location that the software update will be downloaded from is chosen. The choice can be made by any known or inventive method, such as using the first location on the list, using the location that a test message returns from most rapidly, using the first available machine, and so on.

Once a location for the update is known, the software download is attempted from the location of the package computer 548 to the memory 530 of the update server 528. If the download is unsuccessful, then in one inventive method another location from the list of possible locations in the task update list is chosen, and the download of the software update is retried. In some implementations, if the download can't be completed for some reason, the update server 528 waits for a time and tries to download from the package computer 567 again. If the download is successful, then the update server 528 attempts to download 312 the software update to the target computer 500.

Once the download is in cache or other memory in the update computer a second download 312 is attempted to download the software package from the update server to the target computer. In some embodiments of the method, the second download 312 is delayed 310 by some predetermined criterion. This delay may be from the start of the first download, with the delay period based on an estimate of the time needed to download the software update from the package computer to the update server. The second download may also be delayed to a specific time of day when the target computer 500 has less of a chance of being used, such as after a business closes for the day. Other known or inventive delay criteria may also be used.

A monitor 302 checks the installation, performing the role typically played by an administrator, to determine the results of the installation 314. Once the results are known, an administrator can be notified 328. Notification can be by sending an email



ForcePoint LLC
EX-1009
Page379

330, by paging someone, by sending a pre-recorded phone message, or by any other known or inventive method and means.

If the monitoring step detects a failure 316, then the task that failed is suspended 318. The first download 308 to the update server 528 could fail, as could the second download from the update server 528 to the target computer 500. If there are multiple target computers having the software update installed, the Nth installation could fail, and so on. Determining results preferably goes beyond simply ensuring that the software update appears to have installed properly, and in some embodiments of the invention extends for a time beyond the installation. For example, one embodiment of the monitor will test a patch application by having it installed on only one target computer, assuring that it downloads properly, installs it and then watching it for some period of time until the administrator who sets the time delay gains enough confidence in the patch to allow it to be applied to other target computers. Should the application of this patch cause abnormal activity, as noticed by undesirable behaviors either in the program whose software was modified or elsewhere in the computer, the rollout can be automatically suspended until the problem is resolved.

Furthermore, in some instances when failure is detected the software update is disabled or removed 324 from the target computer, and that machine is returned substantially to its pre-update state or another acceptable (working) non-update state. This may mean that the installed software is taken off the target machine 322; or that not only is the software removed, but all the ancillary files (.dll's, .exe's, etc.) are restored to their pre-update state. In other instances it may mean that the target computer or some portion thereof was backed up before the software update was installed, and the backup itself is restored onto the machine.

If there are multiple target computers 500, failure may be detected by the monitor after the software has been installed on one or more machines. In this case, the software update can be removed 322 from not only the target computer 500 where the failure was spotted, but it can also be removed 326 from all of the other target computers 500 where the software was previously installed 326. The removal request can come from an administrator or removal can be performed automatically after failure is detected 316.



ForcePoint LLC
EX-1009
Page380

The monitor 302 may perform more tasks than simply waiting to hear if a software package has installed successfully. For example, in some instances the monitor waits for a time period 400 after the installation and if it has not heard otherwise, assumes that the installation was successful.

Administrators and administrator helpers can benefit greatly from a central repository where they can enter and retrieve information regarding requests for help. One such method is help desk "ticketing". Ticketing records the requestor, the type of request, when help was requested, when the response to the request was completed, and other useful information. A PatchLink HelpDesk service provides facilities for administrators to manage their network requests and network resources, both people and computer resources, via one central repository. PatchLink HelpDesk software provides these facilities across the internet without an invasive application install on the administrator's network that introduces yet another resource that has to be managed, backed up, and updated – this is taken care of transparently at an offsite Management Center.

A website, reachable by a standard web browser or some other known or inventive network connection, provides the facilities to use the help desk services. A preferred implementation is currently at the PatchLink web site, reachable at www.patchlink.com. Simple web forms support the data collection required to begin the enrollment process. Once the enrollment process is complete, the administrator can license one or more services on a recurring subscription basis.

The enrollment process is begun when the administrator initiates the process by signing up and indicating a desired level of participation. A preferred embodiment of the invention has three different user levels: guest, regular, and executive. A guest is allowed to view the web site and can read the user forums but cannot post to the forums. A regular member can perform guest functions and can also chat in chat rooms, and post to forums. An executive member has a subscription to the site. He or she can perform regular member functions, and can also use the more advanced features of the site, such as offsite automatic package updates (e.g., PatchLink Update services), offsite monitoring (e.g., PatchLink Monitor services), and the offsite help desk functions (e.g., PatchLink HelpDesk services).



ForcePoint LLC
EX-1009
Page381

One embodiment of the method entails an email being sent to a customer care agent assigned to the customer's telephone area code. The customer care agent telephones the pending user to complete the enrollment process. The customer care agent collects the necessary identity information and payment information, and then upgrades the pending user's account to permit use of the account, making the pending user an administrator/ user. The areas the administrator/user can participate in or use is controlled by entries in the licensed product's table of the update host's database. These entries are created by the customer care agent during the enrollment process.

Recall that all these services are available to the administrator via a browser or other network connection across the internet. When a monitor enters a ticket into the help desk and initiates a rollout in some instantiations of the inventive method, the monitor then decides whether a failure has occurred 316, 406. To decide the monitor may look at what software updates have recently been installed, how long ago the installations occurred, the current hardware and software configuration, and so on. Which incidents are considered failure can be, without limitation, set by an administrator; defaults can be used, and judgment of the help desk personnel can be taken into account.

In a detect success step 408 the target computer 500 sends a message 410 to the update server after the download from the update server to the target computer has completed successfully. The monitor can presume success 404 if a specified time period has passed without noticing or being notified of a failure.

Failure can be detected in other ways 316, 406. For instance the target computer can notify the monitor that a failure has occurred; a user can notify the monitor through the help desk or through a direct link that a failure has occurred; when a target computer does not contact the monitor within a specified time from the beginning of the second download 312 onto the target machine, a human administrator can declare that a failure has occurred; and so on. Notice that even after the monitor has declared the outcome of a download to be a success, later events, such as an indication of failure from the help desk, can cause the monitor to declare the download to be a failure.



ForcePoint LLC
EX-1009
Page382

In one embodiment of the invention, the update server 528 waits for a confirmation of a successful installation (by the monitor, or by another known or inventive contact method) before the next target computer 500 has the software update placed on its update list 222. The update server checks for a target computer 500 that is eligible for the software update but has not yet received it 412. If one is found 414 the appropriate task identifier, specifying the target machine, the software update, and the location, are added 416 to the update server's 528 task update list. This way, rather than a mass update where all eligible computers have the software installed on them *en masse*, the rollout proceeds one computer at a time until, after a default or user-defined number of successful installations, the rollout is deemed a success; at that juncture the software update is made available to more than one target computer at a time.

It is not always clear by looking at a specific target computer 500 what software packages and patches have been installed. The invention includes a method to analyze a target computer 500 to ensure that a given patch has not already been installed on the computer 500 before the invention attempts to install that patch. The following discussion includes references to Figures 8 and 9 and continuing reference to Figure 5.

A patch fingerprint which defines a specific software update is described in greater detail below. The patch fingerprint is located 800 by monitoring a patch component database location 900 for a new patch fingerprint 902. The word "new" here indicates that the patch has not yet been downloaded into the repository component 600, or for some reason needs to be downloaded into the repository component again, even though it has been downloaded previously. There may be one or many patch component locations; those locations may be on a separate computer connected to the system through a network link, on the update server 528, on the target computer 599, on the package computer 567, on a non-networked location such as a CD, a tape, a floppy disk, etc., or some other known or inventive location.

Once the patch fingerprint 906 is located 800, it is placed 802 into the repository component 600. The usual method of placement is to download 804 the patch fingerprint 906 into the repository component, but in some embodiments the



ForcePoint LLC
EX-1009
Page383

fingerprint 906 will be on the same file system, so the patch fingerprint will be copied without using the network, such as copying between partitions.

The illustrated patch fingerprint comprises one or more general inventory install dependencies 912 that can be used to take a high-level look to see if a specific patch can be installed on a machine. It also includes a signature block 910 that can be used to request specific information from a target computer 500, and an existence test 908 which can use the signature block information to determine if a specific patch has been loaded on a machine.

In some versions of the invention, the inventory install dependencies 912 describe at least some of the necessary software and hardware that must be installed on the target computer 500. These dependencies 912 are compared 808 with information about the target computer 806 previously stored in the inventory library 918. If the install information and the inventory information don't match, then the patch is not installed. In some versions of the invention a message is sent to at least one administrator containing a list of components required (such as necessary hardware and software) for the install.

If the necessary inventory information is on the target computer 500, or if the inventory info is not used, then the signature block is sent 810 from the repository computer 600 to target computer 500. The information requested in the signature block 814, which may consist of more specific install information, is gathered 812 by the discovery agent 548 and then sent back to the repository component 818. In some versions of the invention, the discovery agent also gathers other information 816 about the target computer such as usage statistics, hardware and software installed, configurations, etc. This information can then be used to populate the inventory library 918.

Once the signature information 910 has been sent to the repository component 600, an evaluator 914 evaluates at least a portion of the specific install information requested by the signature block using the existence test 908, and in some instances the inventory install information 912, to determine if the patch is absent 822 on the target computer 500.

As an optional step, once it has been determined if the patch is absent a message is sent 824 to at least one address associated with an administrator. This



ForcePoint LLC
EX-1009
Page384

message may be sent using a variety of methods, including email, pager, fax, voicemail, instant messaging, SNMP notification, and so on.

**Patch Fingerprint**

With continuing reference to Figures 5, 8 and 9, one embodiment of the system verifies that a software package can be or should be installed on a given target computer 500 before attempting installation. To do so, a patch fingerprint 906 is used, e.g., by an agent on a client. The patch fingerprint defines how to determine if a given software package/incremental patch has been previously installed. It may also define a minimum hardware/software configuration necessary for the patch installation. These patch fingerprints 906 are stored in a fingerprint library 904. The fingerprint library 904 is located on a repository component 600. This repository component 600 may be located on the update server 528, or may be in a separate location accessible to the update server 528 and the target computer 500. Some versions of the invention also include an inventory library 918 which contain target inventories. Each target inventory 920 contains the hardware and software information about a defined set of target computers 500. This defined set may include as few as one computer or as many as all of the computers in a given network, or some number in between.

The fingerprint library 904 can be automatically replenished. In some embodiments, at least one, but possibly several, patch component database locations 900 are monitored 800 for new patches 902. In some embodiments of the invention a signal from the locations 900 indicates to the repository component 600 that new patches 902 are available 800. In the preferred implementation the fingerprint library 904 is updated with new patch fingerprints at specific time intervals. After the repository component 600 is aware of the new patch fingerprint, the patch fingerprint is placed into the repository component 802, usually by using a downloader 924 to download the new patch fingerprint. Patch fingerprints may be entered into the repository components in other ways, however. For example, one or more patch fingerprints may be manually installed into the fingerprint library by an administrator.

**Inventory Library**

The repository component 600 also contains an inventory library 918. A discovery agent 548, which in some embodiments initially resides on the update



ForcePoint LLC
EX-1009
Page385

server 528, is installed from the update server 528 to the target computer 500 using known or inventive methods. This discovery agent 548, described in greater detail below, inventories at least some of target computer 500's software information 606, hardware information 608 including specific software updates and patches installed, usage information 604, registry information 612, web information 610, configuration information 614, services 618, file information, patch signatures which have been utilized, etc.

This information, or a subset or superset thereof, is then sent, in some embodiments in compressed form, to the target computer inventory 920 in the inventory library 918. Result information can be quite voluminous, and hence may be compressed for efficient upload and to minimize bandwidth usage on the customers network. A preferred implementation sends the data using an XML data transfer, though any other known or inventive data transfer method can be used. Transfer of inventory information may also be encrypted within a customer network to prevent unwanted wire-level snooping of system configuration information.

**Report Generator**

With this information, a report generator 922 can present a user with detailed reports of the current patch status for all computers within the network, illustrating the number of computers needing the patch, the computers already installed with the patch, computers that can't receive the patch until hardware or software is upgraded and so on. In addition, the report generator 922 can provide a partial or complete inventory of the computers attached to the network. In some embodiments the report generator 922 provides graphical presentations of the inventory for analysis by the administrator, both to track location of hardware as well as to ensure software license compliance. However the repository component 600 also uses the inventory library 918 information as well as detected fingerprint information to distribute relevant signatures 910 from the patch fingerprint 906 to the discovery agent 548, thus greatly optimizing the patch discovery process by eliminating unnecessary scanning work at the target computer 500.



ForcePoint LLC
EX-1009
Page386

**Discovery Agent**

One optional step to decide if a given software program or patch can be installed is by verifying that the necessary hardware, if applicable, is present, and/or the necessary software is present. For example, some programs may require a specific operating system, some programs may require a certain processor. As an example, if an update of Microsoft Word software is to be installed, it is necessary that Microsoft Word software be on the machine. These high-level dependencies are stored, in some versions, in the inventory install block 912 in the patch fingerprint. The information in the inventory install block is generally high level enough that it can be pulled out of the target inventory 920 of the specific target computer 500 stored in the inventory library 918.

In some implementations of the invention the patch fingerprint 906 also includes installation dependency information 912. This, as explained above, is information about the target computer 500 that can be expected to be found in the inventory library, and so can be checked without querying the target computer 500. This includes software that should be present (such as a specific version of a program, a patch, a data file or a driver) a hardware component that should be present, or specific hardware and/or software that shouldn't be present.

If the inventory library does not have an up-to-date inventory for the target computer 500, the discovery agent can be used to scan the target computer 500 for inventory information; it does not necessarily need to also scan simultaneously for signature information. In the preferred implementation, the first time that the discovery agent 548 runs on a given target computer it scans only for inventory information and then loads that information into the inventory library 918; it ignores the patch fingerprint information. At other times when the discovery agent 548 runs it may ignore inventory information and may, rather, be used to look up specific signature information 910 to test for the existence of a specific patch. When the signature block information is looked for, values such as registry entries and INI file values may be inspected for existence, or the actual value may be returned to the repository component 600.

Each Patch fingerprint comprises a signature block 910 and an existence test 908. The patch signature block is a set of information requests, the information itself



ForcePoint LLC
EX-1009
Page387

to be gleaned from a target computer 500 which will then be used to determine if all necessary bug fix and security patches are installed. Examples of patch signature block information include but are not limited to file, hardware, registry and configuration information, a specific file name or directory name, all or part of a path that a file is expected to be found in, a specific version number of a file, a created date of a file, a specific file version of a file, and a specific registry value.

In one implementation the fingerprint library 904 is a SQL database. The patch signatures 910 are extracted from the SQL fingerprint library and then sent to all target computers that meet the dependency criteria for operating system and installed software as specified in the inventory install information 912.

A preferred implementation employs an XML-based request input file. The result file sent back to the update server 528 also employs XML formatting. This result file contains the signature information for the target computer, and may also contain the software and hardware inventory updates. The inventory and signature information sent to the update server can be quite voluminous, and so are compressed and may also be encrypted in the preferred implementation. The following is a sample patch signature that will gather registry information for Microsoft Outlook as well as the EXEs date and time, and information in the registry :

```
<file componentid="1" reportID="1">
        <name>outlook.exe</name>
        <path></path>
        <version>9.0.2416</version>
        <created></created>
        <size></size>
        <root>HKEY_LOCAL_MACHINE</root>
        <Key>SOFTWARE\Microsoft\Windows\CurrentVersion\App
Paths\OUTLOOK.EXE</key>
        <value>Path</value>
</file>
```

Once the discovery agent on the target computer has returned its scan results for the signature, the existence test 908 logic is used by the evaluator 914 to infer whether that particular computer actually has the patch or not. This algorithm



ForcePoint LLC
EX-1009
Page388

minimizes the number of tests that must be done by the evaluator: its sole responsibility is to discover information – allowing the data analysis to be done by the repository component 600 itself. Distributing the workload in this fashion provides a better implementation for scanning and analyzing huge numbers of workstations and servers.

Each existence test is specific to a given patch. A sample existence test might appear as: if registry QQ contains value ZFILEVAL or (if file _Z123.bat was changed on date 12/12/2000 at 11:52 pm and file Z is of size ZFILESIZE) then the patch ZPATCH is present. The preferred embodiment of the patch fingerprint library is an SQL database, but other known or inventive databases can be used.

Note that a patch fingerprint may also contain dependencies to other Finger Print definitions: for example, "MS-023 IIS Vulnerability Fix" patch might hypothetically require the presence of "Microsoft Windows Service Pack 2". This is used to further optimize where the patch signatures are actually sent. These may sometimes be used in the install dependencies info 912 and other times in the signature block 910, depending on circumstances.

In addition, fingerprint definitions 906 are also normally associated with a software package 554 suitable for deployment by the system. Once the need for a particular patch has been established by scanning its signature(s) on a computer or all computers within the network, it can then be quickly deployed by the administrator by merely selecting the date and time.

A fingerprint definition 906 may also contain a logical expression that should be evaluated to assess whether the other elements within the patch signature should be evaluated to TRUE (patched) or FALSE (not patched). The expression is a simple logical statement such as (A AND B) | C where A, B, and C refer to other fingerprint definitions within the patch signature.

In some implementations the downloader 924 regularly checks the patch component database for new patch fingerprints. When a new patch fingerprint is located, it is downloaded into the repository component. The evaluator compares the dependencies needed for the specific patch implementation listed in the install info 912 with each of the target computer 500 specifications listed in the inventory library. Then an update list is created which may identify all of the target computers 500 that



ForcePoint LLC
EX-1009
Page389

need the patch, all of the target computers that don't possess the patch, all of the target computers that can receive the patch, as they have the necessary dependencies, and/or all of the target computers 500 that have already received the patch. This update list may now be used to update the target computers, and/or may be sent to an administrator by a notifier 916.

In some instances of the invention the patch component database is owned by someone other than the target computer 500 owner. Only if this patch update host has given permission to the target computer 500 owner will the downloader be allowed to download the new patch fingerprints into the repository component. The permission may comprise a purchase agreement, a lease agreement, subscription for download permission and an evaluation agreement.

If any modifications are made that may be of interest to the administrator, the notifier 916 will send a notification message containing the new patch updates that have become available or the patch-related state changes that have occurred in his network configuration. Notifications can be sent via e-mail, pager, telephony, SNMP broadcast or Instant Message.

**Target Computer**

In one embodiment, the inventive system comprises three pieces: a target computer 500, an update server 528, and a package computer 548. The target computer 500 has a memory 502, and a network connection 504, which in at least one implementation of the invention is a winsock layer. A socketless protocol can be implemented, or any other known or inventive network connection can be used. The update server 528 has a memory 530 that may include an optional backup storage 534, and a network connection 532. The package computer 567 has a memory 550, and a network connection 552. Figure 5 shows one target computer 500 for convenience but there may be many more in a given embodiment. Likewise, one update server 528, and one package computer 567 is shown for convenience, the invention may only require one but also support two or more. In a preferred embodiment these pieces are all separate computers, but they can be virtual pieces of the same computer, such that they appear to be distinct. For example, the "package computer" piece may reside on a different partition of the update server or the same partition.



ForcePoint LLC
EX-1009
Page390

The target computer contains a network connection 544, which may be protected from the outside by a firewall 526 as is discussed above. Different target computers within a network may run on different platforms; for instance, some may be Windows machines, some Unix machines, etc. The same update server 528 can be used for all the platforms, or different update servers 528 can be specified by platform type, or the update servers 528 can be assigned to target computers 500 using a different schema.

The target computer 500 also contains an update agent 508. The update agent is a software component that can be installed using the inventive method on multiple machines at a time or, in some embodiments of the system, can be installed in the traditional manner on the target computer 500. Once registered, the update agent 508 knows how to perform three basic tasks: 1) how to contact an update server 528 to retrieve its list of tasks from its update list 536, 2) how to start the tasks in the task list received, and 3) how to retrieve policy information received from the update server 528 that control polling interval, hours of operation, and so forth.

**Update agent**

The update agent of target computer 500 contacts the update server 528 to determine if there is work for the agent 508 to do. The update server 528 determines this by analyzing an agent's update list queue 536. This update list 536 contains, at a minimum, a software location reference 538, but can also contain a date 540 that indicates the earliest date that the software package 554 can be installed, and multiple software location references, if the same software package is available from multiple locations. The types of software 554 that can be updated comprise, without restriction, patch files 556 that update a currently installed software application on the target computer, data files 558, script files 562, new application files 564, executable files, 560 driver updates, new software versions and updates to the update agent file itself 566 .

When the update agent discovers an entry on its associated update list 536, with an appropriate date 540, if any, the installer 510 initially checks to see if a copy of the software package already exists in memory 530 on the update server 528. If found, it then downloads the software package directly from the update server. This



ForcePoint LLC
EX-1009
Page391

situation may arise when a previous target computer 500 has requested the software package 554 from the update server 528.

If the software package is not found, the installer 510 then attempts to download the update directly from the package computer location given in the software location reference 538 to the target computer memory 502 using its network connection. This will be possible if there is no firewall 526, or if the update server can connect to the package computer location 548.

When an administrator builds a package that instructs the update agent 508 to retrieve the files from a "non trusted" source such as the package computer 567, the installer 510 will be unable to retrieve the resource directly. However, the update agent can ask the update server 528 to retrieve the package. In some implementations there are multiple update servers and the update agent 508 decides which one of them to access using some predetermined criterion. Examples include selecting the first update server 528 that is available, selecting the least-busy update server, selecting the update server that is "closest" in networking terms, and so on.

In one implementation of the invention, if the update server 528 can reach the offsite package computer 567, it reports to the update agent 508 that it can reach the resource and estimates the time the retrieval will take. This estimate informs the agent how long it should wait before the requested resource is available. If the calculation estimation is not exact, as it probably will not be because of internet traffic fluctuations and server response time variances, then if the agent asks for the resource again the update server will provide another wait time length and the agent will wait once again. This cycle will repeat until the update server 528 has the resource available in memory and can deliver it to the agent upon the next request.

As a particular software package could be requested multiple times by different agents 508, in one implementation of the invention, the update server 528 will store this resource in a local cache 530 from which it can fulfill additional retrieval requests. To prevent the update server 528 from filling up all its available memory with old software packages, one embodiment stores the number of times the package is accessed and the time of the latest access for the stored software package and estimates a "time to live" amount of time for that resource to stay in its cache. A separate task running in the update server 528 will check periodically for resources



ForcePoint LLC
EX-1009
Page392

that have "outlived" their usefulness and recover the update host's storage resources by deleting the stored software package update from the cache 530.

In one embodiment the update server will make the packages available to the list of agents one at a time. If an agent 508 or an outcome finder 512 reports that the application of the patch failed, or if the patch puts the agent's target computer 500 in such a state that it can no longer communicate with the update server, then the update server will suspend the rollout automatically on the administrator's behalf. At this point, the administrator, or some other designated person can be notified 516 of the outcome.

An outcome finder 512 determines if the software package installation was successful and then communicates its finding to the update server 528. If the outcome is unsuccessful, as discussed above, a restorer 514 places the target computer in an acceptable non-updated state. The outcome finder 512 does not necessarily monitor only the actual software installation; rather it can be set up to watch uses of the software that was patched, the entire target computer, and/or computers that are networked to the target computer, for some designated period of time. The outcome finder can also have different levels of success. For instance, the installation itself (file copying) can be considered a low level of success, while the target computer not misbehaving for a period of time thereafter can be considered a higher level of success, with different actions taken according to the success level. Success or failure can then be monitored as described earlier, and installation retried, suspended, etc. as necessary.

Some embodiments store a backup 506, 534 of a target computer 500 or a portion thereof before installing the software package on the target computer 500. Sometimes the backup is stored 534 on the update server, sometimes on the target computer 506, 500 which is having its software updated, and sometimes it is stored offsite at a repository site 600. When the outcome finder 512 reports a problem with a software installation, the restorer 514 can use the backup 534 to return the target computer to a non-updated state.

In one embodiment of the invention, the update server 528 waits for a confirmation of a successful installation (by the outcome finder 512, or by another known or inventive contact method) before the next target computer 500 has the



ForcePoint LLC
EX-1009
Page393

software location reference 538 to the package placed on its update list 536 at the update server 528. In a preferred embodiment, when an installation finishes, an administrator is notified 516 of the results by email 518, pager 520, voice mail 522, SNMP notification 568, instant messaging 570, fax or by some other means. If the installation failed, the specific machine that the installation failed on may be identified. In some embodiments, after a default or user-defined number of successful installations, the package is made available to more than one user at a time.

These update lists 536 facilitate the administrator's designation of pre-built packages, or custom built packages, to be delivered or rolled-out to managed workstations clients or servers, which we refer to as target computers 500. When these packages are to be made available, updates are scheduled by the administrator to be performed by the invention; this may automate a previous task requiring the administrator's visit to a client to install a patch or service pack.

The update agent 508 may be aware of the platform it is operating upon, and may be programmable or scriptable to perform actions on behalf of the administrator. In one implementation these features are exposed to the administrator through Package Builder wizards. A "software package" can be any combination of files, service packs, hotfixes, software installations and scripts. This presents an opportunity for the administration of remote machines, since almost anything that could be performed at a remote machine can be accomplished via the agent acting on behalf of the administrator.

One implementation of the invention allows scripts 562 to be run before (pre-install) and after (post-install) the package installation. An example of a pre-install script may be: (in pseudo-code)

check for available disk space
If disk space available greater than ValueX (where ValueX = room needed for install plus a buffer) then continue with installation.
Else, alert outside administration that an error has occurred, and terminate.
An example of an post-install script (again, in pseudo-code):
If install was successful, then notify an outside source that install successful.
If install was unsuccessful, then notify an outside source that install was unsuccessful.



ForcePoint LLC
EX-1009
Page394

Referring now to Figures 6 and 7, the network 200 may include many different sorts of target computers, each with an agent that may be specifically constructed for the specific target platform. For example, a network running Microsoft Windows PCs, Apple Macintosh computers, and UNIX computers, may have three types of agents. This provides a benefit in that the agent is capable of surveying its target computer and reporting this computer information 602 to the update server 528 and/or to a separate repository site 600 for storage. In some instances of the system, a discovery agent 548 is provided which performs the scan, as discussed elsewhere. In other instances the scan is performed by the update agent 508, or a downloaded script file 562. Hardware configurations 608, software configurations 606, information about the usage of various hardware and software components 604, web sites visited, emails sent and received 610, can all be sent to the offsite location 600. Once this information is available at the update server, an administrator can view the entire managed network from one place.

When the inventive system is implemented on an existing network, the discovery agent 548 may perform a survey of the software in existence at least on the target computer 500, with existing software configurations 700 detected and stored within the repository site 600 memory. Some systems may survey the entire network 200. When updates are called for, the system knows which ones are needed without needing to resurvey the network machines to check their current status.

A recommended configuration 704 for the target computer 500 is placed on the update server 528 or on the repository site 600. The recommended configuration may be decided on in many ways, either inventive or known to those of skill in the database arts, for example, by hardware configuration, by software configuration, by type of computer, by last package update, and so on. The discovery agent 548 then compares the current 700 and recommended 704 configurations and prepares a proposed list of updates 708 for the target computer 500. The update list may include service packs for installed software, previously uninstalled software, updated data files, and the like. The process of preparing the suggested list may take into account not only the current software configuration but also information such as the hardware configuration 608, and how often a particular program, data file, etc. is accessed 604,



ForcePoint LLC
EX-1009
Page395

as well as other information that is known to one of skill in the art. An administrator may be automatically notified of the update list.

Assuming that the target computer current configuration 700 generates a proposed update list 706, an administrator may be automatically notified 708. At this point, the computer use may be restricted until the new target computer is updated at least partially, until the administrator gives permission, or until some other inventive or known condition is met. This proposed update list 706 may also be used to define an update list 536 used to actually update the computer, as explained elsewhere.

Packages are composed of modules representing files, e.g., software files or data files, and scripts, which are sequences of actions to take upon files in the package. Alternatively one or more script file(s) may be included within the package content, and executed by the agent in order to install the patch. In some embodiments of the invention, a human administrator receives notice of the availability of new software patches. In other embodiments, the notices are sent directly to the offsite update server 528 which decides when to roll them out. The offsite update server can be configured to store in permanent memory the packages that have already been stored on each target computer. When a new package becomes available, or during the installation of an existing package, existing evidence of the software packages that need to be installed, as well as information about previous installations, is available in some embodiments at the offsite update server 528, and in other instances at the repository site 600.

The packages that are to be updated do not need to be owned by the target computer 500 user to receive access to it. In one embodiment of the system, the software package is owned by a third party which leases the software to the user. In another embodiment, the software package is owned by the update server user who then leases and provides access to the software package to the target computer 500 user.

**Security and Critical Patch Management, Features**

The present invention provides tools and techniques for managing and distributing critical patches that resolve known security vulnerabilities and other stability issues or enhancements, etc. in various operating systems. Suitable operating



ForcePoint LLC
EX-1009
Page396

systems include, without limitation, all Microsoft operating systems (e.g., 95, 98, ME, NT, W2K, XP, .W2K3), UNIX operating systems (e.g., Linux, Solaris, AIX, HP-UX, SCO, etc), and Novell NetWare operating systems. Operating system product names are the marks of their respective owners.

In the past, in order to manage security or otherwise critical patches, corporations and other computer users have frequently checked vendor web sites, e.g., by reading news reports or textual alerts posted around the world wide web or were sent notifications via email subscription or newsgroup etc, to find out about new patches. Upon learning that a vendor whose software is used by the corporation has released a new patch to fix or enhance application software, driver software, and/or hardware, the corporation's software administrative personnel have generally had to manually download the latest relevant patches, test them for compatibility with the corporation's machines in various layouts and configurations, and then distribute the patch(es) manually or using their traditional software distribution tools.

By contrast, the present invention can provide notification 824 of critical updates to computers in a proactive manner, whether or not they have Internet access. It can operate proactively by performing patch downloads without requiring an express administrator command to perform each download. It can also assist with distribution and installation of software updates, software packages, and other data to networked desktop, server, mobile, and other computers.

One embodiment of the present invention includes content replication through an update server 528 that retrieves the latest critical updates from a master archive such as a package computer 567. Retrieval may use 128-bit SSL or other familiar protocols for secure transmission. As new updates are added to the master archive, the updates' metadata are downloaded automatically to the update servers and/or the fingerprint library 904. If metadata indicates a patch is critical, the patch can be downloaded to the update server and cached there for rapid deployment. Each patch has an associated installer 912, prerequisite signature 910, and other fingerprint identification 906.

In some embodiments information is sent in one direction only, namely, from the master archive to the update server, thereby enhancing security of the master archive. In addition, in some embodiments all transmitted information is encrypted,



ForcePoint LLC
EX-1009
Page397

CRC (cyclic redundancy code) checked, compressed, digitally signed, and downloaded 308 over a 128-bit SSL connection. The SSL connection employs a secure network protocol that validates and confirms the authenticity of the master archive as the patch source. Other secure network protocols may also be used. In other embodiments, some of these elements are omitted, e.g., no CRC check is done and/or no digital signature is used, etc.

The update server 528 acts as the patch source for client target computers 500. The update server, which contains the replication service and administrative tools for managing updates and software packages, can scan clients 500 and schedule patch deliveries to them using protocols such as HTTP, HTTPS, and XML. In some embodiments, the update server uses Microsoft's Internet Information Services. The update server can be implemented to automatically cache critical updates it receives from the master archive. In some embodiments the administrator can set a replication schedule, can trigger replication manually, or can have the replication software in the update server replicate and distribute software automatically in response to expected or measured network inactivity.

In some embodiments, administrators can create software packages 554, which they can then deploy like any other patch. That is, a "patch" in the general sense need not presuppose a previously installed close-related piece of software that is being modified, but may comprise a piece of software new to the target. For example, a package containing Microsoft Office 2000 could be deployed to every desktop. Administrators of custom applications can similarly create packages to rollout custom applications and their patches. Some embodiment administrators may also utilize built-in software distribution features to distribute any software packages to any target computer.

In some embodiments the update server 528 is configured with software and/or hardware which displays an enterprise report matrix or other summary of the patch status of the machines in a corporation or other enterprise. The report is displayed to a network administrator and/or other personnel charged with maintaining the enterprise's computer functionality. The administrator influences (and in some cases totally controls) which updates or packages from the update server are pushed to the clients 500, by setting policies, defining groups, responding to alerts, and/or taking



ForcePoint LLC
EX-1009
Page398

other steps which are discussed here or already familiar. In some embodiments the administrator has full control over the deployment of patches, including control of reboots and the power to set or modify client agent policies.

Patches may be tested internally before they are widely deployed through the enterprise, since a given patch may behave differently in different enterprises. PatchLink.com Corporation ("PatchLink"), which provides commercial software and services for patch management, and which is the assignee of this application and its ancestors, continually researches, tests, and approves patches before they are released by PatchLink. For instance, when a hot fix for the Microsoft W2K (Windows 2000) operating system is released by Microsoft, it may then be installed and tested by PatchLink on two hundred or more different W2K configurations, such as standard W2K, W2K with SQL server, W2K with Office, and W2K with Exchange (marks of Microsoft), and so on, in combination with various service packs and other hot fixes, before it is released by PatchLink to a master archive 567.

In some embodiments, the client agent 508 checks 332 an intranet-hosted update server to determine which updates are needed at the client in question. It reports gathered information, such as the current configuration 700, back to the update server, which creates the report matrix for the administrator. In some embodiments, the administrator specifies and approves patch deployment using a deployment wizard. Administrator-approved updates and packages are downloaded 312 in the background, thereby reducing inconvenience to users of the computers receiving the download, and then auto-installed according to a schedule set by the administrator. Administrator-defined rules can control the behavior of the patch install process.

One embodiment of the present invention provides a proactive service that enables administrators to have the embodiment automatically download 308, 312 and install 510 software packages and updates, such as critical operating system fixes and security patches.

A built-in security feature of some embodiments of the invention uses digital security identification. Before installing 520 a downloaded update on a target 500, this feature verifies the digital certificate, CRC check, compression, and encryption on each file or package. On the update server 528, access to administrative pages and other controls is restricted to authorized administrators. In some embodiments,



ForcePoint LLC
EX-1009
Page399

replication (downloading) of updates uses SSL and the embodiment checks the validity of downloads to the update server; if the SSL certificates do not properly identify a recognized source (e.g., PatchLink.com) then the download fails, and the server sends an email alert to the administrator. In some embodiments, all information in all downloads (master archive to update server, update server to target) is encrypted, CRC checked, compressed, digitally signed, and sent over 128-bit SSL connections only. In other embodiments, these elements are amended (e.g., 40-bit encryption) and/or omitted.

A patch signature 910 feature permits an embodiment to scan the target 500 and determine if the prerequisite(s) for each patch have been met, e.g., by having the agent check for the proper software version and the proper hardware drivers on the target. The patch signature and the patch fingerprinting features may each be used to make a detection report which is viewable in an enterprise report matrix. A workstation inventory feature uses a discovery agent 508 to pinpoint the needed software and hardware drivers for a target computer. The discovery agent may also scan the target for necessary signatures for fingerprints. PatchLink.com has a master archive which now hosts one of the largest automated patch Fingerprinted repositories in the world.

A background download 312 feature in some embodiments provides a secure background transfer service with built-in bandwidth throttling, so the network administrator can decide how the bandwidth should be utilized during large deployments. Some embodiments provide administrators with a configurable agent 508 policy which permits them to define the agent's communication interval and operating hours. For instance, an administrator may set the policy to roll out patches to production servers only between midnight and 2:00 am. In some cases, agents may have more than one policy active at a given time.

A chained installation feature allows an administrator to reduce or minimize repetitive rebooting by using the Microsoft Qchain.exe tool. If multiple updates which require multiple reboots are to be installed 510, the administrator can use the present invention's capabilities in conjunction with Qchain to deploy the updates with fewer reboots; in some cases only a single reboot will be needed. This reduction in reboots increases the uptime of mission critical computers 500 that are being updated.



ForcePoint LLC
EX-1009
Page400

Qchain rearranges the DLLs into an order that will put the latest update in effect. Administrators can choose this option during deployment.

Using a download resumption feature, an embodiment detects interruption 316 of a download, e.g., by a service outage. If the target 500 is a mobile workstation, the user can then simply disconnect it and reconnect it at a different location that is not out of service. If the update server can be accessed (via TCP/IP, for instance), the embodiment will resume its download 312 from at or near the point in the download at which it was interrupted, instead of starting again from the beginning to retransmit the entire package.

A mobile-user support feature allows administrators to deploy patches and software updates to target computers 500 which are not connected to the network when the deployment begins. When a mobile target subsequently connects to the network, the embodiment will automatically scan it and perform the necessary operations to bring that target up to date.

Embodiments feature client agents 508 which communicate with the update server 528 for secure downloads 312. Using agents also permits increased performance and scalability in enterprise-wide embodiments, permitting a single update server to service thousands of clients. The agents can work across firewalls 116, 214, and operate on any computer 500 with a TCP/IP (or other) connection to the enterprise network.

Some embodiments feature support for multi-vendor patches 554, which may also be referred to as "comprehensive patch scanning". The update server 528 is not limited to patches from a single vendor, but instead supports inventive management of patches from multiple vendors. For instance, the update server may coordinate with target agents to scan targets 500 for patch-related security vulnerabilities in software from Microsoft, IBM, Adobe, Corel, Symantec, McAfee, Compaq, WinZip, Citrix, Novell, and many others (marks of the respective companies). This provides a more secure network.

A grouping feature of some embodiments allows administrators to group selected target computers 500 into sets called, e.g., "containers" or "groups". Operations that are applicable to an individual target computer can then also be applied to containers/groups holding a proper subset of the possible target computers,



ForcePoint LLC
EX-1009
Page401

namely, to every target computer 500 (or every suitable target computer in view of patch signatures and fingerprints) belonging to the specified container. This feature facilitates administrator management of deployments, fingerprint reporting, inventory reporting, mandatory patch baseline policy, and/or client agent policies, depending on the embodiment. For instance, each container may have properties that specify its members, its client agent 508 policies, and its mandatory patch baseline policy. Administrators can select individual clients 500, previously-defined client groups, and/or user-defined groups for deployment. In some embodiments computers can be automatically grouped according to the patch(es) they require.

In some embodiments, the administrator can specify Group Managers and delegate limited administrative control to them. From the Group Manager perspective, the view and control of the inventive embodiment is then narrowed to cover only those computers 500 that have been assigned to the managed group by the administrator, all of which preferably use the same update server 528. The administrator can still view and otherwise manage all computers in the network, not merely those in a particular group.

A mandatory patch baseline policy feature in some embodiments permits an administrator to specify a minimal (baseline) configuration for one or more of a network's computers. The embodiment will proactively patch operating systems and/or applications to the organizational standards defined by the baseline policy. Supporting patch policies in an enterprise allows the administrator of an inventive embodiment to set patch policies for his/her company whereby no machine 500 in the company, for instance, can fall below a minimum patch level. For example, if mandatory patch baseline policy for a W2K group includes Microsoft Office 2000, Adobe Acrobat Reader 5.0, and Service Pack 2, then all computers placed in this group (whether placed initially on group definition, or placed later) will have at least those pieces of software installed on them.

A baseline for patches may be associated with a set of computers 500 that is defined by a group (e.g., a user-defined group or an administrator-defined group), or with a set of computers 500 that use a particular operating system (e.g., all W2K computers, regardless of user-or-administrator-defined groups), or with a set of computers 500 that use a particular application (e.g., all computers that use Microsoft



ForcePoint LLC
EX-1009
Page402

Office XP), or with some combination thereof. For example, in some embodiments the administrator could set a baseline policy rule stating that if Microsoft Office XP is installed then the system should automatically patch in Office XP Service Release 1.

When a mandatory patch baseline policy is used, patches 554 that are dropped (removed) from a target 500 by restoring software from a tape backup, mirrored image, or the like, will be automatically reinstalled after the agent 508 determines the new configuration and that configuration is compared 822 (by the client agent and/or the update server) with the baseline required by the policy. Baseline integrity is thus maintained by these embodiments.

A mandatory patch baseline policy can be used according to the invention to perform automated detection of unwanted software and removal of that unwanted software from target computers within a network. The mandatory deployment patch to be applied when unwanted software is detected would be to UNINSTALL the unwanted items. For example, one such patch would be "Uninstall KaZaA" which would detect and remove the KaZaA file sharing application from a corporate network, thereby reducing the risk that corporate employees violate copyright laws during the course of the business day, or that they consume all available network bandwidth for entertainment purposes. With government agencies and other large entities, eliminating popup software and other things that distract users from their assigned duties can be a high priority.

The invention also provides a feature that may be viewed as the logical opposite of mandatory patching to cure vulnerabilities in a network. This logical opposite, which may be termed the "Forbidden Patch" feature, is used to denote a service pack, hotfix or other software that must not ever be installed. Just as the mandatory patch feature is used to auto-fix a vulnerability, the forbidden patch feature is used to prevent the network administrator from installing software that can break an operational configuration. As an example, assume a company has a payroll system that doesn't work with the latest Microsoft Service Pack for Windows2000. If that Service Pack patch is ever deployed manually or automatically to the payroll server(s), the administrator needs to know at once; otherwise nobody gets paid at the end of the week. Some embodiments of the can scan for and detect the presence of "forbidden patches" and alert the administrator. They may also provide rules so that an



ForcePoint LLC
EX-1009
Page403

administrator does not inadvertently deploy a forbidden patch to a machine that should not have that patch installed, regardless of whether the applicable group patching policies say otherwise.

A patch compliance assurance feature in some embodiments provides administrators with the option of locking a set of patches 554 for a particular computer or a group of computers 500. That is, certain patches are required, but in a manner weaker than in the mandatory baseline feature. If an attempt is made to change target 500 configuration in a way that violates the patch requirement, an email alert message 824 is sent to the administrator. For example, several W2K computers may belong to an administrator-defined group of "IIS Servers" which is subject to patch compliance. For security, the embodiment accordingly locks down all operating system patches and all Internet Information Server patches. If at some later point such patches (including without limitation DLLs) are replaced, then the embodiment will send an email alert to the administrator identifying the computer 500 name and/or the modifications done to it. The newly non-compliant computer(s) and the reason(s) for non-compliance – a summary of discrepancies between their configuration and the locked configuration – can be identified. In some cases, this compliance feature may be used by administrators to identify users who install new software or remove existing software from their machine. Note that this compliance locking feature may be used by some embodiments in conjunction with the mandatory patch baseline feature, to automatically patch a target 500 that is non-compliant. When a locked patch or other software component is removed, it is then automatically reinstalled, and the administrator is notified 824 by email.

A service change feature in some embodiments allows administrators to lock down the services provided at client workstations (residing in a group or individually), and to then be informed if a user starts or stops a service item without directly contacting the administrator. As users change and/or attempt to change the status of services on a locked client 500, an email alert 824 is sent to the administrator identifying the computer and the (attempted) service changes.

A hardware change feature in some embodiments allows administrators to lock down the hardware configuration provided at client workstations 500 (e.g., in a group), and to then be informed if a user installs or removes a hardware item from



ForcePoint LLC
EX-1009
Page404

such a workstation without directly contacting the administrator. As users change (or attempt to change) the hardware configuration on a locked client, an email alert is sent 824 to the administrator identifying the computer and the (attempted) hardware changes.

An import/export feature facilitates the updating of computers on networks that are not connected to the Internet, such as highly secure military or government agency computers. Content is transported from the master archive to the target network's update server 528 using a means other than the Internet, such as physically transporting tapes, disks, or other storage media loaded with the content 554 at the master archive, with suitable physical security measures taken during transport. Once the media is accessible to the secure target network's update server 528, the built-in security measures discussed above (encryption, CRC, etc.) can be employed while transmitting the content from the transported media to local storage of that update server. Then that update server can finish updating 304 the secure network's target computers as previously discussed.

A recurring distribution feature in some embodiments facilitates distribution of data or documents 554 that are repeatedly updated, such as an enterprise employee directory or anti-virus definition/data file. One or more such data or document files can be deployed according to a recurring schedule specified by the administrator, to all targets 500, for instance, or to administrator-specified groups or a single target. Other steps, such as recurring server reboots, may also be specified in some cases.

A disaster recovery feature of some embodiments helps administrators recover from system failures such as hard disk crashes or server hardware failures. If an update server 528 fails, the administrator creates another server having the same DNS name as the failed server, and reinstalls the same update server software (with the same serial number if so required) on the new server. Archived, mirrored, or otherwise stored data files 600 used by the embodiment are restored to the new update server as needed. Then the target agents 508 will automatically connect with the new instance of the update server, and normal operations will resume after the target agents provide information (if any) that was lost by the server failure.

An automatic caching feature in some embodiments causes the update server 528 to automatically download and cache in its local update server storage patches



ForcePoint LLC
EX-1009
Page405

554 that are marked as critical, high-priority, and/or security-related. The update server notifies the administrator as to which patches are critical and which are cached, and scans for target computers 500 that need the patch. By contrast, non-critical patches may be cached at the update server only after they are first deployed. Caching the critical and security patches before their initial deployment provides target computers with a readily available source for the patch when the vendor whose software is vulnerable may be overwhelmed by patch requests. During Code Red and Nimda virus attacks, for instance, some users had to wait hours for a connection to the Microsoft web site to get the patches, because of the extremely heavy demand for them. Proactively caching critical and security patches at an inventive update server 528 reduces the risk that operation of target computers 500 will be interrupted or compromised due to a lack of such patches.

Some embodiments have an intelligent multiple patch deployment feature, which matches patches 554 with operating systems, thereby relieving administrators of the need to expressly and fully identify the operating system used on each target computer. For example, assume Microsoft issued a bulletin for its operating systems which specifies different patches 554 for several different operating system platforms. Administrators using this inventive embodiment need only select "Microsoft operating system" for deployment; they can specify target computers 500 regardless of differences in the operating system details of various specified targets. The embodiment compares 820 patch and operating system requirements for compatibility and for the need for a patch, to ensure that the proper patch gets installed on a given target. Thus, the patch for the Microsoft Windows 98 platform will be installed on a target computer that runs the Windows 98 operating system, the patch for the Microsoft NT platform will be installed on a target computer that runs the NT operating system, and so on. This feature speeds up patch deployment by freeing administrators from the need to manually match patches with targets according to the operating systems (or operating system versions, including prior patches) that are involved.

Another feature helps detect applicable patches 554 and manage patch inter-dependencies, thereby helping administrators avoid manually sorting through dozens (or even hundreds) of generally unrelated patches. Instead, the embodiment identifies



ForcePoint LLC
EX-1009
Page406

applicable patches using their metadata, fingerprint, and/or signature data, based on factors such as the operating system involved, the presence (or absence) of other patches, the interdependency of different patches (identifying which patches rely on which other patches to work properly), and the mandatory patch baseline policy (if any). Then the administrator is shown which patches are applicable for the target(s) 500 in question. For example, one embodiment shows IIS patches to administrators only if IIS is installed on a target computer. If used consistently, this feature helps ensure that when a patch is deployed toward a target, that target has the application in question and the patch will install on that target.

As an example of patch interdependencies, on a Microsoft W2K platform one embodiment will recommend Service Pack 2 to the administrator, and once Service Pack 2 is installed it will then recommend a Security Rollup patch, which depends on Service Pack 2. The embodiment reads both the registry and the file information to correctly perform fingerprinting to validate patch 554 identification.

Some embodiments allow an administrator to review a history or log of recent operations, and to also uninstall a patch 554 or portion thereof, and rollback effects of deploying the patch to the network. This allows the administrator to undo a patch installation that has caused problems. Lost user data will not necessarily be recovered, but the usual steps taken by a conventional uninstaller can be taken using a restorer 514, such as deleting a DLL, removing a registry entry, restoring a path or other system variable value, and so forth. In addition, the configuration status particular to the embodiment, such as signatures, fingerprints, alerts, and reports, is updated to reflect the problems encountered and/or the removal of the patch. The administrator can also be notified if the removed patch appears in a patch dependency and/or in the mandatory patch baseline.

Some embodiments have a "directory-neutral" feature, meaning that they are platform neutral and do not require a directory such as Novell's NDS directory or Microsoft's Active Directory product in order to operate. However, some embodiments can integrate with and cooperate with such directories in particular organizations.

Some embodiments operate according to a selective patch feature, under which patches 554 are not automatically installed unless they are required to meet the



ForcePoint LLC
EX-1009
Page407

mandatory patch baseline policy. In some, patches marked as critical and/or security patches are also installed automatically. In such embodiments, other patches are not installed until they administrator selects them and expressly authorizes their installation; this permits administrators to test patches internally within their organization before installing them on the organization's computers. Once the patch is adequately tested, it can be added to the mandatory patch baseline for the group of targets 500 in question, so that it will be automatically installed when needed.

Some embodiments support a security policy patch 554 that prevents applications from running on a target machine 500. This provides a policy-driven way to hook into the target computer's file system and stop a particular file (or multiple files) from executing. This could be implemented by patches that rename the executable/DLL file(s) in question and substitute in place thereof code that does nothing, or code that displays an error message to the user, and/or code that notifies the administrator by email.

Operation of inventive embodiments may be further understood by considering the following example scenarios. In one scenario, as new patches 554 are released by their respective vendors, an update server 528 downloads the corresponding fingerprints from a master archive 567. The embodiment then checks to see if any target computers 500 meet the profile (need the patch in question) by sending the patch's fingerprint to targets for scanning by agents 508. The administrator is notified of the new patch and its potential impact on the network, and a report matrix informs the administrator which targets need the patch and which do not. The administrator selects one or more individual target computers and/or groups, and authorizes deployment. Deployment proceeds as discussed herein. The administrator may set the time of deployment, and decide whether to reboot after the installation.

In a managed data center scenario, the center's administrator creates a patch group from each cluster of data servers. The administrator can test critical updates received from a master archive 567, and then deploy tested patches 554 on network targets, either all at once, or in stages to groups. Agent policies can help the administrator specify the hours of operation for each group.



ForcePoint LLC
EX-1009
Page408

In an embodiment update scenario, the software used by the embodiment is updated by using the embodiment. That is, when a vendor (such as PatchLink.com) provides patches 554 to the software for target agents 508, update servers 528, and/or other embodiment software, those patches can be deployed as discussed herein, using the inventive tools and techniques that would more often be used to deploy patches to operating systems or user applications. For instance, an administrator can select a PatchLink HotFix client patch and deploy it to update client agent software. Client agents may be initially deployed by pushing them to all target computers.

**Implementation notes**

Additional details regarding particular embodiments are provided below. These implementation details are provided in order to err -- if errors are made -- by including too much information rather than including too little. Applicants should not be penalized for being so forthcoming. In particular, the inclusion of details should not be viewed as an assumption or admission that those details, or similar details, or a similar level of detail, are actually required to support the claims ultimately granted. Nor should the inclusion of particular implementation details be misinterpreted by treating as inventors people who simply implemented inventive ideas conceived by others.

Agents

- Microsoft Windows Agents
- NetWare Agent
- Linux Agent
- Java Agent

Package Construction / Package Maintenance

An administrator uses this module to create a package for distribution through the designated Update agents. This package can be a file distribution or a software package, allowing for more flexibility when updating existing installed software, installing new software, file-replication, etc. throughout the designated managed machines.

Below are the steps for proper package creation:



ForcePoint LLC
EX-1009
Page409

1. Enter the Package Specifications

- Package Name – Labels the package throughout the updating process.
- Package Type – When Software Package is selected in the Software Package routines, after the source files for the package are placed in their proper destination sequence, the administrator may immediately finish the package creation (using pre-designated default values for the rest of the options). File Distribution requires the administrator to complete all steps in the package creation routine.
- Operating System - Choose the Operating Systems to which the package can be rolled out. Currently, you may select one operating system per package. These include: Linux , NetWare, Windows 2000/NT, Windows NT, Windows 95/98/ME
- (Optional) Import – Imports a previously exported package. This option is useful for creating the same package for multiple operating systems.

2. Add the Source

- Add File – Adds a file from your local workstation or network location that is reachable.
- Add Dir – Adds a directory from your local workstation or network location that is reachable.
- Add URL – Adds a remote file to the package via well-known protocols. The various types of URLs you can add are: Local File – File://, FTP – ftp://, HTTP – http://, Secure HTTP – https://, Anything else you choose as long as the agent recognizes the protocol (this field is editable).
- Remove – Removes a file from the package.
- Properties – Shows the details of how each file is stored within the update server. Also allows for multiple sources in case one source is busy or slow (due to net lag for example). The agent automatically tries the other sources.
- Import File – Imports a specific list of files from a previously exported package.



ForcePoint LLC
EX-1009
Page410

3. Add the Destination

- Target Computer – A hierarchical tree view of the package file destination. The various default directories shown depend on the operating system for which this package is targeted. The package always displays in the same directory path from which the source files were originally imported (see Step 2). To move the files around simply highlight the directory or file and drag it to its new location.
- Properties – If the directory where the files should install is not displayed, highlight a file and click the Properties button. This displays the base information of where the source file is coming from and an entry field for the destination. Type the new location and click OK and your changes are shown (this may take a while because the paths are reconnected for large package file numbers).
- Export File – Exports a base package to a file (source and destination information) for use later in an import function.

4. Dependencies

- Left Column – A list of existing packages that are ready for rollout (operating system dependent). For example, if you have a Java-based package that must be rolled out to numerous computers, you would select the specific JDK package as your dependency so that the JDK is installed prior to the current package.
- Right Column – The packages placed here (by using the arrow buttons) are the dependencies for your package. Use the + and – buttons to arrange the dependencies in order of importance (most important being the first dependency). Dependencies are processed before your package.
- Asset – If the dependencies are not found the package fails to install. For example, if a Microsoft Office 2000 SR1 package is created, its Asset dependency is Microsoft Office 2000 which must already be installed.
- Install – If the dependencies are not found, install them prior to installing the current package. Using the above example, if MS Office 2000 is not found, it is installed prior to installing the SR1 package.



ForcePoint LLC
EX-1009
Page411

5. Package Settings

- Backup – Backs up any existing package files found on the destination machines. The editable pull-down list contains the most common directories for the operating system in question. If your directory is not found just type it into the list.
- Confidence Level – The default for all new packages is New. The Confidence Level indicates that this package was tested and its performance has determined its confidence level.
- Availability – The default is Available which indicates the package is available for rollouts. Not Available indicates this created package is unavailable for a rollout.

6. Scripts

- There are three types of package scripts you can use: Command Line – The contents of this script are executed as a standard command line. This script is sent after the files are copied to their destinations. Pre-Script – The contents of this script are executed prior to the files being copied onto the machine. Post-Script – The contents of this script are executed after the files are copied onto the machine.

7. System Settings

- Language – Select the languages for which the package is available. The agent then checks that the language is on the machine and that the package matches before the package is installed.
- Processor Type – Select the processor for which the package is available. The agent then checks that the processor is on the machine and that the package matches before the package is installed.

8. Finish – Click Finish to upload the files and assemble the package. When the assembly process ends the button changes from Finish to Done. Click Done to complete the package creation function.

Define a Group / Modify a Group

This module lets an administrator group machines together, making the rollout procedures easier so that a rollout is as easy for one machine as it is for 500 machines.



ForcePoint LLC
EX-1009
Page412

Additionally, an administrator might group machines according to their function or location to make bandwidth utilization more efficient for their network.

1. Group Name – The label designation for the group.
2. Machine List – Select all the machines this group will include. A machine shows up only after the update agent is installed and registered.
3. Finish – After the machines are placed in the group, the Finish button changes to Done. Click Done to complete the group function.

Schedule a Rollout / View Existing Rollouts

The rollout schedule defines the date and time the packages are made available to the designated machines.

1. Choose a Package
   - Package Selection List – Choose a package (only one at this time) to install.
2. Choose Machines
   - Add a Group – This button displays a dialog box showing a list of the available groups. Highlight the groups you wish to deploy then click the OK button.
   - Remove a Group – Highlight the groups you do not want the package rolled out to, then click the Remove a Group button.
   - Add a Machine – This button displays a dialog box showing a list of available machines (with registered update agents on them). Highlight the machines to add then click the OK button.
   - Remove a Machine – Highlight the machines you do not want the package rolled out to, then click the Remove a Machine button.
   - Rollback – Removes the package just installed and returns the backup (if one was designated). This option is available only via View Existing Rollouts.
   - Reapply – Re-installs the package.



ForcePoint LLC
EX-1009
Page413

3. Choose a Rollout Date and Time
   - Calendar – Choose the date for the rollout installation to occur.
   - Time – The time on the server when the package is to be rolled out.
4. Choose Bandwidth and Sequencing
   - Bandwidth – This level determines how much bandwidth on the server downloading of the package will utilize. The minimum value is 30 % and the maximum is 100%.
   - Sequencing – Selecting YES (default value) causes the rollout to go machine by machine throughout the entire rollout process and finish after the last machine is done. If an error occurs anywhere in the rollout process the rollout stops. Selecting NO causes the rollout to install the package on all machines. If an error occurs on one machine, it does not affect the package rollout on another machine.
5. Finish – The rollout is created or updated and is saved after clicking the Done button.

Agent requests will be in the form of HTML Forms using the POST method. Host responses will be well-formed XML 1.0 documents. Most of the returned documents are of such simple structure, a DTD, NameSpace, or Schema will not be included, but they will be syntactically and structurally in compliance with the XML specification. All dates and times are normalized to Coordinated Universal Time (GMT).

This describes the transaction or data flow between the Agent, the requestor, and the Host, the Update Service. All Update transactions will be initiated by the Agent, except for the case where the Host will open, send the agent ID and then close an agreed upon port and protocol at the Agent's IP address to effectively 'Ping' or notify the Agent that it should request a list of work from the host regardless of its request schedule.



ForcePoint LLC
EX-1009
Page414

**First Contact:**

Any Agent needing to converse with the update server 528 service, will always make a request to the designated master site for the /update subdirectory. This subdirectory will be configured to return a '302 Object Moved' and its 'new' location.

As demonstrated in the following example, the agent performs a 'HEAD' request on the /update subdirectory of the www.patchlink.com site.

**Head Request:**
HEAD /update http/1.1

The Host responds that the object is moved, and the new location can be found at the address provided by the Location: header

**InstallShield Agent Registration:**

During the physical installation of the 'update agent', the Administrator will be required to enter some information before the agent is installed. The Admin will be required to enter the Host Name or IP Address, the Account Identifier, a GUID (Globally Unique Identifier), and the User Name and Password that was specified when registering. This data will be sent to the host to validate the ability to install the agent software, and to generate an ID for the agent.

**Agent TaskList**

Once InstallShield has successfully installed the BootStrap Agent software on the computer, it's time for the agent to start working. After the agent resolves the update server 528 host site address, it posts a 'TaskList' request. A 'TaskList' is a simple list of 'Task' items the Admin has scheduled for the Agent to perform.



ForcePoint LLC
EX-1009
Page415

The BootStrap Agent must be able to:

1. Request the initial TaskList.
2. Receive the initial TaskList.
3. Understand the initial TaskList.
4. Download the Full Agent's install file.
5. Run the Agent Install.
6. Report any install problems, if so, continue as instructed
7. Start the full Agent.
8. Poll for new TaskLists
9. Understand SoftPkg IDs and dependencies and download them.
10. Initial "Action Scripts" either by invoking an external Script Engine or by invoking the Script Engine from within the Agent.

The Agent making the initial TaskList request and processing the returned response accomplishes this. For example:

**TaskList Request**

POST ***server_object_returned_in_firstcontact*** http/1.1
Content-Type: text/html
Content-Length: 32

Action=TaskList
&AccountID=AF011203-7A09-4b67-A38E-1CB8D8702A50
&AgentID=D7292F2D-CCFE-46dc-B036-3B318C2952E3
&AgentVer=0.0
&LocalTime=20000628010100
&Status=0

In this request, the Agent's Version is 0.0. This indicates to the host that this is a new installation of the agent and that the host should prepare a 'Task' for the agent that downloads the latest versions of the appropriate agent software. In the following



ForcePoint LLC
EX-1009
Page416

response, this is shown as the first 'Task' - TaskID="C1D50120-FF13-11d3-95B5-000629526438".

Whenever there has been a change to the Agent's policy, the host will include the policy data in the **'TaskList'** – since this is the initial request from the agent, the policy data is included in this response.

**LocalTime** is just that the Local time (NOT GMT). This allows the server to know exactly what time it is on the Agent machine. Format is in YYYYMMDDHHMMSS.

**Status** tells the tasklist processor to just return a simple yes or no status if there are tasks to be done.
Status=0 means to return a normal task list. Status=1 means tell the agent if you have tasks to be done. This allows the agent to come in non-SSL and do a quick check.

**Agent Soft Package Request**

The first task indicates there is a module to be installed. As shown below, the agent requests the detailed installation information from the host:

**Soft Package Request**

POST ***server_object_returned_in_firstcontact*** http/1.1
Content-Type: text/html
Content-Length: nnnn

Action= SOFTPKG
&AccountID=AF011203-7A09-4b67-A38E-1CB8D8702A50
&AgentID=D7292F2D-CCFE-46dc-B036-3B318C2952E3
&AgentVer=0.0
&TaskID=C1D50120-FF13-11d3-95B5-000629526438



ForcePoint LLC
EX-1009
Page417

&PkgID=12340000-1111-0000-0000-000000000000
&LocalTime=20000628010100

Note that in this instance, the Agent's version is 0.0. This indicates to the host that the package to update the Agent software should be included in the TaskList response. This allows the host to dynamically determine when there is a newer version of the agent software that is available and directs the agent to update itself.

The host puts together an "Open Software Distribution" document that details the information the agent will need to be able to complete the task:

LocalTime is just that the Local time (NOT GMT). This allows the server to know exactly what time it is on the Agent machine. Format is in YYYYMMDDHHMMSS.

**Soft Package (All elements)**

A soft package showing all the possible XML components (shows backup).

```
<?xml version="1.0"?>
<!DOCTYPE SOFTPKG SYSTEM
"http://msdn.microsoft.com/standards/osd/osd.dtd">
<SOFTPKG  xmlns:GX="http://www.patchlink.com/standards/osd/update.dtd"
        GX:TaskID="C1D50120-FF13-11d3-95B5-000629526438"
        GX:PkgID="12340000-1111-0000-0000-000000000000"
        Name="12340000-1111-0000-0000-000000000000"
        GX:ReInstall="N" GX:RollBack="N">
  <TITLE>Windows NT update agent</TITLE>
  <IMPLEMENTATION>
    <OS VALUE="win2k"/>
    <OS VALUE="win98"/>
    <DISKSIZE Value="123456"/>
    <CODEBASE>
```



ForcePoint LLC
EX-1009
Page418

<GX:DIR ModuleID="00000104-0000-0000-0000-000000000000">
<GX:Destination>
<GX:URI DateTime="20000415010100">
<GX:URL>FILE://%TEMP%/</GX:URL>
<GX:ACL Attrib="RWXHSMA Name="$OTHER"/>
<GX:ACL Attrib="RWXHSMA" Group="$GROUP"/>
<GX:ACL Attrib="RWXHSMA" Name="$USER"/>
</GX:URI>
</GX:Destination>
</GX:DIR >
<GX:FILE Expand="N" Overwrite="Y" ModuleID="00000100-0000-0000-0000-000000000000">

**Soft Package Status – Success**

The return codes RC and SoftPkgRC are in decimal format. SoftPkgRC denotes the overall completion of the package. Some modules could have been successful (RC=0) but another may have caused the error. If a rollout is attempted with a package that has already been install once then the agent will return (RC=0) for all the modules it installed and return (SoftPkgRC=725003) or 0x000b100b Soft Package already installed.

Upon completion of the task, the agent will update the host with the results:

Request

POST ***server_object_returned_in_firstcontact*** http/1.1
Content-Type: text/html
Content-Length: nnn

Action=Status
&AccountID=AF011203-7A09-4b67-A38E-1CB8D8702A50
&AgentID=D7292F2D-CCFE-46dc-B036-3B318C2952E3
&AgentVer=2.0
&TaskID=C1D50120-FF13-11d3-95B5-000629526438



ForcePoint LLC
EX-1009
Page419

&PKGID=12340000-1111-0000-0000-000000000000
&InstallDate=20000101123456
&SoftPkgRC=0
&SoftPkgRCMsg=Success
&ModuleID=00000100-0000-0000-0000-000000000000
&RC=0
&RCMsg=Success

**File Attributes and ACL's**

This part describes the GX:ACL element found in GX:Destination (GX:URI) element. The attributes in Update are supplied in to the agent in the Super set form defined below.

The problem with doing basic file attributes is that some file systems blur the boundary between Attributes and ACL's. An attribute is the basic ACL of a file and what is defined here is a small cross platform superset. For instance, Windows NTFS has the Read Only attribute flag but it also has the Read ACL. Therefore, if we are going to make generic attribute flags, then we must expect the meaning to be altered somewhat when used as ACL's. Those on the Unix platform will see no difference, except that like other platforms, you would ignore flags that you do not understand or no defined behavior is listed here.

Three default ACL's are defined and act like basic file attributes; $OTHER, $GROUP, $USER.
On Unix the file system all three of these attribute ACL's would use. However, NT, FAT and NetWare would use only $ OTHER for basic file attributes. Any other names in the ACL's element.data will be construed as a name



ForcePoint LLC
EX-1009
Page420

ACL and Attribute flags

| Letter | Short | Definition |
|---|---|---|
| R | Read | Display the file's data, attributes, owner, and permissions |
| W | Write | Write to the file, append to the file. |
| X | Execute | Run the file (if it's a program or has a program associated with it for which you have the necessary permissions) |
| H | Hidden | Hidden file |
| S | System | System file |
| M | Modify | Read, write, modify, execute, and change the file's attributes. |
| A | Archive | File is ready for Archive |

The XML syntax:

This denotes an ACL for a User

<GX:ACL Attrib="RWXHSMA" Name="UserName"/>

ACL for a Group. Note that $GROUP will always use Group=

<GX:ACL Attrib="RWXHSMA" Group="GroupName"/>

**Soft Package - Rollback**

HTTP/1.1 200 OK

Connection: close

Content-Type: text/xml

Content-Length: nnn



ForcePoint LLC
EX-1009
Page421

```
<?xml version="1.0"?>
<!DOCTYPE SOFTPKG SYSTEM
"http://msdn.microsoft.com/standards/osd/osd.dtd">
<SOFTPKG  xmlns:GX="http://www.patchlink.com/standards/osd/update.dtd"
        GX:TaskID="C1D50120-FF13-11d3-95B5-000629526438"
        GX:PkgID="12340000-1111-0000-0000-000000000000"
        Name="12340000-1111-0000-0000-000000000000"
        GX:ReInstall="N" GX:RollBack="Y">
 <TITLE>Windows NT update agent</TITLE>
 <IMPLEMENTATION>
  <DISKSIZE Value="432"/>
  <CODEBASE>
   <GX:FILE Expand="N" Overwrite="Y" ModuleID="00000100-0000-0000-0000-
000000000000">
    <GX:Destination>
     <GX:URI>
      <GX:URL>FILE://%TEMP%/</GX:URL>
      <GX:FILENAME>HelloWorld.txt</GX:FILENAME>
     </GX:URI>
    </GX:Destination>
    <GX:Backup>
     <GX:URI>
      <GX:URL>FILE://%TEMP%/Backup</GX:URL>
     </GX:URI>
    </GX: Backup >
   </GX:FILE>
  </CODEBASE>
 </IMPLEMENTATION>
</SOFTPKG>
```



ForcePoint LLC
EX-1009
Page422

This example rolls back the simple file copy with backup shown above. The agent denotes a rollback from the attribute GX:Rollback="Y" in the SOFTPKG element tag. The GX:Source element tags are not supplied.

The destination MUST contain a filename to restore the backup to. (Note: it is not an error if the backup files do not exist prior to rollback (the destination may not have existed when the package was distributed). It is however, an error if the destination files exists and cannot be deleted prior to the rollback.)

**ProxyGet**

Sometimes an agent may be installed behind a firewall in such a configuration that the agent is only allowed access to the host site. The agent will detect this case when it tries to retrieve a module for a package that is on the vendor's site. When the agent realizes that it cannot establish communications with a standard HTTP get, it can ask the host to retrieve the file on the agent's behalf by using a 'ProxyGet' request – as described below:

**Request**

POST ***server_object_returned_in_firstcontact*** http/1.1
Content-Type: text/html
Content-Length: nnn

Action=ProxyGet
&AccountID=AF011203-7A09-4b67-A38E-1CB8D8702A50
&AgentID=D7292F2D-CCFE-46dc-B036-3B318C2952E3
&AgentVer=2.0
&URL=http://www.Microsoft.com/hotfix/Q12345.exe



ForcePoint LLC
EX-1009
Page423

**ProxyGetStatus**

**Request**

POST ***server_object_returned_in_firstcontact*** http/1.1
Content-Type: text/html
Content-Length: nnn

Action=ProxyGetStatus
&AccountID=AF011203-7A09-4b67-A38E-1CB8D8702A50
&AgentID=D7292F2D-CCFE-46dc-B036-3B318C2952E3
&AgentVer=2.0
&RefID=107045CF06E011D28D6D00C04F8EF8E0

**GetRequest**

POST ***server_object_returned_in_firstcontact*** http/1.1
Content-Type: text/html
Content-Length: nnn

Action=Get
&AccountID=AF011203-7A09-4b67-A38E-1CB8D8702A50
&AgentID=D7292F2D-CCFE-46dc-B036-3B318C2952E3
&AgentVer=2.0
&RefID=107045CF06E011D28D6D00C04F8EF8E0

**HTTP Get**

**Request:**

GET /download/Q12345.EXE http/1.1



ForcePoint LLC
EX-1009
Page424

**Bandwidth Utilization**

**Range specified Get Request:**

HTTP/1.1 allows a client to request that only part (a range of) the response entity be included within the response. HTTP/1.1 uses range units in the Range and Content-Range header fields. An entity may be broken down into subranges according to various structural units.

range-unit = bytes-unit | other-range-unit

bytes-unit = "bytes"

other-range-unit = token

The only range unit defined by HTTP/1.1 is "bytes". HTTP/1.1 implementations may ignore ranges specified using other units. HTTP/1.1 has been designed to allow implementations of applications that do not depend on knowledge of ranges.

Since all HTTP entities are represented in HTTP messages as sequences of bytes, the concept of a byte range is meaningful for any HTTP entity.

Byte range specifications in HTTP apply to the sequence of bytes in the entity-body (not necessarily the same as the message-body). A byte range operation may specify a single range of bytes, or a set of ranges within a single entity.

When the administrator has selected Bandwidth Utilization features, by specifying them in the agent's policy data, the agent will make 'Range' specified Get requests rather than simple Get requests.

Consider the following Agent Profile:



ForcePoint LLC
EX-1009
Page425

<Policy IntervalType="S" Interval="60" Start="000000" End="060000"
Retries="3" BackOff="10%" AlwaysUseProxyGet="Y"
FailAction="Stop" UDPPort="1234" TCPPort="1234"
KeepAliveConns="Y"
DownloadRestartable="Y" DownloadChunkSize="1024"
DownloadWaitSchedule="S" DownloadWaitInterval="10"/>

The following shows a request for the first 1024 bytes of the Q12345.Exe file, and the host's response:

**Request:**

GET /download/Q12345.EXE http/1.1
Range: bytes=0-1023

**ProxyGet Request:**

POST ***server_object_returned_in_firstcontact*** http/1.1
Content-Type: text/html
Content-Length: nnn

Action=Get
&AccountID=AF011203-7A09-4b67-A38E-1CB8D8702A50
&AgentID=D7292F2D-CCFE-46dc-B036-3B318C2952E3
&AgentVer=2.0
&RefID=107045CF06E011D28D6D00C04F8EF8E0
&RangeBytes=0-1023



ForcePoint LLC
EX-1009
Page426

**Response XML Elements**

| Element | **POLICY** |
|---|---|
| Attributes | **IntervalType** – type of time period.<br>• S=Seconds<br>• M=Minutes<br>• H=Hours<br>**Interval** – Number of time periods that agent should check host for Tasklist.<br>**Start** – The time of day the agent should start running and checking for work to do (GMT).<br>**Stop** – The time of day the agent should stop running and checking for work to do (GMT).<br>**Retries** – The number of times to retry a request before applying the Backoff amount.<br>**BackOff** – the amount of IntervalType time to added to the Interval after a failed contact with the host. This may be expressed as a percentage by appending the percent sign (%). **UDPPort**– {nnn} UDP port number used to wake up the Agent.<br>**TCPPort**– {nnn} TCP port number used to wake up the Agent.<br>**TraceLevel** – OFF = 0, INFO = 1, DETAILED = 2, DEBUG = 3<br>**PurgeIntervalType** – type of time period (see IntervalType)<br>**PurgeInterval** – Number of time periods (PurgeIntervalType) that the agent should scan backups and purge those with a time older than the purge interval. |
| Child of | TASKLIST |
| Parent of | |



ForcePoint LLC
EX-1009
Page427

| Element | **TASK** |
|---|---|
| Attributes | **TaskID** – unique task identifier.<br>**PkgID** – The package identifier to be acted upon. |
| Child of | TASTLISK |
| Parent of | |

**Discovery Agent XML tags**

<name> tag - This is the name of the file you want to search for.

- <path> tag - Very versatile. This is the path you want to search for the file in.

<Version> tag - This is the version of the file you are looking for.

<created> tag - This is the date the file was created.

Example <version> > 5/30/2001 12:01:04 PM </version>

Note: This exact date format is preferred.

<Size> tag - This is the size of the file you are looking for. Note: Cannot due < or >

<root> tag - This is the root key to look for the registry entry in.

<Key> tag - This is the key in the registry you are looking for.

<value> tag - this is the value in the key you are looking for.

<Data> tag - this is the data you expecting to find in that key.

<class> tag - You can specify any valid WMI class that makes sense. example win32_services



ForcePoint LLC
EX-1009
Page428

<searchfield> - This is the field that will best determine what wmi entries to look at.

<searchvalue> - This is the value that will best determine what wmi entries to look at.

<checkfield> - This is the field to look in to get the value you are expecting to get.

<checkvalue> - This is the value you are expecting to find.

Example of the <registry> section of the input file.

```
<registry componentid="" reportID="">
        <root> </root>
        <key> </key>
        <value> </value>
        <data> </data>
</registry>
```

**Patch Fingerprint Signature example**

```
<report reportid="22">
    <file componentid="1" reportID="1">
        <name>outlook.exe</name>
        <path></path>
        <version></version>
        <created></created>
        <size>57393</size>
        <root>HKEY_LOCAL_MACHINE</root>
        <Key>SOFTWARE\Microsoft\Windows\CurrentVersion\App
Paths\OUTLOOK.EXE</key>
        <value>Path</value>
    </file>
</report>
```



ForcePoint LLC
EX-1009
Page429

The Above example will find the outlook Path from the registry and then will validate its size.

**Summary**

The invention provides systems, methods, and configured storage media for assuring that software updates are needed, and that the computers have the necessary software and hardware components, then updating the software across a network with little or no need for human oversight, without requiring copies of the software patches on an administrative machine on the network whose clients are being updated, and which removes the updates from the affected machines, leaving them in a usable state when a problem is discovered during installation or after installation with an installed patch.

As used herein, terms such as "a" and "the" and item designations such as "update server" are inclusive of one or more of the indicated item. In particular, in the claims a reference to an item means at least one such item is required. When exactly one item is intended, this document will state that requirement expressly.

The invention may be embodied in other specific forms without departing from its essential characteristics. The described embodiments are to be considered in all respects only as illustrative and not restrictive. Headings are for convenience only. The claims are part of the specification which describes the invention. The scope of the invention is, therefore, indicated by the appended claims rather than by the foregoing description. All changes which come within the meaning and range of equivalency of the claims are to be embraced within their scope.

What is claimed and desired to be secured by patent is:



ForcePoint LLC
EX-1009
Page430

## CLAIMS

1. An automated method for updating software in a system having a first target computer in a non-update state connected across a network to an update server in a pre-update state, the system also having a package computer which may be inaccessible to the first target computer and is accessible to the update server, and a repository component accessible to the first target computer and the update server, the method comprising the steps of:

putting at least one patch fingerprint which defines a specific software update into the repository component;

gathering information about the first target computer;

comparing at least a portion of the gathered information with the patch fingerprint to determine if the specific software update is absent from the target computer;

placing at least one task identifier on an update task list, the task identifier specifying the first target computer, the task identifier also specifying at least one download address which references a location on the package computer that contains a software update for the first target computer;

in response to the task identifier, downloading the software update from the package computer to the update server; and

performing a second download of the software update from the update server to the first target computer.

2. The method of claim 1, further comprising the step of providing a patch definition file which is portable and which can be employed to replicate a patch on update servers in a plurality of networks.

3. The method of claim 1, wherein the method operates proactively by performing the download steps without requiring an express administrator command to perform them.



ForcePoint LLC
EX-1009
Page431

4. The method of claim 1, wherein the method operates proactively by caching a marked patch at the update server before deploying the patch to target computers, the patch marked as at least one of critical, high-priority, and security-related.

5. The method of claim 1, further comprising at least two steps from the following group of security steps: utilizing encryption to secure patch downloads; utilizing cyclic redundancy codes to secure patch downloads; utilizing digital signatures to secure patch downloads; utilizing a secure network protocol such as SSL to secure patch downloads, wherein at least one of the security steps is available in the particular method embodiment.

6. The method of claim 1, wherein the step of downloading the software update from the update server to the first target computer is performed using a background downloading process, thereby reducing inconvenience to a user of the first target computer.

7. The method of claim 1, wherein the step of downloading the software update from the update server to the first target computer is performed using bandwidth-throttled downloading, thereby allowing a network administrator to decide how bandwidth should be employed during a large deployment.

8. The method of claim 1, wherein downloading is performed subject to a policy which limits the hours of operation, and the policy is set by an administrator, thereby allowing the administrator to decide when patch deployments are allowed to occur.

9. The method of claim 1, further comprising preventing downloads of software updates from the update server to the package computer, thereby enhancing security of the package computer.

10. The method of claim 1, wherein the method further comprises use of a chained installation feature permitting an administrator to have downloaded patches installed on the target computer with fewer reboots than would otherwise be required.



ForcePoint LLC
EX-1009
Page432

11. The method of claim 1, wherein the method further comprises use of a download resumption feature which detects interruption of a downloading step and then after a reconnection resumes the downloading step from at or near the point in that downloading step at which the interruption occurred, thereby avoiding repetition of the entire downloading step to achieve the download.

12. The method of claim 1, wherein the method further comprises use of a mobile-user support feature which allows an administrator to deploy a patch to the first target computer even though the first target computer is not connected to the network when the task identifier placing step occurs.

13. The method of claim 1, wherein the method comprises downloading multiple patches which originated from multiple vendors.

14. The method of claim 1, wherein the method further comprises the step of grouping a proper subset of target computers to form a group, whereby an operation that is applicable to an individual target computer can also be applied to the group.

15. The method of claim 14, wherein the grouping step forms a group containing target computers that are specified by an administrator.

16. The method of claim 14, wherein the grouping step forms a group containing target computers that are specified by a non-administrative user.

17. The method of claim 14, wherein the grouping step forms a group containing target computers that are specified by identifying an operating system that is used by all of the target computers which are being placed in the group.

18. The method of claim 14, wherein the grouping step forms a group containing target computers that are specified by identifying an application program that is used by all of the target computers which are being placed in the group.



ForcePoint LLC
EX-1009
Page433

19. The method of claim 14, wherein the method further comprises the step of delegating limited administrative control to a group manager, whereby the group manager receives control over only those target computers that were placed in the group by the grouping step.

20. The method of claim 1, wherein the method further comprises use of a mandatory patch baseline policy which specifies at least in part software that should be installed on the first target computer, and the method proactively downloads and installs on the first target computer a patch that is specified in the mandatory patch baseline policy.

21. The method of claim 20, wherein the mandatory patch baseline policy sets a baseline for target computers that use a particular application.

22. The method of claim 20, wherein the mandatory patch baseline policy mandates removal of unwanted software from a target computer.

23. The method of claim 1, wherein the method further comprises use of a forbidden patch feature which specifies software that should not be installed on the first target computer, and the method attempts to prevent such installation from occurring.

24. The method of claim 20, wherein the method further comprises automatically reinstalling a patch that is specified in the mandatory patch baseline policy after software in the patch was dropped from a target computer that is subject to the mandatory patch baseline policy.

25. The method of claim 1, wherein the method further comprises the steps of grouping a proper subset of target computers to form a group, and using a mandatory patch baseline policy to specify at least in part software that should be installed on the target computers in the group.



ForcePoint LLC
EX-1009
Page434

26. The method of claim 1, wherein the method further comprises use of a patch compliance assurance feature which specifies software that is locked on the first target computer, and the method proactively notifies an administrator if locked software is removed from the first target computer.

27. The method of claim 1, wherein the method further comprises use of a change control feature which specifies at least one item that is locked on the target computer, and the method proactively notifies an administrator if a locked item is changed on the target computer, wherein the item is at least one of: a hardware item, a service item, and a software item.

28. The method of claim 1, wherein at least the step of downloading the software update from the update server to the first target computer recurs, thereby repeatedly updating a particular file on at least the first target computer.

29. The method of claim 1, further comprising at least one step from a group of disaster recovery steps, the step helping an administrator recover and continue operation after a system failure, wherein the group of disaster recovery steps comprises: creating another server with the same domain name as a failed server, reinstalling update server software on a server, restoring archived data, and restoring mirrored data, and wherein at least one of the disaster recovery steps is available in the particular method embodiment.

30. The method of claim 1, further comprising the steps of maintaining a record of recent operations, and rolling back deployment of a patch, thereby allowing an administrator to undo a target computer patch installation that has caused problems.

31. The method of claim 1, wherein the method further comprises use of a intelligent multiple patch deployment feature which matches patches with target computer operating systems, thereby relieving an administrator of the need to expressly and fully identify the operating system used on the target computer.



ForcePoint LLC
EX-1009
Page435

32. The method of claim 1, wherein the method installs a security patch on the first target computer, thereby providing an administrator with a policy-driven way to hook into the target computer's file system and stop at least one particular file from executing on the target computer.

33. A configured program storage medium having a configuration that represents data and instructions which will cause at least a portion of a computer system to perform method steps of an automated method for updating software in the system, the system having a first target computer in a non-update state connected across a network to an update server in a pre-update state, the system also having a package computer which is inaccessible to the first target computer and is accessible to the update server, and a repository component accessible to the first target computer and the update server, the method comprising the steps of:

putting at least one patch fingerprint which defines a specific software update into the repository component;

gathering information about the first target computer;

comparing at least a portion of the gathered information with the patch fingerprint to determine if the specific software update is absent from the target computer;

placing at least one task identifier on an update task list, the task identifier specifying the first target computer, the task identifier also specifying at least one download address which references a location on the package computer that contains a software update for the first target computer;

in response to the task identifier, downloading the software update from the package computer to the update server; and

performing a second download of the software update from the update server to the first target computer.

34. The configured storage medium of claim 33, wherein the method further comprises the step of providing a patch definition file which is portable and which can be employed to replicate a patch on update servers in a plurality of networks.



ForcePoint LLC
EX-1009
Page436

35. The configured storage medium of claim 33, wherein the method operates proactively by performing the download steps without requiring an express administrator command to perform them.

36. The configured storage medium of claim 33, wherein the method operates proactively by caching a marked patch at the update server before deploying the patch to target computers, the patch marked as at least one of critical, high-priority, and security-related.

37. The configured storage medium of claim 33, wherein the method further comprises at least two steps from the following group of security steps: utilizing encryption to secure patch downloads; utilizing cyclic redundancy codes to secure patch downloads; utilizing digital signatures to secure patch downloads; utilizing a secure network protocol such as SSL to secure patch downloads, wherein at least one of the security steps is available in the particular method embodiment.

38. The configured storage medium of claim 33, wherein the step of downloading the software update from the update server to the first target computer is performed using a background downloading process, thereby reducing inconvenience to a user of the first target computer.

39. The configured storage medium of claim 33, wherein the step of downloading the software update from the update server to the first target computer is performed using bandwidth-throttled downloading, thereby allowing a network administrator to decide how bandwidth should be employed during a large deployment.

40. The configured storage medium of claim 33, wherein downloading is performed subject to a policy which limits the hours of operation, and the policy is set by an administrator, thereby allowing the administrator to decide when patch deployments are allowed to occur.



ForcePoint LLC
EX-1009
Page437

41. The configured storage medium of claim 33, wherein the method further comprises preventing downloads of software updates from the update server to the package computer, thereby enhancing security of the package computer.

42. The configured storage medium of claim 33, wherein the method further comprises use of a chained installation feature permitting an administrator to have downloaded patches installed on the target computer with fewer reboots than would otherwise be required.

43. The configured storage medium of claim 33, wherein the method further comprises use of a download resumption feature which detects interruption of a downloading step and then after a reconnection resumes the downloading step from at or near the point in that downloading step at which the interruption occurred, thereby avoiding repetition of the entire downloading step to achieve the download.

44. The configured storage medium of claim 33, wherein the method further comprises use of a mobile-user support feature which allows an administrator to deploy a patch to the first target computer even though the first target computer is not connected to the network when the task identifier placing step occurs.

45. The configured storage medium of claim 33, wherein the method comprises downloading multiple patches which originated from multiple vendors.

46. The configured storage medium of claim 33, wherein the method further comprises the step of grouping a proper subset of target computers to form a group, whereby an operation that is applicable to an individual target computer can also be applied to the group.

47. The configured storage medium of claim 46, wherein the grouping step forms a group containing target computers that are specified by an administrator.



ForcePoint LLC
EX-1009
Page438

48. The configured storage medium of claim 46, wherein the grouping step forms a group containing target computers that are specified by a non-administrative user.

49. The configured storage medium of claim 46, wherein the grouping step forms a group containing target computers that are specified by identifying an operating system that is used by all of the target computers which are being placed in the group.

50. The configured storage medium of claim 46, wherein the grouping step forms a group containing target computers that are specified by identifying an application program that is used by all of the target computers which are being placed in the group.

51. The configured storage medium of claim 46, wherein the method further comprises the step of delegating limited administrative control to a group manager, whereby the group manager receives control over only those target computers that were placed in the group by the grouping step.

52. The configured storage medium of claim 33, wherein the method further comprises use of a mandatory patch baseline policy which specifies at least in part software that should be installed on the first target computer, and the method proactively downloads and installs on the first target computer a patch that is specified in the mandatory patch baseline policy.

53. The configured storage medium of claim 52, wherein the mandatory patch baseline policy sets a baseline for target computers that use a particular application.

54. The configured storage medium of claim 52, wherein the method further comprises automatically reinstalling a patch that is specified in the mandatory patch baseline policy after software in the patch was dropped from a target computer that is subject to the mandatory patch baseline policy.



ForcePoint LLC
EX-1009
Page439

55. The configured storage medium of claim 33, wherein the method further comprises the steps of grouping a proper subset of target computers to form a group, and using a mandatory patch baseline policy to specify at least in part software that should be installed on the target computers in the group.

56. The configured storage medium of claim 33, wherein the method further comprises use of a patch compliance assurance feature which specifies software that is locked on the first target computer, and the method proactively notifies an administrator if locked software is removed from the first target computer.

57. The configured storage medium of claim 33, wherein the method further comprises use of a change control feature which specifies at least one item that is locked on the target computer, and the method proactively notifies an administrator if a locked item is changed on the target computer, wherein the item is at least one of: a hardware item, a service item, and a software item.

58. The configured storage medium of claim 33, wherein at least the step of downloading the software update from the update server to the first target computer recurs, thereby repeatedly updating a particular file on at least the first target computer.

59. The configured storage medium of claim 33, wherein the method further comprises at least one step from a group of disaster recovery steps, the step helping an administrator recover and continue operation after a system failure, wherein the group of disaster recovery steps comprises: creating another server with the same domain name as a failed server, reinstalling update server software on a server, restoring archived data, and restoring mirrored data, and wherein at least one of the disaster recovery steps is available in the particular method embodiment.

60. The configured storage medium of claim 33, wherein the method further comprises the steps of maintaining a record of recent operations, and rolling back



ForcePoint LLC
EX-1009
Page440

deployment of a patch, thereby allowing an administrator to undo a target computer patch installation that has caused problems.

61. The configured storage medium of claim 33, wherein the method further comprises use of a intelligent multiple patch deployment feature which matches patches with target computer operating systems, thereby relieving an administrator of the need to expressly and fully identify the operating system used on the target computer.

62. The configured storage medium of claim 33, wherein the method installs a security patch on the first target computer, thereby providing an administrator with a policy-driven way to hook into the target computer's file system and stop at least one particular file from executing on the target computer.



ForcePoint LLC
EX-1009
Page441

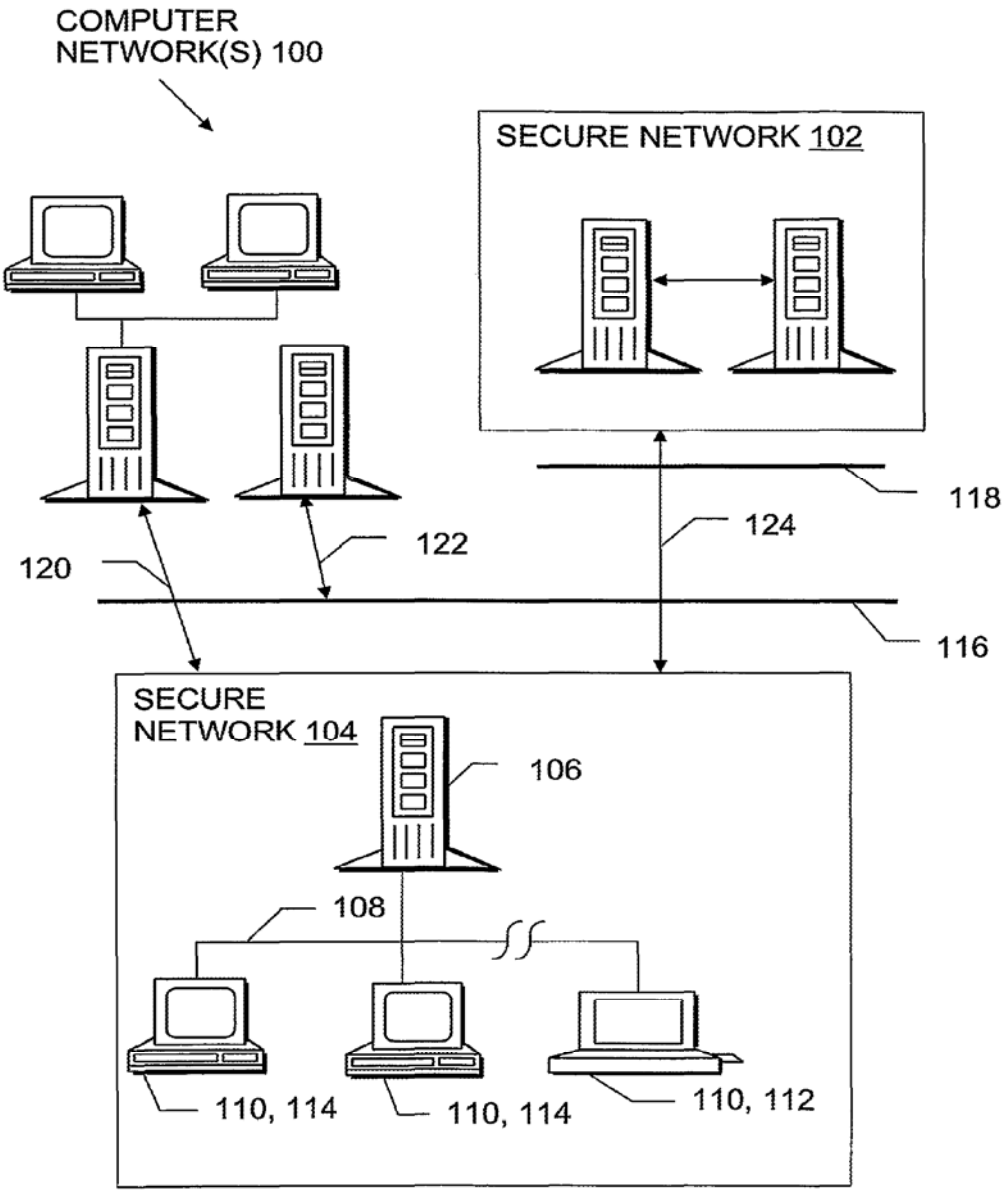

COMPUTER
NETWORK(S) 100
SECURE NETWORK 102
118
124
122
120
116
SECURE
NETWORK 104
106
108
110, 114
110, 114
110, 112

Fig. 1

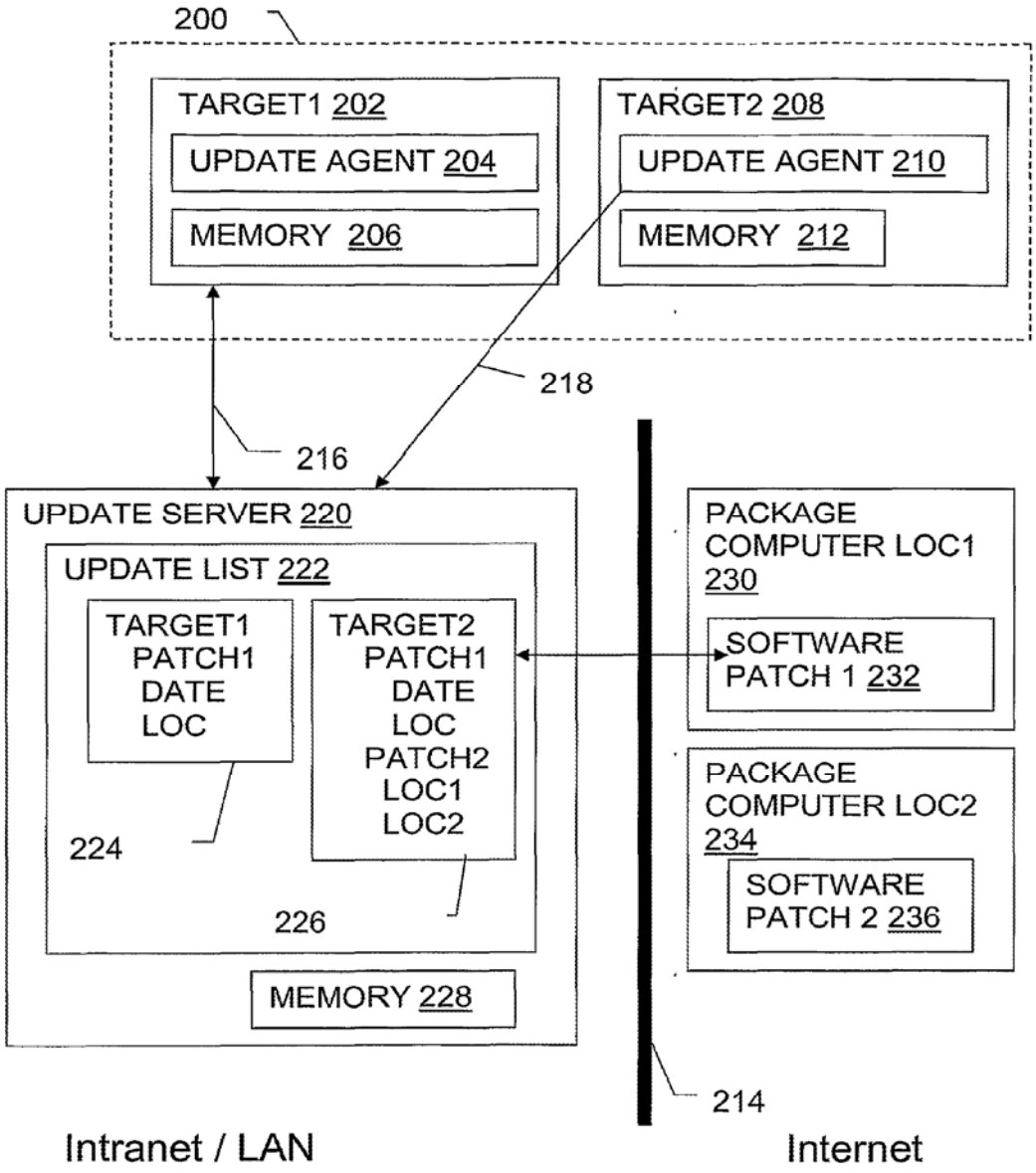

200
TARGET1 202
UPDATE AGENT 204
MEMORY 206
TARGET2 208
UPDATE AGENT 210
MEMORY 212
218
216
UPDATE SERVER 220
UPDATE LIST 222
TARGET1
PATCH1
DATE
LOC
TARGET2
PATCH1
DATE
LOC
PATCH2
LOC1
LOC2
224
226
MEMORY 228
PACKAGE
COMPUTER LOC1
230
SOFTWARE
PATCH 1 232
PACKAGE
COMPUTER LOC2
234
SOFTWARE
PATCH 2 236
214
Intranet / LAN
Internet

Fig. 2

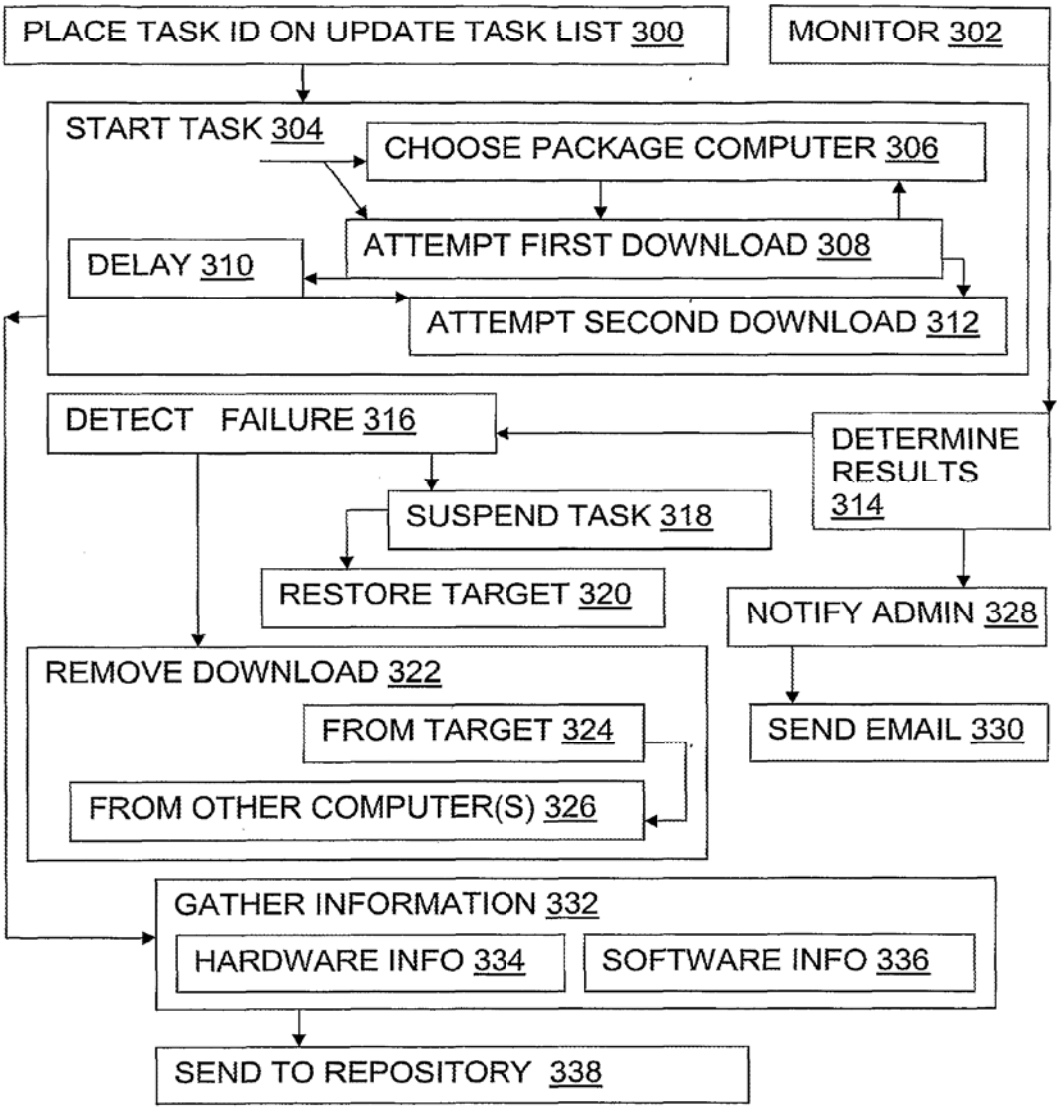

PLACE TASK ID ON UPDATE TASK LIST 300
MONITOR 302
START TASK 304
CHOOSE PACKAGE COMPUTER 306
ATTEMPT FIRST DOWNLOAD 308
DELAY 310
ATTEMPT SECOND DOWNLOAD 312
DETECT FAILURE 316
DETERMINE RESULTS 314
SUSPEND TASK 318
RESTORE TARGET 320
NOTIFY ADMIN 328
REMOVE DOWNLOAD 322
FROM TARGET 324
SEND EMAIL 330
FROM OTHER COMPUTER(S) 326
GATHER INFORMATION 332
HARDWARE INFO 334
SOFTWARE INFO 336
SEND TO REPOSITORY 338

Fig. 3

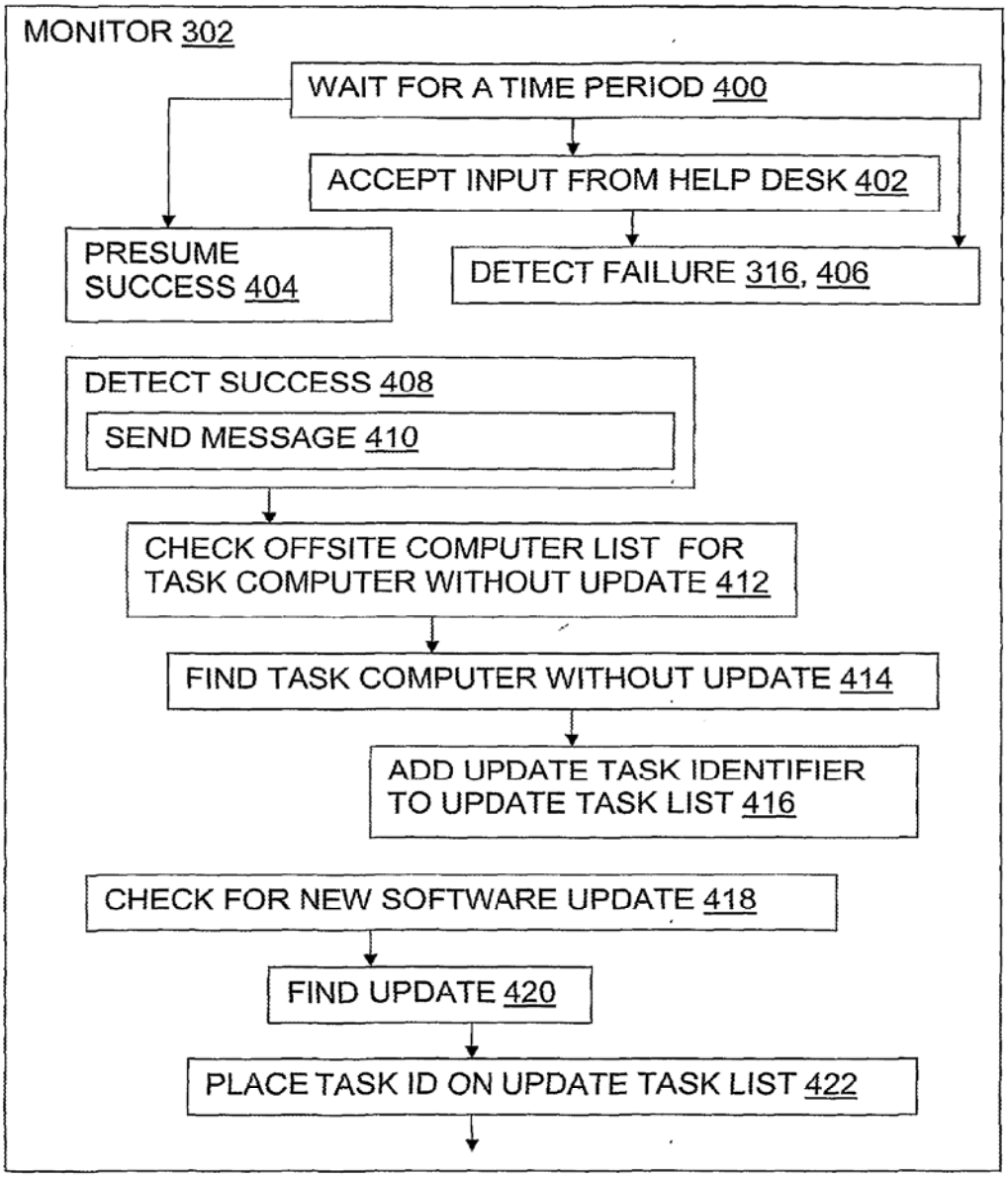

MONITOR 302
WAIT FOR A TIME PERIOD 400
ACCEPT INPUT FROM HELP DESK 402
PRESUME SUCCESS 404
DETECT FAILURE 316, 406
DETECT SUCCESS 408
SEND MESSAGE 410
CHECK OFFSITE COMPUTER LIST FOR TASK COMPUTER WITHOUT UPDATE 412
FIND TASK COMPUTER WITHOUT UPDATE 414
ADD UPDATE TASK IDENTIFIER TO UPDATE TASK LIST 416
CHECK FOR NEW SOFTWARE UPDATE 418
FIND UPDATE 420
PLACE TASK ID ON UPDATE TASK LIST 422

Fig. 4

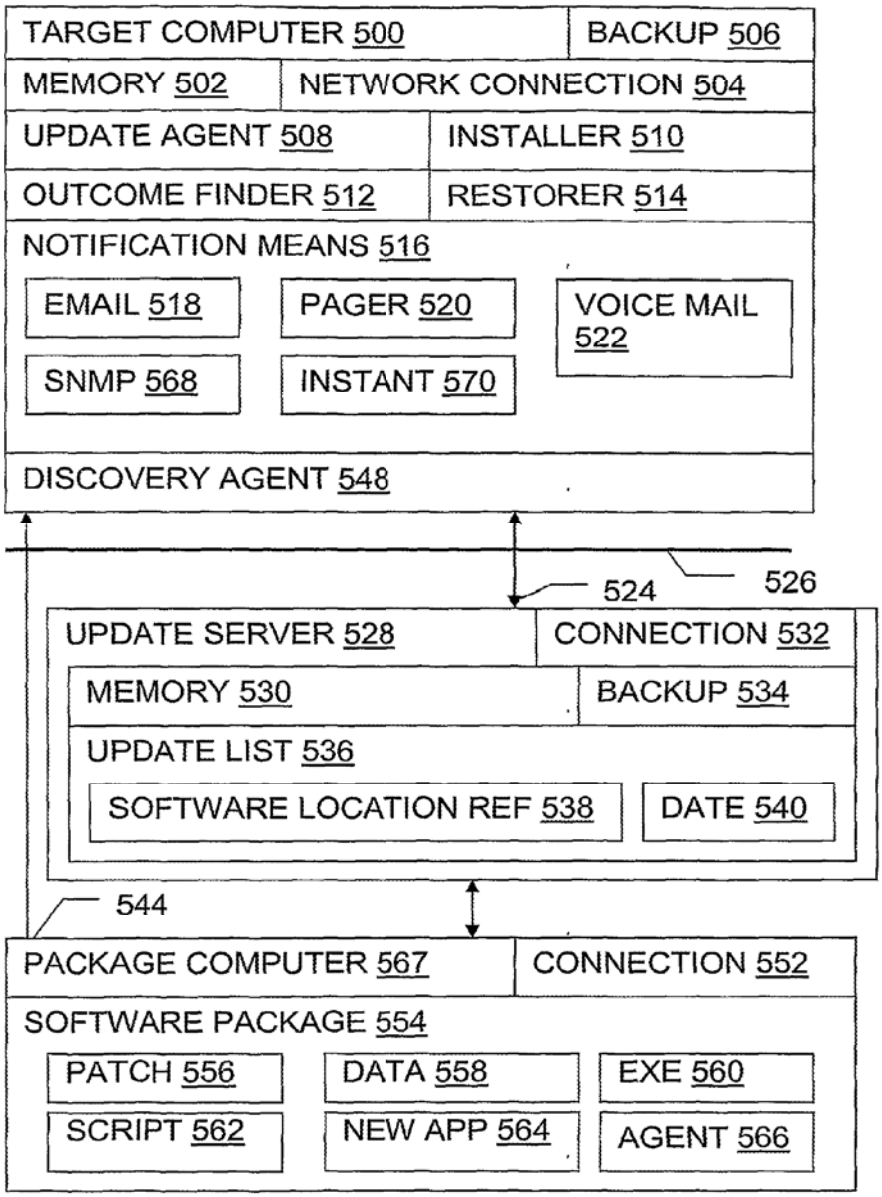

TARGET COMPUTER 500
BACKUP 506
MEMORY 502
NETWORK CONNECTION 504
UPDATE AGENT 508
INSTALLER 510
OUTCOME FINDER 512
RESTORER 514
NOTIFICATION MEANS 516
EMAIL 518
PAGER 520
VOICE MAIL 522
SNMP 568
INSTANT 570
DISCOVERY AGENT 548
524
526
UPDATE SERVER 528
CONNECTION 532
MEMORY 530
BACKUP 534
UPDATE LIST 536
SOFTWARE LOCATION REF 538
DATE 540
544
PACKAGE COMPUTER 567
CONNECTION 552
SOFTWARE PACKAGE 554
PATCH 556
DATA 558
EXE 560
SCRIPT 562
NEW APP 564
AGENT 566

Fig. 5

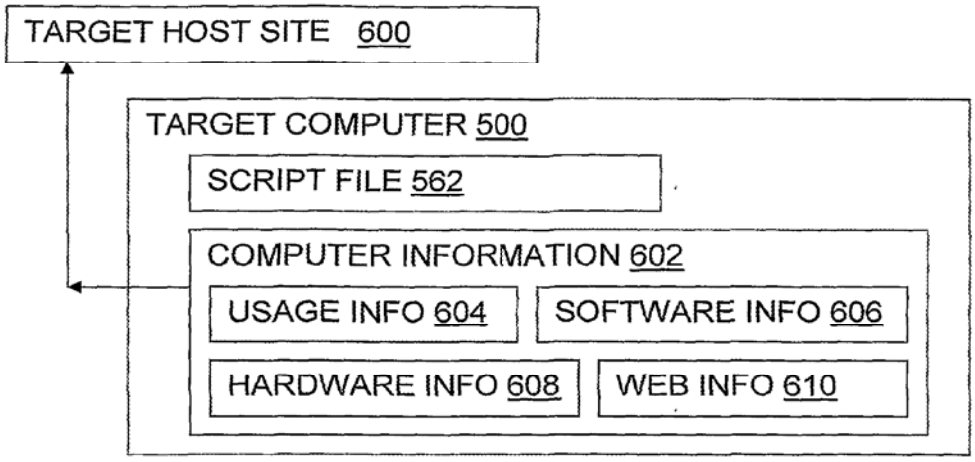

TARGET HOST SITE 600
TARGET COMPUTER 500
SCRIPT FILE 562
COMPUTER INFORMATION 602
USAGE INFO 604
SOFTWARE INFO 606
HARDWARE INFO 608
WEB INFO 610

Fig. 6

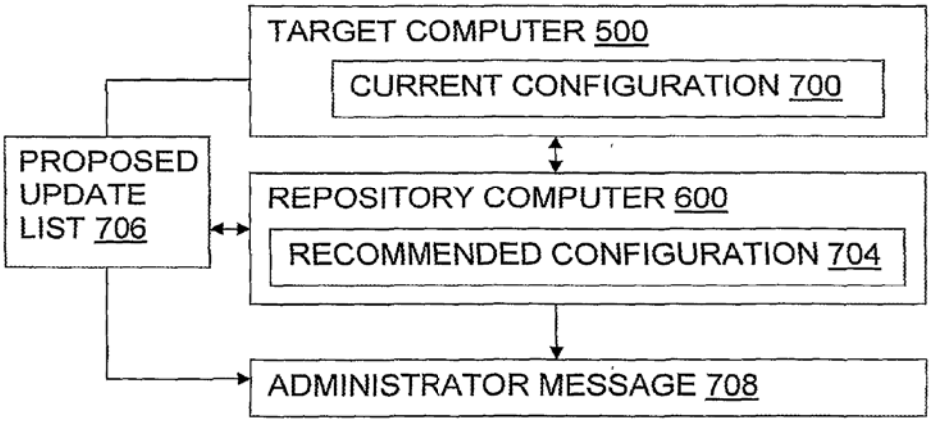

TARGET COMPUTER 500
CURRENT CONFIGURATION 700
PROPOSED UPDATE LIST 706
REPOSITORY COMPUTER 600
RECOMMENDED CONFIGURATION 704
ADMINISTRATOR MESSAGE 708

Fig. 7

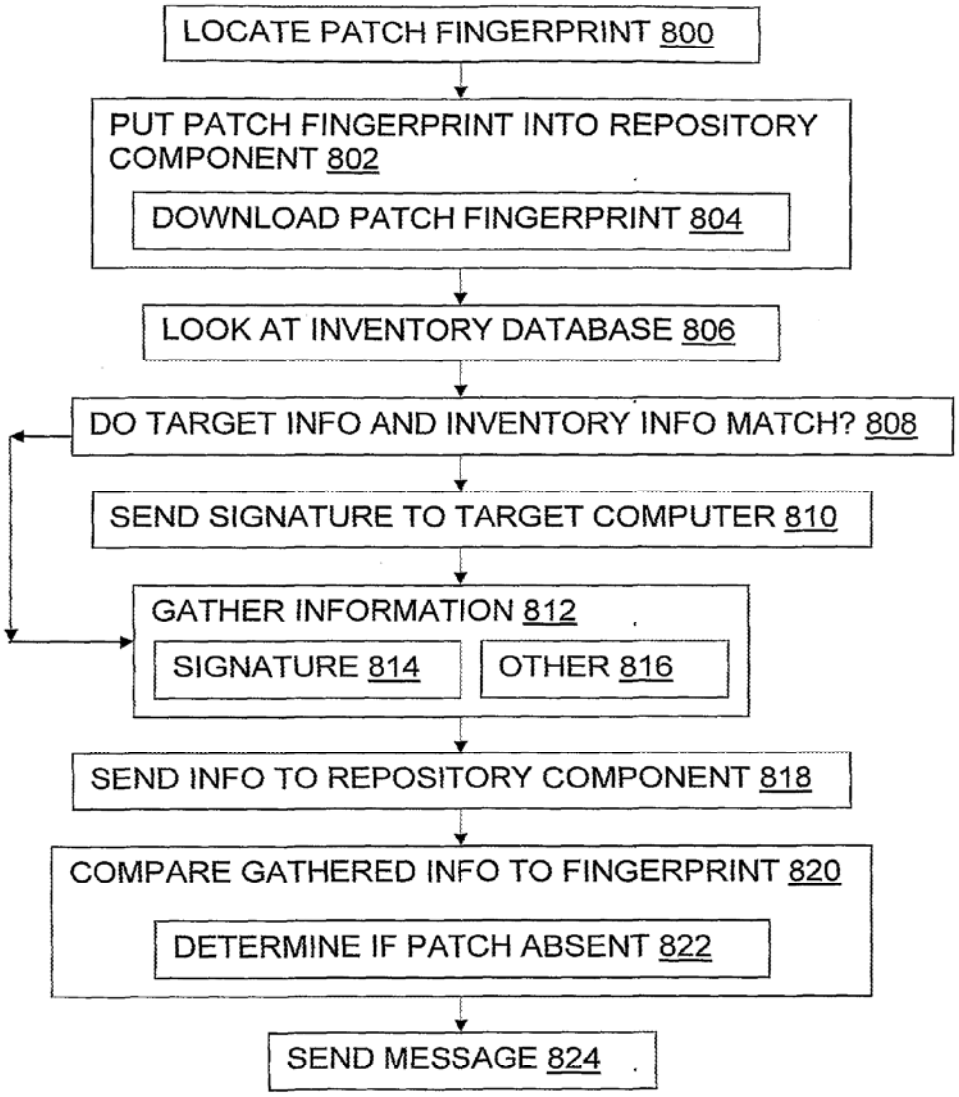

LOCATE PATCH FINGERPRINT 800
PUT PATCH FINGERPRINT INTO REPOSITORY COMPONENT 802
DOWNLOAD PATCH FINGERPRINT 804
LOOK AT INVENTORY DATABASE 806
DO TARGET INFO AND INVENTORY INFO MATCH? 808
SEND SIGNATURE TO TARGET COMPUTER 810
GATHER INFORMATION 812
SIGNATURE 814
OTHER 816
SEND INFO TO REPOSITORY COMPONENT 818
COMPARE GATHERED INFO TO FINGERPRINT 820
DETERMINE IF PATCH ABSENT 822
SEND MESSAGE 824

Fig. 8

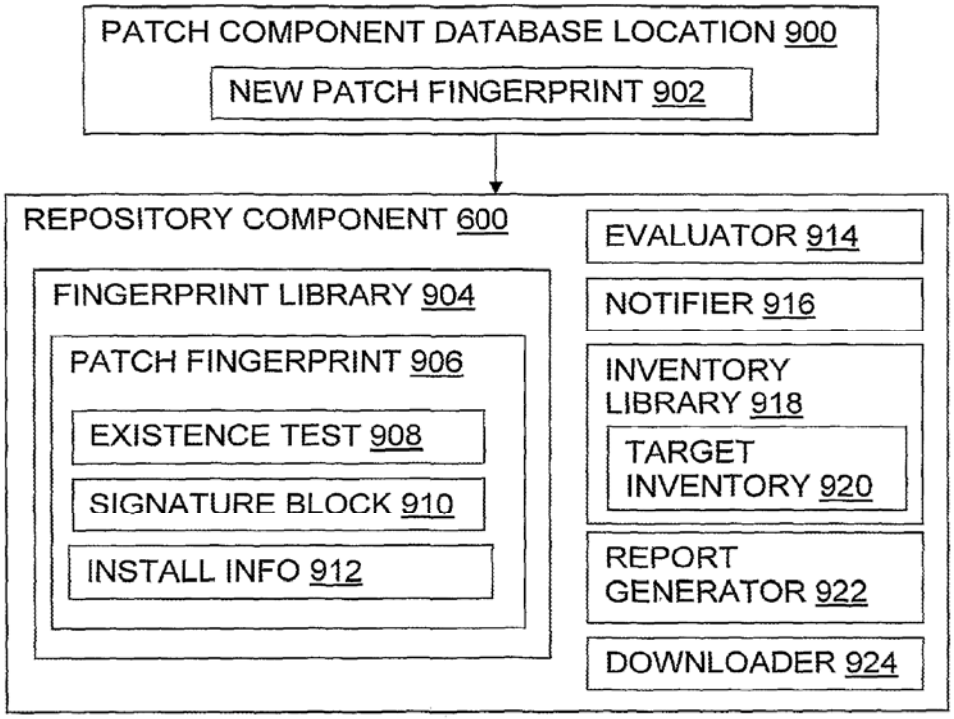

PATCH COMPONENT DATABASE LOCATION 900
NEW PATCH FINGERPRINT 902
REPOSITORY COMPONENT 600
FINGERPRINT LIBRARY 904
PATCH FINGERPRINT 906
EXISTENCE TEST 908
SIGNATURE BLOCK 910
INSTALL INFO 912
EVALUATOR 914
NOTIFIER 916
INVENTORY LIBRARY 918
TARGET INVENTORY 920
REPORT GENERATOR 922
DOWNLOADER 924

Fig. 9

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
29 December 2004 (29.12.2004)

PCT

(10) International Publication Number
WO 2004/114663 A1

(51) International Patent Classification[7]: H04N 7/16

(21) International Application Number:
PCT/GB2004/002570

(22) International Filing Date: 16 June 2004 (16.06.2004)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
0314043.1 17 June 2003 (17.06.2003) GB

(71) Applicant *(for all designated States except US)*: **IMAGINE BROADBAND LIMITED** [GB/GB]; Carmelite, 50 Victoria Embankment, Blackfriars, London EC4Y 0DX (GB).

(72) Inventor; and

(75) Inventor/Applicant *(for US only)*: **PATEL, Dipan**; c/o Imagine Broadband Limited, Aldwych House, 81 The Aldwych, London WC2B 4HN (GB).

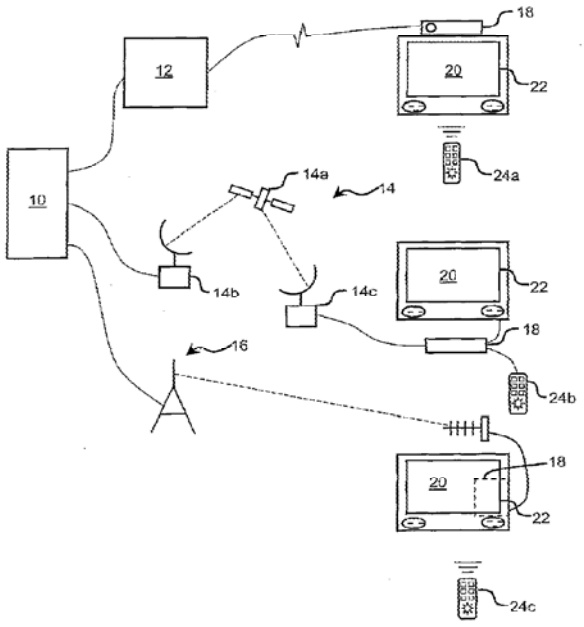
(74) Agents: **KAZI, Ilya** et al.; Mathys & Squire, 100 Grays Inn Road, London WC1X 8AL (GB).

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

*[Continued on next page]*

(54) Title: METHOD AND SYSTEM FOR SELECTIVELY DISTRIBUTING DATA TO A SET OF NETWORK DEVICES




(57) Abstract: A method selectively distributes data to a set of devices linked by a network, where each device has at least one unique identifier. The method comprises selecting devices to be members of the set, the set devices being selected to receive the data based on the at least one unique identifier, and distributing data to at least the selected devices, the data including at least one matching key for matching at least a portion of the unique identifier of selected devices. The unique identifier is a device hardware identifier, being a string of bits and the key correlates to at least a portion of the string of bits.

WO 2004/114663 A1

ForcePoint LLC
EX-1009
Page450

FR, GB, GR, HU, IE, IT, LU, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

**Published:**

— *with international search report*

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments*

*For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.*

# METHOD AND SYSTEM FOR SELECTIVELY DISTRIBUTING DATA TO A SET OF NETWORK DEVICES

## Technical Field

The present invention relates to a method and system for selectively distributing data to a set of network devices and will be described in relation to selectively modifying instruction code of devices of subscriber television networks. However, it will be appreciated that the invention is not limited to this particular field of use, but is also applicable to other systems of data processing equipment of which mobile telephone networks and computer networks are further non-limiting examples.

## Background Art

Networks, such as subscriber television service networks, are typically controlled and run by complex computer networks, where end users access the networks' programming via "set top boxes" (STB) connected to their television, and control the programming using a remote control device. STBs may be used merely for accessing different audiovisual entertainment channels on the end user's television, or for datacasting, internet access, pay per view programs, video telephony, and so on. These STBs typically include flash memory for storing data such as software code to process information sent by the subscriber network, and to provide to the connected television a front end graphical user interface (GUI) for the user to access some functions of the software. Due to the changing and improving nature of the services provided by such networks, the STB software may be updated from time to time.

However, updating software often causes problems. For example, even though new software is thoroughly tested under workshop conditions, unforeseen problems often occur once the software is deployed to live STBs; or functionality of the software may change where end users require additional services to learn how to use the changed software. Since the updating of software is not a common event, the service network operators are typically not resourced to cope with calls from users for help with dealing with the changes or problems. For this reason, it is desirable for software updates to be deployed to a trial batch of users to detect problems before mass installation. This is often achieved in practice by using employees (and perhaps friends of employees) to trial the updated software for a

ForcePoint LLC
EX-1009
Page452

period of time to determine the software's robustness in the field. However, this may not provide a large enough or representative sample of the user base.

Furthermore, as equipment may come from a number of sources over a long period of time, there may be numerous versions of software employed by devices. It is not normally practical to maintain the complex database that would be required to track existing software versions.

Similar problems occur when rolling out computer software updates and patches for computer systems. Typical methods of providing updates and patches include using the internet or CD-ROMs distributed with trade magazines and the like.

Preferred embodiments of the present invention aim to ameliorate one or more of the problems of the prior art or to provide an alternative thereto.

**Summary of the Invention**

According to one aspect of the invention there is provided a method of selectively distributing data to a set of devices linked by a network, each said device having at least one unique identifier, the method comprising the steps of:

selecting devices to be members of the set, the set devices being selected to receive the data based on the at least one unique identifier; and

distributing data to at least the selected devices, the data including at least one matching key for matching at least a portion of the unique identifier of selected devices.

This method is advantageous in that the distribution of data is controllable and flexible. Preferably the data is broadcast to a plurality of the devices on the network but only selected devices selectively install the data. The data to be distributed may include updates or improvements to code for running the devices.

In a complementary method aspect, there is provided a method of selectively installing data at one of a set of devices linked by a network, each device having a unique identifier, the method comprising determining whether to install the data based on matching at least a portion of the unique identifier to at least one received matching key associated with the data.

Preferably, the unique identifier is independent of the content of the data, and most preferably is a device hardware identifier. Surprisingly, although device hardware identifiers do not have a logical connection with the decision whether or not to update software, and indeed because of this, they can provide a useful means of

selecting subsets of a population of devices.

Preferably, the set of devices comprises one of a plurality of substantially arbitrarily partitioned subsets of a population of devices. Alternatively, the set of devices may be randomly selected, or selected by one or more predetermined geographic regions common to the set devices.

Preferably, the data is instruction code, preferably a software update.

In one embodiment, the unique identifier may be a string of characters.

Preferably the identifier is a string of bits or characters, the string being preferably of a first length and the set of devices preferably determined from a shorter length subset of the string of the first length. Typically, the first length is at least about 32 bits. The shorter length subset of the string preferably has a selectable length. By using a selectable length bit string to define the set is advantageous, in that the number of matching devices to be updated can be adjusted. Alternatively the shorter length subset may be a fixed length, such as 16 bits.

Preferably, the step of installing the data is performed after performing one or more predetermined action steps on each device to determine whether or not to install the data. The step(s) may include determining whether a controller of the network has the data and whether the data includes a key or flag corresponding to the device, the key or flag preferably corresponding to the shorter length subset of the longer, first length device identifier.

The action step may be performed only when a device initialisation instruction is performed by the device, or when switching to a power on or standby state. Alternatively, the action step may be performed periodically, or at regular intervals while the device is in communication with the network, or when a user of the device performs a manual check of the network while the device is in communication with the network.

The method may include a confirmation step, after the one or more predetermined action steps, of a user of the device performing a confirmation instruction to download and run the data, if the device has been determined as a member of the set of devices.

Preferably, the one or more predetermined action steps include the steps of:

determining the version of a device data of each respective set device;

comparing the version of the device data with the version of the data to be relayed; and

ForcePoint LLC
EX-1009
Page454

determining whether or not to perform the step of downloading the data, based on the outcome of the step of comparing the versions of the device and modification data. Preferably, the predetermined action steps are performed by the device. Preferably each device is arranged to run a manual update routine for allowing the user to decide whether to download data flagged as user selectable from the networks wherein the manual update routine runs a text routine to determine whether to download data flagged as user selectable. Also preferably the test routine comprises comparing the matching key to the unique identifier.

According to a third aspect of the invention there is provided a device for running data, the device being linked to other devices by a network, the device including:

a processor;

a memory with stored data processable by the processor; and

at least one unique identifier,

wherein the data stored by the memory includes a routine for checking for update data for the device, and for selectively downloading the data based on at least a portion of the unique identifier, and at least one key associated with the update data. Preferably, the data includes the key or flag and the device selectively downloads the update data when the key correlates to the at least the portion of the unique identifier. Also preferably the unique identifier is a device hardware identifier, and is a string of bits of characters.

Preferably, the devices are set top boxes, and the network is a subscription television service. The set of devices may be selected by determining one or more channels subscribed by and common to users of the respective set devices. In this way, an operator of the subscription television service can send data to users, or subscribers of particular channel packages, such as movie channel packages, or sport channel packages. Examples of such data may include a broadcast message to the subscribers about a particular relevant event on a particular channel, or code for allowing preview access for a limited time period to another channel to which the user does not have a subscription.

Alternatively, the devices may be mobile telephones or telephone base stations and the network may be a mobile telephone network.

Alternatively, the devices may be computers and the network may be a computer network. In this embodiment, the computer network is preferably one or

ForcePoint LLC
EX-1009
Page455

more of a local area network, a wide area network or the Internet.

In another alternate arrangement, the devices may be back to base home security devices and the network may be a back to base security network.

According to a fourth aspect of the invention there is provided a system for selectively distributing data to of a set of devices linked by a network, each said device having at least one unique identifier, the system comprising:

means for selecting devices to be members of the set, the set of devices being selected to receive the data based on respective unique identifiers of the devices;

means for distributing data to at least each selected device, the data including at least one matching key for matching at least a portion of the unique identifier of selected devices; and

means for running the data on each respective selected device.

According to a fifth aspect of the invention there is provided a method of updating a device which has an update routine which checks for available updates over a network and selects available updates which are applicable to the device for download, each update having associated therewith an update condition selected from a plurality of pre-determined conditions which conditions include at least (1) at least one power change condition and (2) a user-selection condition, the device being arranged to execute a user confirmation routine on selection of an available applicable update associated with a user-selection condition, the method comprising executing a routine which performs a test to determine whether a selective update criterion is met in place of the user confirmation routine and which automatically opts to install the available applicable update if the selective update criterion is met.

This method is advantageous where an operator of the network needs to perform a "retrofit" of a new code or data routine to devices already on the network.

Preferably the device is arranged to execute the routine which performs the test to determine whether a selective update criterion is met. Also preferably performing the test comprises comparing additional update data associated with the available applicable update to device data obtained from the device, and the additional update data. Preferably the additional update data comprises matching key data and the device data comprises a unique identifier of the device. Also preferably performing the test comprises detecting at least a partial match between said matching key data and said unique identifier. The extent of match required for a positive test result may be specified in the additional updated data.

ForcePoint LLC
EX-1009
Page456

According to a sixth aspect of the invention there is provided a method of providing updated data to a population of devices, the method comprising partitioning the population of devices into subsets, making the updated data available to a first subset, and subsequently making the data available to further subsets, characterised by monitoring the response following making available to the first subset and setting the size of further subsets and/or the rate of making data available to further subsets based on the response.

The size of further subsets may be increased in the event of a low level of response.

Preferably the subsets are defined by specifying a portion of a matching key to match to a unique identifier of each device. The size of each subset may be determined by setting the length of the portion of matching key to match, wherein a shorter matching portion will match a larger number of devices.

Preferably, a longer portion of matching key is used to define a smaller first subset and a shorter portion of matching key is used to define at least one larger further subset.

Preferably, monitoring the response comprises obtaining a measure of response based on user feedback.

According to another aspect of the invention there is provided a computer program or computer program product for selectively distributing data of a set of devices of a network comprising means for performing the method aspects of the invention.

In the specification, unless otherwise stated or apparent from the context, references to STBs are intended to encompass any form of decoder which receives (preferably digital) signals from a medium (satellite, cable, DSL, terrestrial being non-limiting examples) and provides (typically analogue, but digital output is not excluded) video signals for a display device (television, plasma screen, video projector being non-limiting examples). Whilst stand-alone STBs are commonplace, the functionality of the STB may be integrated into a display device (for example a television with built in digital decoding) or other device (for example cable modem or other communication device).

Typical STBs in current use only allow of the order of about 8 megabytes for applications and only have a processor capable of the order of 100 MIPS (million instructions per second) which is an order of magnitude less than typical personal

ForcePoint LLC
EX-1009
Page457

computers. However, the architecture is applicable to other devices, such as hand-held devices (Personal Digital Assistants, or PDA being a non-limiting example), mobile devices, dedicated games or entertainment devices and even in fact to general purpose computers where it is desired to provide an efficient processing architecture.

As will be understood, the term "matching" in the context of the specification and claims is preferably to be understood as a direct match where, for a match to occur, the matching key is identical to a portion of the unique identifier.

**Brief Description of the Drawings**

Preferred embodiments of the invention will now be described, by way of example only, with reference to the accompanying drawings, in which:

Figure 1 is a schematic overview of a system for employing the present invention;

Figure 2 is a schematic view of the architecture of a STB for employing the present invention; and

Figure 3 is flow diagram of a preferred embodiment of the present invention.

**Detailed Description of the Preferred Embodiments**

In the following description, an embodiment is described in the context of a device in the form of a STB for decoding video signals, which is a particularly preferred application. However, the architecture (and portions thereof) may be applied to other network systems, for devices such as mobile telephone devices, handheld devices, games platforms, general purpose computers, GPS automobile security systems, back to base building security systems, or indeed any device having a processor, memory and input and output functions.

Referring to Fig. 1, a preferred embodiment of a method, system, device and computer program or computer program product according to the invention may comprise at least one network controller in the form of a server 10 connected via a communication medium such as a digital subscriber link (DSL, or cable) headend 12, a satellite link 14, a terrestrial transmitter 16, or other communication link (not shown) to devices in the form of STBs 18. The satellite link typically includes a satellite 14a and transmitting and receiving dishes 14b and 14c respectively. The STBs 18 provide video signals to a display device such as a television 20 to be displayed on a

ForcePoint LLC
EX-1009
Page458

screen 22. The STBs 18 receive input signals, for example from an infra-red remote control 24a, a direct coupled joystick or input device 24b or, in the case of a STB 18a integrated with a television 20a, from a master control 24c for the television.

In normal use, the STB 18 decodes video signals received over the communication medium to provide video output, as is well-known in the art. The STBs normally include limited functionality for providing a graphics display, for example to display static configuration and programme menus. Alternatively, the graphics capability is made use of to provide real-time moving video (although the invention can be used to provide static video as well), for example to enable a user to play an arcade-style game.

The architecture of the STB 18 which makes this feasible will be explained with reference to Figure 2. As is well-known, a STB 18 generally has a hardware platform 26 which provides a variety of interfaces 27, including a graphics interface 28, a network interface 29, an input interface 30 and normally a sound interface (not shown). A variety of standard drivers 31 including a graphics driver 32 and input driver 34 are provided for controlling the interfaces 27. The hardware has a central processing unit (CPU) 36 and memory 38, such as flash memory or other applicable memory types, which runs the software 40, including a real time operating system (RTOS) 42.

In the preferred embodiment of the invention, each STB has a unique hardware identifier being a string of a first length in the form of at least a 32 bit linear string, resulting in a maximum of $2^{32}$, or approximately 4 billion unique identifiers for the STB population.

In alternative embodiments, the unique identifier is in the form of an 8 bit x 8 bit (1 byte x 1 byte), 2 dimensional array string, or is in the form of a 64 bit linear string, each resulting in a STB population maximum of $2^{64}$ unique identifiers. Such a configuration would ensure no two STBs share the same unique identifier. Other embodiments of STBs may have unique identifiers with larger or smaller bit strings, or with other strings, such as alphanumeric strings. The identifier may not be truly globally unique in certain cases.

In another alternative embodiment, each STB 18 may be supplied a downloaded unique identifier, such as a subscriber number, to be written to its memory.

In yet another alternative embodiment, the unique identifier may be a whole or

ForcePoint LLC
EX-1009
Page459

part hash value of a predetermined portion of data on the STB 18. The predetermined portion of data may include user details held by the memory 38, or data relating to the user's subscription of service that they are using the STB for.

A preferred embodiment of a method according to the present invention will now be described with reference to Figures 1 to 3. The server 10 sends multiple data streams at different frequencies relating to different channels throughout the network to connected STBs. A channel may be for broadcasting audiovisual data of popular television programs, or alternatively for data streaming of teletext information, the Internet or pay-per-view programs. Typically, television programs and teletext are continuously broadcast, whereas the Internet is interactive, allowing uploading as well as downloading of data. Pay-per-view programs may be "on demand", requested by the user to be broadcast, continuously broadcast, but restricted to access by relevant subscribers.

Depending on a subscription type or level of a STB user, each respective STB contains software code to decode the data streams or channels to which the user has subscribed. In addition to subscribed channels, the Internet, etc, the server 10 may from time to time send data streams of STB data in the form of software code updates on one or more channels, where the software relates to a change in user subscription, or a change to the functionality of the STB, such as its graphical user interface. It is the method and system by which the latter data stream is transmitted which is the principal concern of the preferred embodiment of the present invention.

Prior to sending software code updates, the modification or updated code is normally tested by the network operator for robustness. Depending on the nature of the updated code, the server will roll out the code, in accordance with the preferred embodiment, to sets of subscribers over time. This has many advantages. For example, depending on the change to functionality of the STB made by the new code, network users will invariably contact the network operator with questions regarding the use of the STB. Therefore, in spreading out the roll out of updated code over time, the impact on the network operator's resources for helping the subscribers is also spread out over time, ensuring a more consistent use of such resources. Or, if there are problems with updated code, the number of updated STBs affected by the update is controlled to update a manageable portion of network subscribers. The rollout can then be stopped, and new update code repaired.

Prior to rolling out the updated code, the server 10 determines a 16-bit string

ForcePoint LLC
EX-1009
Page460

to be a flag included with the updated code, and then streams 50 the update code on a preselected channel. As illustrated in Figure 3, after occurrences of a STB trigger 52, resident software code on the STB tells the STB to run a check 54 of the server 10 to see if there are any code updates where the 16-bit string flag matches the first 16 bits of the unique identifier of the STB. If there is no match, the STB continues to function as normal. If there is a flag match, and the software code is an update of code existing on the STB 18, the STB then checks 56 whether the version of the update code is newer, older or the same as the STB's version of the existing code. If newer, the STB 18 downloads and runs the code update from a predetermined channel where the code is being constantly streamed. During or once the code has been downloaded, the version of the updated code is written to the STB memory 38. Alternatively, the code may be an initial instruction to the STB 18 that another instruction code will be available on a predetermined channel at a future predetermined time, such as a week later for example, and that the STB in question is to download and run the other instruction code from the predetermined channel at the predetermined time. Alternatively, steps 54 and 56 occur simultaneously.

There are three main triggers 52 for the STB to check the network for code updates. The first trigger is on boot or reboot of the STB 18. The second trigger is when, with the STB 18 in use, resident code on the STB 18 runs a timed routine asking the server 10 whether a code update is available containing a matching flag. The third trigger is where the user manually asks the server, via a STB GUI function or physical button on the STB 18 or remote controller 24a-c, whether a code update is available containing a matching flag.

In all the triggers 52 described above, the user may be given the option to confirm the downloading and running of the code update. Alternatively, the STB 18 will automatically download and run the code update without the user's input or knowledge. In an alternative embodiment, the code includes a secondary flag which tells the STB 18 a predetermined future time when the update code will be available for download.

Depending on the number of subscribers, approximately 1 in 65536 ($2^{16}$) subscribers would receive the code with the above method, due to the flag being 16-bit. This would allow the network operator to roll out new or updated code to a set, or portion, of the users, where the set is approximately 1 in 65536. Once feedback as to the success of the take-up of code of the set has been determined, the flag can be

ForcePoint LLC
EX-1009
Page461

changed to another 16-bit string to allow download and running of the code by a different set of users.

As will be appreciated by the skilled person, depending on the set size required, in alternative embodiments the flag may be larger or smaller, as required. This covers the situation where, if the unique identifier is large enough, a single STB can be targeted for downloading of code, where the flag is identical to the unique identifier of the single STB to be targeted. It also covers the situation where the flag is 1-bit, thus targeting approximately half the STB population. In the example of targeting a single STB, this may be used where only a single user requires a code update, for example should an unusual error have occurred in the download and/or running of an update code, or should that user elect to change their subscription service.

The same or shorter length subset of the string may alternatively be in the form of a string having a variable length over the period of the roll out of the new data. For example, the network operator may determine to rollout new data to a 1 in $2^{16}$ set of STBs, by determining a 16 bit flag. If over a predetermined time frame the network operator receives more complaints or query from STB users than it is possible for resources to effectively deal with, the operator may choose to roll out the next stage of the new data to a smaller set portion of the users, say 1 in $2^{18}$ users, or to delay or halt roll out. If over a predetermined time frame the network operator receives less feedback from STB users than a threshold amount, the operator may choose to roll out the next stage of the new data to a larger set portion of the users and advantageously the software automatically chooses the subset, say 1 in $2^4$ users, using a 4 bit flag. This can provide effective control of network resources in dealing with rolling out of new data.

In another alternative embodiment of the invention, STBs 18 are distributed to users such that the first 16-bit string of the unique identifier is determined based on more systematically distributed data, known by the network operator to correspond to a known criteria, such as geographical region or, particularly in the example of STBs used for television subscriptions, preferred program genre. In this way code updates can be rolled out by region or by program preference. This has the advantage that regional or program specific code updates can be targeted to users who fall into appropriate regional or program specific categories.

In reality, there are already many STBs 18 in use in the market place. Many of

these have resident factory installed software code that determines when the STB will check for and download updated data or software code. The STB checks for new code on each boot, reboot/reset, or when manually requested by the user. If, on boot, etc, a newer version of code is available, a download condition is checked to see if the STB is in the correct state to download the new software. The available states for downloading, depending on the needs of the network operator, include: immediate download; on power up or next boot or reset of the STB; when the STB is put into a standby state; or when manually requested by the user.

Manual requesting of new data or software is a feature rarely used, as it is rarely related to the user as being an available feature.

This resident boot code is therefore quite limiting to the network operator who may want or need to have STB data or software updated, but will be unsure as to when the STB is likely to be booted or rebooted/reset to enable initiation of a download.

Therefore, in another embodiment of the invention, there is a method of updating a STB 18 which has an update routine which checks for available updates over the network and selects available updates which are applicable to the STB for download. Each update has associated therewith an update condition selected from a plurality of pre-determined conditions. The conditions include at least:

1. at least one power change condition
   (for example, detection of a switch to or from a standby condition or a hard power-up condition);
2. a user-selection condition; and
3. an immediate installation condition signifying that an available applicable update is to be applied without waiting for a power change condition or user selection.

The STB is arranged to execute a user confirmation routine on selection of an available applicable update associated with a user-selection condition. In this method, the STB is also arranged to execute a routine, in place of the user confirmation routine, the routine performing a test to determine whether a selective update criterion is met. Also, this routine automatically opts to install the available applicable update if the selective update criterion is met. In an alternative

ForcePoint LLC
EX-1009
Page463

embodiment, the test performing routine is arranged to revert to requesting user confirmation in the absence of data specifying the selective update criterion.

Performing the test also comprises comparing a matching key in the form of a flag associated with the available applicable update to the unique identifier of the STB in question. The extent of positive match between the flag and unique identifier is specified in the additional update data. As with previously described embodiments, the unique identifier may be 32 bits, and the flag specified may be a 16 bit flag, to be potentially matched to the first 16 bits of the unique identifier.

This embodiment can be used in conjunction with earlier described embodiments to allow for downloading of new data to a set of STBs, where the set is based on a correlation between the STBs' respective unique identifiers and a key in the new data.

It will be apparent to those skilled in the art that alternative embodiments of the invention are applicable to different types of networks. For example, the invention may be applied to a mobile telephone network. In this embodiment, the server communicates with a plurality of subscribed mobile telephones via landline telephone substations and mobile telephone tower substations. The server can then roll out updated software, code or data using a method corresponding to the method described above with respect to STBs. Similarly, each telephone would preferably have a 2-byte, or 16 bit unique identifier, and there would be three main triggers for a mobile telephone connected to the network to download data, software, etc where the triggers correspond to the three triggers described above in reference to STBs. The first trigger would be turning on the telephone, the second trigger would be a timed routine run by the telephone and the third trigger would be a manual request by the telephone user, via the user interface on the telephone.

Other embodiments suited to the use of the invention include local area computer networks (LANs); wide area computer networks (WANs); computers connected to the Internet; landline (terrestrial) video/telephones; handhelds, or personal digital assistants (PDAs); back to base alarm systems for houses and commercial properties; GPS, satellite or mobile telephone connected back to base car alarm systems, and so on.

While the present invention has been described in its preferred embodiments, it is to be understood that the words which have been used are words of description rather than limitation and that changes may be made to the invention without

ForcePoint LLC
EX-1009
Page464

departing from its scope as defined by the appended claims.

The text of the abstract filed herewith is repeated here as part of the specification.

A method selectively distributes data to a set of devices linked by a network, where each device has at least one unique identifier. The method comprises selecting devices to be members of the set, the set devices being selected to receive the data based on the at least one unique identifier, and distributing data to at least the selected devices, the data including at least one matching key for matching at least a portion of the unique identifier of selected devices. The unique identifier is a device hardware identifier, being a string of bits and the key correlates to at least a portion of the string of bits.

ForcePoint LLC
EX-1009
Page465

**CLAIMS:**

1. A method of selectively distributing data to a set of devices linked by a network, each said device having at least one unique identifier, the method comprising the steps of:
selecting devices to be members of the set, the set devices being selected to receive the data based on the at least one unique identifier; and
distributing data to at least the selected devices, the data including at least one matching key for matching at least a portion of the unique identifier of selected devices.

2. A method according to claim 1 wherein the data is broadcast to a plurality of the devices over the network but wherein only selected devices selectively install the data.

3. A method of selectively installing data at one of a set of devices linked by a network, each device having a unique identifier, the method comprising determining whether to install the data based on matching at least a portion of the unique identifier to at least one received matching key associated with the data.

4. The method of any preceding claim wherein the unique identifier is independent of the content of the data.

5. The method of any preceding claim wherein the unique identifier is a device hardware identifier.

6. The method of any preceding claim wherein the set of devices comprises one of a plurality of substantially arbitrarily partitioned subsets of a population of devices.

7. The method of any of claims 1 to 4 wherein the set of devices is selected by one or more predetermined geographic regions common to the set devices.

8. The method of any preceding claim wherein the data comprises instruction

ForcePoint LLC
EX-1009
Page466

code or a software update.

9. The method of any preceding claim wherein the unique identifier is a string of bits or characters.

10. The method of claim 9 wherein the unique identifier is a string of a first length and the set of devices is determined from a shorter length subset of the string of the first length.

11. The method of claim 10 wherein the first length is at least about 32 bits.

12. The method of claim 10 or 11 wherein the shorter length subset of the string is of a selectable length.

13. The method of claim 2 or any claim dependent thereon wherein the step of installing the data is performed after performing one or more predetermined action steps to determine whether or not to install the data.

14. The method of claim 13 wherein the one or more action steps include determining whether the data includes the key corresponding to the device.

15. The method of any one of claims 10 to 14 wherein the key corresponds to the shorter length subset.

16. The method of any one of claims 13 to 15 wherein the one or more action steps are performed when a device initialisation instruction is performed by the device when the device is switched to a power on state or when the device is switched to a standby state.

17. The method of any one of claims 13 to 15 wherein the one or more action steps are performed periodically or at regular intervals while the device is in communication with the network.

18. The method of any one of claims 13 to 17 wherein the one or more

ForcePoint LLC
EX-1009
Page467

predetermined action steps include the steps of:
determining the version of a device data of a set device;
comparing the version of the device data with the version of the data to be distributed; and
determining whether or not to perform the step of downloading the data to be distributed, based on the outcome of the step of comparing the versions of the device data and the data to be distributed.

19. The method of any one of claims 13 to 18 wherein the predetermined action steps are performed by the device.

20. The method of any one of the preceding claims wherein each device is arranged to run a manual update routine for allowing a user to decide whether to download data flagged as user selectable from the network, wherein the manual update routine is modified so that, in place of user decision, the routine runs a test routine to determine whether to download data flagged as user selectable.

21. The method according to claim 20 wherein the test routine comprises comparing the matching key to the unique identifier.

22. A device for running data, the device being linked to other devices by a network, the device including:
a processor;
a memory with stored data processable by the processor; and
at least one unique identifier,
wherein the data stored by the memory includes a routine for checking for update data for the device, and for selectively downloading the data based on at least a portion of the unique identifier, and at least one key associated with the update data.

23. The device of claim 22 wherein the update data includes the key and the device selectively downloads the update data when the key correlates to the at least a portion of the unique identifier.

ForcePoint LLC
EX-1009
Page468

24. The device of claim 22 or 23 wherein the unique identifier is a device hardware identifier.

25. The device of any one of claims 22 to 24 wherein the unique identifier is a string of bits or characters.

26. The device of claim 25 wherein the unique identifier is a string of a first length and the at least a portion of the unique identifier is determined from a shorter length of the first length.

27. The device of claim 26 wherein the first length is at least about 32 bits.

28. The device of claim 26 or 27 wherein the shorter length subset of the string is of a selectable length.

29. The device of any one of claims 22 to 28 wherein the device is a set top box.

30. The device of any one of claims 22 to 29 wherein the network is a subscription television service.

31. The device of any one of claims 22 to 28 wherein the devices are:
mobile telephones and the network is a mobile telephone network; or
telephone base stations and the network is a telephone network; or
computers and the network is a computer network; or
back to base home security devices and the network is a back to base security network.

32. A system for selectively distributing data to of a set of devices linked by a network, each said device having at least one unique identifier, the system comprising:
means for selecting devices to be members of the set, the set of devices being selected to receive the data based on respective unique identifiers of the devices;

ForcePoint LLC
EX-1009
Page469

means for distributing data to at least each selected device, the data including at least one matching key for matching at least a portion of the unique identifier of selected devices; and

means for running the data on each respective selected device.

33. A method of updating a device which has an update routine which checks for available updates over a network and selects available updates which are applicable to the device for download, each update having associated therewith an update condition selected from a plurality of pre-determined conditions which conditions include at least (1) at least one power change condition and (2) a user-selection condition, the device being arranged to execute a user confirmation routine on selection of an available applicable update associated with a user-selection condition, the method comprising executing a routine which performs a test to determine whether a selective update criterion is met in place of the user confirmation routine and which automatically opts to install the available applicable update if the selective update criterion is met.

34. The method of claim 33 wherein the device is arranged to execute the routine which performs the test to determine whether a selective update criterion is met.

35. The method of claim 33 or 34 wherein performing the test comprises comparing additional update data associated with the available applicable update to device data obtained from the device.

36. The method of claim 35 wherein the additional update data comprises matching key data and the device data comprises a unique identifier of the device.

37. The method of claim 36 wherein performing the test comprises detecting at least a partial match between said matching key data and said unique identifier.

ForcePoint LLC
EX-1009
Page470

38. The method of any of claims 35 to 37 wherein the extent of match required for a positive test result is specified in the additional update data.

39. The method of any of claims 33 to 38 wherein said at least one power change condition includes (a) detection of a switch to or from a standby condition and (b) a hard power-up condition.

40. The method of any of claims 33 to 39 wherein the conditions further include (3) an immediate installation condition signifying that an available applicable update is to be applied without waiting for a power change condition or user selection.

41. The method of any of claims 33 to 40 wherein the routine which performs the test is arranged to revert to requesting user confirmation in the absence of data specifying the selective update criterion.

42. A method of providing updated data to a population of devices, the method comprising partitioning the population of devices into subsets, making the updated data available to a first subset, and subsequently making the data available to further subsets, characterised by monitoring the response following making available to the first subset and setting the size of further subsets and/or the rate of making data available to further subsets based on the response.

43. The method of claim 42 wherein the size of further subsets is increased in the event of a low level of response.

44. The method of claim 42 or 43 wherein the subsets are defined by specifying a portion of a matching key to match to a unique identifier of each device.

45. The method of claim 44 wherein the size of each subset is determined by setting the length of the portion of matching key to match, wherein a shorter matching portion will match a larger number of devices.

ForcePoint LLC
EX-1009
Page471

46. The method of claim 45 wherein a longer portion of matching key is used to define a smaller first subset and a shorter portion of matching key is used to define at least one larger further subset.

47. The method of any of claims 42 to 46 wherein monitoring the response comprises obtaining a measure of response based on user feedback.

48. The method of any one of claims 1 to 20 or 33 to 47 wherein the devices are set top boxes.

49. The method of any one of claims 1 to 20 or 33 to 47 wherein the network is a subscription television service.

50. The method of claim 48 or 49 wherein the set of devices is selected by determining one or more channels subscribed by and common to users of the respective set devices.

51. The method of any one of claims 1 to 20 or 33 to 47 wherein the devices are:
mobile telephones and the network is a mobile telephone network; or
telephone base stations and the network is a telephone network; or
computers and the network is a computer network; or
back to base home security devices and the network is a back to base security network.

52. A computer program or computer program product comprising means for performing the method of any of claims 1 to 20 or 33 to 51.

53. A method of selectively distributing data to a set of devices linked by a network substantially as herein described with reference to the accompanying drawings.

54. A method of selectively installing data at one of a set of devices linked by a network substantially as herein described with reference to the accompanying drawings.

55. A device for running data substantially as herein described with reference to the accompanying drawings.

56. A system for selectively distributing data substantially as herein described with reference to the accompanying drawings.

57. A method of updating a device substantially as herein described with reference to the accompanying drawings.

58. A method of providing updated data to a population of devices substantially as herein described with reference to the accompanying drawings.

59. A computer program or a computer program product substantially as herein described with reference to the accompanying drawings.

ForcePoint LLC
EX-1009
Page473

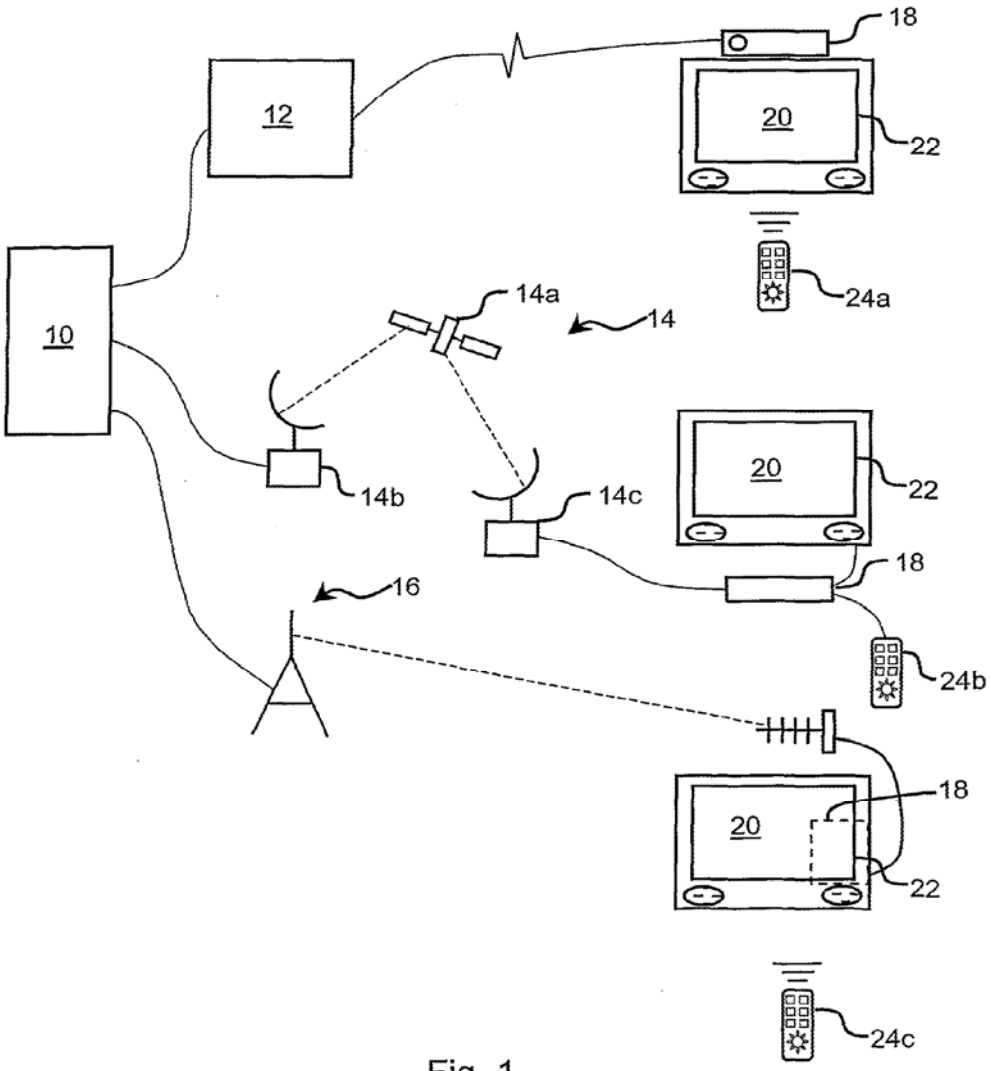



Fig. 1

ForcePoint LLC
EX-1009
Page474

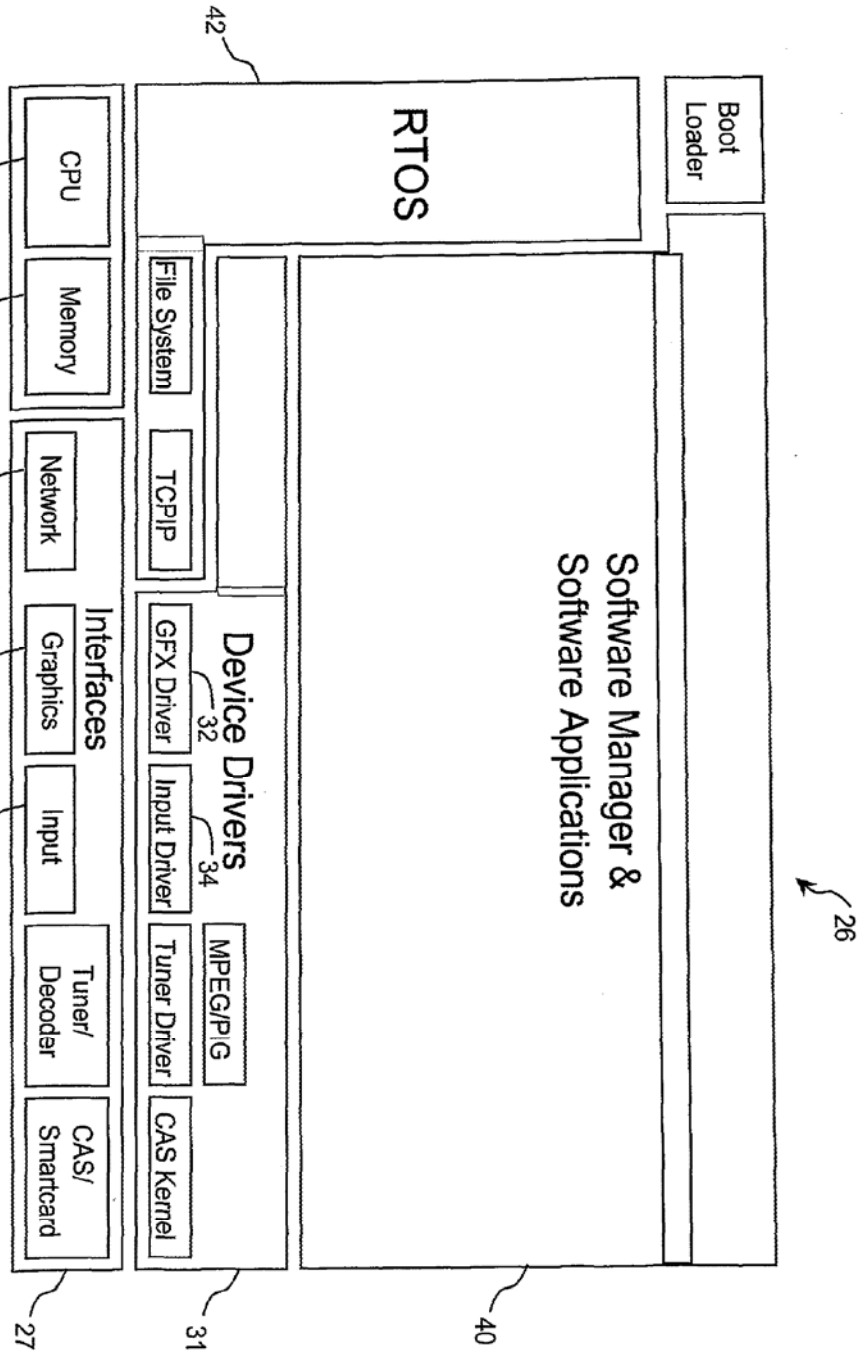

Boot Loader
RTOS
Software Manager & Software Applications
File System
TCP/IP
Device Drivers
GFX Driver
Input Driver
MPEG/PIG
Tuner Driver
CAS Kernel
Interfaces
CPU
Memory
Network
Graphics
Input
Tuner/ Decoder
CAS/ Smartcard
26
27
29
30
31
32
34
36
38
40
42

Fig. 2

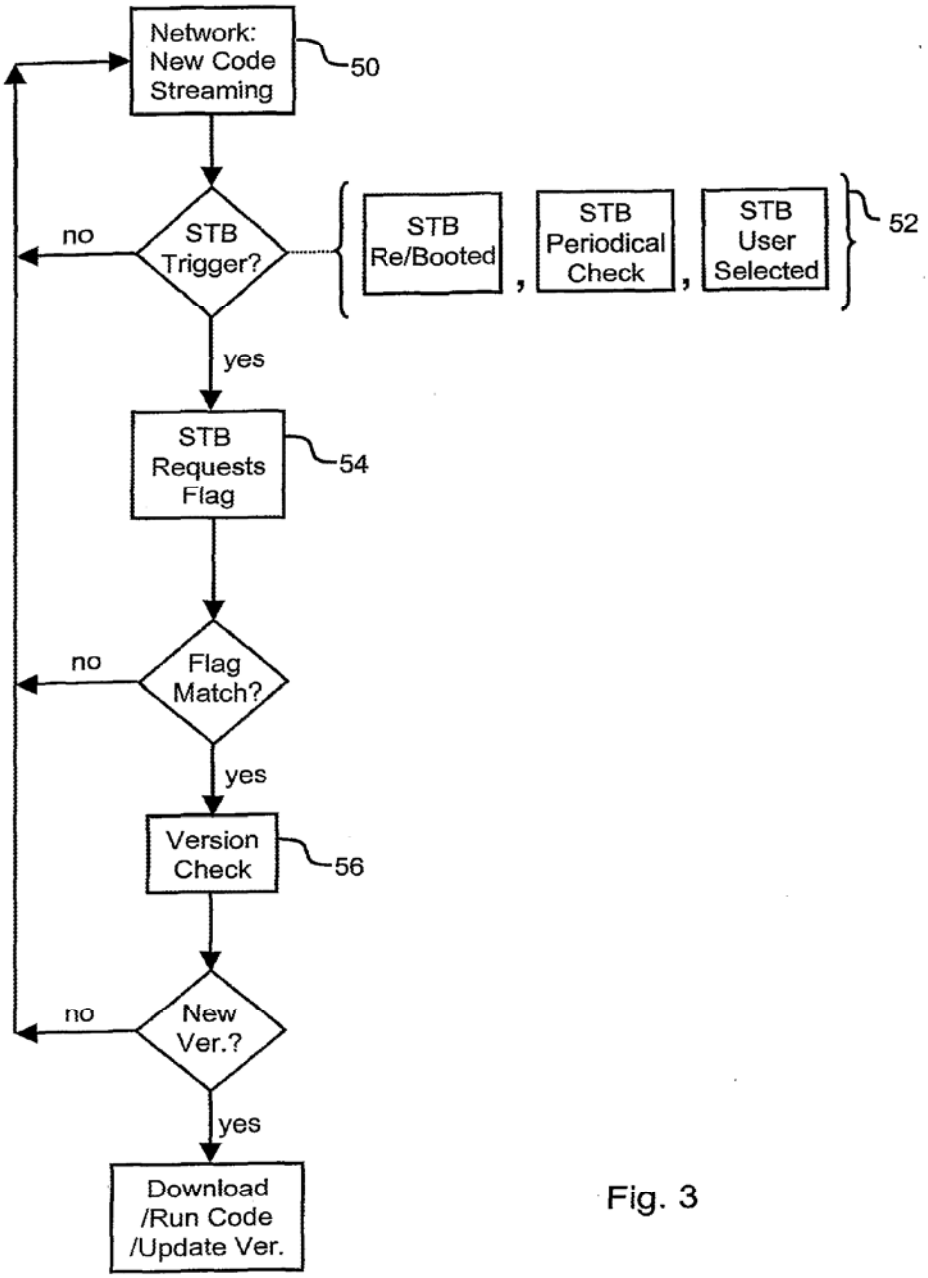



Fig. 3

ForcePoint LLC
EX-1009
Page476

# INTERNATIONAL SEARCH REPORT

International Application No
PCT/GB2004/002570

**A. CLASSIFICATION OF SUBJECT MATTER**
IPC 7 H04N7/16

According to International Patent Classification (IPC) or to both national classification and IPC

**B. FIELDS SEARCHED**

Minimum documentation searched (classification system followed by classification symbols)
IPC 7 H04N

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)
EPO-Internal

**C. DOCUMENTS CONSIDERED TO BE RELEVANT**

| Category ° | Citation of document, with indication, where appropriate, of the relevant passages | Relevant to claim No. |
|---|---|---|
| X | WO 00/64178 A (GEN INSTRUMENT CORP)<br>26 October 2000 (2000-10-26) | 1-6,<br>8-30,<br>32-50,<br>52-59 |
| Y | abstract<br>page 1, line 13 - line 18<br>page 3, line 11 - line 17<br>page 3, line 23 - page 5, line 10<br>page 6, line 3 - page 7, line 24<br>page 13, line 3 - line 7<br>page 13, line 21 - line 28<br>page 16, line 19 - page 18, line 14<br>figures 1,3,4 | 7,31,51 |

-/--

[X] Further documents are listed in the continuation of box C.

[X] Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

| Date of the actual completion of the international search | Date of mailing of the international search report |
|---|---|
| 2 November 2004 | 10/11/2004 |

| Name and mailing address of the ISA | Authorized officer |
|---|---|
| European Patent Office, P.B. 5818 Patentlaan 2<br>NL – 2280 HV Rijswijk<br>Tel. (+31-70) 340-2040, Tx. 31 651 epo nl,<br>Fax: (+31-70) 340-3016 | Ricardo, P |

Form PCT/ISA/210 (second sheet) (January 2004)

ForcePoint LLC
EX-1009
Page477

# INTERNATIONAL SEARCH REPORT

International Application No
PCT/GB2004/002570

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

| Category ° | Citation of document, with indication, where appropriate, of the relevant passages | Relevant to claim No. |
|---|---|---|
| Y | WO 96/41472 A (KLOSTERMAN BRIAN L ; MILNES KENNETH A (US); STARSIGHT TELECAST INC (US) 19 December 1996 (1996-12-19) | 7,31,51 |
| A | page 2, line 35 - page 3, line 24<br>page 6, line 37 - page 8, line 3<br>figures 1,2<br>claims 1,16,26 | 1-6,<br>8-30,<br>32-50,<br>52-59 |
| A | US 2002/083442 A1 (ELDERING CHARLES A)<br>27 June 2002 (2002-06-27)<br>paragraphs '0012! - '0014!<br>paragraphs '0028! - '0031!<br>figure 1A | 1-59 |
| A | US 2003/041335 A1 (GAUTIER ERIC ET AL)<br>27 February 2003 (2003-02-27)<br>paragraphs '0032! - '0034!<br>paragraphs '0062!, '0063!<br>paragraphs '0067!, '0068!<br>figure 2 | 1-59 |

Form PCT/ISA/210 (continuation of second sheet) (January 2004)

ForcePoint LLC
EX-1009
Page478

## INTERNATIONAL SEARCH REPORT

tional Application No
/GB2004/002570

| Patent document cited in search report | Publication date | Patent family member(s) | Publication date |
|---|---|---|---|
| WO 0064178 A | 26-10-2000 | AU 4241500 A | 02-11-2000 |
| | | AU 4241600 A | 02-11-2000 |
| | | AU 4348600 A | 02-11-2000 |
| | | AU 770984 B2 | 11-03-2004 |
| | | AU 4459600 A | 02-11-2000 |
| | | BR 0009901 A | 12-03-2002 |
| | | BR 0009902 A | 12-03-2002 |
| | | BR 0009903 A | 12-03-2002 |
| | | CA 2370214 A1 | 26-10-2000 |
| | | CA 2370764 A1 | 26-10-2000 |
| | | CA 2371144 A1 | 26-10-2000 |
| | | CA 2382509 A1 | 26-10-2000 |
| | | CN 1355996 T | 26-06-2002 |
| | | CN 1355997 T | 26-06-2002 |
| | | EP 1181824 A1 | 27-02-2002 |
| | | EP 1181825 A1 | 27-02-2002 |
| | | EP 1172005 A1 | 16-01-2002 |
| | | EP 1172006 A1 | 16-01-2002 |
| | | JP 2002542736 T | 10-12-2002 |
| | | TW 480887 B | 21-03-2002 |
| | | TW 472489 B | 11-01-2002 |
| | | TW 503662 B | 21-09-2002 |
| | | TW 472490 B | 11-01-2002 |
| | | WO 0064178 A1 | 26-10-2000 |
| | | WO 0064179 A1 | 26-10-2000 |
| | | WO 0064180 A1 | 26-10-2000 |
| | | WO 0064181 A1 | 26-10-2000 |
| | | US 6718374 B1 | 06-04-2004 |
| WO 9641472 A | 19-12-1996 | AU 714491 B2 | 06-01-2000 |
| | | AU 5990596 A | 30-12-1996 |
| | | BR 9608610 A | 25-04-2000 |
| | | CA 2223424 A1 | 19-12-1996 |
| | | EP 0830785 A1 | 25-03-1998 |
| | | JP 2001527702 T | 25-12-2001 |
| | | WO 9641472 A1 | 19-12-1996 |
| | | US 2003167473 A1 | 04-09-2003 |
| US 2002083442 A1 | 27-06-2002 | AU 8699801 A | 13-03-2002 |
| | | WO 0219581 A1 | 07-03-2002 |
| | | US 2002083435 A1 | 27-06-2002 |
| | | US 2002072966 A1 | 13-06-2002 |
| | | US 2002026638 A1 | 28-02-2002 |
| | | US 2002083444 A1 | 27-06-2002 |
| | | US 2002083439 A1 | 27-06-2002 |
| | | US 2002083443 A1 | 27-06-2002 |
| | | US 2002083445 A1 | 27-06-2002 |
| | | US 2002083441 A1 | 27-06-2002 |
| | | US 2001032333 A1 | 18-10-2001 |
| | | US 2002087975 A1 | 04-07-2002 |
| | | US 2002056107 A1 | 09-05-2002 |
| | | US 2002144263 A1 | 03-10-2002 |
| US 2003041335 A1 | 27-02-2003 | FR 2809838 A1 | 07-12-2001 |
| | | FR 2809839 A1 | 07-12-2001 |
| | | AU 3030901 A | 16-07-2001 |
| | | CN 1415170 T | 30-04-2003 |
| | | EP 1245116 A1 | 02-10-2002 |



ForcePoint LLC
EX-1009
Page479

# INTERNATIONAL SEARCH REPORT

International Application No
PCT/GB2004/002570

| Patent document cited in search report | Publication date | Patent family member(s) | Publication date |
|---|---|---|---|
| US 2003041335 A1 | | WO 0150762 A1 | 12-07-2001 |
| | | JP 2003534676 T | 18-11-2003 |

Form PCT/ISA/210 (patent family annex) (January 2004)

| Form PTO-1449 (Modified) | U.S. Department of Commerce Patent and Trademark Office | Attorney Docket No. SP1P0012 | Application No. 14/499,230 |
|---|---|---|---|
| **Information Disclosure Statement by Applicant** (Use several sheets if necessary) | | Applicant Brett M. Oliphant | |
| (37 CFR §1.98(b)) | | Filing Date 09/28/2014 | Group Art Unit |

| U.S. Patent Documents | | | | | | | |
|---|---|---|---|---|---|---|---|
| Examiner Initial | Desig. ID | Document Number | Publication Date | Patentee | Class | Subclass | Filing Date If Appropriate |
| | 1 | 5,210,854 | 05/11/1993 | Beaverton et al. | | | |
| | 2 | 5,249,705 | 10/05/1993 | Gates | | | |
| | 3 | 5,253,782 | 10/19/1993 | Gates et al. | | | |
| | 4 | 5,272,754 | 12/21/1993 | Boerbert | | | |
| | 5 | 5,276,735 | 01/04/1994 | Boebert et al. | | | |
| | 6 | 5,361,359 | 11/01/1994 | Tajalli et al. | | | |
| | 7 | 5,499,297 | 03/12/1996 | Boebert | | | |
| | 8 | 5,502,766 | 03/26/1996 | Boebert et al. | | | |
| | 9 | 5,557,346 | 09/17/1996 | Lipner et al. | | | |
| | 10 | 5,557,765 | 09/17/1996 | Lipner et al. | | | |
| | 11 | 5,596,718 | 01/21/1997 | Boebert et al. | | | |
| | 12 | 5,612,503 | 03/18/1997 | Sepp | | | |
| | 13 | 5,640,454 | 06/17/1997 | Lipner et al. | | | |
| | 14 | 5,662,291 | 09/02/1997 | Sepp et al. | | | |
| | 15 | 5,696,907 | 12/09/1997 | Tom | | | |
| | 16 | 5,745,573 | 04/28/1998 | Lipner et al. | | | |
| | 17 | 5,796,836 | 08/18/1998 | Markham | | | |
| | 18 | 5,819,275 | 10/06/1998 | Badger et al. | | | |
| | 19 | 5,822,435 | 10/13/1998 | Boebert et al. | | | |
| | 20 | 5,831,987 | 11/03/1998 | Spilo | | | |
| | 21 | 5,864,683 | 01/26/1999 | Boebert et al. | | | |
| | 22 | 5,867,647 | 02/02/1999 | Haigh et al. | | | |
| | 23 | 5,913,024 | 06/15/1999 | Green et al. | | | |
| | 24 | 5,915,087 | 06/22/1999 | Hammond et al. | | | |
| | 25 | 5,918,018 | 06/29/1999 | Gooderum et al. | | | |

| Examiner Signature | Date Considered |
|---|---|
| EXAMINER: Initials citation considered. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant. | |

Form (PTO-1449)

| Form PTO-1449 (Modified) | U.S. Department of Commerce Patent and Trademark Office | Attorney Docket No. SP1P0012 | Application No. 14/499,230 |
|---|---|---|---|
| **Information Disclosure Statement by Applicant** (Use several sheets if necessary) | | Applicant Brett M. Oliphant | |
| (37 CFR §1.98(b)) | | Filing Date 09/28/2014 | Group Art Unit |

**U.S. Patent Documents**

| Examiner Initial | Desig. ID | Document Number | Publication Date | Patentee | Class | Subclass | Filing Date If Appropriate |
|---|---|---|---|---|---|---|---|
| | 26 | 5,950,195 | 09/07/1999 | Stockwell et al. | | | |
| | 27 | 5,956,403 | 09/21/1999 | Lipner et al. | | | |
| | 28 | 5,968,133 | 10/19/1999 | Latham et al. | | | |
| | 29 | 5,983,350 | 11/09/1999 | Minear et al. | | | |
| | 30 | 5,991,406 | 11/23/1999 | Lipner et al. | | | |
| | 31 | 5,991,856 | 11/23/1999 | Spilo et al. | | | |
| | 32 | 6,003,084 | 12/14/1999 | Green et al. | | | |
| | 33 | 6,006,035 | 12/21/1999 | Nabahi | | | |
| | 34 | 6,028,514 | 02/22/2000 | Lemelson deceased et al. | | | |
| | 35 | 6,029,256 | 02/22/2000 | Kouznetsov | | | |
| | 36 | 6,035,423 | 03/07/2000 | Hodges et al. | | | |
| | 37 | 6,055,527 | 04/25/2000 | Badger et al. | | | |
| | 38 | 6,064,811 | 05/16/2000 | Spilo et al. | | | |
| | 39 | 6,072,942 | 06/06/2000 | Stockwell et al. | | | |
| | 40 | 6,084,510 | 07/04/2000 | Lemelson et al. | | | |
| | 41 | 6,122,657 | 09/19/2000 | Hoffman Jr. et al. | | | |
| | 42 | 6,134,601 | 10/17/2000 | Spilo et al. | | | |
| | 43 | 6,144,934 | 11/07/2000 | Stockwell et al. | | | |
| | 44 | 6,151,643 | 11/21/2000 | Cheng et al. | | | |
| | 45 | 6,173,180 | 01/09/2001 | Hussain et al. | | | |
| | 46 | 6,182,226 | 01/30/2001 | Reid et al. | | | |
| | 47 | 6,185,689 | 02/06/2001 | Todd Sr. et al. | | | |
| | 48 | 6,195,352 | 02/27/2001 | Cushman et al. | | | |
| | 49 | 6,205,552 | 03/20/2001 | Fudge | | | |
| | 50 | 6,208,999 | 03/27/2001 | Spilo et al. | | | |

| Examiner Signature | Date Considered |
|---|---|
| EXAMINER: Initials citation considered. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant. | |

Form (PTO-1449)

| Form PTO-1449 (Modified) | U.S. Department of Commerce Patent and Trademark Office | Attorney Docket No. SP1P0012 | Application No. 14/499,230 |
|---|---|---|---|
| **Information Disclosure Statement by Applicant** (Use several sheets if necessary) | | Applicant Brett M. Oliphant | |
| (37 CFR §1.98(b)) | | Filing Date 09/28/2014 | Group Art Unit |

| U.S. Patent Documents | | | | | | | |
|---|---|---|---|---|---|---|---|
| Examiner Initial | Desig. ID | Document Number | Publication Date | Patentee | Class | Subclass | Filing Date If Appropriate |
| | 51 | 6,209,101 | 03/27/2001 | Mitchem et al. | | | |
| | 52 | 6,219,707 | 04/17/2001 | Gooderum et al. | | | |
| | 53 | 6,226,372 | 05/01/2001 | Beebe et al. | | | |
| | 54 | 6,230,199 | 05/08/2001 | Revashetti et al. | | | |
| | 55 | 6,230,288 | 05/08/2001 | Kuo et al. | | | |
| | 56 | 6,240,531 | 05/29/2001 | Spilo et al. | | | |
| | 57 | 6,249,585 | 06/19/2001 | McGrew et al. | | | |
| | 58 | 6,266,774 | 07/24/2001 | Sampath et al. | | | |
| | 59 | 6,266,811 | 07/24/2001 | Nabahi | | | |
| | 60 | 6,267,039 | 07/31/2001 | Czarnecki | | | |
| | 61 | 6,269,456 | 07/31/2001 | Hodges et al. | | | |
| | 62 | 6,272,632 | 08/07/2001 | Carman et al. | | | |
| | 63 | 6,275,942 | 08/14/2001 | Bernhard et al. | | | |
| | 64 | 6,282,546 | 08/28/2001 | Gleichauf et al. | | | |
| | 65 | 6,298,422 | 10/02/2001 | Spilo et al. | | | |
| | 66 | 6,301,658 | 10/09/2001 | Koehler | | | |
| | 67 | 6,301,668 | 10/09/2001 | Gleichauf et al. | | | |
| | 68 | 6,301,699 | 10/09/2001 | Hollander et al. | | | |
| | 69 | 6,314,190 | 11/06/2001 | Zimmermann | | | |
| | 70 | 6,321,293 | 11/20/2001 | Fabrizio et al. | | | |
| | 71 | 6,321,336 | 11/20/2001 | Applegate et al. | | | |
| | 72 | 6,324,656 | 11/27/2001 | Gleichauf et al. | | | |
| | 73 | 6,332,195 | 12/18/2001 | Green et al. | | | |
| | 74 | 6,336,186 | 01/01/2002 | Dyksterhouse et al. | | | |
| | 75 | 6,343,362 | 01/29/2002 | Ptacek et al. | | | |

| Examiner Signature | Date Considered |
|---|---|
| EXAMINER: Initials citation considered. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant. | |

Form (PTO-1449)

| Form PTO-1449 (Modified) | U.S. Department of Commerce Patent and Trademark Office | Attorney Docket No. SP1P0012 | Application No. 14/499,230 |
|---|---|---|---|
| **Information Disclosure Statement by Applicant** (Use several sheets if necessary) | | Applicant Brett M. Oliphant | |
| (37 CFR §1.98(b)) | | Filing Date 09/28/2014 | Group Art Unit |

**U.S. Patent Documents**

| Examiner Initial | Desig. ID | Document Number | Publication Date | Patentee | Class | Subclass | Filing Date If Appropriate |
|---|---|---|---|---|---|---|---|
| | 76 | 6,351,794 | 02/26/2002 | Spilo et al. | | | |
| | 77 | 6,353,446 | 03/05/2002 | Vaughn et al. | | | |
| | 78 | 6,357,010 | 03/12/2002 | Viets et al. | | | |
| | 79 | 6,370,578 | 04/09/2002 | Revashetti et al. | | | |
| | 80 | 6,385,458 | 05/07/2002 | Papadimitriou et al. | | | |
| | 81 | 6,393,424 | 05/21/2002 | Hallman et al. | | | |
| | 82 | 6,402,689 | 06/11/2002 | Scarantino et al. | | | |
| | 83 | 6,415,321 | 07/02/2002 | Gleichauf et al. | | | |
| | 84 | 6,442,686 | 08/27/2002 | McArdle et al. | | | |
| | 85 | 6,453,347 | 09/17/2002 | Revashetti et al. | | | |
| | 86 | 6,453,419 | 09/17/2002 | Flint et al. | | | |
| | 87 | 6,457,076 | 09/24/2002 | Cheng et al. | | | |
| | 88 | 6,460,126 | 10/01/2002 | Spilo et al. | | | |
| | 89 | 6,470,384 | 10/22/2002 | O'Brien et al. | | | |
| | 90 | 6,477,550 | 11/05/2002 | Balasubramaniam et al. | | | |
| | 91 | 6,477,651 | 11/05/2002 | Teal | | | |
| | 92 | 6,493,756 | 12/10/2002 | O'Brien et al. | | | |
| | 93 | 6,496,875 | 12/17/2002 | Cheng et al. | | | |
| | 94 | 6,499,107 | 12/24/2002 | Gleichauf et al. | | | |
| | 95 | 6,499,109 | 12/24/2002 | Balasubramaniam et al. | | | |
| | 96 | 6,510,448 | 01/21/2003 | Churchyard | | | |
| | 97 | 6,513,122 | 01/28/2003 | Magdych et al. | | | |
| | 98 | 6,523,023 | 02/18/2003 | Sonnenberg | | | |
| | 99 | 6,535,227 | 03/18/2003 | Fox et al. | | | |
| | 100 | 6,542,943 | 04/01/2003 | Cheng et al. | | | |

| Examiner Signature | Date Considered |
|---|---|
| EXAMINER: Initials citation considered. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant. | |

Form (PTO-1449)

| Form PTO-1449 (Modified) | U.S. Department of Commerce Patent and Trademark Office | Attorney Docket No. SP1P0012 | Application No. 14/499,230 |
|---|---|---|---|
| **Information Disclosure Statement by Applicant** (Use several sheets if necessary) | | Applicant Brett M. Oliphant | |
| (37 CFR §1.98(b)) | | Filing Date 09/28/2014 | Group Art Unit |

| U.S. Patent Documents | | | | | | | |
|---|---|---|---|---|---|---|---|
| Examiner Initial | Desig. ID | Document Number | Publication Date | Patentee | Class | Subclass | Filing Date If Appropriate |
| | 101 | 6,546,493 | 04/08/2003 | Magdych et al. | | | |
| | 102 | 6,550,012 | 04/15/2003 | Villa et al. | | | |
| | 103 | 6,553,377 | 04/22/2003 | Eschelbeck et al. | | | |
| | 104 | 6,553,378 | 04/22/2003 | Eschelbeck | | | |
| | 105 | 6,567,808 | 05/20/2003 | Eschelbeck et al. | | | |
| | 106 | 6,584,504 | 06/24/2003 | Choe | | | |
| | 107 | 6,584,508 | 06/24/2003 | Epstein et al. | | | |
| | 108 | 6,587,888 | 07/01/2003 | Chieu et al. | | | |
| | 109 | 6,601,091 | 07/29/2003 | Spilo | | | |
| | 110 | 6,604,139 | 08/05/2003 | Sajina et al. | | | |
| | 111 | 6,608,817 | 08/19/2003 | Ivory | | | |
| | 112 | 6,611,869 | 08/26/2003 | Eschelbeck et al. | | | |
| | 113 | 6,611,925 | 08/26/2003 | Spear | | | |
| | 114 | 6,615,266 | 09/02/2003 | Hoffman Jr. et al. | | | |
| | 115 | 6,622,150 | 09/16/2003 | Kouznetsov et al. | | | |
| | 116 | 6,633,835 | 10/14/2003 | Moran et al. | | | |
| | 117 | 6,636,972 | 10/21/2003 | Ptacek et al. | | | |
| | 118 | 6,640,307 | 10/28/2003 | Viets et al. | | | |
| | 119 | 6,654,751 | 11/25/2003 | Schmugar et al. | | | |
| | 120 | 6,654,782 | 11/25/2003 | O'Brien et al. | | | |
| | 121 | 6,658,571 | 12/02/2003 | O'Brien et al. | | | |
| | 122 | 6,663,000 | 12/16/2003 | Muttik et al. | | | |
| | 123 | 6,665,269 | 12/16/2003 | Schmitz | | | |
| | 124 | 6,668,054 | 12/23/2003 | Contino et al. | | | |
| | 125 | 6,668,289 | 12/23/2003 | Cheng et al. | | | |

| Examiner Signature | Date Considered |
|---|---|
| EXAMINER: Initials citation considered. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant. | |

Form (PTO-1449)

| Form PTO-1449 (Modified) | U.S. Department of Commerce Patent and Trademark Office | Attorney Docket No. SP1P0012 | Application No. 14/499,230 |
|---|---|---|---|
| **Information Disclosure Statement by Applicant** (Use several sheets if necessary) | | Applicant Brett M. Oliphant | |
| (37 CFR §1.98(b)) | | Filing Date 09/28/2014 | Group Art Unit |

| U.S. Patent Documents | | | | | | | |
|---|---|---|---|---|---|---|---|
| Examiner Initial | Desig. ID | Document Number | Publication Date | Patentee | Class | Subclass | Filing Date If Appropriate |
| | 126 | 6,671,368 | 12/30/2003 | Contino et al. | | | |
| | 127 | 6,671,812 | 12/30/2003 | Balasubramaniam et al. | | | |
| | 128 | 6,674,738 | 01/06/2004 | Yildiz et al. | | | |
| | 129 | 6,675,042 | 01/06/2004 | Swerdlow et al. | | | |
| | 130 | 6,678,887 | 01/13/2004 | Hallman et al. | | | |
| | 131 | 6,684,329 | 01/27/2004 | Epstein et al. | | | |
| | 132 | 6,687,833 | 02/03/2004 | Osborne et al. | | | |
| | 133 | 6,693,888 | 02/17/2004 | Cafarelli et al. | | | |
| | 134 | 6,697,337 | 02/24/2004 | Cafarelli et al. | | | |
| | 135 | 6,697,870 | 02/24/2004 | Cafarelli III et al. | | | |
| | 136 | 6,697,950 | 02/24/2004 | Ko | | | |
| | 137 | 6,701,440 | 03/02/2004 | Kim et al. | | | |
| | 138 | 6,701,441 | 03/02/2004 | Balasubramaniam et al. | | | |
| | 139 | 6,704,874 | 03/09/2004 | Porras et al. | | | |
| | 140 | 6,707,794 | 03/16/2004 | Leong et al. | | | |
| | 141 | 6,708,292 | 03/16/2004 | Mangasarian | | | |
| | 142 | 6,711,127 | 03/23/2004 | Gorman et al. | | | |
| | 143 | 6,714,513 | 03/30/2004 | Joiner et al. | | | |
| | 144 | 6,718,469 | 04/06/2004 | Pak et al. | | | |
| | 145 | 6,721,847 | 04/13/2004 | Hursey | | | |
| | 146 | 6,725,377 | 04/20/2004 | Kouznetsov | | | |
| | 147 | 6,728,219 | 04/27/2004 | Leong et al. | | | |
| | 148 | 6,728,885 | 04/27/2004 | Taylor et al. | | | |
| | 149 | 6,732,157 | 05/04/2004 | Gordon et al. | | | |
| | 150 | 6,735,629 | 05/11/2004 | Cafarelli III et al. | | | |

| Examiner Signature | Date Considered |
|---|---|
| EXAMINER: Initials citation considered. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant. | |

Form (PTO-1449)

| Form PTO-1449 (Modified) | U.S. Department of Commerce Patent and Trademark Office | Attorney Docket No. SP1P0012 | Application No. 14/499,230 |
|---|---|---|---|
| **Information Disclosure Statement by Applicant** (Use several sheets if necessary) | | Applicant Brett M. Oliphant | |
| (37 CFR §1.98(b)) | | Filing Date 09/28/2014 | Group Art Unit |

| U.S. Patent Documents | | | | | | | |
|---|---|---|---|---|---|---|---|
| Examiner Initial | Desig. ID | Document Number | Publication Date | Patentee | Class | Subclass | Filing Date If Appropriate |
| | 151 | 6,735,703 | 05/11/2004 | Kilpatrick et al. | | | |
| | 152 | 6,742,124 | 05/25/2004 | Kilpatrick et al. | | | |
| | 153 | 6,742,128 | 05/25/2004 | Joiner | | | |
| | 154 | 6,745,192 | 06/01/2004 | Libenzi | | | |
| | 155 | 6,745,311 | 06/01/2004 | Fabrizio et al. | | | |
| | 156 | 6,748,534 | 06/08/2004 | Gryaznov et al. | | | |
| | 157 | 6,751,627 | 06/15/2004 | Sternin | | | |
| | 158 | 6,751,718 | 06/15/2004 | Manin | | | |
| | 159 | 6,754,488 | 06/22/2004 | Won et al. | | | |
| | 160 | 6,754,705 | 06/22/2004 | Joiner et al. | | | |
| | 161 | 6,757,727 | 06/29/2004 | Ivory | | | |
| | 162 | 6,757,822 | 06/29/2004 | Feiertag et al. | | | |
| | 163 | 6,757,830 | 06/29/2004 | Tarbotton et al. | | | |
| | 164 | 6,760,845 | 07/06/2004 | Cafarelli et al. | | | |
| | 165 | 6,763,403 | 07/13/2004 | Cheng et al. | | | |
| | 166 | 6,763,466 | 07/13/2004 | Glover | | | |
| | 167 | 6,766,458 | 07/20/2004 | Harris et al. | | | |
| | 168 | 6,768,721 | 07/27/2004 | Schmitz et al. | | | |
| | 169 | 6,768,991 | 07/27/2004 | Hearnden | | | |
| | 170 | 6,769,016 | 07/27/2004 | Rothwell et al. | | | |
| | 171 | 6,772,332 | 08/03/2004 | Boebert et al. | | | |
| | 172 | 6,772,334 | 08/03/2004 | Glawitsch | | | |
| | 173 | 6,772,345 | 08/03/2004 | Shetty | | | |
| | 174 | 6,775,704 | 08/10/2004 | Watson et al. | | | |
| | 175 | 6,775,780 | 08/10/2004 | Muttik | | | |

| Examiner Signature | Date Considered |
|---|---|
| EXAMINER: Initials citation considered. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant. | |

Form (PTO-1449)

| Form PTO-1449 (Modified) | U.S. Department of Commerce Patent and Trademark Office | Attorney Docket No. SP1P0012 | Application No. 14/499,230 |
|---|---|---|---|
| **Information Disclosure Statement by Applicant** (Use several sheets if necessary) | | Applicant Brett M. Oliphant | |
| (37 CFR §1.98(b)) | | Filing Date 09/28/2014 | Group Art Unit |

| U.S. Patent Documents | | | | | | | |
|---|---|---|---|---|---|---|---|
| Examiner Initial | Desig. ID | Document Number | Publication Date | Patentee | Class | Subclass | Filing Date If Appropriate |
| | 176 | 6,779,033 | 08/17/2004 | Watson et al. | | | |
| | 177 | 6,782,421 | 08/24/2004 | Soles et al. | | | |
| | 178 | 6,782,527 | 08/24/2004 | Kouznetsov et al. | | | |
| | 179 | 6,785,237 | 08/31/2004 | Sufleta | | | |
| | 180 | 6,785,820 | 08/31/2004 | Muttik et al. | | | |
| | 181 | 6,785,821 | 08/31/2004 | Teal | | | |
| | 182 | 6,789,082 | 09/07/2004 | Sternin et al. | | | |
| | 183 | 6,789,117 | 09/07/2004 | Joiner et al. | | | |
| | 184 | 6,789,201 | 09/07/2004 | Barton et al. | | | |
| | 185 | 6,789,202 | 09/07/2004 | Ko et al. | | | |
| | 186 | 6,792,543 | 09/14/2004 | Pak et al. | | | |
| | 187 | 6,799,197 | 09/28/2004 | Shetty et al. | | | |
| | 188 | 6,801,756 | 10/05/2004 | Agrawal et al. | | | |
| | 189 | 6,801,940 | 10/05/2004 | Moran et al. | | | |
| | 190 | 6,802,012 | 10/05/2004 | Smithson et al. | | | |
| | 191 | 6,810,017 | 10/26/2004 | Leong et al. | | | |
| | 192 | 6,814,842 | 11/09/2004 | Yago et al. | | | |
| | 193 | 6,816,973 | 11/09/2004 | Gleichauf et al. | | | |
| | 194 | 6,823,460 | 11/23/2004 | Hollander et al. | | | |
| | 195 | 6,826,608 | 11/30/2004 | Sheth et al. | | | |
| | 196 | 6,826,698 | 11/30/2004 | Minkin et al. | | | |
| | 197 | 6,834,301 | 12/21/2004 | Hanchett | | | |
| | 198 | 6,836,860 | 12/28/2004 | Muttik et al. | | | |
| | 199 | 6,839,852 | 01/04/2005 | Pantuso et al. | | | |
| | 200 | 6,842,860 | 01/11/2005 | Branstad et al. | | | |

| Examiner Signature | Date Considered |
|---|---|
| EXAMINER: Initials citation considered. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant. | |

Form (PTO-1449)

| Form PTO-1449 (Modified) | U.S. Department of Commerce Patent and Trademark Office | Attorney Docket No. SP1P0012 | Application No. 14/499,230 |
|---|---|---|---|
| **Information Disclosure Statement by Applicant** (Use several sheets if necessary) | | Applicant Brett M. Oliphant | |
| (37 CFR §1.98(b)) | | Filing Date 09/28/2014 | Group Art Unit |

| U.S. Patent Documents | | | | | | | |
|---|---|---|---|---|---|---|---|
| Examiner Initial | Desig. ID | Document Number | Publication Date | Patentee | Class | Subclass | Filing Date If Appropriate |
| | 201 | 6,842,861 | 01/11/2005 | Cox et al. | | | |
| | 202 | 6,843,190 | 01/18/2005 | LaPierre-McAfee | | | |
| | 203 | 6,845,449 | 01/18/2005 | Carman et al. | | | |
| | 204 | 6,851,058 | 02/01/2005 | Gartside | | | |
| | 205 | 6,851,061 | 02/01/2005 | Holland III et al. | | | |
| | 206 | 6,859,793 | 02/22/2005 | Lambiase | | | |
| | 207 | 6,862,581 | 03/01/2005 | Lambiase | | | |
| | 208 | 6,868,069 | 03/15/2005 | Knobbe et al. | | | |
| | 209 | 6,868,413 | 03/15/2005 | Grindrod et al. | | | |
| | 210 | 6,871,279 | 03/22/2005 | Sames et al. | | | |
| | 211 | 6,874,018 | 03/29/2005 | Wu | | | |
| | 212 | 6,879,812 | 04/12/2005 | Agrawal et al. | | | |
| | 213 | 6,886,099 | 04/26/2005 | Smithson et al. | | | |
| | 214 | 6,892,227 | 05/10/2005 | Elwell et al. | | | |
| | 215 | 6,892,241 | 05/10/2005 | Kouznetsov et al. | | | |
| | 216 | 6,895,387 | 05/17/2005 | Roberts et al. | | | |
| | 217 | 6,895,513 | 05/17/2005 | Balasubramaniam et al. | | | |
| | 218 | 6,898,712 | 05/24/2005 | Vignoles et al. | | | |
| | 219 | 6,898,715 | 05/24/2005 | Smithson et al. | | | |
| | 220 | 6,904,521 | 06/07/2005 | Jivsov | | | |
| | 221 | 6,907,396 | 06/14/2005 | Muttik et al. | | | |
| | 222 | 6,915,426 | 07/05/2005 | Carman et al. | | | |
| | 223 | 6,920,493 | 07/19/2005 | Schwab | | | |
| | 224 | 6,920,558 | 07/19/2005 | Sames et al. | | | |
| | 225 | 6,922,776 | 07/26/2005 | Cook et al. | | | |

| Examiner Signature | Date Considered |
|---|---|
| EXAMINER: Initials citation considered. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant. | |

Form (PTO-1449)