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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
8 March 2007 (08.03.2007)

PCT

(10) International Publication Number
WO 2007/027746 A1

(51) International Patent Classification:
G06F 15/00 (2006.01)

(21) International Application Number:
PCT/US2006/033823

(22) International Filing Date: 29 August 2006 (29.08.2006)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
11/217,748 1 September 2005 (01.09.2005) US

(71) Applicant *(for all designated States except US)*: **MICROSOFT CORPORATION** [US/US]; One Microsoft Way, Redmond, Washington 98052-6399 (US).

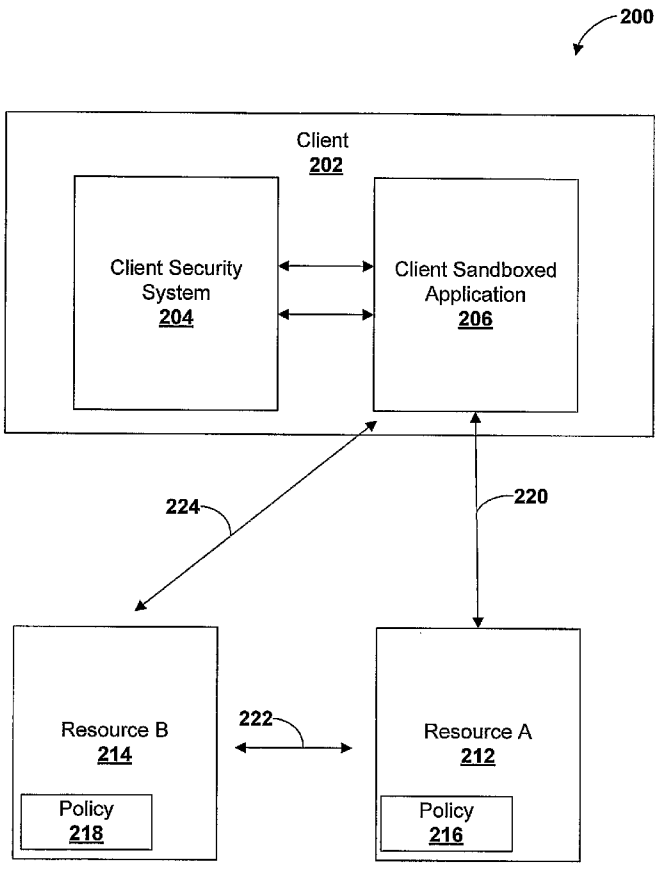
(72) Inventors: **COOPERSTEIN, Jeffrey M.**; One Microsoft Way, Redmond, Washington 98052-6399 (US). **GOLDFEDER, Aaron R.**; One Microsoft Way, Redmond, Washington 98052-6399 (US). **FEE, Gregory D.**; One Microsoft Way, Redmond, Washington 98052-6399 (US). **HAWKINS, John M.**; One Microsoft Way, Redmond, Washington 98052-6399 (US). **KUDALLUR, Venkatraman**; One Microsoft Way, Redmond, Washington 98052-6399 (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, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LV, LY, MA, MD, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, SV, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

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

[Continued on next page]

(54) Title: RESOURCE BASED DYNAMIC SECURITY AUTHORIZATION



(57) Abstract: Access to a resource by sandboxed code is dynamically authorized by a client security system based on a resource based policy. A sandboxed application running on a client is granted access to a resource based on a resource based policy despite denial of the access based on a static policy associated with the client security system. The granting of access coincides with the determination that the threat to a user or the user's information is not increased should the access be granted.

WO 2007/027746 A1

Declarations under Rule 4.17:

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

— *as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(iii))*

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.

RESOURCE BASED DYNAMIC SECURITY AUTHORIZATION

BACKGROUND

Security is a primary concern when using an Internet browser to navigate the World Wide Web. The applications that run on a browser may institute actions or execute code that could compromise security. Some browser applications may in fact include malicious code that can infect a computer or gain access to confidential information stored on the computer.

In order to prevent such malicious code from infecting the computer or gaining access to a user's information, mechanisms have been developed to wall-off or contain this code so that it has restricted access to the computer's resources. The security restrictions are often instituted when executable code comes from an unknown or un-validated source and allows the user to run un-secure code. However, a blanket restriction of access to resources also prevents certain legitimate actions from being executed for some applications. For example, two domains on the Internet may be associated with the same source or Internet resource even though they have different domain names (e.g., http://www.site1.com and http://www.site2.com). With current security restrictions, the application is unable to obtain resources from the second domain based on the security access permission granted to the first domain.

SUMMARY

Aspects of the present invention are generally related to providing a resource based security system for dynamic authorization of access to a resource by sandboxed code. Sandboxed code is code associated with an environment in which there are strict limitations on what system resources the code can request or access. Access to the resources is authorized when a threat to the user is shown as not increasing if the authorization took place. Client security policy is static and ensures that users are safe under all circumstances. In many cases however, the client security policy restricts access from resource to resource in an overly restrictive manner. For example, web pages from http://www.site1.com cannot script http://www.site2.com even if they are owned and operated by the same company. The present invention solves this problem

by recognizing that the threat to the user or machine is not increased if these sites did script each other. Since the sites are owned and operated by the same company the security access between each site may be assumed. Similarly, even when two domains are not owned or operated by the same company or even related companies, one domain may affirmatively state that the sandboxed application associated with the other company is allowed access to the their associated resource. In accordance with one aspect of the present invention, a resource based policy statement is issued that allows the sandboxed application of one site to script another site despite the presence of a static policy preventing such scripting.

This Summary is provided to introduce a selection of concepts in a simplified form that are further described below in the Detailed Description. This Summary is not intended to identify key features or essential features of the claimed subject matter, nor is it intended to be used as an aid in determining the scope of the claimed subject matter.

BRIEF DESCRIPTION OF THE DRAWINGS

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

FIGURE 1 illustrates an exemplary computing device that may be used in accordance with one exemplary embodiment;

FIGURE 2 illustrates a functional diagram of an exemplary system for providing dynamic security authorization for access to a resource;

FIGURE 3 illustrates a flow diagram representing an exemplary embodiment of a process for providing dynamic security authorization for access to a resource; and

FIGURE 4 illustrates a flow diagram representing another exemplary embodiment of a process for providing dynamic security authorization for access to a resource, in accordance with one embodiment of the present invention.

DETAILED DESCRIPTION

Embodiments of the present invention are described more fully below with reference to the accompanying drawings, which form a part hereof, and which show

specific exemplary embodiments for practicing the invention. However, embodiments may be implemented in many different forms and should not be construed as limited to the embodiments set forth herein; rather, these embodiments are provided so that this disclosure will be thorough and complete, and will fully convey the scope of the invention to those skilled in the art. Embodiments of the present invention may be practiced as methods, systems or devices. Accordingly, embodiments of the present invention may take the form of an entirely hardware implementation, an entirely software implementation or an implementation combining software and hardware aspects. The following detailed description is, therefore, not to be taken in a limiting sense.

The logical operations of the various embodiments of the present invention are implemented (1) as a sequence of computer implemented steps running on a computing system and/or (2) as interconnected machine modules within the computing system. The implementation is a matter of choice dependent on the performance requirements of the computing system implementing the invention. Accordingly, the logical operations making up the embodiments of the present invention described herein are referred to alternatively as operations, steps or modules.

With reference to FIGURE 1, one exemplary system for implementing the invention includes a computing device, such as computing device 100. Computing device 100 may be configured as a client, a server, mobile device, or any other computing device. In a very basic embodiment, computing device 100 typically includes at least one processing unit 102 and system memory 104. Depending on the exact configuration and type of computing device, system memory 104 may be volatile (such as RAM), non-volatile (such as ROM, flash memory, etc.) or some combination of the two. System memory 104 typically includes an operating system 105, one or more applications 106, and may include program data 107. In one embodiment, application 106 includes a client security system 120 for implementing the system of the present invention. This basic embodiment is illustrated in FIGURE 1 by those components within dashed line 108.

Computing device 100 may have additional features or functionality. For example, computing device 100 may also include additional data storage devices

(removable and/or non-removable) such as, for example, magnetic disks, optical disks, or tape. Such additional storage is illustrated in FIGURE 1 by removable storage 109 and non-removable storage 110. Computer storage media may include volatile and nonvolatile, removable and non-removable media implemented in any method or technology for storage of information, such as computer readable instructions, data structures, program modules, or other data. System memory 104, removable storage 109 and non-removable storage 110 are all examples of computer storage media. Computer storage media includes, but is not limited to, RAM, ROM, EEPROM, flash memory or other memory technology, CD-ROM, digital versatile disks (DVD) or other optical storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices, or any other medium which can be used to store the desired information and which can be accessed by computing device 100. Any such computer storage media may be part of device 100. Computing device 100 may also have input device(s) 112 such as keyboard, mouse, pen, voice input device, touch input device, etc. Output device(s) 114 such as a display, speakers, printer, etc. may also be included.

Computing device 100 also contains communication connections 116 that allow the device to communicate with other computing devices 118, such as over a network. Communication connection 116 is one example of communication media. Communication media may typically be embodied by computer readable instructions, data structures, program modules, or other data in a modulated data signal, such as a carrier wave or other transport mechanism, and includes any information delivery media. The term "modulated data signal" means a signal that has one or more of its characteristics set or changed in such a manner as to encode information in the signal. By way of example, and not limitation, communication media includes wired media such as a wired network or direct-wired connection, and wireless media such as acoustic, RF, infrared and other wireless media. The term computer readable media as used herein includes both storage media and communication media.

An embodiment executed by computing device 100 provides for providing a resource based security system for dynamic authorization of access to a resource by

sandboxed code. The access is evaluated by comparing evidence of the authenticity of the sandboxed code with a policy associated with the resource. If the evidence is sufficient according to the resource based policy, then access to the resource is authorized. For example, web pages from http://www.site1.com cannot script http://www.site2.com even if they are affiliated or owned and operated by the same company or even if they contain an affirmative statement that an application associated with one web page may access data from the other web page. The present invention solves this problem by recognizing that the threat to the user or machine is not increased if these sites did script each other. A policy statement may be included at site2 which partially states:

```
<AllowScriptAccess>
        <Sites>
                http://www.site1.com
        </Sites>
</AllowScriptAccess>
```

The policy statement further includes requirements for evidence of the authenticity of the sandboxed code. When the evidence is sufficient for the requirements, the policy authorizes web pages of site1 to script web pages of site2. Other examples for granting access to a resource based on a resource based dynamical authorization are provided in the discussion below.

Except for a broader meaning that allows the disclosure and claims herein to encompass a wider variety of alternative embodiments, the following terms as used herein and throughout the claims are generally defined as follows:

"Dynamic authorization" is generally defined as authorization based on policies that determine the access to resources on a resource-by-resource basis. Stated differently, the authorization is dynamic because there is no fixed set of resources that a sandboxed application may access.

"Evidence" is generally defined as any information about an application that identifies the application or that provides support for the application's authenticity.

Evidence may include information about the source of the application, the type of application, and other information that relays that the application may be trusted to operate correctly. When an application has evidence referred to as sufficient, then the application has successfully identified itself to a resource as an application that may be trusted.

"Resource" is generally defined as a web page, a file, a database, or the like. Resources are accessed and used by users and applications during executions or web browsing events.

"Resource based policy" generally refers to a policy wherein the resources decide which sandboxed application or sandboxed code may access that resource.

"Sandboxed code" or a "sandboxed application" refers to code that has been walled-off or restricted from the other applications, code, and resources on a client. The restricted code corresponds to a restricted environment within which the sandboxed code runs. In one embodiment corresponding to managed code, the sandboxed code refers to code managed under code access security that prevents the code from accessing resources that are protected. In another example, during web browsing, the code associated with a web page is operated as sandboxed code and is allowed to access only limited resources of the computing device. Sandboxed code may be limited to accessing the data created or provided in association with the sandboxed code, some limited user interface functions, and the resource associated with the source (e.g., source web server) of the sandboxed code.

FIGURE 2 illustrates a functional diagram of an exemplary system for providing dynamic security authorization for access to a resource in accordance with one embodiment. System 200 includes client 202, resource A 212, and resource B 214. Client 202 includes client security system 204 and client sandboxed application 206. In one embodiment, client 202 corresponds to the computing device described in relation to FIGURE 1. Furthermore, although resource A 212 and resource B 214 are shown as separate from client 202, these resources may also be located within client 202 (e.g., within memory or on disk).

Client security system 204 provides the control and enforcement of the restrictions of client sandboxed application 206. The restrictions correspond to limitations on actions that client sandboxed application 206 may take, as well as limitations on resources that the application may access.

In one implementation of system 200, client restriction application is sourced by resource A 212. For example, client restriction application 206 is an application running on client 202 (e.g., within a browser application), and client restriction application 206 was provided by a web server that corresponds to resource A 212. When running, client restriction application 206 is allowed to access resource A 212 using connection 220 since a static security setting exists according to client security system 204 that states that a sandboxed application may access a resource that corresponds to the source of the sandboxed application.

In one example however, it may be that client sandboxed application 206 desires to access resource B 214 using unrestricted connection 224. Resource B 214 corresponds to another web server providing a separate web page. Access of this type, sometimes referred to as "cross-domain" access is usually among the activities prevented by client security system 204. However, under certain circumstances, resource A 212 and resource B 214 are owned and operated by the same entity. Since the resources are commonly owned, communication may take place between the resources. Similarly, using the communication between resources, communication may be rigged between resource B 214 and client sandboxed application 206 across connections 222 and 220 by redirecting or shuffling communication through resource A 212. Often however, this communication may be unidirectional or suffer from other inefficiencies. However, due to the common ownership between resources, the threat to a user or a user's information would not be increased by allowing a communication between client sandboxed application 206 and resource B 214 across connection 224. Despite this lack of threat, the sandbox environment still prevents unrestricted communication between resource B 214 and client sandboxed application 206 across connection 224.

One embodiment provided herein solves this restriction by allowing client sandboxed application 206 to check a resource based policy (e.g., policy 216, policy 218) that dynamically authorizes the communication across connection 224. In one embodiment, when resource A 212 is the source of client sandboxed application 206, the application checks for the allowance of the communication with resource B 214 against a policy (e.g., policy 216) associated with resource A 212. In another embodiment, a service connection, separate from a full communication connection, is established between client sandboxed application 206 and resource B 214 to check for permission against a policy (e.g., policy 218) associated with resource B 214. Exemplary methods for dynamically authorizing access to a resource are further described in the discussion of FIGURES 3 and 4 below.

In another implementation of system 200, resource B 214 corresponds to a database server that may or may not be associated with resource A 212. Resource B 214 may be configured with a resource based policy that associates resource B 214 with client sandboxed application 206. Resource B 214 may then communicate to client 202 that client sandboxed application 206 is authorized to access the database server.

Other embodiments provide other uses for when a resource may be accessed by a sandboxed application based on a resource based policy. Additionally, although two resources are shown (resource A 212 and resource B 214) the present invention is equally applicable to a scenario involving only one resource or multiple resources.

FIGURE 3 illustrates a flow diagram representing an exemplary embodiment of a process for providing dynamic security authorization for access to a resource. Process 300 starts where a client sandboxed application or sandboxed code is included on a client machine, and an attempt has been made for the sandboxed code to access a particular resource. Processing continues at access decision operation 302.

Access decision operation 302 determines whether the sandboxed code has access granted according to static client security policy. A static client security policy corresponds to a policy dictated by the client security system enforcing the restrictions on the sandboxed code. For example, it may be that the resource to be accessed is the source

of the sandboxed code present on the client. A policy may already be in place by the client security application that authorizes access to the resource based on the resource corresponding to the source of the sandboxed code. In one embodiment, the policies set by the client security system are static policies that do not change, but may however be overridden by external policies. If access to the resource by the sandboxed code is granted by a static policy, processing advances to access allowance operation 312. However, if a static policy does not exist that grants access to the resource, then processing continues with service connection operation 304.

Service connection operation 304 establishes a service connection between the resource and the sandboxed code. In one embodiment, the service connection is separate from other communication connections such as an unrestricted web connection. The service connection is limited in its functionality to confirming whether the sandboxed code meets any existing resource based policies that grant the code access to the resource. In another embodiment, the service connection corresponds to a connection to another resource related to the resource for which access is being sought. Access to the other resource is granted according to a static policy set by the client security system. A resource based policy associated with the other resource may exist that authorizes access to the resource according to the relation between the resource and the other resource. Once a service connection is established, processing continues to evidence transmission operation 306.

Evidence transmission operation 306 sends evidence of the sandboxed code to the resource across the service connection. Evidence of the sandboxed code refers to evidence that shows whether the sandboxed code corresponds to a legitimate application that is running on the client. The evidence may include the source of the sandboxed code, as well as other identifying factors for the sandboxed code. Once the evidence is sent across the service connection to the resource, processing continues at sufficiency determination operation 308.

Sufficiency determination operation 308 determines whether the evidence provided to the resource is sufficient to identify the sandboxed code as code that meets a

resource based policy. The resource is able to compare the evidence against the resource based policy and dynamically authenticate access to the resource when the evidence indicates that the sandboxed code meets the policy. If the evidence is insufficient when compared to the resource based policy, then processing moves to denial operation 310.

Denial operation 310 prevents the sandboxed code from accessing the resource since the evidence supplied by the sandboxed code did not meet the policy. After the denial of access, processing ends and process 300 moves to other tasks related to running the sandboxed code.

Alternatively, if the evidence provided by the sandboxed code to the resource is found to be sufficient, processing moves to allowance operation 312. Allowance operation 312 grants the sandboxed code access to the resource. The grant of access to the resource coincides with the determination that the risk to a user's information is not increased by the authorization to access the resource. Once access is granted to the resource, processing ends and process 300 moves to other tasks related to running the sandboxed code.

FIGURE 4 illustrates a flow diagram representing another exemplary embodiment of a process for providing dynamic security authorization for access to a resource. Process 400 starts similarly to process 300 of FIGURE 3, where a client sandboxed application or sandboxed code is present on a client and a request has been made for the sandboxed code to access a particular resource. Processing continues at access decision operation 402.

Access decision operation 402 determines whether the sandboxed code has access granted according to static client security policy. A static client security policy corresponds to a policy dictated by the client security system enforcing the restrictions on the sandboxed code. For example, it may be that the resource to be accessed is the source of the sandboxed code present on the client. A policy may already be in place by the client security application that authorizes access to the resource based on the resource corresponding to the source of the sandboxed code. In one embodiment, the policies set by the client security system are static policies that do not change, but may however be

overridden by external policies. If access to the resource by the sandboxed code is granted by a static policy, processing advances to access allowance operation 414. However, if a static policy does not exist that grants access to the resource, then processing continues with service connection operation 404.

Service connection operation 404 establishes a service connection between the resource and the sandboxed code. The service connection established for process 400 is similar to the service connection established for process 300 described in accordance with FIGURE 3. Once a service connection is established, processing continues to policy query operation 406.

Policy query operation 406 queries the resource over the service connection to determine the resource based policy that may authorize the sandboxed code to access the resource. The query is sent by the sandboxed code to the resource. Once the query is sent, processing continues to policy receipt operation 408.

Policy receipt operation 408 receives the resource based policy from the resource. The policy may correspond to a set of criteria that the sandboxed code must meet to access the resource. Once the policy is received by the sandboxed code on the client, processing continues with sufficiency determination operation 410.

Sufficiency determination operation 410 determines whether the sandboxed code meets the requirements of the received resource based policy. The client security system is able to compare the attributes of the sandboxed code against the resource based policy and dynamically authenticate access to the resource when the comparison indicates that the sandboxed code meets the policy. In another embodiment, the client security system compares the attributes of the sandboxed code to the received resource based policy to determine if the sandboxed code meets the policy. If the sandboxed code does not meet the resource based policy, then processing moves to denial operation 412.

Denial operation 412 prevents the sandboxed code from accessing the resource since the sandboxed code did not meet the received policy. After the denial of access, processing ends and process 400 moves to other tasks related to running the sandboxed code.

Alternatively, if the sandboxed code is found to meet the resource based policy requirements, processing moves to allowance operation 414. Allowance operation 414 grants the sandboxed code access to the resource. The grant of access to the resource coincides with the determination that the risk to a user's information is not increased by the authorization to access to the resource. Once access is granted to the resource, processing ends and process 400 moves to other tasks related to running the sandboxed code.

In both process 300 and 400 of FIGURES 3 and 4, the resource based policy is shown as controlling the access of the sandboxed code to the resource. In other embodiments, a determination may be made that despite the presence of a resource based policy, a static policy provided by the client security system should still control. In still other embodiments, the static policy and the dynamic resource based policy may be applied together for the access determination of the resource.

Although the above discussion concentrates mainly on the situation where dynamic authorization is provided for access to a resource that is otherwise restricted by a static policy, the above described invention may also be used for the denial of access as well. For example, a resource such as http://www.example.com/page1 may have a policy that does not allow access by http://www.example.com/page2 though a static policy would allow access. Correspondingly to how the present invention dynamically grants access to a resource, the present invention may also dynamically deny access to resource.

Although the invention has been described in language that is specific to structural features and/or methodological steps, it is to be understood that the invention defined in the appended claims is not necessarily limited to the specific features or steps described. Rather, the specific features and steps are disclosed as forms of implementing the claimed invention. Since many embodiments of the invention can be made without departing from the spirit and scope of the invention, the invention resides in the claims hereinafter appended.

CLAIMS

What is claimed is:

1. A computer-implemented method for managing access to a resource by sandboxed code included on a client, the method comprising:

establishing a service connection (220) between the resource (212) and the client (202);

comparing evidence of the sandboxed code (206) against a resource based policy (216) associated with the resource (212); and

providing the sandboxed code (206) access to the resource when the evidence is determined as sufficient according to the resource based policy (216).

2. The computer-implemented method of claim 1, wherein the evidence is determined as sufficient according to the resource based policy (216) when granting the sandboxed code access (206) to the resource does not increase a threat level to user data.

3. The computer-implemented method of claim 2, wherein granting the sandboxed code (206) access to the resource (214) does not increase a threat level to user data when the resource (214) is affiliated with a source resource (212) that provided the sandboxed code to the client.

4. The computer-implemented method of claim 1, wherein the evidence is transmitted to the resource (214) for comparing the evidence against the resource based policy (218).

5. The computer-implemented method of claim 1, wherein the resource based policy (218) is transmitted to the client (202) for comparing the evidence against the resource based policy (218).

6. The computer-implemented method of claim 1, further comprising alternatively denying access by the sandboxed code (206) to the resource (214) when the resource based policy (218) dictates that access to the resource (214) should be denied and the access to the resource (214) is otherwise allowed.

7. The computer-implemented method of claim 1, wherein the resource based policy (218) is applied for managing access to the resource (214) along with a static policy.

8. The computer-implemented method of claim 1, wherein the service connection (220) corresponds to a communication connection other than an unrestricted connection (224).

9. A computer-readable medium having stored thereon instructions that when executed implement the computer-implemented method of claim 1.

10. A system, comprising:

a sandboxed application (206) that is included on a client (202);

a resource (212), wherein the resource (212) is unaccessible to the sandboxed application (206) according to a static policy;

a resource based policy (216), wherein the resource based policy is associated with resource;

a client security system (204) that is configured to evaluate evidence of the authenticity of the sandboxed application (206), wherein the client security system (204) grants the sandboxed code access to the resource (212) when the evidence is sufficient according to the resource based policy (216).

11. The system of claim 10, wherein the static policy is supplied by the client security system (204).

12. The system of claim 10, wherein the client security system (204) is further configured to determine that the evidence is sufficient according to the resource based policy (216) when granting the sandboxed code access to the resource (212) does not increase a threat level to user data.

13. The system of claim 10, further comprising a service connection (220) that is configured to transmit data between the resource (212) and the sandboxed application (206) for the purposes of evaluating the evidence against the resource based policy (216).

14. The system of claim 13, wherein the service connection is further configured to transmit the evidence to the resource for the evaluation.

15. The system of claim 13, wherein the service connection (220) is further configured to transmit the resource based policy (216) to the client security system (204) for the evaluation.

16. The system of claim 10, wherein the resource (212) corresponds to web server.

17. The system of claim 10, wherein the resource (212) corresponds to database server.

18. The system of claim 10, wherein the client security system (204) is further configured to determine that the evidence is sufficient according to the resource based policy (216) when the sandboxed application (206) is identified as associated with a source resource that is affiliated with the resource.

19. A computer-readable medium having stored thereon instructions that when executed implement the system of claim 10.

20. A computer-readable medium having stored thereon computer-executable instructions for dynamically managing access to a resource by sandboxed code included on a client, the computer-executable instructions comprising:

establishing a service connection (220) between the resource (212) and the client (202), wherein the service connection (220) is separate from an unrestricted connection;

comparing evidence of the sandboxed code (206) against a resource based policy (216) associated with the resource (212); and

dynamically providing the sandboxed code (206) access to the resource (212) when the evidence is determined as sufficient according to the resource based policy (216), wherein the evidence is determined as sufficient when granting the sandboxed code (206) access to the resource (212) does not increase a threat level to a user.

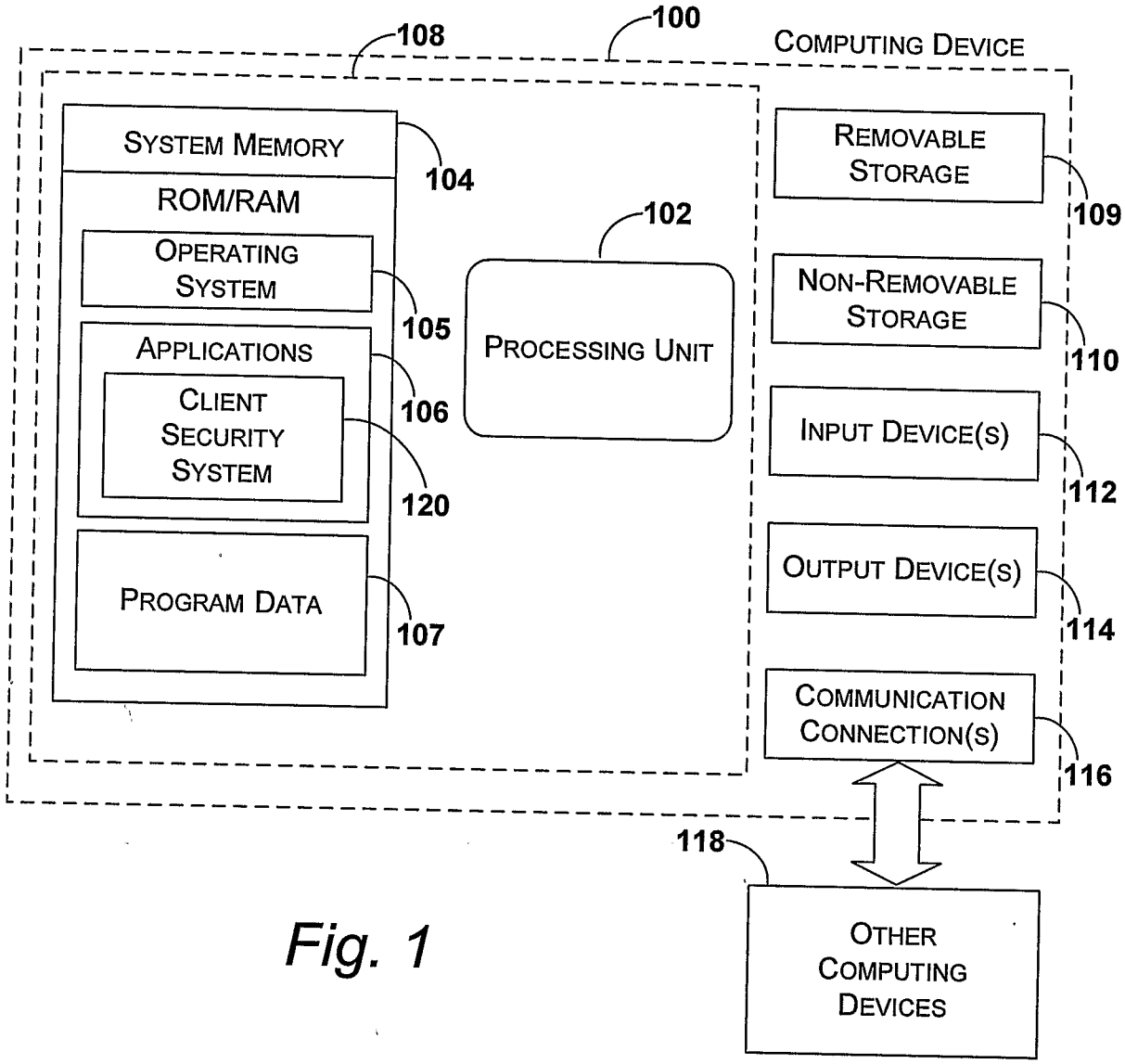

100
COMPUTING DEVICE
108
SYSTEM MEMORY
104
ROM/RAM
OPERATING SYSTEM
105
APPLICATIONS
106
CLIENT SECURITY SYSTEM
120
PROGRAM DATA
107
102
PROCESSING UNIT
REMOVABLE STORAGE
109
NON-REMOVABLE STORAGE
110
INPUT DEVICE(S)
112
OUTPUT DEVICE(S)
114
COMMUNICATION CONNECTION(S)
116
118
OTHER COMPUTING DEVICES

Fig. 1

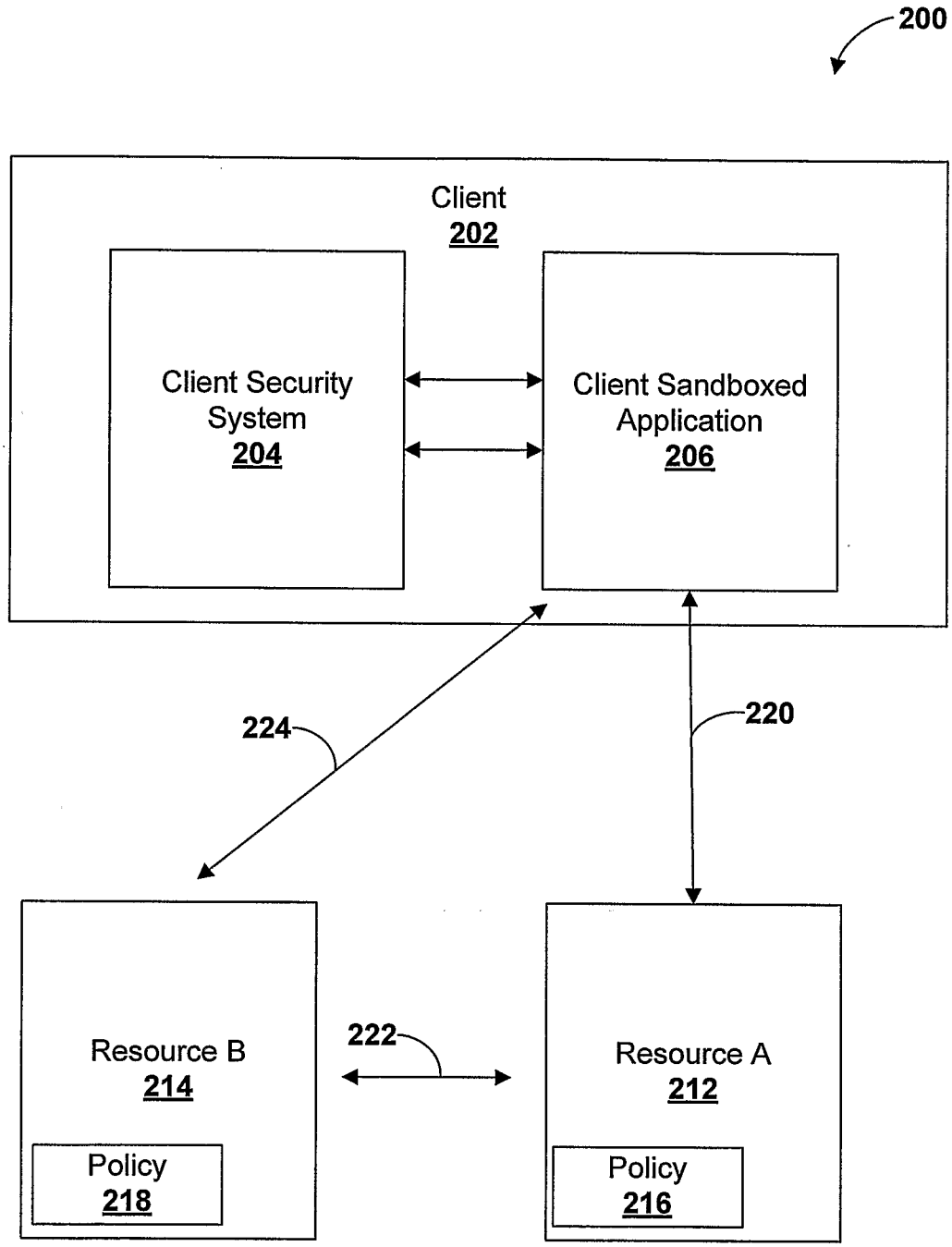

200
Client
202
Client Security System
204
Client Sandboxed Application
206
224
220
Resource B
214
Policy
218
222
Resource A
212
Policy
216

Fig. 2

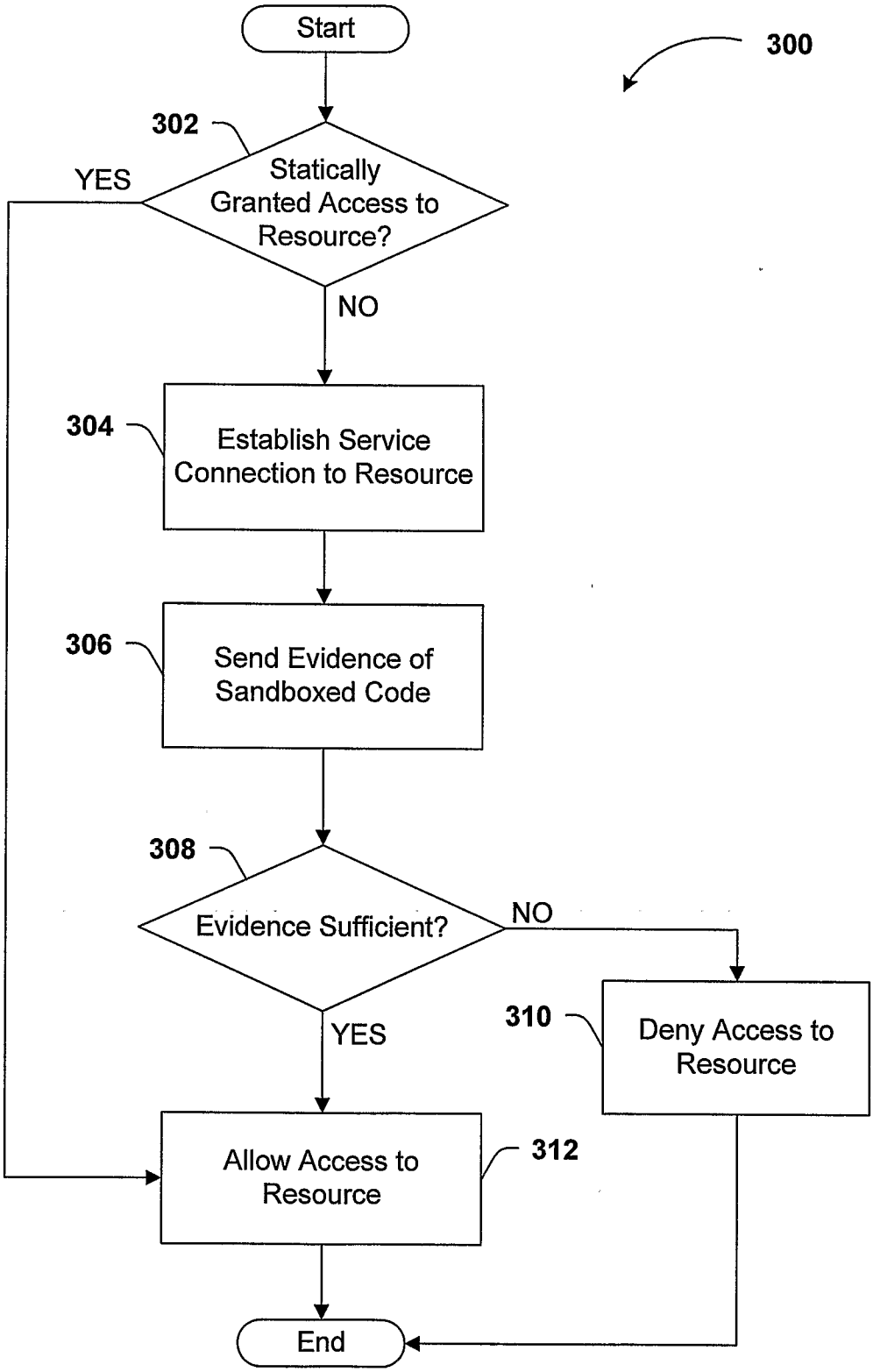

Start
300
302
Statically Granted Access to Resource?
YES
NO
304
Establish Service Connection to Resource
306
Send Evidence of Sandboxed Code
308
Evidence Sufficient?
NO
YES
310
Deny Access to Resource
Allow Access to Resource
312
End

Fig. 3

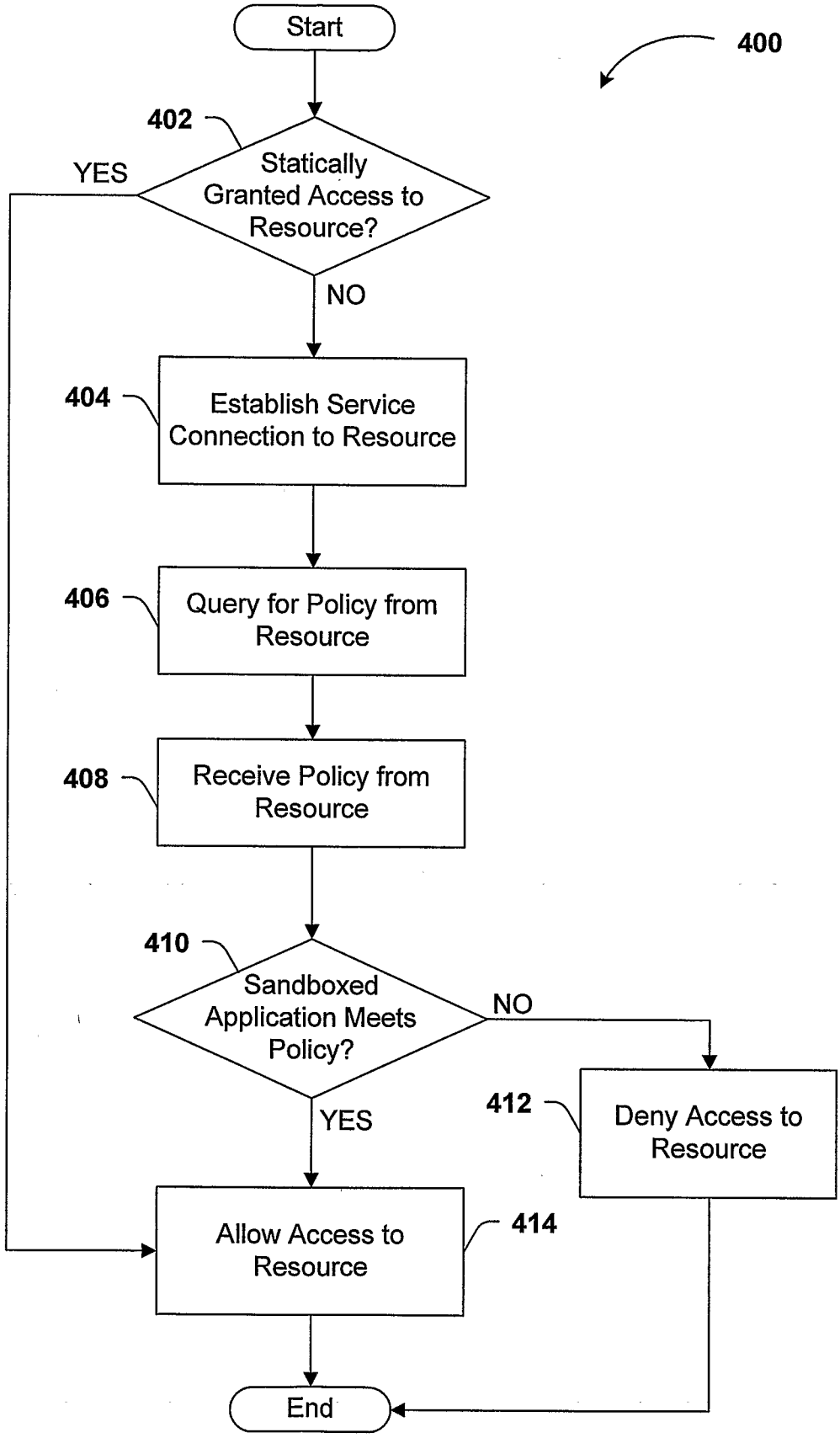

Start
400
402
Statically Granted Access to Resource?
YES
NO
404
Establish Service Connection to Resource
406
Query for Policy from Resource
408
Receive Policy from Resource
410
Sandboxed Application Meets Policy?
NO
YES
412
Deny Access to Resource
Allow Access to Resource
414
End

Fig. 4

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2006/033823

A. CLASSIFICATION OF SUBJECT MATTER

G06F 15/00(2006.01)i

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)

IPC8 G06F, H04L

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

Korean Patents and applications for inventions since 1975
Korean Utility models and applications for Utility models since 1975
Japanese Utility models and application for Utility models since 1975

Electronic data base consulted during the intertnational search (name of data base and, where practicable, search terms used)

eKIPASS(KIPO internal) "sandboxed code client"

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2005/0091536 A1 (Ray Whitmer et al.) 28 APRIL 2005 See abstract & Fig 1	1-20
A	US 2005/0177635 A1 (Roland Schmidt et al.) 11 AUGUST 2005 See abstract	1-20
A	US 6,799,208 B1 (MICROSOFT CORPORATION) 28 SEPTEMBER 2004 See abstract	1-20
A	US 6,092,194 A (FINJAN SOFTWARE, LTD.) 18 JULY 2000 See the whole document	1-20
A	US 2004/0109410 A1 (IBM CORP.) 10 JUNE 2004 See column 8-11 & Fig 1-3	1-20

☐ Further documents are listed in the continuation of Box C.

☒ See patent family 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 application or patent 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 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
05 JANUARY 2007 (05.01.2007)	**08 JANUARY 2007 (08.01.2007)**

Name and mailing address of the ISA/KR

Korean Intellectual Property Office
920 Dunsan-dong, Seo-gu, Daejeon 302-701,
Republic of Korea

Facsimile No. 82-42-472-7140

Authorized officer

YEO, Won Hyeon

Telephone No. 82-42-481-5696

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.
PCT/US2006/033823

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2005/0091536A1	28.04.2005	None	None
US 2005/0177635A1	11.08.2005	CA2547825A1	07.07.2005
		EP01700214A2	13.09.2006
		WO2005062571A2	07.07.2005
		WO2005062571A3	23.03.2006
US 06799208B1	28.09.2004	AU200149769A1	12.11.2001
		EP01342156A2	10.09.2003
		JP2004508611T2	18.03.2004
		US2005033846A1	10.02.2005
		US2005044205A1	24.02.2005
		WO200184301A2	08.11.2001
		WO200184301A3	17.04.2003
US 06092194A	18.07.2000	CA2275771AA	22.05.1998
		EP965094A2	22.12.1999
		EP965094A4	29.12.2004
		IL129729A0	19.03.2001
		JP2002514326T2	14.05.2002
		US2005005107A1	06.01.2005
		US2005108554A1	19.05.2005
		US2005240999AA	27.10.2005
		US2006149968AA	06.07.2006
		US6804780B1	12.10.2004
US 2004/0109410A1	10.06.2004	AU2003286240A1	30.06.2004
		CN1742261A	01.03.2006
		JP2006520937T2	14.09.2006
		WO2004053693A3	16.09.2004