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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
9 December 2004 (09.12.2004)

PCT

(10) International Publication Number
WO 2004/107647 A1

(51) International Patent Classification⁷: H04L 9/00

(21) International Application Number: PCT/US2003/015709

(22) International Filing Date: 17 May 2003 (17.05.2003)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: **MICROSOFT CORPORATION** [US/US]; One Microsoft Way, Redmond, WA 98052 (US).

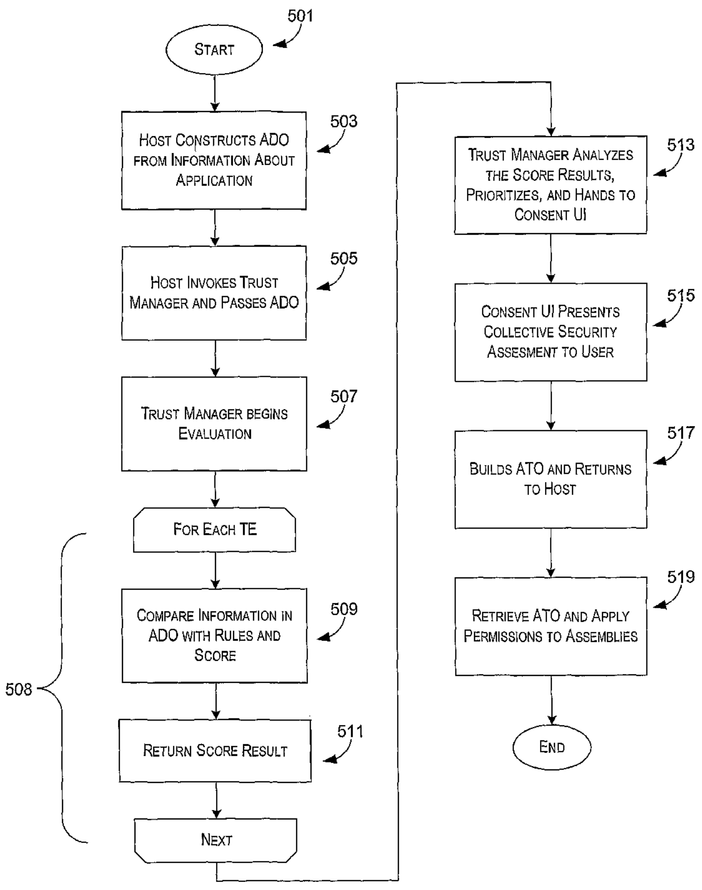
(72) Inventors: **GOLDFEDER, Aaron**; 6571A 5th Ave NE, Seattle, WA 98115 (US). **HAWKINS, John**; 14140 Batten Road, Duvall, WA 98019 (US). **KHORUN, Serge**; 16529 NE 36th Ave MM103, Redmond, WA 98052 (US). **RAMDATMISIER, Viresh**; 105 25th Avenue, Seattle, WA 98122 (US). **FARRO, Joe**; 15235 108th Place NE, Bothell, WA 98011 (US). **FEE, Gregory, Darrell**; 500 Wall St, Apt. 1521, Seattle, WA 98121 (US). **EPLING, Jeremy**; 6529 Latona Avenue NE, Seattle, WA 98115 (US). **BYBEE, Andrew**; 15715 NE 111th St, Redmond, WA 98052 (US). **XU, Jingyang**; 9539 172nd Ave NE, Redmond, WA 98052 (US). **SCHREINER, Tony**; 18666 Redmond Way Apt. TT 2156, Redmond, WA 98052 (US). **COOL, Jamie**; 10023 177th Ave NE, Redmond, WA 98052 (US).

(74) Agent: **DAIGNAULT, Ronald, A.**; Merchant & Gould P.C., P.O. Box 2903, Minneapolis, MN 55402-0903 (US).

(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, 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, NO, NZ, OM, PH, PL, PT, RO, RU, SC, SD, SE,

[Continued on next page]

(54) Title: MECHANISM FOR EVALUATING SECURITY RISKS


501
Start
503
Host Constructs ADO from Information About Application
505
Host Invokes Trust Manager and Passes ADO
507
Trust Manager begins Evaluation
For Each TE
509
Compare Information in ADO with Rules and Score
511
Return Score Result
Next
508
513
Trust Manager Analyzes the Score Results, Prioritizes, and Hands to Consent UI
515
Consent UI Presents Collective Security Assesment to User
517
Builds ATO and Returns to Host
519
Retrieve ATO and Apply Permissions to Assemblies
End

(57) Abstract: Described is a mechanism for collectively evaluating security risks associated with loading an application. A hosting environment associated with loading the application invokes a trust manager (505) to evaluate the security risks. The trust manager invokes a plurality of trust evaluators, where each trust evaluator is responsible for analyzing and assessing a different security risk (507). Upon completion of each security risk evaluation, results of those individual security risk evaluations are returned to the trust manager.(511) The trust manager aggregates the variety of security risk evaluation results and makes a security determination based on the aggregated evaluation results.(513) That determination may be to move forward with loading the application, to block the load of the application, or perhaps to prompt the user for a decision about whether to move forward with the load.

WO 2004/107647 A1

SG, SK, SL, TJ, TM, TN, TR, TT, TZ, UA, UG, UZ, VC, VN, YU, ZA, ZM, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IT, LU, MC, NL, PT, RO, SE, SI, SK, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Declaration under Rule 4.17:
— *as to non-prejudicial disclosures or exceptions to lack of novelty (Rule 4.17(v)) for all designations*

Published:
— *with international search report*
— *with a declaration as to non-prejudicial disclosures or exceptions to lack of novelty*

MECHANISM FOR EVALUATING SECURITY RISKS

This application is being filed as a PCT application filed May 17, 2003 by MICROSOFT CORPORATION., a United States national and resident, designating all countries except US.

Field of the Invention

The present invention relates to computer security systems. More particularly, the present invention relates to a mechanism for evaluating and aggregating security assessment information for computing systems.

Background of the Invention

Computer users today have access to a multitude of different applications and utilities. The typical computer user may install dozens of computer programs on a computer over the course of a year. Most times, computer users knowingly install programs on their computers. For instance, a user may purchase a software program and install it manually. Sometimes a user may install a program unknowingly, such as by visiting a particular Web site that is configured to install an applet or small program on the users computer. Installing programs on computers has become so commonplace today that some users are unaware of the security issues involved with installing new software. Other users are keenly aware of the security issues in general, but are typically uncertain about the particular issues that may surround installing a particular program.

Most users understand that new programs can introduce viruses or other malicious code on their computers. Users also understand that some software developers make programs freely available that have an overt function or purpose, such as enhancing e-mail messages, and a covert function or purpose, such as recording information about the user that is later returned a marketing entity. This particular type of software is often referred to as "spyware." So users often try to protect themselves from these security threats in various way. For instance, many users install anti-virus utilities to protect themselves against viruses. Fewer users also install anti-spyware utilities to address the spyware security issues.

Unfortunately, each security utility operates separately from each other and without knowledge of each other's results, thus burdening the user with assimilating the information from each security utility. Security systems today operate in a vacuum with respect to each other, and each reports to the user only on its specific security risk. Most users do not want separate notifications of different security risks from several disparate systems. Rather, they want their security systems just to work. The patchwork nature of security utilities today typically leaves users in fear that they have left a hole in their defenses, and that malicious or undesirable programs will slip through. Because of that fear, many users are reluctant to try new programs, especially in online environments.

Unfortunately, there are currently no mechanisms that can protect a user from multiple disparate security risks presented by a particular software program when it is being downloaded, installed, or executed. An adequate mechanism for evaluating security risks has eluded those skilled in the art.

Summary of the Invention

The present invention is directed at a system and method for accumulating security assessment information about a program and operating on that information in a convenient and usable fashion. Briefly stated, a hosting environment is responsible for loading an application. In response to the initiation of the application load, the hosting environment invokes a trust manager to evaluate any security risks associated with that application. The trust manager invokes a plurality of trust evaluators, where each trust evaluator is responsible for analyzing and assessing a different security risk. Upon completion of each security risk evaluation, results of those individual security risk evaluations are returned to the trust manager. The trust manager aggregates the variety of security risk evaluation results and makes a security determination based on the aggregated evaluation results. That determination may be to move forward with loading the application, to block the load of the application, or perhaps to prompt the user for a decision about whether to move forward with the load. Advantageously, if prompted, the user can make a decision based on the collective security assessment of the application, which provides the user with a greater sense of protection about his computer system in general.

Brief Description of the Drawings

FIGURE 1 is a functional block diagram that illustrates a computing device that may be used in implementations of the present invention.

FIGURE 2 is a functional block diagram generally illustrating components of a system for performing a security evaluation of an application and for presenting a user with a collective security assessment of that evaluation.

FIGURE 3 is a graphical representation of one illustrative grant set for an application that associates particular permissions with components of the application in the context of the application.

FIGURE 4 is a graphical representation of one illustrative user interface that may be used to present collective security assessment information to a user.

FIGURE 5 is a logical flow diagram generally illustrating a process for evaluating the security risks associated with an application and for presenting to a user the collective security assessment.

Detailed Description of the Preferred Embodiment

The invention will be described here first with reference to one example of an illustrative computing environment in which embodiments of the invention can be implemented. Next, a detailed example of one specific implementation of the invention will be described. Alternatives implementations may also be included with respect to certain details of the specific implementation. It will be appreciated that embodiments of the invention are not limited to those described here.

Illustrative Computing Environment of the Invention

FIGURE 1 illustrates a computing device that may be used in illustrative implementations of the present invention. With reference to FIGURE 1, one exemplary system for implementing the invention includes a computing device, such as computing device **100**. In a very basic configuration, 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 program modules **106**, and may include program data **107**. This basic configuration of computing device **100** 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 **122**, mouse **123**, pen, voice input device, touch input device, scanner, etc. Output device(s) **114** such as a display, speakers, printer, etc. may also be included. These devices are well known in the art and need not be discussed at length here.

Computing device **100** may also contain communication connections **116** that allow the device to communicate with other computing devices **118**, such as over a network. Communication connections **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.

Discussion of Specific Implementation

FIGURE 2 is a functional block diagram generally illustrating components of an environment implementing the present invention. As shown in FIGURE 2, a trust evaluation system **200** is configured to evaluate an application **201** and identify any security risks associated with the application **201**. The application **201** may be any executable code that is available to the computing device **100**. There are inherently some security risks associated with executing the application **201** on the computing device **100**. For instance, the application **201** may contain a virus or it may constitute spyware. Accordingly, the system **200** is configured to analyze the application **201** to assess and quantify those risks in a meaningful way. The trust evaluation system **200** then makes a decision about loading the application **201**.

The application **201** may be composed of several components operating in conjunction. For instance, the application **201** may include multiple modules or assemblies, such as assembly A **202** and assembly B **203**. The application **201** may include metadata that describes the application and each of its constituent components. That metadata may be contained in a manifest **205** or otherwise stored in association with the application **201**. The metadata may include information such as the name of the application, the version of the application, what rights and permissions the constituent components of the application desire, privacy policy information, digital signature information, and the like.

The application **201** may be first loaded onto the computing device **100** in one of many ways. For instance, the application **201** may be downloaded during an Internet session, it may be obtained on an optical disk or other permanent storage, it may be received in an e-mail message, or through some other mechanism. In this implementation, the application **201** is loaded by and executed in a hosting environment **220**. For the purpose of this discussion, the hosting environment **220** includes any environment in which the application **201** will be executed. For instance, the hosting environment **220** may be a managed code runtime environment, a shell, another application, or the like. In this particular embodiment, the hosting environment **220** may include a priority rating based on the type of host it is. For instance, it may be determined that a hosting environment associated with an optical disk drive may pose a lower security risk than a hosting environment associated with a network session, such as the Internet. The priority rating may be used later when assigning a security score to the application **201**.

The hosting environment **220** is configured to create an Application Description Object (ADO) **221** based on the metadata about the application **201**. The hosting environment **220** includes in the ADO **221** sufficient information about the application **201** to effectively evaluate the security risks associated with the application **201**. Accordingly, the ADO **221** may include, in object form, the name of the application, the version of the application, what rights and permissions the constituent components of the application desire, privacy policy information, digital signature information, and the like. The hosting environment **220** is further configured to invoke a Trust Manager **210** to perform the evaluation.

The Trust Manager **210** may be a trusted component of an operating system resident on the computing device **100**. In this particular embodiment, the Trust Manager **210** exposes an interface that is called by the hosting environment **220** to initiate the security evaluation of the application **201**. The Trust Manager **210** receives the ADO **221** from the hosting environment **220** via the interface. The Trust Manager **201** is further configured to invoke a series of trust evaluation engines to assess the security risk associated with the application **201**. Each evaluation engine is configured to evaluate a particular class of threat based on information in the ADO **221** or on the components of the application **201** itself. For instance, evaluation engine **240** may be a scoring engine that evaluates evidence about the application, as may be contained in the ADO **221** or elsewhere, to determine the ability of the application to perform malicious acts on the computing device **100**. Evaluation engine **241** may be a virus checker and evaluation engine **242** may be configured to evaluate privacy concerns about the application **201**. Each of the evaluation engines may derive from a base class, or may be implemented as an interface.

Each evaluation engine is configured to assess the application **201** against its particular rules or criteria to determine a score **245**. Examples of the score include a numerical value between a minimum and maximum, or a discrete value from a set of alternative security levels. These are only examples and not an exhaustive list. The score **245** may then be returned to the Trust Manager **210** by each evaluation engine at the conclusion of its assessment. The Trust Manager **210** is configured to aggregate the individual scores into a score collection **250**, which represents the collective security assessment of the application in each of the areas for which an evaluation engine exists. Any priorities that may exist, such as priorities associated with the particular type of hosting environment **220**, may be applied to the score collection **250** to further refine the collective security assessment. Based on the collective security assessment, the Trust Manager **210** may have sufficient information to make a loading decision without involving the user. For instance, pre-determined thresholds (either set by default or perhaps provided by the user) may govern what programs are loaded without seeking user acceptance, or what programs are blocked without prompting the user. If the collective security assessment for the particular application being loaded falls between those two thresholds, the user may be prompted for a loading decision.

The Trust Manager **210** constructs a Trust Object **261** that describes the level of permissions with which the application will be loaded, if at all. The Trust Object **261** may include data that defines a permission grant set **262** for the application on a component-by-component basis. One example of an illustrative permission grant set **262** is illustrated in Figure 3 and described below. If the collective security assessment for the application **201** falls between the two thresholds mentioned above, the Trust Manager **210** may pass the Trust Object **261** to a User Interface **260** so that the user may be prompted.

The User Interface **260** is a mechanism for presenting the collective security assessment to the user in a meaningful way so that the user can make an informed decision about proceeding. The User Interface **260** may take many forms, such as a dialog box, an audible signal, an iconic indicator, or the like. One example of a potential User Interface **260** is illustrated in Figure 4 and described below. In essence, the User Interface **260** represents a single point of presentation for various and disparate security information that, in conventional systems, does not exist.

The User Interface **260** may prompt the user with the potential security ramifications of allowing the application load to proceed, and possibly presenting the user with various levels of permissions that may be assigned to the application. The user is asked to make a determination whether to proceed with loading the application or not. The User Interface **260** adds the user's response information to the Trust Object **261** and returns it to the Trust Manager **210**.

Each time the application **201** is launched or executed, it's hosting environment **220** could invoke the Trust Manager **210** to retrieve the security assessment of the application **201**. In the case where the grant set **262** has already been created, the Trust Manager **210** may return that grant set **262** to the hosting environment **220**. Alternatively, the hosting environment **220** could cache the security assessment information for subsequent use without involving the Trust Manager **210**. The hosting environment **220** will then apply any access permissions identified in the grant set **262** to the application **201**. More specifically, the hosting environment **220** may apply the access permissions to each individual component, such as assembly A **202**, of the application **201**. It is equally feasible that the hosting environment **220** or some other application may present a component to the Trust Manager **210** for a security assessment without the specific intent of then executing the component.

FIGURE 3 is a graphical representation of one illustrative grant set **301** that may be generated by implementations of the present invention. It should be noted that the term "grant set," as used in this document, means any collection of information that is used to define the security environment in which an application may execute. The term "grant set" used in this document is not limited to a particular security environment, such as a Common Language Runtime environment, but rather is intended to cover information used to define the security environment within which an application executes regardless of the particular operating environment.

In this particular example, the grant set **301** may be data within an object, such as a Trust Object or the like. In this example, the grant set **301** includes information that identifies each component of the application. In addition, the grant set **301** includes information that defines the permissions for each component of the application. In this case, a components table **310** identifies the components Assembly A, Assembly B, and Assembly C and associates each of those components with a permission set. For instance, in the grant set **301**, Assembly A is identified as having permission set PS1.

A permissions table **320** is also included in the grant set **301** to define specifically those permissions are security rights that are associated with each permission set. In this example, permission set PS1 includes those permissions and rights identified in the example as Permissions 1. It will be appreciated that, as described above, when the hosting environment **220** begins to load the components of the application, by referring to the grant set **301** the appropriate permissions may be applied to each component of the application in the context of the application. In other words, some other application may also include Assembly B, but in the context of that other application, Assembly B may have a different permission set. In that case, when the other application was executed, and Assembly B was loaded, it would have the permission set defined by a grant set associated with the other application.

FIGURE 4 is an illustrative User Interface dialogue that may be presented to a user based on a security assessment of an application. In this particular example, the dialog **401** is presented based on an evaluation of an application that has requested access to the file system and the network. In addition, a virus evaluator has determined that the application does not contain a virus. In iconic indication of the risk level **405** may also be included. The user is presented with the option of allowing the load to proceed, such as by clicking an OK button **410**, or to abort the load. The User Interface shown in FIGURE 4 is for the purpose of illustration only, and is not to be viewed as limiting or the exclusive mechanism for presenting security information to the user. Indeed, it is envisioned that very many different forms of collective security assessment presentation will become apparent from the teachings of this document.

FIGURE 5 is a logical flow diagram generally illustrating a process for identifying and collectively presenting, in meaningful way, information about security risks posed by an application. The process begins at starting block **501**, where an application is being loaded for execution on a computing system. As discussed above, an application may be loaded in many ways through various types of hosts. Accordingly, at starting block **501**, a particular application is being loaded through use of a particular host. The process continues at block **503**.

At block **503**, the host constructs an Application Description Object (ADO) based on information about the application. As described above, the information may be obtained from a manifest included with the application, or through any other metadata associated with the application. The ADO contains descriptive information about the application, such as the name and version of the application, any rights being requested by the application, any code access permissions being requested by the application, digital signature information related to the application, privacy policy information, and the like. The process continues at block **505**.

At block **505**, the host invokes a Trust Manager with an instruction to evaluate the security risks associated with the application. The host passes the ADO to the Trust Manager for use in the evaluation.

At block **507**, the Trust Manager begins evaluating the security risks of the application by invoking a series of Trust Evaluators that each evaluate a specific area of security risk. For instance a virus evaluator may be configured to examine each component of an application for the possibility that the application contains a virus. A privacy evaluator may evaluate the permissions requested by the application to determine what level of threat to privacy the application presents. Many other Trust Evaluators may also be used, as will be apparent to those skilled in the art.

Loop **508** is performed for each Trust Evaluator in the system. The Loop **508** begins at block **509**, where the current Trust Evaluator examines the information in the ADO and/or the components of the application to assess the security risk. The information in the ADO may be compared against a set of rules or other criteria to build a score that quantifies the security risk of the application. In one example, a score may be a value from zero (maximum risk) to one (minimum risk). The score may also include a priority and a string descriptor.

It will be appreciated that the evaluations being performed by each Trust Evaluator are analogous to similar security risk evaluations that may be performed by conventional mechanisms. However, in accordance with the invention, each Trust Evaluator assesses its respective security risk and returns the score collection to the Trust Manager (block **511**). When each Trust Evaluator has returned its score collection to the Trust Manager, the loop **508** terminates and the process continues to block **513**.

At block **513**, the Trust Manager analyzes the score collections from the Trust Evaluators. The Trust Manager may prioritize the score collections based on some pre-determined criteria, such as a priority associated with a particular Trust Evaluator, or some other prioritization scheme. For instance, a high risk that a virus is present may outweigh a low risk that a privacy violation may occur. The Trust Manager determines, from the prioritized score collections, an aggregate security impact on the computing system. If the aggregate security impact on the system exceeds some pre-determined threshold, the Trust Manager may simply block the load of the application. If the aggregate security impact is below some other threshold, the Trust Manager may simply build a Trust Object that includes sufficient permissions for the application to execute. If however, neither of these cases exists, the Trust Manager may invoke a User Interface to prompt the user to make the determination.

At block **515**, the Trust Manager passes the prioritized score collection and aggregate impact information to the User Interface for final evaluation if required from the user. If so, the aggregate security impact is presented to the user. The presentation may be in the form of a dialog box that summarizes or details specifically the security risks associated with loading the application. For instance, a scoring engine may have determined that the application has requested sufficient permissions to read and modify files on the computer, and to transmit data over a network connection. Based on that information, together with perhaps other evidence, a privacy evaluator may have determined that the application is likely to share the user's information over the network. Accordingly, that information may be combined to inform the user that loading the application is likely to result in the user being targeting by telemarketing campaigns or other inappropriate uses of the user's personal information. Advantageously, the user is presented with disparate security information collected into a common notification, such as a dialog box or the like.

At block **517**, with any input from the User Interface, the Trust Manager modifies the Trust Object to describe the security environment in which the application may be executed. In one embodiment, the Trust Object includes data that associates the application, or components of the application, with a permission grant set. The permission grant set describes the level of security that will be applied to the application when executed. In one specific environment, a permission grant set is associated with each component of the application. In that way, a component that is shared among different applications may be executed with different permissions depending on the application context in which it is executing. The process may idle at block **517** until the application is actually executed, thereby causing the host to begin loading components of the evocation. At that point, the process continues to block **519**.

At block **519**, the application is being loaded by the host. As part of a security policy that applies to applications being loaded, the host queries the Trust Manager for the Trust Object associated with the application. As each component of the application is loaded, the permission grant set associated with that component is applied. In this way, applications that have been loaded in accordance with the invention are only allowed those permissions which the user has, in an informed way, directly and comprehensively established. If sufficient privileges to execute have not been granted to the application, the Trust Manager may block the execution of the application.

The above specification, examples and data provide a complete description of the concepts and illustrative implementations of the 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.

WE CLAIM:

1. A computer-readable medium having computer-executable components, comprising:

a trust manager configured to receive a notification that an application is being loaded and, in response, to cause the application to be evaluated for a plurality of security risks, the trust manager being further configured to aggregate scores associated with each security risk evaluation to determine a collective security assessment based on the aggregated scores; and

a user interface configured to present the collective security assessment determined by the trust manager.

2. The computer-readable medium of claim 1, wherein the application is being loaded by a hosting environment, and wherein the hosting environment issues the notification to the trust manager.

3. The computer-readable medium of claim 2, wherein the hosting environment is configured to create an Application Description Object that includes descriptive information about the application.

4. The computer-readable medium of claim 3, wherein the descriptive information includes the name of the application and the version of the application.

5. The computer-readable medium of claim 3, wherein the descriptive information identifies rights and permissions requested by components of the application.

6. The computer-readable medium of claim 3, wherein the descriptive information includes privacy policy information.

7. The computer-readable medium of claim 3, wherein the descriptive information includes digital signature information about the application.

8. The computer-readable medium of claim 1, wherein the security risk evaluations are performed by a separate trust evaluator, each trust evaluator being configured to analyze a particular security risk associated with the application.

9. The computer-readable medium of claim 8, wherein the particular security risk comprises the presence of a virus.

10. The computer-readable medium of claim 8, wherein the particular security risk comprises a violation of privacy.

11. The computer-readable medium of claim 1, wherein the collective security assessment identifies one or more security risks associated with loading the application.

12. A computer-readable medium encoded with a data structure, comprising:

a grant set associated with an application, the grant set including a first table and a second table, the first table including a list of components constituting the application, each component being associated with a permission set, the second table including a list of permission sets and a description for each permission set.

13. A computer-implemented method, comprising:

receiving a notification that an application is being loaded by a hosting environment;

receiving an application description object that includes information about the application;

causing the application to be evaluated to determine a plurality of security risks associated with the application;

aggregating results from the evaluation of the plurality of security risks; and

presenting the aggregated results as a collective security assessment of the application.

14. The computer-implemented method of claim 13, wherein causing the application to be evaluated further comprises invoking a plurality of trust evaluators, each trust evaluator being associated with a different possible security risk, each trust evaluator being operative to assess a likelihood that the application suffers from the particular possible security risk corresponding to that trust evaluator.

15. The computer-implemented method of claim 13, further comprising assigning a permission set to the application, the permission set defining permissions with which the application will be allowed to execute.

16. The computer-implemented method of claim 15, further comprising, in response to a notification to execute the application, retrieving the permission set and causing the application to be executed with the appropriate permissions.

17. The computer-implemented method of claim 15, wherein the permission set defines permissions for components of the application.

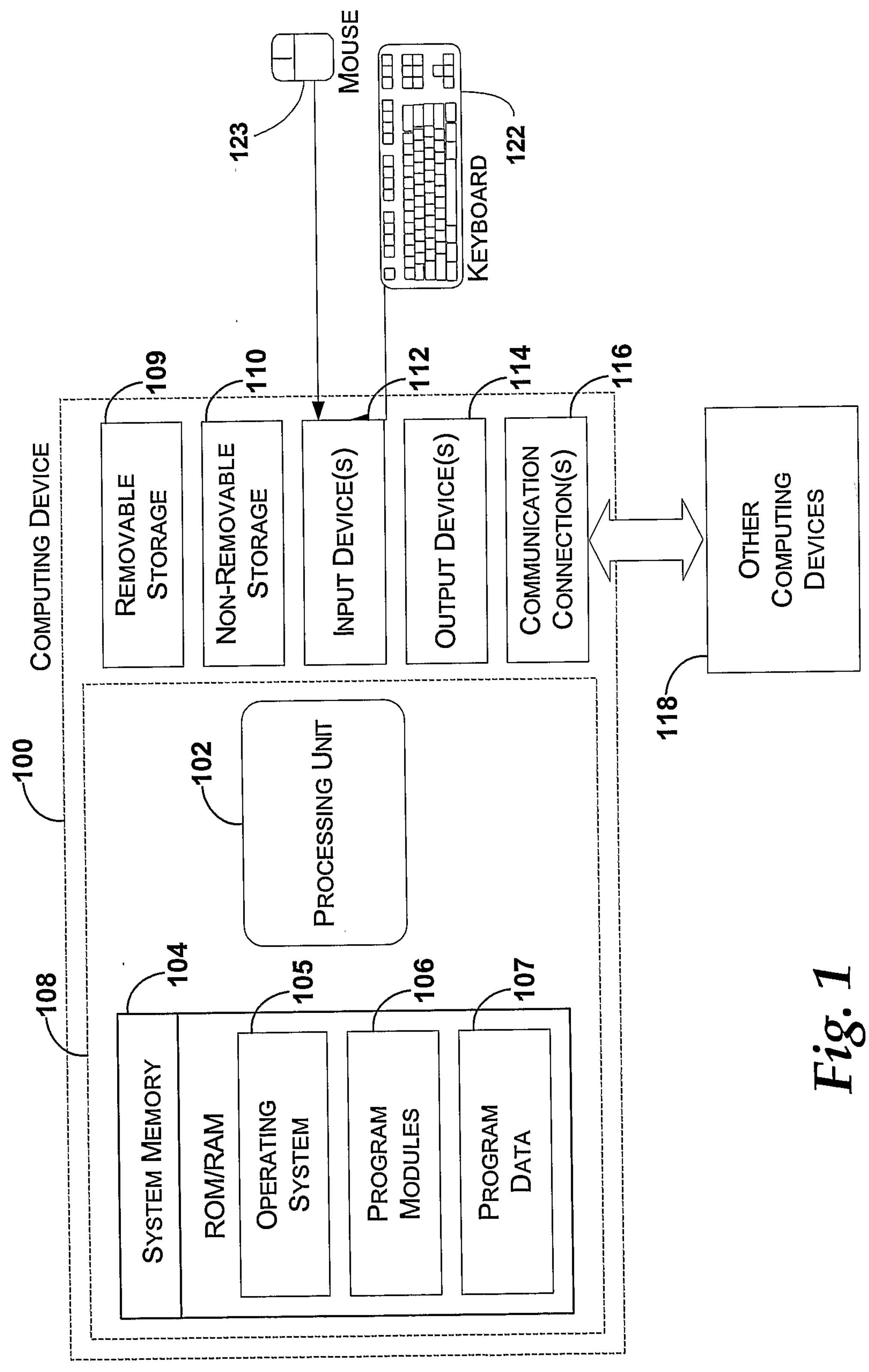

Computing Device
100
108
System Memory
104
ROM/RAM
Operating System
105
Program Modules
106
Program Data
107
Processing Unit
102
Removable Storage
109
Non-Removable Storage
110
Input Device(s)
112
Output Device(s)
114
Communication Connection(s)
116
Other Computing Devices
118
Mouse
123
Keyboard
122

Fig. 1

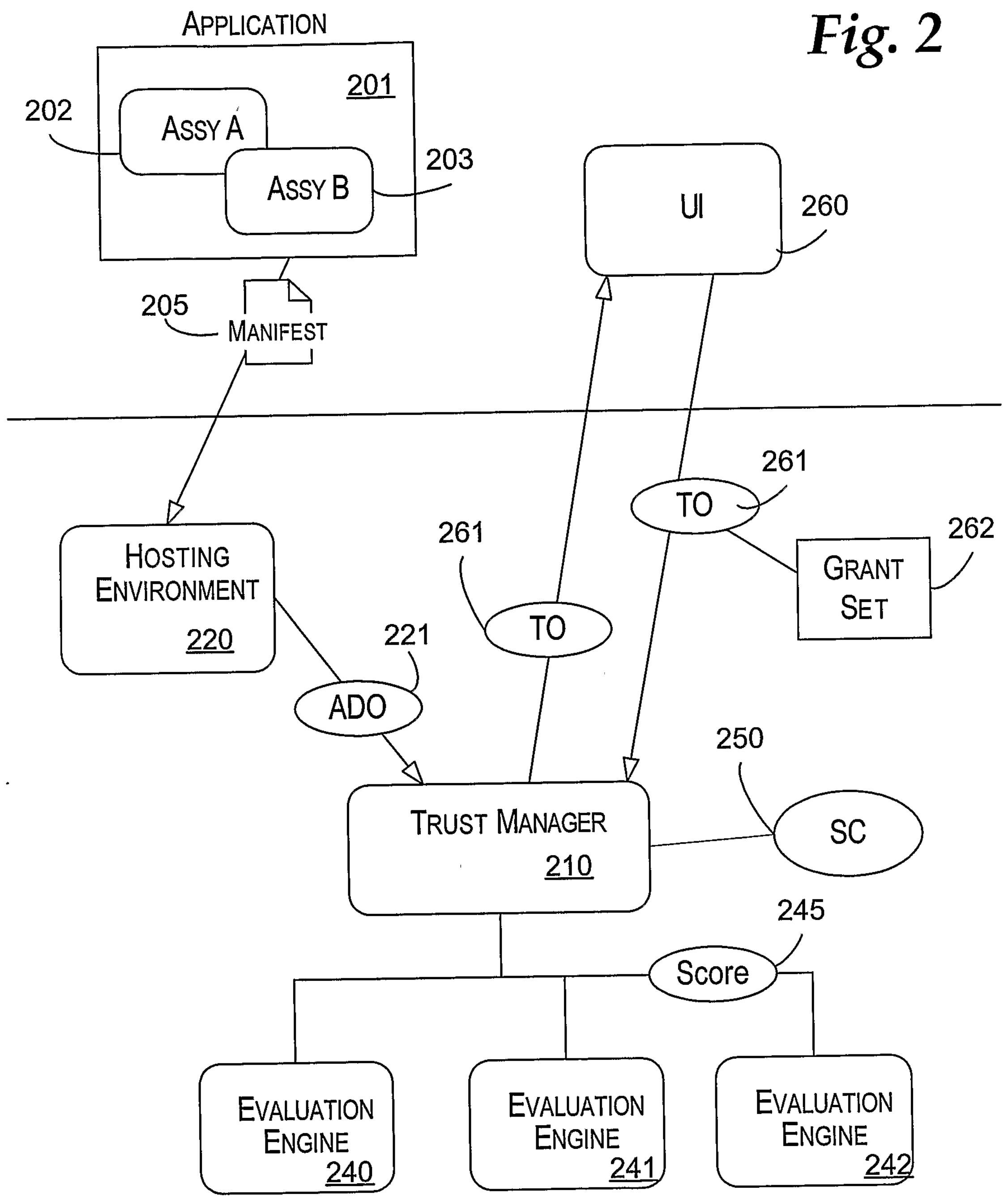

APPLICATION
201
202
ASSY A
ASSY B
203
205
MANIFEST
UI
260
261
TO
GRANT SET
262
HOSTING ENVIRONMENT
220
261
TO
221
ADO
250
SC
TRUST MANAGER
210
245
Score
EVALUATION ENGINE
240
EVALUATION ENGINE
241
EVALUATION ENGINE
242

Fig. 2

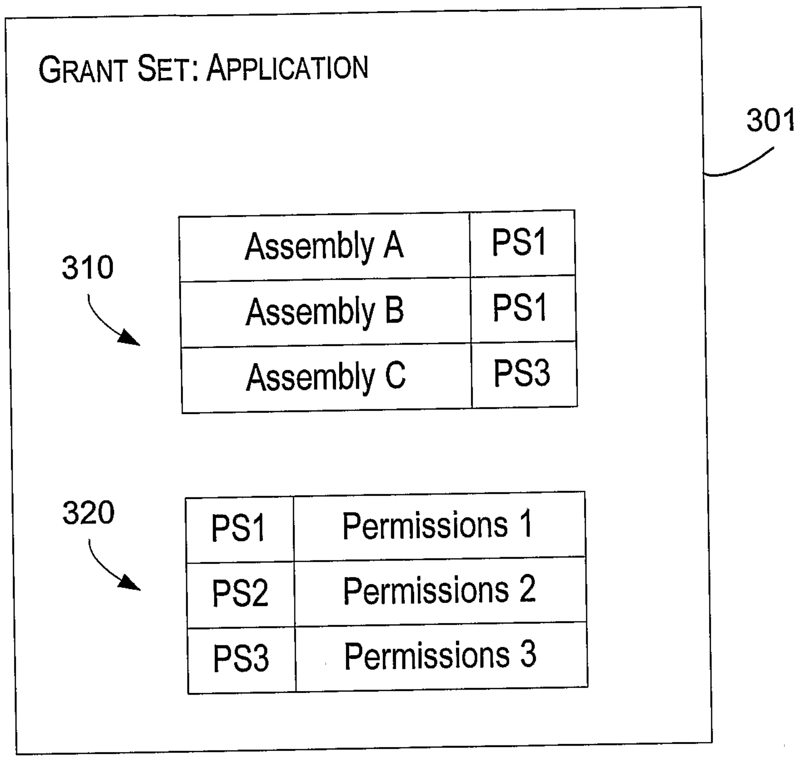
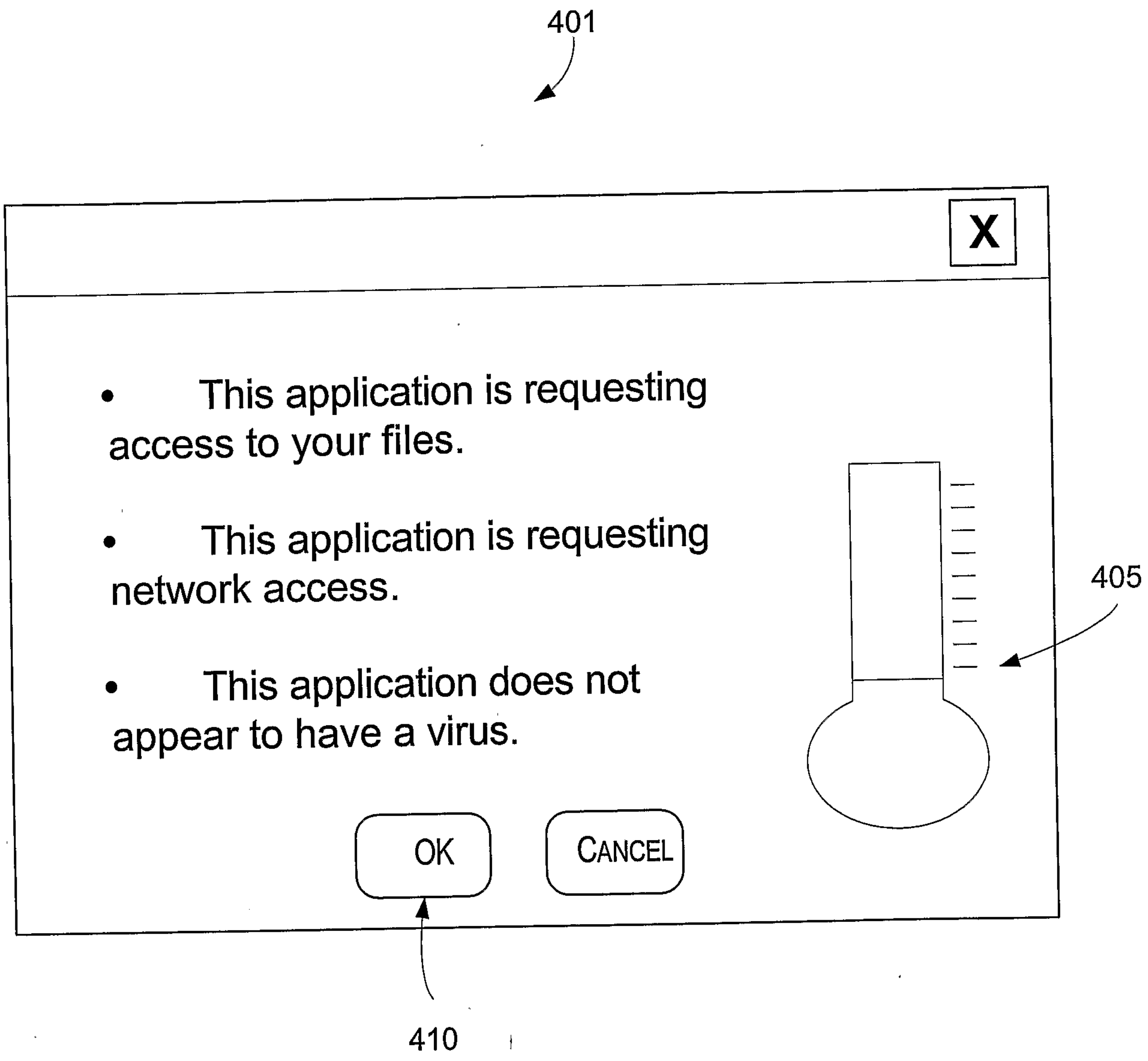

GRANT SET: APPLICATION
301
310
Assembly A
PS1
Assembly B
PS1
Assembly C
PS3
320
PS1
Permissions 1
PS2
Permissions 2
PS3
Permissions 3

FIG. 3



- This application is requesting access to your files.
- This application is requesting network access.
- This application does not appear to have a virus.

FIG. 4

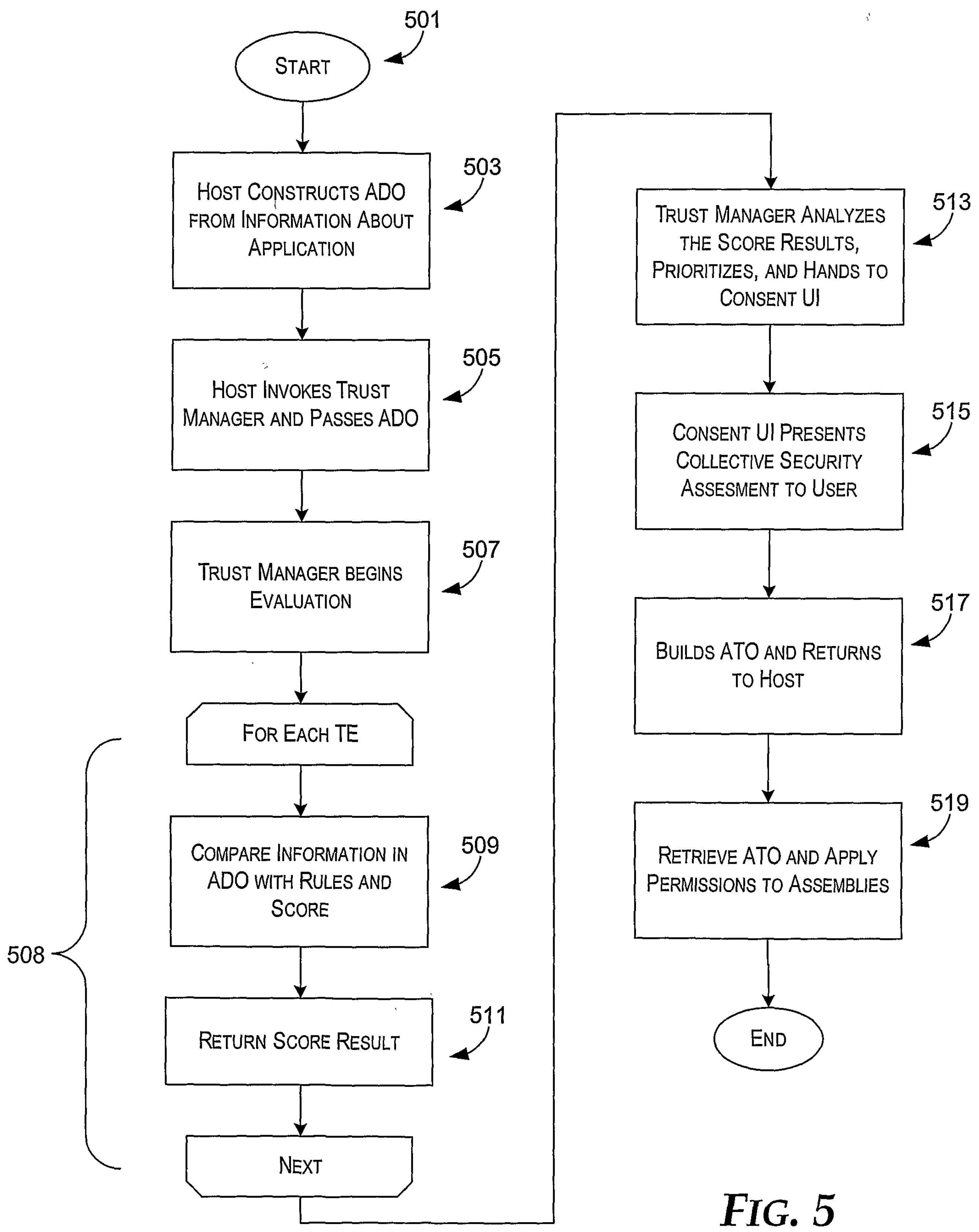

501
START
503
HOST CONSTRUCTS ADO FROM INFORMATION ABOUT APPLICATION
505
HOST INVOKES TRUST MANAGER AND PASSES ADO
507
TRUST MANAGER BEGINS EVALUATION
FOR EACH TE
508
509
COMPARE INFORMATION IN ADO WITH RULES AND SCORE
511
RETURN SCORE RESULT
NEXT
513
TRUST MANAGER ANALYZES THE SCORE RESULTS, PRIORITIZES, AND HANDS TO CONSENT UI
515
CONSENT UI PRESENTS COLLECTIVE SECURITY ASSESMENT TO USER
517
BUILDS ATO AND RETURNS TO HOST
519
RETRIEVE ATO AND APPLY PERMISSIONS TO ASSEMBLIES
END

FIG. 5

Supplemental Box *If the Supplemental Box is not used, this sheet should not be included in the request.*

1. If in any of the Boxes, **the space is insufficient** to furnish all the information: in such case, write "Continuation of Box No. .." [indicate the number of the Box] and furnish the information in the same manner as required according to the captions of the Box in which the space was insufficient in particular:

(i) **if more than two persons are involved as applicants and/or inventors** and no "continuation sheet" is available: in such case, write "Continuation of Box No. III" and indicate for each additional person the same type of information as required in Box No. III. The country of the address indicated in this Box is the applicant's State (that is, country) of residence if no State of residence is indicated below;

(ii) if, in Box No. II or in any of the sub-boxes of Box No. III, the indication **"the States indicated in the Supplemental Box"** is checked: in such case, write "Continuation of Box No. II" or "Continuation of Box No. III" or "Continuation of Boxes No. II and No. III" (as the case may be), indicate the name of the applicant(s) involved and, next to (each) such name, the State(s) (and/or, where applicable, ARIPO, Eurasian, European or OAPI patent) for the purposes of which the named person is applicant;

(iii) if in Box No. II or in any of the sub-boxes of Box No. III, **the inventor or the inventor/applicant is not inventor for the purposes of all designated States or for the purposes of the United States of America:** in such case, write "Continuation of Box No. II" or "Continuation of Box No. III" or "Continuation of Boxes No. II and No. III" (as the case may be), indicate the name of the inventor(s) and, next to (each) such name, the State(s) (and/or, where applicable, ARIPO, Eurasian, European or OAPI patent) for the purposes of which the named person is inventor;

(iv) if, in addition to the agent(s) indicated in Box No. IV, there are **further agents**: in such case, write "Continuation of Box No. IV" and indicate for each further agent the same type of information as required in Box No. IV;

(v) if in Box No. V, the name of any State (or OAPI) is accompanied by the indication **"patent of addition,"** or **"certificate of addition,"** or if, in Box No. V, the name of the United States of America is accompanied by an indication **"continuation"** or **"continuation-in-part"**: in such case, write "Continuation of Box No. V" and the name of each State involved (or OAPI), and after the name of each such State (or OAPI), the number of the parent title or parent application and the date of grant of the parent title or filing of the parent application;

(vi) if there are more than three earlier applications whose priority is claimed: in such case, write "Continuation of Box No. VI" and indicate for each additional earlier application the same type of information as required in Box No. VI.

(vii) if, in Box No. VI, the earlier application is an ARIPO application: in such case, write "Continuation of Box No. VI," specify the number of the item corresponding to that earlier application and indicate at least one country party to the Paris Convention for the Protection of Industrial Property for which that earlier application was filed.

2. If, with regard to the precautionary designation statement contained in Box No. V, the applicant wishes to exclude any State(s) from the scope of that statement: in such case, write Designation(s) excluded from precautionary designation statement" and indicate the name or two-letter code of each State so excluded.

3. If the applicant claims, in respect of any designated Office, the benefits of provisions of the national law concerning non-prejudicial disclosures or exceptions to lack of novelty: in such case, write "Statement Concerning Non-Prejudicial Disclosures or Exceptions to Lack of Novelty" and furnish that statement below.

Declaration as to non-prejudicial disclosures or exceptions to lack of novelty (Rules 4.17(v) and 51bix.1(a)(v)):
in relation to this international application No. PCT/US03/15709
MICROSOFT CORPORATION declares that the subject matter claimed in this international application was disclosed as follows:
i. Kind of Dislcosure: Abusive Disclosure on the Internet by a third party
ii. Date of Disclosure: 19 November 2002
iii. Title of Disclosure: Windows Longhorn Alpha Leaks to the Web
iv Place of Disclosure: eWeek Enterprise News & Reviews on the internet
v. This Declaration is made for the purpose of the following designations for national and/or regional patents: All designations

INTERNATIONAL SEARCH REPORT

International application No.

PCT/US03/15709

A. CLASSIFICATION OF SUBJECT MATTER

IPC(7) : HO4L 9/00
US CL : 713/176
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)
U.S. : 707/9; 713/176, 200, 201

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 practicable, search terms used)
Please See Continuation Sheet

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X --- Y	US 6,282,546 B1 (GLEICHAUF et al) 28 August 2001 (28.08.2001), column 4, lines 1-12; column 4, lines 43-50; column 5, lines 9-36.	1-5, 8, 11, 13, 14 ---------- 6, 7, 9, 10, 15-17
Y	US 5,649,185 (ANTOGNINI et al) 15 July 1997 (15.07.1997), column 15, lines 18-46.	6, 10, 15-17
Y	SCHNEIER, B., Applied Cryptography, John Wiley & Sons, Inc., 1996, pages 34-38.	7
Y	US 5,832,208 (CHEN et al) 3 November 1998 (03.11.1998), column 3, lines 48-60.	9
X, P	US 6,446,069 B1 (YAUNG et al) 3 September 2002 (03.09.2002), column 5 line 16 to column 8 line 67.	12
A	Anonymous, ForeScout Joins Symantec Technology Partner Program; ActiveScout Certified Interoperable with Symantec Enterprise Security Architecture, Business Wire, July 2003, pages 1-3.	1-17
A	Anonymous, ISS Ships New Version of Market-Leading Security Vulnerability Detection Solution, Business Wire, October 1998, pages 1-3.	1-17

☒ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

* Special categories of cited documents:	"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
"A" document defining the general state of the art which is not considered to be of particular relevance	"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
"E" earlier application or patent published on or after the international filing date	"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
"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)	"&" document member of the same patent family
"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	

Date of the actual completion of the international search 30 July 2003 (30.07.2003)	Date of mailing of the international search report 22 JUL 2004
Name and mailing address of the ISA/US Mail Stop PCT, Attn: ISA/US Commissioner for Patents P.O. Box 1450 Alexandria, Virginia 22313-1450 Facsimile No. (703)305-3230	Authorized officer for Ayaz R Sheikh [illegible] Telephone No. 703-305-9648

INTERNATIONAL SEARCH REPORT

PCT/US03/15709

C. (Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT		
Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	Connolly, Julie, Operation Chain Link The Deployment of a Firewall at Hanscom Air Force Base, 1996, pages 170-177.	1-17
A	Shostack et al, Towards a Taxonomy of Network Security Assessment Techniques, July 1999, pages 1-11.	1-17

Continuation of B. FIELDS SEARCHED Item 3:
EAST, Proquest Direct, GOOGLE, IEEE
search terms: network assessment, network vulnerability, network scanning, digital signatures, access rights, access policies, virus detection