US007174566B2

(12) **United States Patent**
Yadav

(10) **Patent No.: US 7,174,566 B2**
(45) **Date of Patent: Feb. 6, 2007**

(54) **INTEGRATED NETWORK INTRUSION DETECTION**

(75) Inventor: **Satyendra Yadav**, Portland, OR (US)

(73) Assignee: **Intel Corporation**, Santa Clara, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 847 days.

(21) Appl. No.: **10/066,140**

(22) Filed: **Feb. 1, 2002**

(65) **Prior Publication Data**

US 2003/0149888 A1 Aug. 7, 2003

(51) **Int. Cl.**
G06F 21/22 (2006.01)
G06F 21/20 (2006.01)
G06F 21/06 (2006.01)

(52) **U.S. Cl.** **726/26**; 726/22; 713/152; 713/188

(58) **Field of Classification Search** 713/200
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,421,006	A	5/1995	Jablon et al.	
5,802,275	A	9/1998	Blonder	
5,919,257	A	7/1999	Trostle	
5,948,104	A	9/1999	Gluck et al.	
5,970,143	A	10/1999	Schneier et al.	
5,978,936	A	11/1999	Chandra et al.	
5,983,348	A	11/1999	Ji	
6,219,706	B1 *	4/2001	Fan et al.	709/225
6,266,811	B1	7/2001	Nabahi	
6,272,641	B1	8/2001	Ji	
6,279,113	B1	8/2001	Vaidya	
6,282,546	B1	8/2001	Gleichauf et al.	
6,301,668	B1	10/2001	Gleichauf et al.	
6,370,584	B1	4/2002	Bestavros et al.	

(Continued)

FOREIGN PATENT DOCUMENTS

WO 00/34867 6/2000

(Continued)

OTHER PUBLICATIONS

Stallings, William. "Intruders." Cryptography and Network Security-Principles and Practices. Upper Saddle River, NJ:Prentice Hall, 2003.580-581.*

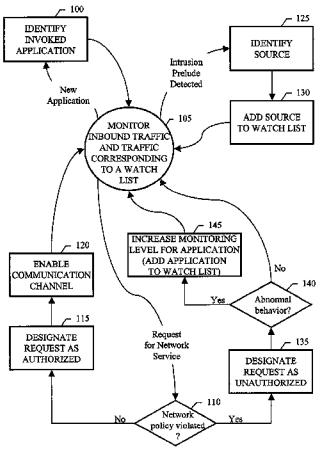
(Continued)

Primary Examiner—Kambiz Zand
Assistant Examiner—Venkat Perungavoor
(74) *Attorney, Agent, or Firm*—Fish & Richardson P.C.

(57) **ABSTRACT**

Intrusion preludes may be detected (including detection using fabricated responses to blocked network requests), and particular sources of network communications may be singled out for greater scrutiny, by performing intrusion analysis on packets blocked by a firewall. An integrated intrusion detection system uses an end-node firewall that is dynamically controlled using invoked-application information and a network policy. The system may use various alert levels to trigger heightened monitoring states, alerts sent to a security operation center, and/or logging of network activity for later forensic analysis. The system may monitor network traffic to block traffic that violates the network policy, monitor blocked traffic to detect an intrusion prelude, and monitor traffic from a potential intruder when an intrusion prelude is detected. The system also may track behavior of applications using the network policy to identify abnormal application behavior, and monitor traffic from an abnormally behaving application to identify an intrusion.

35 Claims, 8 Drawing Sheets



U.S. PATENT DOCUMENTS

6,463,470	B1	10/2002	Mohaban et al.	
6,466,984	B1	10/2002	Naveh et al.	
6,496,483	B1 *	12/2002	Kung et al.	370/252
6,501,752	B1 *	12/2002	Kung et al.	370/352
6,553,377	B1	4/2003	Eschelbeck et al.	
6,694,436	B1	2/2004	Audebert	
6,742,015	B1	5/2004	Bowman-Amuah	
6,751,659	B1	6/2004	Fenger et al.	
6,816,903	B1	11/2004	Rakoshitz et al.	
6,816,973	B1	11/2004	Gleichauf et al.	
6,842,861	B1	1/2005	Cox et al.	
6,851,057	B1	2/2005	Nachenberg	
6,892,303	B2	5/2005	Le Pennec et al.	
6,996,843	B1 *	2/2006	Moran	726/23
6,996,845	B1 *	2/2006	Hurst et al.	726/25
2001/0052012	A1	12/2001	Rinne et al.	
2002/0010771	A1	1/2002	Mandato	
2002/0120853	A1 *	8/2002	Tyree	713/188
2002/0129278	A1 *	9/2002	Elgressy et al.	713/201
2002/0143911	A1	10/2002	Vicente et al.	
2002/0143914	A1	10/2002	Cihula	
2002/0194317	A1	12/2002	Kanada et al.	
2003/0126468	A1 *	7/2003	Markham	713/201
2003/0149887	A1	8/2003	Yadav	
2003/0204596	A1	10/2003	Yadav	

FOREIGN PATENT DOCUMENTS

WO	01/17161	3/2001
WO	01/37511	5/2001
WO	01/84270	11/2001

OTHER PUBLICATIONS

Sean Boran, "Personal Firewalls Tests" Sygate, Internet Publication , 'Online!, Apr. 23, 2001, XP-002248365; http://www.boran.com/security/sp/pf/p.

Sygate: "Sygate Personal Firewall Pro User Guide", Sygate User Manual, 'Online! 2001, XP002248366; http.//smb/sygate.com/support/userguides/pspf/pspf42_userguide.pdf, pp. 1-77.

Nokia, "Combining Network Intrusion Detection with Firewalls for Maximum Perimeter Protection", Nokia White Paper, 'Online!, Apr. 2001, XP002248367; http://www.nwfusionpartners.com/nokia/preffered/media/Nokia%20WP%20IDS%20and%20Firewall.pdf.

Jerry T. Oney, "Intrusion Detection Systems, IDS", Lecture Materials, 'Online!, Aug. 10, 2001, XP002248368; http://www.nvcc.edu/home/joney/Intrusion%20Detection.ppt, pp. 34-62.

Secure Hash Standard (available at http://csrc.nist.gov/encryption/tkhash.html); Federal Information Processing Standards Publication 108-1; Apr. 17, 1995.

Windows 2000 Network Architecture, NDIS Driver Types; http://www.microsoftcom/windows2000/techinfo/reskit/en-us/cnet/cnad_arc_zjmj.asp?fra; Jan. 8, 2002.

Windows 2000 Network Architecture, Overview of Windows 2000 Network Architecture; http://www.microsoftcom/windows2000/techinfo/reskit/en-us/cnet/cnad_arc_jypf.asp?fra; Jan. 8, 2002.

http://www.ietf.org/rfc.html; Network Working Group; RFC #2828; May 2000.

http://www.ietf.org/rfc.html; Network Working Group; RFC #2979; Oct. 2000.

A. Ghanwani et al, "A Framework for Integrated Services Over Shared and Switched IEEE 802 LAN Technologies," Network Working Group, Request for Comments: 2816, May 2000.

G. Huston, "Next Steps for the IP Qos Architecture," Network Working Group, Request for Comments: 2990, Nov. 2000.

K, Nichols et al, "A Two-Bit Differentiated Services Architecture for the Internet," Network Working Group, Request for Comments: 2638, Jul. 1999.

K. Nichols et al, "Definition of the Differentiated Services Field (DS Field) in the Ipv4 and Ipv6 Headers," Network Working Group, Request for Comments: 2474, Dec. 1998.

M. Seaman et al, "Integrated Service Mappings on IEEE 802 Networks," Network Working Group, Request for Comments: 2815, May 2000.

R. Yavatkar et al, "A Framework for Policy-Based Admission Control," Network Working Group, Request for Comments: 2753, Jan. 2000.

R. Yavatkar et al, "SBM (Subnet Bandwidth Manager): A Protocol for RSVP-based Admission Control over IEEE 802-style Networks," Network Working Group, Request for Comments: 2814, May 2000.

S. Blake et al, "An Architecture for Differentiated Services," Network Working Group, Request for Comments: 2475, Dec. 1998.

S. Herzog, "Signaled Preemption Priority Policy Element," Network Working Group, Request for Comments: 2751, Jan. 2000.

Smith, James E., "A Study of Branch Prediction Strategies", IEEE, pp. 135-148 (1981).

Y. Bernet, "A Framework for Integrated Services Operation over Diffserv Networks," Network Working Group, Request for Comments: 2998, Nov. 2000.

* cited by examiner

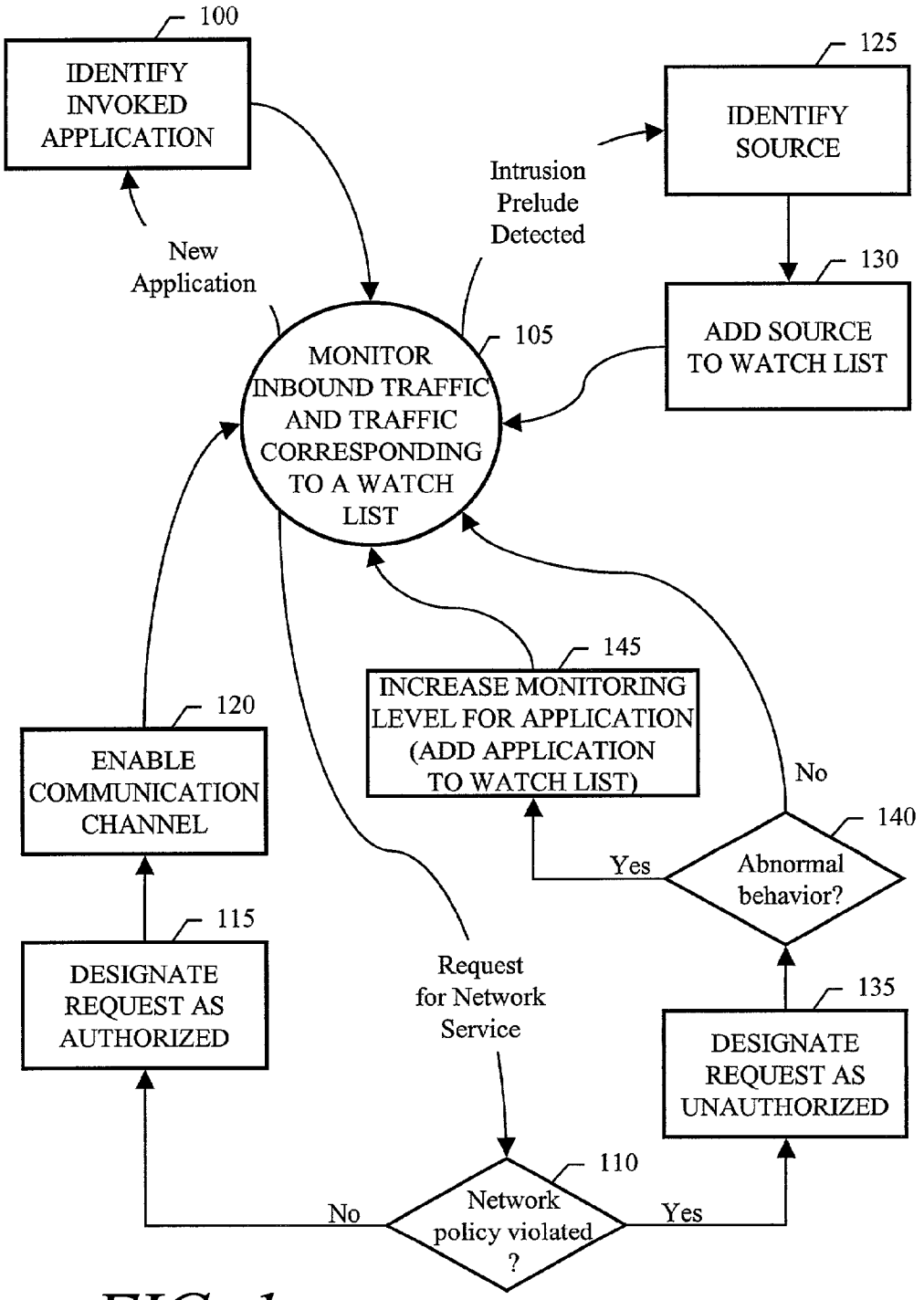

100
IDENTIFY INVOKED APPLICATION
New Application
125
IDENTIFY SOURCE
Intrusion Prelude Detected
130
ADD SOURCE TO WATCH LIST
105
MONITOR INBOUND TRAFFIC AND TRAFFIC CORRESPONDING TO A WATCH LIST
145
INCREASE MONITORING LEVEL FOR APPLICATION (ADD APPLICATION TO WATCH LIST)
120
ENABLE COMMUNICATION CHANNEL
No
140
Abnormal behavior?
Yes
115
DESIGNATE REQUEST AS AUTHORIZED
Request for Network Service
135
DESIGNATE REQUEST AS UNAUTHORIZED
110
Network policy violated ?
No
Yes

FIG. 1

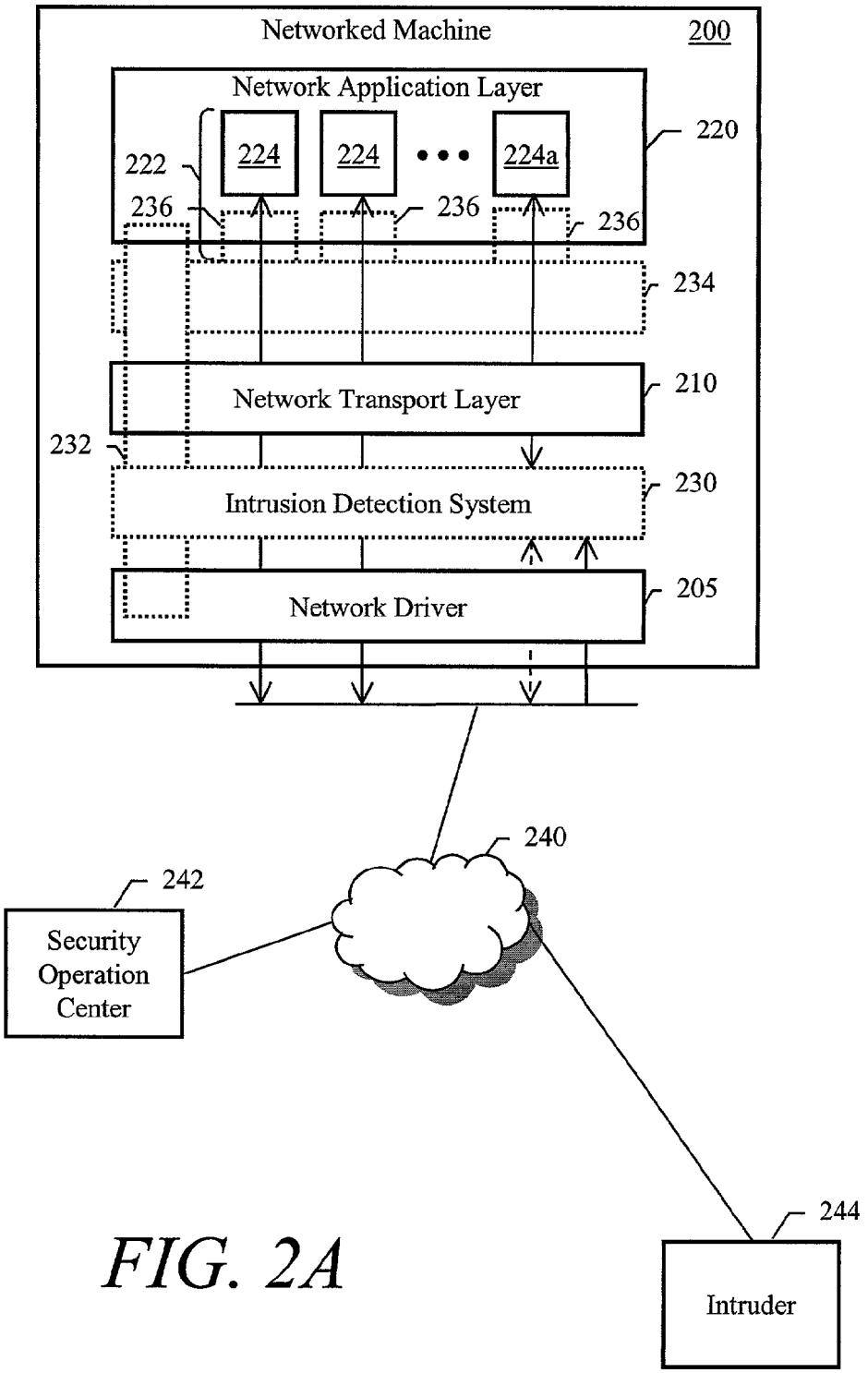

Networked Machine
200
Network Application Layer
220
222
224
224
224a
236
236
236
234
Network Transport Layer
210
232
Intrusion Detection System
230
Network Driver
205
240
242
Security Operation Center
244
Intruder

FIG. 2A

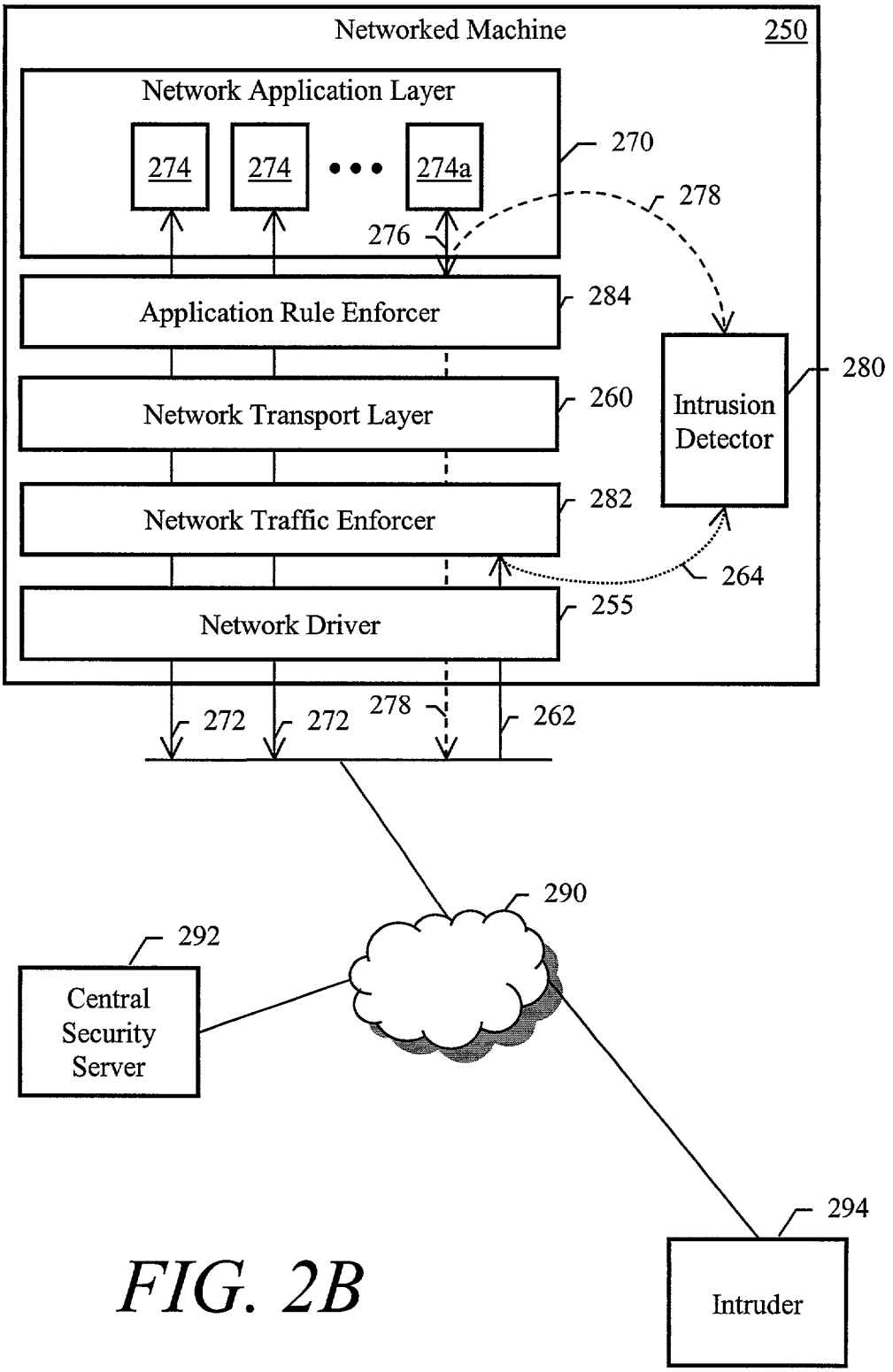

Networked Machine
250
Network Application Layer
274
274
274a
270
278
276
Application Rule Enforcer
284
Network Transport Layer
260
Intrusion Detector
280
Network Traffic Enforcer
282
264
Network Driver
255
272
272
278
262
290
292
Central Security Server
294
Intruder

FIG. 2B

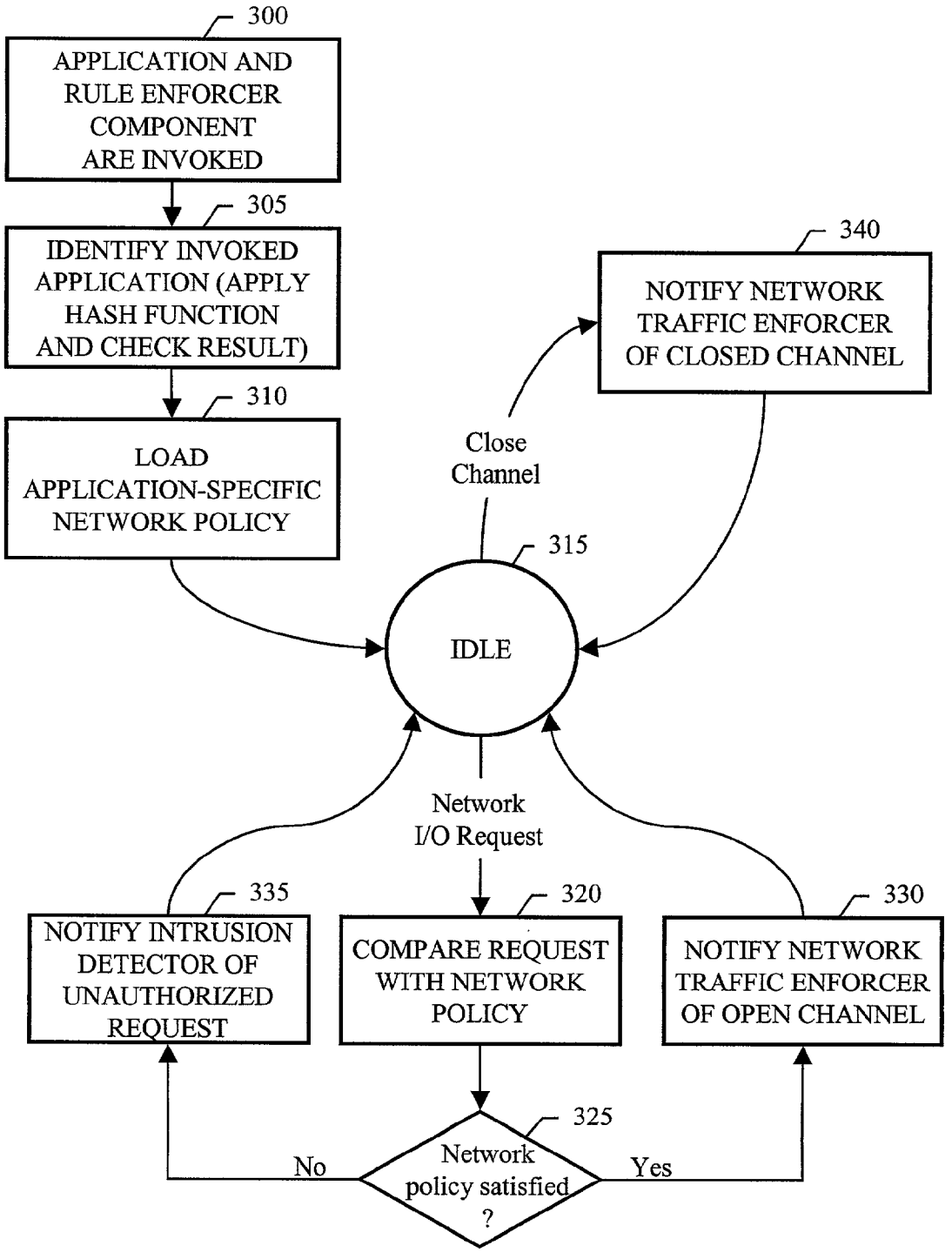

300
APPLICATION AND RULE ENFORCER COMPONENT ARE INVOKED
305
IDENTIFY INVOKED APPLICATION (APPLY HASH FUNCTION AND CHECK RESULT)
310
LOAD APPLICATION-SPECIFIC NETWORK POLICY
340
NOTIFY NETWORK TRAFFIC ENFORCER OF CLOSED CHANNEL
Close Channel
315
IDLE
Network I/O Request
335
NOTIFY INTRUSION DETECTOR OF UNAUTHORIZED REQUEST
320
COMPARE REQUEST WITH NETWORK POLICY
330
NOTIFY NETWORK TRAFFIC ENFORCER OF OPEN CHANNEL
325
Network policy satisfied ?
No
Yes

FIG. 3

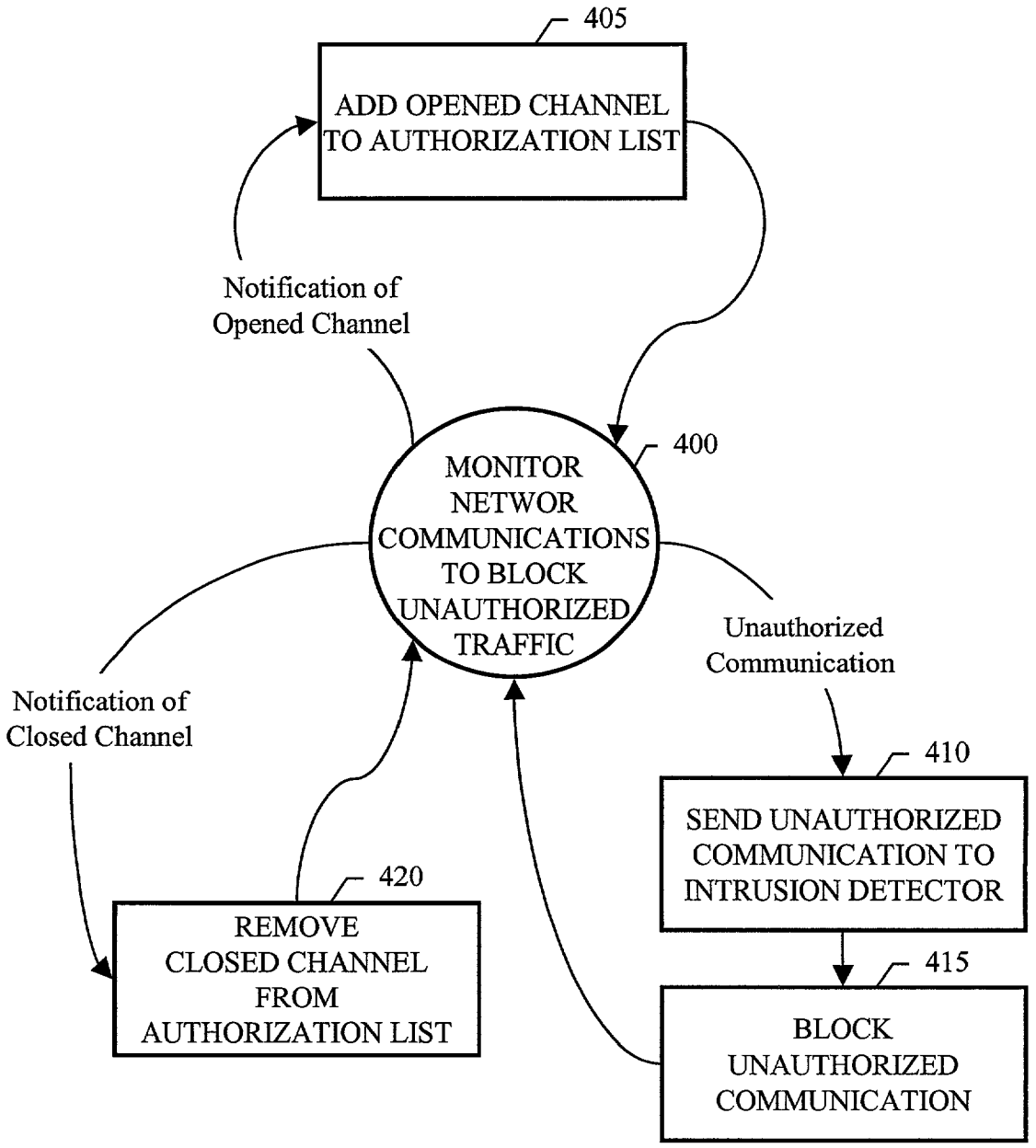

405
ADD OPENED CHANNEL TO AUTHORIZATION LIST
Notification of Opened Channel
400
MONITOR NETWOR COMMUNICATIONS TO BLOCK UNAUTHORIZED TRAFFIC
Unauthorized Communication
Notification of Closed Channel
410
SEND UNAUTHORIZED COMMUNICATION TO INTRUSION DETECTOR
420
REMOVE CLOSED CHANNEL FROM AUTHORIZATION LIST
415
BLOCK UNAUTHORIZED COMMUNICATION

FIG. 4

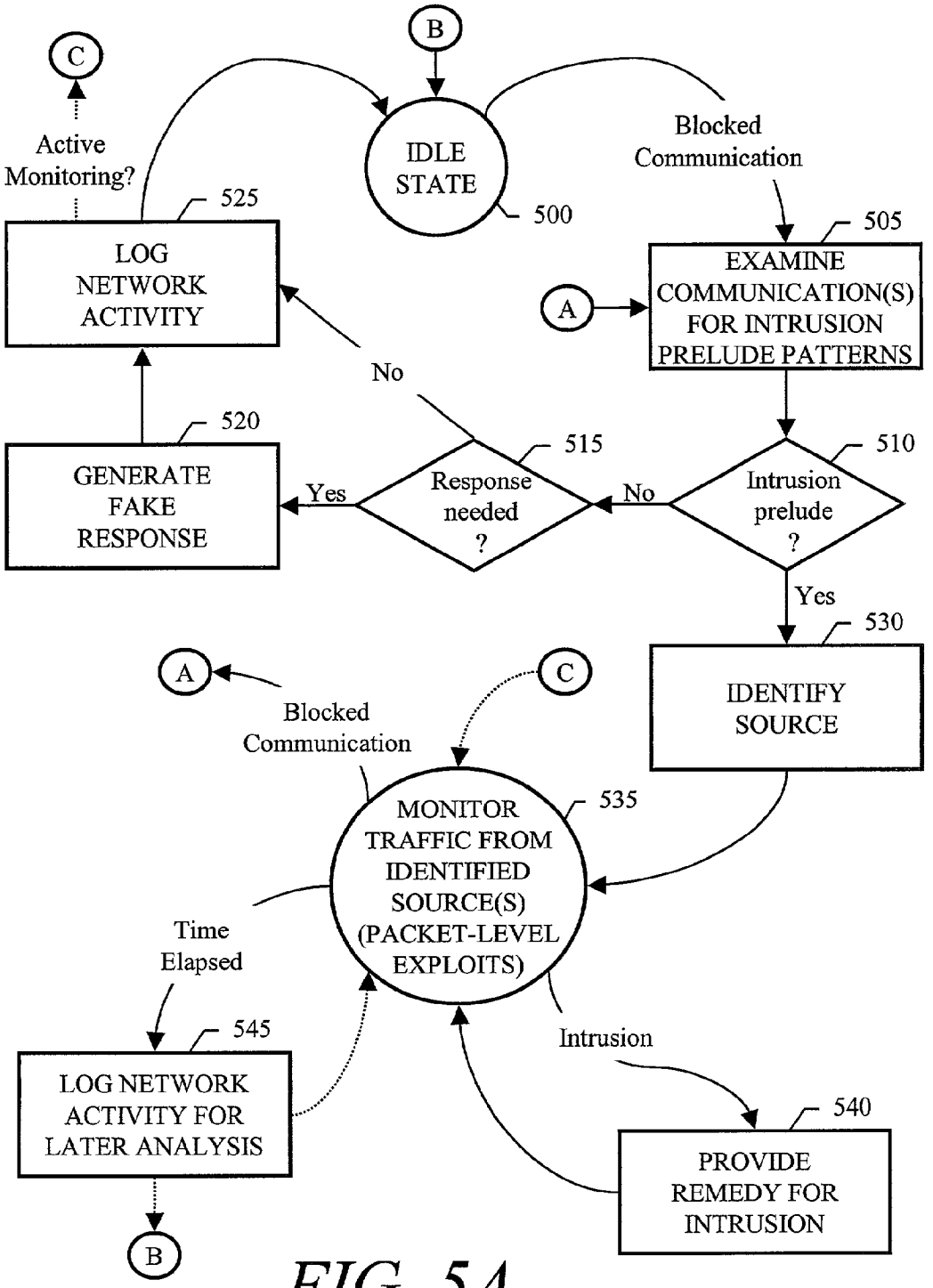

B
C
Active Monitoring?
IDLE STATE
500
Blocked Communication
525
LOG NETWORK ACTIVITY
505
EXAMINE COMMUNICATION(S) FOR INTRUSION PRELUDE PATTERNS
A
No
520
GENERATE FAKE RESPONSE
Yes
515
Response needed ?
No
510
Intrusion prelude ?
Yes
530
IDENTIFY SOURCE
A
C
Blocked Communication
535
MONITOR TRAFFIC FROM IDENTIFIED SOURCE(S) (PACKET-LEVEL EXPLOITS)
Time Elapsed
545
LOG NETWORK ACTIVITY FOR LATER ANALYSIS
Intrusion
540
PROVIDE REMEDY FOR INTRUSION
B

FIG. 5A

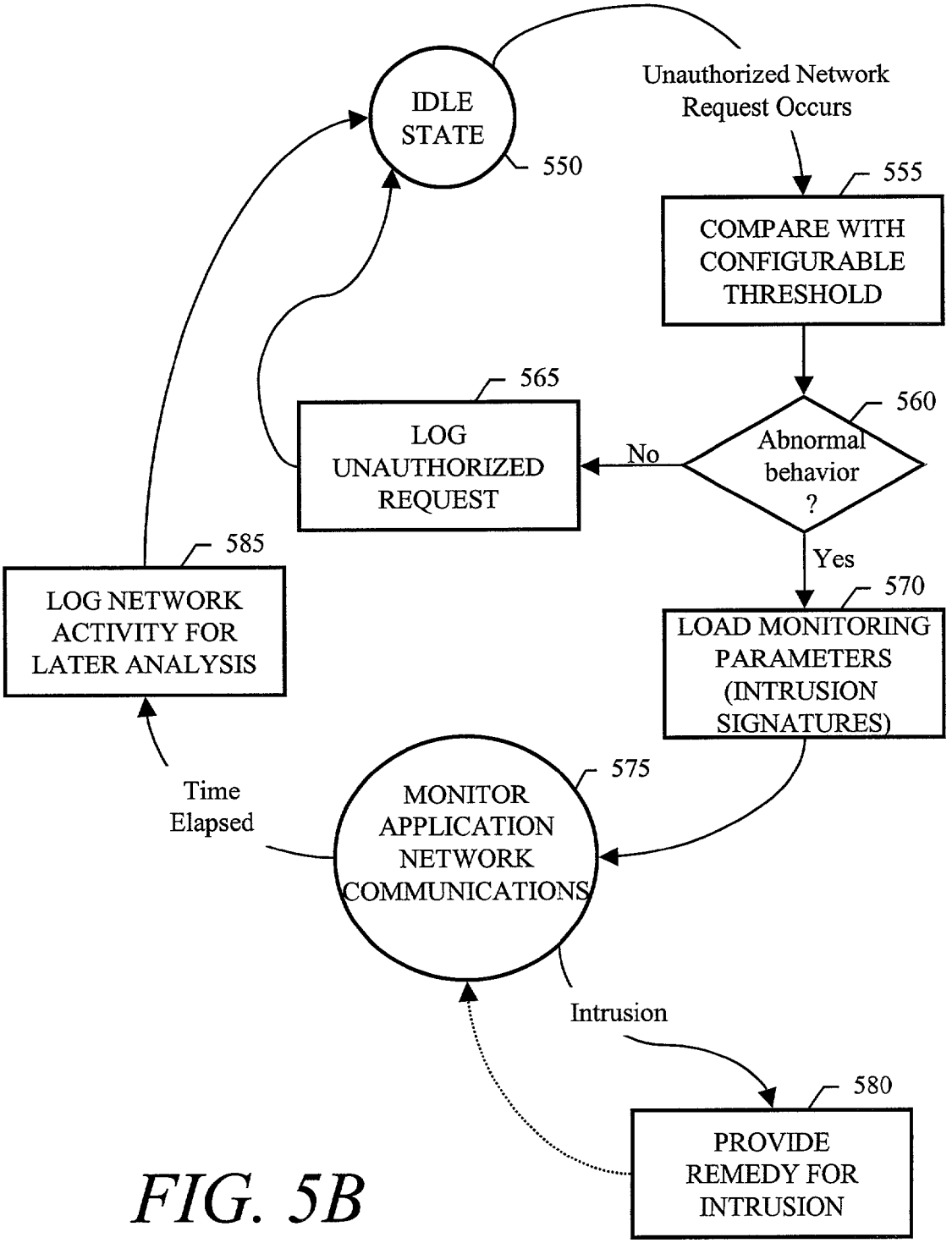

IDLE STATE
550
Unauthorized Network Request Occurs
555
COMPARE WITH CONFIGURABLE THRESHOLD
565
LOG UNAUTHORIZED REQUEST
No
560
Abnormal behavior ?
Yes
570
LOAD MONITORING PARAMETERS (INTRUSION SIGNATURES)
585
LOG NETWORK ACTIVITY FOR LATER ANALYSIS
Time Elapsed
575
MONITOR APPLICATION NETWORK COMMUNICATIONS
Intrusion
580
PROVIDE REMEDY FOR INTRUSION

FIG. 5B

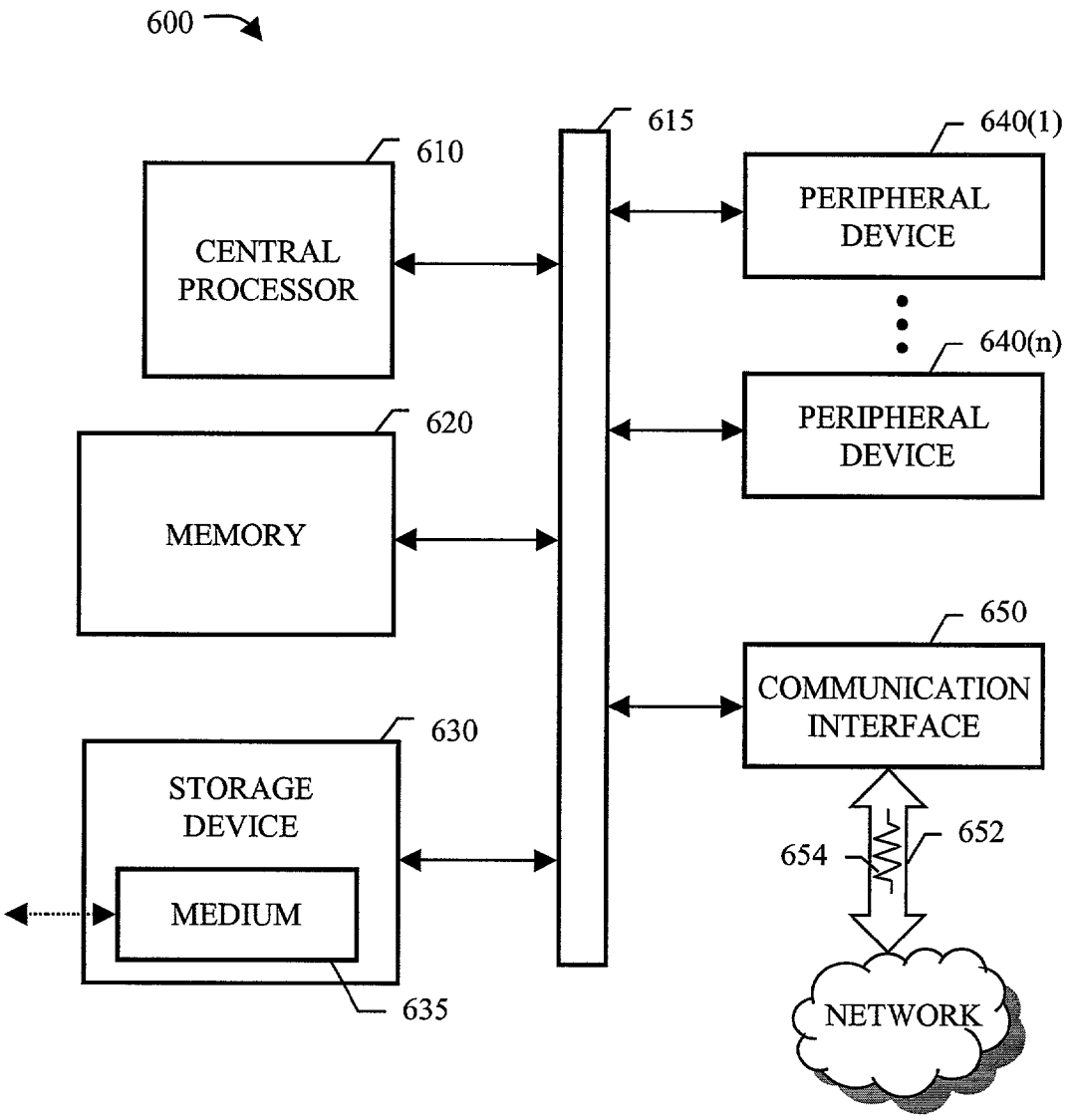

600
615
610
CENTRAL PROCESSOR
640(1)
PERIPHERAL DEVICE
640(n)
PERIPHERAL DEVICE
620
MEMORY
650
COMMUNICATION INTERFACE
630
STORAGE DEVICE
MEDIUM
635
654
652
NETWORK

FIG. 6

INTEGRATED NETWORK INTRUSION DETECTION

BACKGROUND

The present application describes systems and techniques relating to network intrusion detection, for example, integrated network intrusion detection.

A machine network is a collection of nodes coupled together with wired and/or wireless communication links, such as coax cable, fiber optics and radio frequency bands. A machine network may be a single network or a collection of networks (e.g., an internetwork), and may use multiple networking protocols, including internetworking protocols (e.g., Internet Protocol (IP)). These protocols define the manner in which information is prepared for transmission through the network, and typically involve breaking data into segments generically known as packets (e.g., IP packets, ATM (Asynchronous Transfer Mode) cells) for transmission. A node may be any machine capable of communicating with other nodes over the communication links using one or more of the networking protocols.

These networking protocols are typically organized by a network architecture having multiple layers, where each layer provides communication services to the layer above it. A layered network architecture is commonly referred to as a protocol stack or network stack, where each layer of the stack has one or more protocols that provide specific services. The protocols may include shared-line protocols such as in Ethernet networks, connection-oriented switching protocols such as in ATM networks, and/or connectionless packet-switched protocols such as in IP.

As packets travel through a network, they are typically encapsulated within other packets multiple times. Encapsulation enables data to travel from a source process on one node to a destination process on another node, through multiple networks using different protocols and addressing schemes, without the two end nodes knowing anything about the intermediate addressing schemes and protocols.

Machine networks may provide powerful communication capabilities, but also may increase the difficulty of maintaining computer system security by making systems and data more accessible. Most networks are susceptible to attacks or improper use, both from inside and from outside the network. Attacks include attempts to gain unauthorized access to data, destroy or bring down a computer system, prevent others from accessing a system and attempts to take control of a system. For example, some network intrusions exploit application anomalies to gain access to a system and infect it with a computer virus, such as Code Red or Nimba.

A common technique used to improve network security is to install a firewall, which restricts and controls the flow of traffic between networks, typically between an enterprise network and the Internet. Firewalls typically monitor incoming and outgoing traffic and filter, redirect, repackage and/or discard packets. A firewall may serve as a proxy and may enforce an organization's security policies.

Frequently, network administrators employ systems to detect network intrusions to improve network security. Traditional network intrusion detection (NID) systems attempt to examine every packet on a network in order to detect intrusions. These NID systems may be implemented as standalone systems (e.g., NFR (Network Flight Recorder), provided by Cisco Systems, Inc. of San Jose, Calif.), or they may be implemented as distributed node-based systems (e.g., BlackICE, provided by Network Ice Corporation of San Mateo Calif.).

DRAWING DESCRIPTIONS

FIG. **1** is a combined flowchart and state diagram illustrating a method of monitoring network traffic to detect intrusions.

FIG. **2**A is a block diagram illustrating a system implementing integrated network intrusion detection.

FIG. **2**B is a block diagram illustrating another system implementing integrated network intrusion detection.

FIG. **3** is a combined flowchart and state diagram illustrating a method of servicing network requests in an application rule enforcer component of an integrated network intrusion detection system.

FIG. **4** is a combined flowchart and state diagram illustrating a method of filtering network communications in a network traffic enforcer component of an integrated network intrusion detection system.

FIG. **5**A is a combined flowchart and state diagram illustrating a method of detecting intrusion preludes and intrusions in a first detector component of an integrated network intrusion detection system.

FIG. **5**B is a combined flowchart and state diagram illustrating a method of detecting intrusions in a second detector component of an integrated network intrusion detection system.

FIG. **6** is a block diagram illustrating an example data processing system.

Details of one or more embodiments are set forth in the accompanying drawings and the description below. Other features and advantages may be apparent from the description and drawings, and from the claims.

DETAILED DESCRIPTION

The systems and techniques described here relate to integrated network intrusion detection. The description that follows frequently discusses intrusion detection in the context of IP networks, but the systems and techniques described apply equally to multiple types of machine communication networks and operating system environments.

As used herein, the term "application" means a software program, which is a collection of computing operations embodied by a set of instructions (e.g., one or more binary objects, one or more scripts, and/or one or more interpretable programs). The term "component" means a software program designed to operate with other components and/or applications. The term "process" means an executing software program. The term "execution context" means a set of processing cycles given to a process, such as a task in a multitasking operating system. Both an invoked application and an invoked component are a separate process, even if their functionality is interrelated and they share a single execution context. For example, an applet and a Web browser in which the applet runs are each a process. The term "applet" means a component designed specifically to be run from within an application. The term "thread" means a part of a software program that is given its own execution context.

The term "intrusion" means an attempt to break into and/or misuse a computing system. The term "intrusion prelude" means communication activities that typically precede an intrusion. The term "intrusion signature" means a communication pattern identified as corresponding to a known type of intrusion, including patterns that may be found in individual packets and patterns that may be gleaned from analyzing multiple packets.

The present inventor recognized the potential advantages of integrating firewall filtering information with network intrusion analysis. In typical network environments, most network traffic is legitimate and only a small portion of network communications may contain intrusions. By performing intrusion analysis on packets blocked by a firewall, intrusion preludes may be detected (including detection using fabricated responses to blocked network requests), and particular sources of network communications may be singled out for greater scrutiny. Thus, an overall amount of network traffic that needs to be monitored may be reduced, real-time intrusion detection may be improved, and more information about an intruder and the intruder's system and/or network may be obtained.

In addition, firewall functionality may be integrated with intrusion detection on end nodes (e.g., servers and hosts) in a network, such as an enterprise network, to further improve intrusion detection and network security. For example, a networked machine may include an intrusion detection system that functions in part as a dynamic firewall for the networked machine.

The intrusion detection system may include three components. The first component may be an application rule enforcer that authorizes network service requests from applications invoked on the networked machine and identifies abnormal behavior by an invoked application. The second component may be a network traffic enforcer that monitors inbound network communications and blocks those communications that fail to correspond to an authorized network service request. The third component may be an intrusion detector that monitors the blocked communications and identifies abnormal application behavior to determine when additional traffic monitoring is needed to detect an intrusion. Thus, the total number of communications (e.g., packets) that are examined may be reduced while intrusions may be detected more effectively.

FIG. **1** is a combined flowchart and state diagram illustrating a method of monitoring network traffic to detect intrusions. The method begins by identifying one or more applications invoked on a machine (**100**). This identification may be performed for an application by examining network communications generated by the application, system records for the application, and/or a set of instructions embodying the application.

Next a default state **105** is entered, in which inbound traffic (i.e., inbound network communications) and traffic corresponding to a watch list are monitored. These network communications are monitored to detect an intrusion prelude or an intrusion. Moreover, multiple levels of monitoring may be implemented in the default monitoring state **105**.

When a new application is invoked, the new application is identified (**100**). When a request is received for network service (i.e., a network input/output (I/O) request) from an invoked application, a check is made as to whether the request violates a network policy (**110**). The network policy may include a system policy and/or an application-specific policy.

For example, the request may include information such as destination IP address, destination port, source port and type of request (e.g., bind, connect, accept, listen, send, receive, etc.). The network policy may include application-specific rules such as Application=Internet Explorer, destination port=Any, destination address=Any, source port=**80**, request=Listen, action=Allow. This rule states that the network policy allows any inbound traffic for the Internet Explorer application from any remote server through port **80**. In addition to permissive rules that specify allowed communications, the network policy may also include restrictive rules that specify communications that are not allowed (e.g., a Deny action).

If the received request does not violate the network policy, the request is designated as authorized (**115**). Then, a communication channel for the request is enabled (**120**), and monitoring continues.

Rules similar to the policy rule above may be dynamically added to and removed from a network filter driver to open and close communication channels. Such filtering rules identify authorized network flows associated with invoked applications. In an IP network, a channel may be created by specifying an open channel for a network flow using five values: (1) source IP address, (2) source port, (3) destination IP address, (4) destination port, and (5) protocol. Additional and/or alternative values may be used to specify an open channel.

Following the creation of an open channel, inbound traffic that corresponds to the open channel is allowed, whereas inbound traffic that fails to correspond to an open channel is blocked in the monitoring state **105**. Moreover, outbound traffic may also be monitored in the monitoring state **105**, and disabled channels may also be created, such as by using the Deny action discussed above. Blocked traffic is monitored to detect an intrusion prelude, for example, a system scan, a port scan and/or an operating system (OS) fingerprinting. The blocked traffic may be checked for patterns that span multiple communications and/or multiple communication channels (e.g., multiple TCP/IP (Transmission Control Protocol/Internet Protocol)connections).

When an intrusion prelude is detected, a source of the intrusion prelude is identified (**125**). For example, a source IP addresses may be extracted from a packet that is part of the intrusion prelude. This source is then added to a watch list for increased monitoring (**130**), and monitoring continues. All packets from the identified source may then be monitored and these packets may be checked for intrusion signature(s). Additionally, multiple sources may be associated with each other, both in intrusion prelude detection and in subsequent intrusion detection, to counter distributed attacks.

If a received request violates the network policy, the request is designated as unauthorized (**135**). A determination is then made as to whether the application that generated the unauthorized request is behaving abnormally (**140**). This determination may be based on the number of unauthorized requests and/or on the severity of the unauthorized request generated by the application. For example, in one implementation, a single unauthorized request may be treated as abnormal behavior by an application. If the requesting application is behaving normally, monitoring continues.

When an application behaves abnormally, a level of monitoring for the application is increased (**145**), and monitoring continues. For example, the application may be added to a watch list to initiate monitoring of network communications both to and from the application. This monitoring may include searching packets for application-specific intrusion signatures.

FIG. **2**A is a block diagram illustrating a system implementing integrated network intrusion detection. A networked machine **200** includes a network stack, which is a set of layered software modules implementing a defined protocol stack. The number and composition of layers in the network stack will vary with machine and network architecture, but generally includes a network driver **205**, a network transport layer **210** (e.g., TCP/IP) and an application layer **220**.

An intrusion detection system (IDS) **230** may be implemented between the network driver **205** and the network transport layer **210** so that all incoming packets may be monitored. Packet-level intrusion detection may be implemented in an NDIS (Network Driver Interface Specification) intermediate driver in a Windows environment. In addition, the IDS **230** may have additional components **232** placed elsewhere in the network stack. System-level intrusion detection may be implemented in one or more TDI (Transport Driver Interface) filter drivers, and application-level intrusion detection may be implemented in one or more components placed just below and/or just inside the application layer **220** (i.e., as part of a network interface library).

If an application-level component **234** is used as part of the IDS **230**, network services requested by applications **224** go to the application-level component **234** first. As a result, the application-level component **234** knows which application requested which network service. In a Windows operating system environment, the application-level component **234** may be implemented as a WinSock (Windows Socket) Layer Service Provider (LSP) and/or as a TDI filter driver. WinSock is an Application Programming Interface (API) for developing Windows programs that communicate over a network using TCP/IP.

Alternatively, or in addition, application-level components **236** may be used for intrusion detection. Such components **236** load and run with each new network application **224** in an execution context **222** for that network application. These components **236** may perform authorization of network requests and application-specific intrusion signature detection such that the processing time consumed by these techniques affects only corresponding network applications.

The networked machine **200** is coupled with a network **240** that may provide communication links to a security operation center **242** and a potential intruder **244**. The security operation center **242** may include a central security server. Various alert levels may be used in the IDS **230**. These alert levels may trigger heightened monitoring states, cause alerts to be sent to the security operation center **242**, and/or initiate logging of network activity, locally and/or with the central security server, for later forensic analysis.

The IDS **230** functions as a dynamic firewall for the networked machine **200**. The IDS **230** monitors network traffic to block traffic that violates a network policy and monitors blocked traffic to detect an intrusion prelude. The IDS **230** monitors traffic from the potential intruder **244** when an intrusion prelude is detected. The IDS **230** may track behavior of applications **224** using a network policy that specifies behavior criteria (which may be application-specific) to identify abnormal application behavior. The IDS **230** may monitor traffic from an abnormally behaving application **224***a* to identify an intrusion, including e.g. an intrusion connected with a Trojan Horse in the application.

FIG. **2**B is a block diagram illustrating a system implementing integrated network intrusion detection. A networked machine **250** includes a network stack, as described above, and generally includes a network driver **255**, a network transport layer **260** (e.g., TCP/IP) and an application layer **270**. The networked machine **250** also includes an intrusion detection system divided into three components: an intrusion detector **280**, a network traffic enforcer **282**, and an application rule enforcer **284**.

These components **280**, **282**, **284** may reside in fewer or greater than three software modules. For example, the intrusion detector **280** may include a kernel component that resides in a first module with the network traffic enforcer **282**, and the intrusion detector **280** also may include a user component that resides in a second module with the application rule enforcer **284**. Additionally, the application rule enforcer **284** may be a component that is invoked separately with each of multiple invoked applications **274**, as described above.

The networked machine **250** is coupled with a network **290** that may provide communication links to a central security server **292** and a potential intruder **294**.

As each application **274** requests network I/O service, the request is either authorized or rejected by the application rule enforcer **284**. If the request is authorized, corresponding authorized communications **272** are allowed to pass from the application **274** to the network **290**, and from the network **290** to the application **274**. If a request is rejected, this rejected request is communicated to the intrusion detector **280**.

If a request **276** is rejected, the intrusion detector **280** may determine that an application **274***a* is behaving abnormally, and the intrusion detector **280** may then begin monitoring other communications **278** for the suspect application **274***a*. This additional monitoring of communications **278** may involve checking for application-specific intrusion signatures, which may be dynamically loaded from the central security server **292**.

The network traffic enforcer **282** monitors incoming network traffic. If an inbound communication **262** fails to correspond to an authorized request (i.e., the inbound communication was not effectively pre-approved by the application rule enforcer), the communication is dropped (i.e., blocked from passage to another layer in the network stack). Additionally, the network traffic enforcer **282** may monitor outbound communications in a similar manner. For example, the network traffic enforcer **282** may check all packets (both from the network **290** and from the applications **274**) to see if they match a channel opened by the application rule enforcer **284**.

Moreover, the network traffic enforcer **282** may also drop other communications in a manner similar to a firewall. For example, the network traffic enforcer **282** may drop malformed packets and packets used for system scanning (e.g., ICMP (Internet Control Message Protocol) echo requests). Communications that are dropped by the network traffic enforcer are sent to the intrusion detector **280**.

The intrusion detector **280** examines the dropped communications to look for patterns signaling an intrusion prelude. For example, the intrusion detector **280** may look for system scans (e.g., ping), port scans (e.g., TCP-SYN (synchronization), TCP-FIN (finished), etc.), and OS fingerprinting. Frequently an intruder **294** will perform scanning operations on a system, or make some missteps, before an intrusion is launched. These operations may be detected by the intrusion detector **280** as an intrusion prelude.

Additionally, the intrusion detector **280** may encourage these operations by generating fabricated responses to the dropped communications to catch the intruder **294**. A fabricated response to blocked traffic may be used to gain knowledge about a potential intruder and their system for later use. For example, by selectively generating one or more fabricated responses to blocked inbound traffic, which would otherwise be blocked silently, significant information concerning a potential intruder's system and network may be obtained for use in later forensic analysis. If the potential intruder later turns out to be an actual intruder, this collected information may be associated with the detected intrusion and may be especially useful, such as for use in prosecution of the intruder and/or other legal action (e.g., legal action

requiring an intruder's Internet Service Provider (ISP) to take action, such as denying future network services to the intruder).

When an intrusion prelude is detected, the intrusion detector **280** then identifies and registers a source address for the intruder **294** and begins examining communications from that source to detect an intrusion. For example, the intrusion detector **280** may watch traffic from a potential intruder to look for packet level exploits such as launching intrusions using packet fragments (e.g., tear drop, Boink, etc.). Thus, the intrusion detector **280** may support packet reassembly to detect fragmentation related intrusions.

If the intrusion detector **280** detects an intrusion (e.g., a packet exploit), it may block the traffic and/or report the intrusion to the central security server **292**. Additionally, the intrusion detector **280** may log the communications associated with a detected intrusion and intrusion prelude for forensic analysis.

FIG. **3** is a combined flowchart and state diagram illustrating a method of servicing network requests in an application rule enforcer (ARE) component of an integrated network intrusion detection system. The method begins when an application and the ARE component are invoked (**300**). The ARE component then identifies the invoked application (**305**).

To do so, the ARE component may determine the full path (directory and file name) of the loading application executable (e.g., "C:/Program Files/Application/application.exe"), examine machine instructions embodying the application (e.g., "application.exe") to identify the application, and/or may crosscheck this identification with file properties information, such as name, size and version number. Examining the machine instructions may involve applying a hash function to the application's executable to generate a condensed representation (or hash value) of the executable. This hash value may then be compared with predefined hash values for known applications to identify the invoked application.

The hash function may be a message digest algorithm with a mathematical property that effectively guarantees that for any size message, a unique value of a fixed size (e.g., 128 bits) is returned. The hash function may be part of a standardized message digest specification (e.g., Secure Hash Standard (SHA-1), defined in Federal Information Processing Standards Publication 180-1).

Once the invoked application is identified, an application-specific network policy is loaded (**310**). This network policy information may be loaded from a local repository and/or from a remote repository of network policy information (including dynamic loading from the remote repository to the local repository to keep the network policy information up to date as network policies change). Then, the ARE component enters an idle state **315**.

When a network I/O request is made by the application, the request is compared with the application-specific network policy (**320**). If the policy is satisfied (**325**), a network traffic enforcer (NTE) component is notified to open a channel (**330**). For example, a message may be sent specifying a source IP address, a source port, a destination IP address, a destination port and a protocol for the opened channel.

If the policy is not satisfied, an intrusion detector component is notified of the rejected request (**335**). Alternatively, the notice may be that the application is behaving abnormally. For example, a single violation of network policy may be considered abnormal behavior for the application. Alternatively, the application-specific network policy may be multi-tiered, such that certain violations are logged, but repeated and/or more severe violations of network policy constitute abnormal application behavior.

Such policies may include configurable thresholds for one or more characteristics of network communications. The configurable thresholds may be set directly by the intrusion detector, and/or by a network administrator, after analysis of communication statistics for the application. Thus, network administrators may set the configurable thresholds, such as by including them with intrusion signatures provided by security service providers, and/or the configurable thresholds may be auto-configurable, such as by monitoring communications during a defined time window.

When an open channel is closed, the NTE component is notified of this closing channel (**340**).

FIG. **4** is a combined flowchart and state diagram illustrating a method of filtering network communications in a network traffic enforcer (NTE) component of an integrated network intrusion detection system. The method begins in a monitoring state **400**, where communications are monitored to block unauthorized communications. When a notification of an opened channel is received, the opened channel is added to an authorization list (**405**), and monitoring continues.

When an unauthorized communication is received, a copy of the communication is sent to an intrusion detector component (**410**). Then the unauthorized communication is blocked (i.e., dropped) (**415**), and monitoring continues. When a notification of a closed channel is received, the closed channel is removed from the authorization list (**420**), and monitoring continues. Thus, network communications that have not been pre-approved by the ARE component are blocked and copied to the intrusion detector.

FIG. **5**A is a combined flowchart and state diagram illustrating a method of detecting intrusion preludes and intrusions in a first detector component of an integrated network intrusion detection system. The method begins in an idle state **500**. When a blocked communication is received, it is checked for intrusion prelude patterns (**505**). Such patterns may include system scan, port scan and OS fingerprinting.

A check is made to determine if an intrusion prelude is present (**510**). If not, a check is made to determine if a response is needed to encourage an intruder (**515**). If so, a fabricated response is generated and sent to the potential intruder (**520**). Then, or if a fabricated response was not needed, the present communication activity is logged for future use in detecting intrusion preludes (**525**).

If an intrusion prelude is detected, the source of the intrusion prelude is identified (**530**). The identified source is a potential intruder, and thus communications from the potential intruder are monitored in an active monitoring state **535**. This active monitoring may involve checking for packet level exploits, such as intrusions using packet fragments, as described above. When a blocked communication is received, it is checked for intrusion prelude patterns, as before (**505**). Thus, additional sources may be added to a list of potential intruders to be monitored in the active monitoring state **535**.

If an intrusion is detected, a remedy is provided (**540**). For example, the intrusion activity may be logged, the traffic may be cut, countermeasures may be employed and/or an alert may be sent to a security operation center.

If a pre-defined time elapses for an identified source, the monitored activity for the source is logged for later analysis, and the source-specific monitoring for that source is termi-

nated (**545**). If this is the last source being monitored in the active monitoring state **535**, the method returns to the idle state **500**.

FIG. **5**B is a combined flowchart and state diagram illustrating a method of detecting intrusions in a second detector component of an integrated network intrusion detection system. The method begins in an idle state **550**. When an unauthorized request occurs, the unauthorized request is compared with one or more configurable thresholds (**555**). These configurable thresholds specify the type and/or number of requests that constitute abnormal application behavior. The configurable thresholds may be set as described above.

A check is then made for abnormal behavior (**560**). If the application is not behaving abnormally, the unauthorized request is logged for later use (**565**). If the application is behaving abnormally, monitoring parameters for the application are loaded (**570**). These parameters may include application-specific intrusion detection signatures.

Then, a monitoring state **575** is entered, in which network communications for the application are monitored using the loaded parameters. If an intrusion is detected, a remedy is provided (**580**). For example, the intrusion activity may be logged, the traffic may be cut, countermeasures may be taken, and/or an alert may be sent to a security operation center. This remedy may be application-specific.

If a predefined time elapses, in which no intrusion is detected, the monitored communications for the application are logged for later analysis (**585**). Then, the method returns to the idle state **550**.

Although FIGS. **3** to **5**C show methods being performed in four separate components, these methods may also be combined into a single component or two or more components. For example, a first component, being a combination of the NTE component and the first intrusion detector component, may perform a combination of the methods shown in FIGS. **4** and **5**A. A second component, being a combination of the ARE component and the second intrusion detector component, may perform a combination of the methods shown in FIGS. **3** and **5**B.

Various implementations of the systems and techniques described here may be realized in digital electronic circuitry, integrated circuitry, specially designed ASICs (application specific integrated circuits), computer hardware, firmware, software, and/or combinations thereof. These various implementations may include implementation in one or more computer programs that are executable/interpretable on a programmable system including at least one programmable processor, which may be special or general purpose, coupled to receive data and instructions from, and to transmit data and instructions to, a storage system, at least one input device, and at least one output device.

FIG. **6** is a block diagram illustrating an example data processing system **600**. The data processing system **600** includes a central processor **610**, which executes programs, performs data manipulations and controls tasks in the system **600**, thereby enabling the features and function described above. The central processor **610** is coupled with one or more communication busses **615**.

The data processing system **600** includes a memory **620**, which may be volatile and/or non-volatile memory, and is coupled with the communications bus **615**. The system **600** may also include one or more cache memories. These memory devices enable storage of instructions and data close to the central processor **610** for retrieval and execution.

The data processing system **600** may include a storage device **630** for accessing a medium **635**, which may be removable. The medium **635** may be read-only or read/write media and may be magnetic-based, optical-based or magneto-optical-based media. The data processing system **600** may also include one or more peripheral devices **640**(**1**)–**640**(*n*) (collectively, devices **640**), and one or more controllers and/or adapters for providing interface functions. The devices **640** may be additional storage devices and media as described above, other storage interfaces and storage units, input devices and/or output devices.

The system **600** may further include a communication interface **650**, which allows software and data to be transferred, in the form of signals **654** over a channel **652**, between the system **600** and external devices, networks or information sources. The signals **654** may embody instructions for causing the system **600** to perform operations. The communication interface **650** may be a network interface designed for a particular type of network, protocol and channel medium, or may be designed to serve multiple networks, protocols and/or channel media.

The system **600** represents a programmable machine, and may include various devices such as embedded controllers and Programmable Logic Devices (PLDs). Machine instructions (also known as programs, software, software applications or code) may be stored in the machine **600** or delivered to the machine **600** over a communication interface. These instructions, when executed, enable the machine **600** to perform the features and function described above.

As used herein, the term "machine-readable medium" refers to any medium or device used to provide machine instructions and/or data to the machine **600**. The various implementations described above have been presented by way of example only, and not limitation. Thus, other embodiments may be within the scope of the following claims.

What is claimed is:

1. A machine-implemented method comprising:

receiving requests for network communication services from an invoked application;

selectively designating each of the received requests as authorized or unauthorized based on an application-specific network policy; and

monitoring inbound network communications, based on the authorized requests, to detect an intrusion.

2. The method of claim **1**, wherein monitoring inbound network communications comprises:

blocking the inbound network communications that fail to correspond to an authorized request; and

monitoring the blocked inbound network communications to detect an intrusion.

3. The method of claim **2**, wherein monitoring the blocked inbound network communications comprises:

examining the blocked inbound network communications to detect an intrusion prelude;

identifying a source for a detected intrusion prelude; and

initiating monitoring of inbound network communications from the identified source.

4. The method of claim **3**, wherein examining the blocked inbound network communications comprises checking for patterns spanning multiple communications.

5. The method of claim **4**, wherein monitoring the blocked inbound network communications further comprises generating fabricated responses to the blocked inbound network communications.

6. The method of claim **3**, wherein the monitoring of inbound network communications from the identified source comprises checking the inbound network communications from the identified source for packet-level exploits.

7. The method of claim 1, further comprising increasing a monitoring level for network communications for the invoked application in response to one or more unauthorized requests.

8. The method of claim 7, wherein increasing a monitoring level for network communications for the invoked application comprises initiating monitoring of the network communications for the invoked application using an application-specific intrusion signature.

9. The method of claim 8, further comprising identifying the invoked application by examining a set of instructions embodying the invoked application.

10. The method of claim 9, wherein monitoring of the network communications for the invoked application comprises monitoring in an intrusion detection system component invoked with the invoked application.

11. The method of claim 10, wherein the intrusion detection system component and the invoked application run within a single execution context.

12. The method of claim 9, wherein examining the set of instructions comprises:

applying a hash function to the set of instructions to generate a condensed representation; and

comparing the condensed representation with existing condensed representations for known applications.

13. A machine-implemented method comprising:

identifying an invoked application;

receiving requests for network communication services from the invoked application;

selectively designating each of the received requests as authorized or unauthorized based on an application-specific network policy;

blocking inbound network communications that fail to correspond to an authorized request;

monitoring the blocked inbound network communications to detect an intrusion; and

initiating monitoring of network communications for the invoked application using an application-specific intrusion signature in response to one or more unauthorized requests.

14. The method of claim 13, wherein monitoring the blocked inbound network communications comprises:

examining the blocked inbound network communications to detect an intrusion prelude;

identifying a source for a detected intrusion prelude; and

initiating monitoring of inbound network communications from the identified source.

15. The method of claim 14, wherein identifying the invoked application comprises examining a set of instructions embodying the invoked application.

16. The method of claim 15, wherein examining the blocked inbound network communications comprises checking for patterns spanning multiple communication.

17. The method of claim 16, wherein monitoring the blocked inbound network communications further comprises generating fabricated responses to the blocked inbound network communications.

18. The method of claim 15, wherein monitoring of inbound network communications from the identified source comprises checking the inbound network communications from the identified source for packet-level exploits.

19. The method of claim 18, wherein examining the set of instructions comprises:

applying a hash function to the set of instructions to generate a condensed representation; and

comparing the condensed representation with existing condensed representations for known applications.

20. The method of claim 19, wherein monitoring of the network communications for the invoked application comprises monitoring in an intrusion detection system component invoked with the invoked application.

21. The method of claim 20, wherein the intrusion detection system component and the invoked application run within a single execution context.

22. A system comprising:

an application network policy enforcer, which services network requests from an application invoked on a machine, identifies the network requests that fail to satisfy an application-specific network policy, and identifies the network requests that satisfy the application-specific network policy;

a network traffic enforcer, which blocks inbound network traffic that does not correspond to the network requests identified by the application network policy enforcer as satisfying the application-specific network policy; and

an intrusion detector, which responds to the network requests identified by the application network policy enforcer as failing to satisfy the application-specific network policy, and which responds to the inbound network traffic blocked by the network traffic enforcer.

23. The system of claim 22, wherein the intrusion detector comprises;

a first component that responds to the network requests identified as failing to satisfy the application-specific network policy by monitoring traffic for the invoked application, wherein the first component shares a software module with the application network policy enforcer; and

a second component that responds to the blocked traffic by monitoring traffic for an identified source of an intrusion prelude detected in the blocked traffic, wherein the second component shares a software module with the network traffic enforcer.

24. A system comprising:

means for servicing network requests from an application invoked on a machine;

means for authorizing the network requests using an application-specific network policy;

means for blocking traffic that does not correspond to an authorized request;

means for monitoring blocked traffic to identify an intrusion prelude and to identify abnormal, application behavior;

means for detecting an intrusion in response to an identified intrusion prelude; and

means for detecting an intrusion in response to identified abnormal application behavior.

25. The system of claim 24, wherein the means for detecting an intrusion in response to an identified intrusion prelude comprises mean for detecting packet-level exploits for traffic from an identified source of the identified intrusion prelude, and wherein the means for detecting an intrusion in response to identified abnormal application behavior comprises means for detecting application-specific intrusion signatures for traffic corresponding to an abnormally behaving application, the system further comprising:

means for generating a fabricated response to blocked traffic to gain knowledge about a potential intruder; and

means for responding to a detected intrusion.

26. A machine-readable medium embodying machine instructions for causing one or more machines to perform operations comprising:

identifying an invoked application;

receiving requests for network communication services from the invoked application;

selectively designating each of the received requests as authorized or unauthorized based on an application-specific network policy;

blocking inbound network communications that fail to correspond to an authorized request;

monitoring the blocked inbound network communications to detect an intrusion; and

initiating monitoring of network communications for the invoked application using an application-specific intrusion signature in response to one or more unauthorized requests.

27. The machine-readable medium of claim **26**, wherein monitoring the blocked inbound network communications comprises:

examining the blocked inbound network communications to detect an intrusion prelude;

identifying a source for a detected intrusion prelude; and

initiating monitoring of inbound network communications from the identified source.

28. The machine-readable medium of claim **27**, wherein identifying the invoked application comprises examining a set of instructions embodying the invoked application.

29. The machine-readable medium of claim **28**, wherein monitoring of inbound network communications from the identified source comprises checking the inbound network communications from the identified source for packet-level exploits.

30. The machine-readable medium of claim **29**, wherein examining the set of instructions comprises:

applying a hash function to the set of instructions to generate a condensed representation; and

comparing the condensed representation with existing condensed representations for known applications.

31. A machine-implemented method comprising:

blocking inbound network communications that fail to correspond to a network policy;

detecting a potential intrusion prelude from the blocked inbound network communications;

selectively generating a fabricated response to the detected potential intrusion prelude; and

receiving information about a potential intruder in response to the generated fabricated response;

wherein the detecting comprises detecting communication activities including system scans, port scans, and operating system fingerprinting.

32. The method of claim **31**, wherein the network policy comprises an application-specific network policy, the method further comprising:

receiving requests for network communication services from an invoked application;

selectively designating each of the received requests as authorized or unauthorized based on the application-specific network policy;

monitoring the blocked inbound network communications to detect an intrusion; and

associating the information about the potential intruder with a detected intrusion.

33. The method off claim **32**, wherein monitoring the blocked inbound network communications comprises:

examining the blocked inbound network communications to detect an intrusion prelude;

identifying a source for a detected intrusion prelude; and

initiating monitoring of inbound network communications from the identified source.

34. The method of claim **13**, wherein the application-specific intrusion signature is loaded from a central security server.

35. The machine-readable medium of claim **26**, wherein the application-specific intrusion signature is loaded from a central security server.

* * * * *