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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
25 April 2013 (25.04.2013)

WIPO | PCT

(10) International Publication Number
WO 2013/058940 A1

(51) International Patent Classification:
H04L 12/56 (2006.01) *H04L 9/30* (2006.01)
H04L 9/32 (2006.01) *H04L 29/06* (2006.01)

(21) International Application Number:
PCT/US2012/057153

(22) International Filing Date:
25 September 2012 (25.09.2012)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
13/275,196 17 October 2011 (17.10.2011) US

(71) Applicant *(for all designated States except US)*: **MCAFEE, INC.** [US/US]; 2821 Mission College Blvd., Santa Clara, CA 95054 (US).

(72) Inventors; and
(71) Applicants *(for US only)*: **COOPER, Geoffrey** [US/US]; 2542 Webster St., Palo Alto, CA 94301 (US). **GREEN, Michael, W.** [US/US]; 1389 Rice Creek Trail, Shoreview, MN 55126 (US). **GUZIK, John, Richard** [US/US]; 706 San Rafael St., Sunnyvale, CA 94085 (US).

(74) Agent: **FRAME, Thomas, J.**; Patent Capital Group, 2816 Lago Vista Lane, Rockwall, TX 75032 (US).

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

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK,

[Continued on next page]

(54) Title: SYSTEM AND METHOD FOR HOST-INITIATED FIREWALL DISCOVERY IN A NETWORK ENVIRONMENT

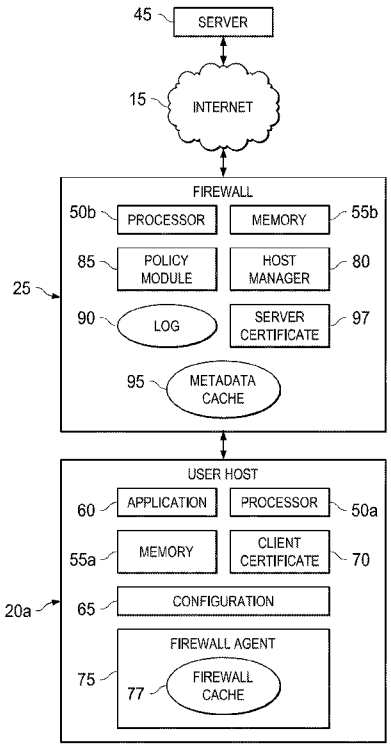

FIG. 2

(57) Abstract: A method is provided in one example embodiment that includes intercepting a network flow to a destination node having a network address and sending a discovery query based on a discovery action associated with the network address in a firewall cache. A discovery result may be received and metadata associated with the flow may be sent to a firewall before releasing the network flow. In other embodiments, a discovery query may be received from a source node and a discovery result sent to the source node, wherein the discovery result identifies a firewall for managing a route to a destination node. Metadata may be received from the source node over a metadata channel. A network flow from the source node to the destination node may be intercepted, and the metadata may be correlated with the network flow to apply a network policy to the network flow.

WO 2013/058940 A1

EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

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

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

Published:

— *with international search report (Art. 21(3))*

SYSTEM AND METHOD FOR HOST-INITIATED FIREWALL DISCOVERY IN A NETWORK ENVIRONMENT

TECHNICAL FIELD

[0001] This specification relates in general to the field of network security, and more particularly, to a system and method for host-initiated firewall discovery in a network environment.

BACKGROUND

[0002] The field of network security has become increasingly important in today's society. The Internet has enabled interconnection of different computer networks all over the world. However, the Internet has also presented many opportunities for malicious operators to exploit these networks. Certain types of malicious software can be configured to receive commands from a remote operator once the software has infected a host computer. The software can be instructed to perform any number of malicious actions, such as sending out spam or malicious emails from the host computer, stealing sensitive information from a business or individual associated with the host computer, propagating to other host computers, and/or assisting with distributed denial of service attacks. In addition, the malicious operator can sell or otherwise give access to other malicious operators, thereby escalating the exploitation of the host computers. Thus, the ability to effectively protect and maintain stable computers and systems continues to present significant challenges for component manufacturers, system designers, and network operators.

BRIEF DESCRIPTION OF THE DRAWINGS

[0003] To provide a more complete understanding of the present disclosure and features and advantages thereof, reference is made to the following description, taken in conjunction with the accompanying figures, wherein like reference numerals represent like parts, in which:

[0004] FIGURE 1 is a simplified block diagram illustrating an example embodiment of a network environment in which a firewall may be discovered by a host according to this specification;

[0005] FIGURE 2 is simplified block diagram illustrating additional details that may be associated with potential embodiments of the network environment;

[0006] FIGURE 3 is a simplified table illustrating example entries in a firewall cache that may be associated with some embodiments of the network environment;

[0007] FIGURE 4 is a simplified interaction diagram illustrating potential operations that may be associated with example embodiments of the network environment that transport firewall discovery messages in ICMP packets;

[0008] FIGURE 5 is a simplified interaction diagram illustrating potential operations that may be associated with other example embodiments of the network environment that transport firewall discovery messages in ICMP packets;

[0009] FIGURE 6 is a simplified interaction diagram illustrating potential operations that may be associated with example embodiments of the network environment that transport firewall discovery messages in UDP packets;

[0010] FIGURE 7 is a simplified interaction diagram illustrating potential operations that may be associated with other example embodiments of the network environment that transport firewall discovery messages in UDP packets; and

[0011] FIGURE 8 is an example packet data unit format that may be associated with exchanging metadata in example embodiments of the network environment.

DETAILED DESCRIPTION OF EXAMPLE EMBODIMENTS

OVERVIEW

[0012] A method is provided in one example embodiment that includes intercepting a network flow to a destination node having a network address and sending a discovery query based on a discovery action associated with the network address in a firewall cache. A discovery result may be received, which can identify a firewall responsible for managing a route to the destination node, and metadata associated with the flow may be sent to the firewall before releasing the network flow.

[0013] In other embodiments, a discovery query may be received from a source node and a discovery result sent to the source node, wherein the discovery result identifies a firewall for managing a route to a destination node. Metadata may be received from the source node over a metadata channel. A network flow from the source node to the destination node may be intercepted, and the metadata may be correlated with the network flow to apply a network policy to the network flow.

EXAMPLE EMBODIMENTS

[0014] Turning to FIGURE 1, FIGURE 1 is a simplified block diagram of an example embodiment of a network environment 10 in which a firewall may be discovered through host redirection. In the embodiment illustrated in FIGURE 1, network environment 10 can include Internet 15, a user host 20a and a user host 20b, a firewall 25, a policy server 30, a mail server 35, and a web server 40. In general, user hosts 20a-20b may be any type of termination node in a network connection, including but not limited to a desktop computer, a server, a laptop, a mobile telephone, or any other type of device that can receive or establish a connection with another node, such as mail server 35 or web server 40. Firewall 25 may control communications between user hosts 20a-20b and other nodes attached to Internet 15, such as by blocking unauthorized access while permitting authorized communications. In some instances, firewall 25 may be coupled to or integrated with an intrusion prevention system, network access control device, web gateway, email gateway, or any other type of gateway between Internet 15 and user hosts 20a-20b. Moreover, the location of firewall 25 close to user hosts 20a-20b in the network topology of FIGURE 1 is arbitrary. Policy server 30 may be coupled to or integrated with firewall 25, and may be used to manage user hosts 20a-20b and to administer and distribute network policies. Thus, in this example embodiment, user hosts 20a-20b may communicate with servers attached to Internet 15 or another network, such as mail server 35 or web server 40, by establishing a connection through firewall 25 if permitted by policies implemented in firewall 25 and managed by policy server 30.

[0015] Each of the elements of FIGURE 1 may couple to one another through simple interfaces or through any other suitable connection (wired or wireless), which provides a viable pathway for network communications. Additionally, any one or more of these

elements may be combined or removed from the architecture based on particular configuration needs. Network environment 10 may include a configuration capable of transmission control protocol/Internet protocol (TCP/IP) communications for the transmission or reception of packets in a network. Network environment 10 may also operate in conjunction with a user datagram protocol/IP (UDP/IP) or any other suitable protocol where appropriate and based on particular needs.

[0016] For purposes of illustrating the techniques for providing network security in example embodiments, it is important to understand the activities occurring within a given network. The following foundational information may be viewed as a basis from which the present disclosure may be properly explained. Such information is offered earnestly for purposes of explanation only and, accordingly, should not be construed in any way to limit the broad scope of the present disclosure and its potential applications.

[0017] Typical network environments used in organizations and by individuals include the ability to communicate electronically with other networks using the Internet, for example, to access web pages hosted on servers connected to the Internet, to send or receive electronic mail (i.e., email) messages, or to exchange files. However, malicious users continue to develop new tactics for using the Internet to spread malware and to gain access to confidential information. Malware generally includes any software designed to access and/or control a computer without the informed consent of the computer owner, and is most commonly used as a label for any hostile, intrusive, or annoying software such as a computer virus, bot, spyware, adware, etc. Once compromised, malware may subvert a host and use it for malicious activity, such as spamming or information theft. Malware also typically includes one or more propagation vectors that enable it to spread within an organization's network or across other networks to other organizations or individuals. Common propagation vectors include exploiting known vulnerabilities on hosts within the local network and sending emails having a malicious program attached or providing malicious links within the emails.

[0018] One way in which malware may operate is to deceive a user by using a different network protocol exchange than the user expects. The malware may be packaged so as to convince the user to allow access to run it in some innocuous way, thus allowing it access to the network, which often may require passing through a firewall or other security

measure. The malware may then exploit the access to engage in alternative or additional activities not contemplated by the user. For example, a game may send email messages or a word processor may open a web connection, using standard protocols to deceive the firewall into permitting the malware to establish remote connections.

[0019] Botnets, for example, use malware and are an increasing threat to computer security. In many cases they employ sophisticated attack schemes that include a combination of well-known and new vulnerabilities. Botnets generally use a client-server architecture where a type of malicious software (i.e., a bot) is placed on a host computer and communicates with a command and control (C&C) server, which may be controlled by a malicious user (e.g., a botnet operator). Usually, a botnet is composed of a large number of bots that are controlled by the operator using a C&C protocol through various channels, including Internet Relay Chat (IRC) and peer-to-peer (P2P) communication. The bot may receive commands from the C&C server to perform particular malicious activities and, accordingly, may execute such commands. The bot may also send any results or pilfered information back to the C&C server.

[0020] A bot is often designed to initiate communication with the C&C server and to masquerade as normal web browser traffic. For example, a bot may use a port typically used to communicate with a web server. Such bots, therefore, may not be detected by existing technologies without performing more detailed packet inspection of the web traffic. Moreover, once a bot is discovered, the botnet operator may simply find another way to masquerade network traffic by the bot to continue to present as normal web traffic. More recently, botnet operators have crafted bots to use encryption protocols such as, for example, secure socket layer (SSL), thereby encrypting malicious network traffic. Such encrypted traffic may use a Hypertext Transfer Protocol Secure (HTTPS) port so that only the endpoints involved in the encrypted session can decrypt the data. Thus, existing firewalls and other network intrusion prevention technologies may be unable to perform any meaningful inspection of the web traffic, and bots may continue to infect host computers within networks.

[0021] Other software security technology focused on preventing unauthorized program files from executing on a host computer may have undesirable side effects for end users or employees of a business or other organizational entity. Network or Information

Technology (IT) administrators may be charged with crafting extensive policies relevant to all facets of the business entity to enable employees to obtain software and other electronic data from desirable and trusted network resources. Without extensive policies in place, employees may be prevented from downloading software and other electronic data from network resources that are not specifically authorized, even if such software and other data facilitate legitimate and necessary business activities. Such systems may be so restrictive that if unauthorized software is found on a host computer, any host computer activities may be suspended pending network administrator intervention. Moreover, at the network level there may simply be too many applications to effectively track and incorporate into policies. Large whitelists or blacklists can be difficult to maintain and may degrade network performance, and some applications may not be susceptible to easy identification.

[0022] Information may be shared between a host and a firewall to collectively and mutually achieve better security, though. For example, a host may understand an application as an executable file that is running a process with specific authentication, while the firewall may understand the application as a protocol in a TCP connection, which may also be correlated to a particular user authentication. The host may share session descriptors and other metadata with the firewall, and the firewall may share network policy with the host as needed to correlate application activities with expected network behavior. Network policy may include elements of security policy as well as other network specific parameters, such as quality of service (QoS) and routing. A host may also be associated with a universally unique identifier (UUID), which can be used to correlate connections and activities originating behind network address translators.

[0023] A host may also notify the firewall of additional network connections to the host. If a host has both wireless and wired connections active simultaneously, for example, there may be a risk of data received on one connection being transmitted on the other, so it may be desirable to restrict access to sensitive data. A host may also notify the firewall if the connection is associated with a virtual machine, or if the host has mountable read/write media, such as a USB stick attached.

[0024] In some embodiments of network environment 10, a host may include multiple attachment points, causing it to have multiple IP addresses. In other embodiments, a host may use the IP version 6 (IPv6), perhaps including Privacy Extensions (RFC4941),

causing it to have one or more registered and known IPv6 addresses and one or more hidden or private IPv6 addresses. In these embodiments, an interlocked firewall may readily use dynamic information sharing to discover the user-to-host mapping for all the addresses on a host.

[0025] This dynamic information sharing between an interlocked host and firewall in network environment 10 may provide several benefits over conventional architectures. For example, by coordinating firewall policy with a host, a firewall can manage routes differently, such as by allowing or denying traffic depending on which of multiple users on a host may be attempting to establish a connection. Moreover, only applications that may need to be granularly controlled need to be controlled by the firewall. Thus, the firewall may control arbitrary or evasive applications, provide higher effective throughput, and control mobile-user traffic. In addition, traffic that does not need to be completely allowed or denied can be rate-limited. Arbitrary or evasive applications can also be rate-limited with process information available on a firewall, and differentiated services can be provided for managed and unmanaged hosts.

[0026] Many hosts may only use a single firewall for all routes, and an agent running on a host may maintain a firewall cache that can identify this firewall. In a more complex scenario, a host may use more than one firewall, in which case it is important to understand which firewall can process a given flow. The firewall cache can provide routes through more than one firewall by mapping a given network route to a particular firewall, or it may provide discovery actions for dynamic mapping. Routes are generally managed or unmanaged. A "managed route" generally refers to a route to a network through a firewall configured to accept metadata and apply network policy to network flows, while an "unmanaged route" is a route to a network through a firewall not configured to accept metadata or apply network policy. A firewall cache may identify a managed route by associating a destination network with a firewall designated for managing flows to the network, for example, or may identify an unmanaged route by associating a destination network with a null value, for example. The firewall cache may be initialized or configured by an administrator, providing separate configurations for each named network and/or default configurations for the first time a network is used. Some configurations may define

one firewall for Internet addresses, initially based on an assumption that all global IP addresses are on the Internet.

[0027] Session descriptors generally include information about a host and an application associated with a given network session. For example, a session descriptor may include a UUID associated with the host and the user credentials of a process owner. Since a user can run separate processes with different user credentials, such information may be particularly advantageous for Citrix and terminal services. A session descriptor may additionally include a filename, pathname or other unique identifier of an application file (e.g., C:\...\WINWORD.EXE) that corresponds to the process attempting to establish a network connection. For example, in some embodiments the application may be identified by a hash function of the application's executable file, so as to make it more difficult for a malicious user to spoof the application name. A firewall may correlate this information with an application identifier or protocol to ensure that the application is performing as expected.

[0028] A session descriptor may also contain information about the host environment, such as software installed on the host and the current configuration and state of the software, permitting the firewall to act as a network access control device. For example, a session descriptor may indicate whether the local anti-virus system is up to date and running. If Host-based Data Loss Prevention (HDLP) software is available, a session descriptor may also include file-typing information for file transfer. HDLP normally determines the type of file being transmitted out of the network (e.g., PDF, Word, etc.). The firewall may have additional policies about certain file types being transmitted over particular protocols, which may not be visible directly to an HDLP program.

[0029] Session descriptors and other metadata may be exchanged over an out-of-band communication channel (a "metadata channel") in some embodiments of network environment 10, which may be implemented with a protocol that provides authentication and/or encryption for communication privacy. In more particular embodiments, a Datagram Transport Layer Security (DTLS) protocol may be used to provide a metadata channel with communication privacy, and a host and a firewall may use certificates based on a common certificate authority. A policy server may distribute certificates to a host and firewall in some embodiments, while an external certificate authority may be used in other

embodiments. Some protocols, including DTLS, may also be used to establish a back channel from a firewall to a host, which may be used for error messages and diagnostics, for example.

[0030] A host can send metadata to a firewall before opening a new network flow such that, in general, metadata arrives at the firewall before the first packet of a new flow. More particularly, a firewall agent on the host may intercept the first packet of a new flow and send a session descriptor and other metadata associated with the flow, such as source IP address and port, destination IP address and port, and protocol. The firewall may maintain a metadata cache and correlate a network flow with metadata if the firewall agent releases the flow. More particularly, the firewall may correlate metadata with network flow data, which broadly refers to information that associates a given network flow with a source node (i.e., a node sending or attempting to send a packet) and a destination node (i.e., a node to which a packet is addressed) or destination nodes (e.g., broadcast or multicast address). Flow data may also include other information about the flow, such as a protocol family or protocol, for example.

[0031] For example, TCP generally opens a new flow (generally referred to as a "connection" in the context of a TCP flow) with a handshake – a host sends the first packet with one of the TCP flag bits (i.e., the SYN bit) set to indicate that a three-way handshake is in progress. Thus, an agent on a source node may intercept a new TCP connection by detecting an application on the source node sending a SYN packet (i.e., a packet with the SYN bit set) and holding the SYN packet. The agent may be able to identify a firewall for managing the route to a destination node associated with the new connection, such as by locating the route and its associated firewall in a firewall cache, and send metadata to the firewall (which the firewall can cache) over a secure metadata channel. The connection request may then be released by sending the SYN packet to the firewall, and the firewall may correlate the source IP, destination IP, protocol, etc.

[0032] Flows are not limited to communications using a reliable protocol such as TCP; a flow may also provide communications using an unreliable protocol such as UDP or IP. In other embodiments, an agent may track flows that use an unreliable protocol and intercept a new flow by holding the first packet of a flow while it transmits metadata. The agent may also be able to retransmit the metadata by caching a hash of the first packet of a

flow and comparing the hash to the hash of subsequent packets to determine if the first packet is being retransmitted by an application. In yet other embodiments, a firewall may track flows and cache the first packet until metadata arrives. In still yet other embodiments, metadata may be sent with every packet in a flow using an unreliable protocol, or never sent. Caches of first packet data can be very short-lived (e.g. less than one second to five seconds).

[0033] In accordance with embodiments disclosed herein, network environment 10 may provide a system and method for host-initiated discovery of an interlocked firewall. For example, a firewall agent may consult a firewall cache if it detects a new flow to determine which firewall should receive metadata for the flow. If no cache entry is found, the firewall agent may consult a discovery table to determine if the address is local or lies along the Internet path. Alternatively, the firewall cache may include discovery actions for certain routes. The firewall agent may send a firewall discovery query based on an appropriate entry in the discovery table or firewall cache. A firewall along the route may receive the firewall discovery query and return a firewall discovery result, which can provide the firewall address.

[0034] Managed hosts and firewalls may also maintain a shared secret (e.g., a password, key, etc.) for authentication of discovery messages. The shared secret may be distributed by a policy server or manually configured, for example, and a firewall may share the same secret with more than one host, including all hosts within a site. In certain embodiments, the shared secret may be a function of time. In yet other embodiments, managed hosts and firewalls may use asymmetric key cryptography (i.e., public key cryptography) to secure discovery messages. Key pairs may also be distributed by a policy server in some embodiments, or manually configured in others, and the same key pair may be shared with more than one firewall.

[0035] In more particular embodiments, firewall discovery messages may be implemented with protocol data units (PDUs) packaged in an Internet Control Message Protocol (ICMP) or UDP packet. For example, a firewall discovery query PDU may include the IP address of the firewall agent host, the IP address of the discovery target (which may be retrieved from the firewall cache), a firewall query (e.g., a magic number, subtype code, and discovery target address), and a message authentication code (MAC), which may be

packaged in an ICMP echo message and sent to the discovery target address. A firewall discovery result PDU in an ICMP echo reply message may return the IP address of the host and the IP address of the discovery target, along with firewall information (e.g., magic number, subtype code, firewall IP address, host manager port) and a message authentication code. In such embodiments, the magic number may be a 32-bit identifier (e.g., 0x46484131 or "FHA1"), for example, which may also act as a protocol version number. A discovery target may be a globally known address designated for discovery queries in ICMP echo messages, such that it may be used to discover an Internet route, for example, or it may be a node within a given network designated for discovery of internal or external routes.

[0036] In certain example embodiments, each MAC may be a hash-based message authentication code (HMAC). In general, an HMAC is a MAC involving a cryptographic hash function, such as MD5 or SHA-1, in combination with a shared secret (e.g., a secret key), which can be used to verify data integrity and authenticity of a message. For example, a firewall discovery query PDU may provide an HMAC-SHA1 based on a key shared with the host initiating discovery and data in the firewall query, while a firewall discovery result PDU may provide an HMAC-SHA1 based on the shared key and firewall information.

[0037] In other embodiments, a firewall may have a public/private key pair that it can use to establish a metadata channel (e.g., a DTLS connection). A host may also use the firewall's public key to encrypt a hash of a discovery query message (using RSA for example), and the firewall's private key may be used to encrypt a hash of a discovery reply message. The encrypted hashes can be inserted into the discovery query or reply, and the discovery messages can be validated with the appropriate key. For example, an ICMP DU packet may be used as described above, but replacing the HMAC with the encrypted hash.

[0038] In example embodiments, a firewall agent on a host may send a firewall discovery query, tracking a discovery rule and target. A host manager of the firewall may intercept what appears to be an innocuous ICMP echo message from the firewall agent if it detects a firewall discovery query PDU and can validate the query by computing a MAC for the query using the shared secret. If the query is valid, the host manager may construct a firewall discovery result and send it to the firewall agent in an ICMP echo reply. The firewall agent on the host may receive the ICMP echo reply and process it if it detects a firewall

discovery result PDU, and can authenticate the query based on its MAC. For example, if the message is authentic, a host may update its firewall cache to reflect the firewall information in the result.

[0039] In other embodiments, a firewall agent may send a firewall discovery query in a UDP packet. A host manager of the firewall may be configured to inspect UDP packets from the firewall agent and to block UDP-based firewall discovery queries from leaving the network to prevent inadvertent leakage. If the host manager detects a firewall discovery query PDU, it can validate the query by computing a MAC for the query using the shared secret. If the query is valid, the host manager may construct a firewall discovery result and send it to the firewall agent in a UDP packet. The firewall agent on the host may receive the UDP packet and process it if it detects a firewall discovery result PDU, and can authenticate the query based on its MAC. For example, if the message is authentic, a host may update its firewall cache to reflect the firewall information in the result.

[0040] Turning to FIGURE 2, FIGURE 2 is a simplified block diagram illustrating additional details that may be associated with potential embodiments of network environment 10. FIGURE 2 includes Internet 15, user host 20a, firewall 25, and a server 45. Each of user host 20a and firewall 25 may include a respective processor 50a-50b, a respective memory element 55a-55b, and various hardware and/or software modules. More particularly, user host 20a may include an application 60, a configuration database 65, a client certificate 70, and a firewall agent 75, which may maintain a firewall cache 77. Firewall 25 may include a host manager 80 and a policy module 85, as well as a log 90, a metadata cache 95, and a server certificate 97.

[0041] In one example implementation, user hosts 20a-20b, firewall 25, and/or policy server 30 are network elements, which are meant to encompass network appliances, servers, routers, switches, gateways, bridges, loadbalancers, processors, modules, or any other suitable device, component, element, or object operable to exchange information in a network environment. Network elements may include any suitable hardware, software, components, modules, or objects that facilitate the operations thereof, as well as suitable interfaces for receiving, transmitting, and/or otherwise communicating data or information in a network environment. This may be inclusive of appropriate algorithms and communication protocols that allow for the effective exchange of data or information.

However, user hosts 20a-20b may be distinguished from other network elements, as they tend to serve as a terminal point for a network connection, in contrast to a gateway or router that tends to serve as an intermediate point in a network connection. User hosts 20a-20b may also be representative of wireless network nodes, such as an i-Phone, i-Pad, Android phone, or other similar telecommunications devices.

[0042] In regards to the internal structure associated with network environment 10, each of user hosts 20a-20b, firewall 25, and/or policy server 30 can include memory elements for storing information to be used in the operations outlined herein. Each of user hosts 20a-20b, firewall 25, and/or policy server 30 may keep information in any suitable memory element (e.g., random access memory (RAM), read-only memory (ROM), erasable programmable ROM (EPROM), electrically erasable programmable ROM (EEPROM), application specific integrated circuit (ASIC), etc.), software, hardware, or in any other suitable component, device, element, or object where appropriate and based on particular needs. Any of the memory items discussed herein (e.g., memory elements 55a-55b) should be construed as being encompassed within the broad term 'memory element.' The information being used, tracked, sent, or received by user hosts 20a-20b, firewall 25, and/or policy server 30 could be provided in any database, register, queue, table, cache, control list, or other storage structure, all of which can be referenced at any suitable timeframe. Any such storage options may be included within the broad term 'memory element' as used herein.

[0043] In certain example implementations, the functions outlined herein may be implemented by logic encoded in one or more tangible media (e.g., embedded logic provided in an ASIC, digital signal processor (DSP) instructions, software (potentially inclusive of object code and source code) to be executed by a processor, or other similar machine, etc.), which may be inclusive of non-transitory media. In some of these instances, memory elements (as shown in FIGURE 2) can store data used for the operations described herein. This includes the memory elements being able to store software, logic, code, or processor instructions that are executed to carry out the activities described herein.

[0044] In one example implementation, user hosts 20a-20b, firewall 25, and/or policy server 30 may include software modules (e.g., firewall agent 75 and/or host manager 80) to achieve, or to foster, operations as outlined herein. In other embodiments, such

operations may be carried out by hardware, implemented externally to these elements, or included in some other network device to achieve the intended functionality. Alternatively, these elements may include software (or reciprocating software) that can coordinate in order to achieve the operations, as outlined herein. In still other embodiments, one or all of these devices may include any suitable algorithms, hardware, software, components, modules, interfaces, or objects that facilitate the operations thereof.

[0045] Additionally, each of user hosts 20a-20b, firewall 25, and/or policy server 30 may include a processor that can execute software or an algorithm to perform activities as discussed herein. A processor can execute any type of instructions associated with the data to achieve the operations detailed herein. In one example, the processors (as shown in FIGURE 2) could transform an element or an article (e.g., data) from one state or thing to another state or thing. In another example, the activities outlined herein may be implemented with fixed logic or programmable logic (e.g., software/computer instructions executed by a processor) and the elements identified herein could be some type of a programmable processor, programmable digital logic (e.g., a field programmable gate array (FPGA), an EPROM, an EEPROM) or an ASIC that includes digital logic, software, code, electronic instructions, or any suitable combination thereof. Any of the potential processing elements, modules, and machines described herein should be construed as being encompassed within the broad term 'processor.'

[0046] FIGURE 3 is a simplified table illustrating example entries in a firewall cache 300 that may be associated with some embodiments of network environment 10. Table 300 includes a discovery target column 305 and a discovery action column 310. Discovery target column 305 generally identifies target hosts or subnets (e.g., by destination address and network mask), and discovery action column 310 associates a discovery action with the host or subnet in the corresponding entry of discovery target column 305. In certain embodiments, a discovery action may indicate that the target is exempt from discovery, that a discovery query should be sent directly to the target, that discovery should be sent directly to the target and the result should be associated with the default (e.g., Internet) route, or that the default path in the cache should be used. In general, no discovery queries are sent for exempt targets. If the discovery action is direct discovery, a discovery query

may be addressed to the target, but the intervening firewall may intercept and reply, allowing a host to update the firewall cache accordingly.

[0047] For example, rows 315 and 320 indicate that targets in the 10.0.0.0 or 192.168.0.0 subnets are exempt from firewall discovery action. Row 325 associates hosts in the 172.17.0.0 subnet with a direct discovery action, such that a host opening a connection to a server with an address of 172.17.2.100, for example, may send a firewall discovery query addressed to that server. Row 330 indicates that a host opening a connection to example.com should send a firewall discovery query to example.com, and to use any returned firewall discovery result as the default Internet firewall. The last entry, in row 335, indicates that all other addresses should use the default Internet firewall.

[0048] In general, rules (i.e., discovery targets and associated discovery actions) are evaluated in order. In a typical small network environment, private address space may be exempt when a single firewall to the Internet is available and there are no internal interlocked firewalls. In the example of table 300, however, a firewall may be involved for anything in the 172.0.0.0 subnet. In some embodiments, a prefix length parameter (e.g., /24 for IPv4 and /64 for IPv6) may be added to control the granularity of the cache entry.

[0049] FIGURE 4 is a simplified interaction diagram illustrating potential operations that may be associated with example embodiments of network environment 10 that transport firewall discovery messages in ICMP packets. FIGURE 4 includes application 60, firewall agent 75, and firewall cache 300, which may be installed in a user host, such as user host 20a, for example. FIGURE 4 also illustrates host manager 80, metadata cache 95, and policy module 85, which may be installed in a firewall such as firewall 25, for example. An intranet 78, Internet 15, and server 45 are also depicted in FIGURE 4.

[0050] An application such as application 60 may attempt to open a new TCP connection at 405, with a server such as server 45, for example. In this scenario, server 45 is in the example.com domain. Firewall agent 75 may intercept and hold the new connection, consulting firewall cache 300 at 410 to identify a firewall or discovery action associated with the route to server 45. Based on the example entry 330 in cache 300, firewall agent 75 may send a firewall discovery query in an ICMP echo message addressed to example.com at 415. Such a discovery query may, for example, include the IP address of user host 20a, the address of server 45 (i.e., example.com or the IP address of server 45), a firewall query

(FHA1 | 0 | ::example.com | 0), and an HMAC-SHA1 (shared key, firewall query). In this example, the first field of the firewall query ("FHA1") is a literal that may be used to identify the protocol and version number, the second field (0) indicates the message is a query, the third field (::example.com) represents the IPv6 address or the IPv4 mapped form of example.com, and the fourth field (0) is a constant placeholder that can be used to keep the structure of query and result messages the same, which allows the protocol to be reflective and efficient.

[0051] Host manager 80 may receive and inspect the ICMP echo message in route to server 45. In this example, host manager 80 recognizes the ICMP echo message payload as a firewall discovery query PDU, validates the MAC by computing an HMAC-SHA1 on the shared key and the firewall query, and constructs a firewall discovery result at 420. Such a discovery result may, for example, include the IP address of user host 20a, the address of server 45 (i.e., example.com or the IP address of server 45), firewall information (FHA1 | 1 | firewall address | host manager port), and an HMAC-SHA1(shared key, firewall information). Host manager 80 may send the discovery result in an ICMP echo reply to firewall agent 75 at 425.

[0052] Firewall agent 75 may receive and inspect the ICMP echo reply from host manager 80. In this example, firewall agent 75 may recognize the ICMP echo reply payload as a firewall discovery result PDU and update firewall cache 300 at 430, such as by substituting the discovery action in row 330 with the firewall address and port for host manager 80. Moreover, in this particular example the discovery action in row 330 specifies that the discovered firewall should also be used as the default Internet firewall, so firewall agent may also substitute any discovery action in row 335 with the firewall address and port for host manager 80.

[0053] Firewall 75 may then open a connection (e.g., a DTLS connection) to the firewall at 435, using a certificate (e.g., client certificate 70) distributed by a policy server, for example. The connection may also be added to firewall cache 300 at 440 for future connections. Firewall agent 75 may send metadata for the connection initiated by application 60 to host manager 80 at 445a via a DTLS packet, for example. Host manager 80 may store the metadata in metadata cache 95 at 445b. Firewall agent 75 may release the connection at 450a, allowing data from application 60 to flow to host manager 80. Host

manager 80 may provide connection data (i.e., TCP flow data, such as source IP address/port, destination IP address/port, protocol, etc.) to policy module 85 at 450b, and policy module 85 may correlate the connection data with metadata from metadata cache 95 at 455 to apply appropriate network policy at 460. In the example of FIGURE 4, network policy permits the connection, so the connection may be released to server 45 at 465 and data may flow between server 45 and application 60 at 470.

[0054] FIGURE 5 is a simplified interaction diagram illustrating potential operations that may be associated with other example embodiments of network environment 10 that transport firewall discovery messages in ICMP packets. FIGURE 5 includes application 60, firewall agent 75, and firewall cache 300, which may be installed in a user host, such as user host 20a, for example. FIGURE 5 also illustrates host manager 80, metadata cache 95, and policy module 85, which may be installed in a firewall such as firewall 25, for example. Intranet 78, a network 88, and server 45 are also depicted in FIGURE 5.

[0055] An application such as application 60 may attempt to open a new flow at 505, with a server such as server 45, for example. In this scenario, server 45 is in the 172.17.0.0 subnet, with an IP address of 172.17.2.100, for example. Firewall agent 75 may intercept and hold the new flow, consulting firewall cache 300 at 510 to identify a firewall or discovery action associated with the route to server 45. Based on the example entry 325 in cache 300, firewall agent 75 may send a firewall discovery query in an ICMP echo message addressed to server 45 at 515. Such a discovery query may, for example, include the IP address of user host 20a, the IP address of server 45, a firewall query (FHA1 | 0 | ::172.17.2.100 | 0), and an HMAC-SHA1 (shared key, firewall query).

[0056] Host manager 80 may receive and inspect the ICMP echo message in route to server 45. In this example, host manager 80 recognizes the ICMP echo payload as a firewall discovery query PDU, and may validate the MAC by computing an HMAC-SHA1 on the shared key and the firewall query before constructing a firewall discovery result at 520. Such a discovery result may, for example, include the IP address of user host 20a, the address of server 45 (e.g., 172.17.2.100), firewall information (FHA1 | 1 | firewall address | host manager port), and an HMAC-SHA1(shared key, firewall information). Host manager 80 may send the discovery result in an ICMP echo reply to firewall agent 75 at 525.

[0057] Firewall agent 75 may receive and inspect the ICMP echo reply from host manager 80. In this example, firewall agent 75 may recognize the ICMP echo reply payload as a firewall discovery result PDU and update firewall cache 300 at 530, such as by substituting the discovery action in row 330 with the firewall address and port for host manager 80.

[0058] Firewall 75 may then open a flow (e.g., a DTLS connection) to the firewall at 535, using a certificate (e.g., client certificate 70) distributed by a policy server, for example. The flow may also be added to firewall cache 300 at 540 for future flows. Firewall agent 75 may send metadata for the flow initiated by application 60 to host manager 80 at 545a via a DTLS packet, for example. Host manager 80 may store the metadata in metadata cache 95 at 545b. Firewall agent 75 may release the flow at 550a, allowing data from application 60 to flow to host manager 80. Host manager 80 may provide flow data (e.g., source IP address/port, destination IP address/port, protocol, etc.) to policy module 85 at 550b, and policy module 85 may correlate the connection data with metadata from metadata cache 95 at 555 to apply appropriate network policy at 560. In the example of FIGURE 5, network policy permits the flow, so the flow may be released to server 45 at 565 and data may flow between server 45 and application 60 at 570.

[0059] FIGURE 6 is a simplified interaction diagram illustrating potential operations that may be associated with example embodiments of network environment 10 that transport firewall discovery messages in UDP packets. FIGURE 6 includes application 60, firewall agent 75, and firewall cache 300, which may be installed in a user host, such as user host 20a, for example. FIGURE 6 also illustrates host manager 80, metadata cache 95, and policy module 85, which may be installed in a firewall such as firewall 25, for example. Intranet 78, Internet 15, and server 45 are also depicted in FIGURE 6.

[0060] An application such as application 60 may attempt to open a new flow at 605, with a server such as server 45, for example. In this scenario, server 45 is in the example.com domain. Firewall agent 75 may intercept and hold the new flow, consulting firewall cache 300 at 610 to identify a firewall or discovery action associated with the route to server 45. Based on the example entry 330 in cache 300, firewall agent 75 may send a firewall discovery query in a UDP packet to a preconfigured UDP port on server 45 at 615. Such a discovery query may, for example, include the IP address of user host 20a, the

address of server 45 (i.e., example.com or the IP address of server 45), a firewall query (FHA1 | 0 | ::example.com | 0), and an HMAC-SHA1 (shared key, firewall query).

[0061] Host manager 80 may receive and inspect the UDP packet in route to the preconfigured UDP port. In this example, host manager 80 recognizes the UPD payload as a firewall discovery query PDU, validates the MAC by computing an HMAC-SHA1 on the shared key and the firewall query, and constructs a firewall discovery result at 620. Such a discovery result may, for example, include the IP address of user host 20a, the address of server 45 (i.e., example.com or the IP address of server 45), firewall information (FHA1 | 1 | firewall address | host manager port), and an HMAC-SHA1(shared key, firewall information). Host manager 80 may send the discovery result in a UDP packet to firewall agent 75 at 625.

[0062] Firewall agent 75 may receive and inspect the UDP packet from host manager 80. In this example, firewall agent 75 may recognize the UDP payload as a firewall discovery result PDU and update firewall cache 300 at 630, such as by substituting the discovery action in row 330 with the firewall address and port for host manager 80. Moreover, in this particular example the discovery action in row 330 specifies that the discovered firewall should also be used as the default Internet firewall, so firewall agent may also substitute any discovery action in row 335 with the firewall address and port for host manager 80.

[0063] Firewall 75 may then open a connection (e.g., a DTLS connection) to the firewall at 635, using a certificate (e.g., client certificate 70) distributed by a policy server, for example. The connection may also be added to firewall cache 300 at 640 for future connections. Firewall agent 75 may send metadata for the flow initiated by application 60 to host manager 80 at 645a via a DTLS packet, for example. Host manager 80 may store the metadata in metadata cache 95 at 645b. Firewall agent 75 may release the flow at 650a, allowing data from application 60 to flow to host manager 80. Host manager 80 may provide flow data (e.g., source IP address/port, destination IP address/port, protocol, etc.) to policy module 85 at 650b, and policy module 85 may correlate the flow data with metadata from metadata cache 95 at 655 to apply appropriate network policy at 660. In the example of FIGURE 6, network policy permits the flow, so the flow may be released to server 45 at 665 and data may flow between server 45 and application 60 at 670.

[0064] FIGURE 7 is a simplified interaction diagram illustrating potential operations that may be associated with other example embodiments of network environment 10 that transport firewall discovery messages in UDP packets. FIGURE 7 includes application 60, firewall agent 75, and firewall cache 300, which may be installed in a user host, such as user host 20a, for example. FIGURE 7 also illustrates host manager 80, metadata cache 95, and policy module 85, which may be installed in a firewall such as firewall 25, for example. Intranet 78, network 88, and server 45 are also depicted in FIGURE 7.

[0065] An application such as application 60 may attempt to open a new TCP connection at 705, with a server such as server 45, for example. In this scenario, server 45 is in the 172.17.0.0 subnet, with an IP address of 172.17.2.100, for example. Firewall agent 75 may intercept and hold the new connection, consulting firewall cache 300 at 710 to identify a firewall or discovery action associated with the route to server 45. Based on the example entry 325 in cache 300, firewall agent 75 may send a firewall discovery query in a UDP packet to a preconfigured port on server 45 at 715. Such a discovery query may, for example, include the IP address of user host 20a, the IP address of server 45, a firewall query (FHA1 | 0 | ::172.17.2.100 | 0), and an HMAC-SHA1 (shared key, firewall query).

[0066] Host manager 80 may receive and inspect the UDP packet in route to server 45. In this example, host manager 80 recognizes the UDP payload as a firewall discovery query PDU, validates the MAC by computing an HMAC-SHA1 on the shared key and the firewall query, and constructs a firewall discovery result at 720. Such a discovery result may, for example, include the IP address of user host 20a, the address of server 45 (e.g., 172.17.2.100), firewall information (FHA1 | 1 | firewall address | host manager port), and an HMAC-SHA1(shared key, firewall information). Host manager 80 may send the discovery result in an UDP packet to firewall agent 75 at 725.

[0067] Firewall agent 75 may receive and inspect the UDP packet from host manager 80. In this example, firewall agent 75 may recognize the UDP payload as a firewall discovery result PDU and update firewall cache 300 at 730, such as by substituting the discovery action in row 330 with the firewall address and port for host manager 80.

[0068] Firewall 75 may then open a flow (e.g., a DTLS connection) to the firewall at 735, using a certificate (e.g., client certificate 70) distributed by a policy server, for example. The flow may also be added to firewall cache 300 at 740 for future connections. Firewall

agent 75 may send metadata for the connection initiated by application 60 to host manager 80 at 745a via a DTLS packet, for example. Host manager 80 may store the metadata in metadata cache 95 at 745b. Firewall agent 75 may release the connection at 750a, allowing data from application 60 to flow to host manager 80. Host manager 80 may provide connection data (e.g., source IP address/port, destination IP address/port, protocol, etc.) to policy module 85 at 750b, and policy module 85 may correlate the connection data with metadata from metadata cache 95 at 755 to apply appropriate network policy at 760. In the example of FIGURE 7, network policy permits the connection, so the connection may be released to server 45 at 765 and data may flow between server 45 and application 60 at 770.

[0069] FIGURE 8 is an example packet data unit (PDU) format 800 that may be associated with exchanging metadata over a metadata channel in example embodiments of network environment 10. PDU format 800 may include, for example, network flow data 805 and session descriptor data 810. Network flow data 805 may provide information associated with a new flow from a source, such as an application on a managed host. In PDU format 800, for instance, network flow data 805 may identify a protocol (short protocol) (e.g., TCP, UDP, ICMP, GRE, IPSec, etc.), the IP address of the source node (IPaddress source_address), the port number of the process opening the connection (short source_port), the IP address of the destination node (IPaddress dest_address), and the port number of the process receiving the connection on the destination node (short dest_port). Session descriptor 810 may provide information about a user associated with the application opening the connection, such as a secure ID (string sid), a domain associated with the user (string domain), and a user name (string user), as well as information about the application, such as the full path of the application (string application_path). Other information in session descriptor 810 may provide data about the state of the source node (e.g., a host), including the state of a host firewall (boolean FW_enabled) and antivirus software running on the host (boolean AV_enabled), and information about interfaces on the source node (Interface interfaces[]). PDU format 800 is merely illustrative, though, and may be readily adapted to provide alternative or additional metadata, such as information about an intrusion prevention system, routing information, additional vendor information, etc.

[0070] Network environment 10 may provide significant advantages, some of which have already been discussed. For example, network environment 10 can provide authentication of host/firewall interlock data with low protocol overhead and minimal configuration. Network environment 10 may be readily adapted to reuse standard code packages, leveraging configuration data, protocols such as DTLS, and timers in TCP and application layer protocols, while co-existing within existing network policy and security infrastructure.

[0071] Network environment 10 may also operate seamlessly with unmanaged routes. For example, a firewall agent may intercept a new flow from an application to a server and determine from a firewall cache that the route is unmanaged. The firewall agent may release the flow and the flow with the server may be established with no additional packet overhead.

[0072] In the examples provided above, as well as numerous other potential examples, interaction may be described in terms of two, three, or four network elements. However, the number of network elements has been limited for purposes of clarity and example only. In certain cases, it may be easier to describe one or more of the functionalities of a given set of operations by only referencing a limited number of network elements. It should be appreciated that network environment 10 is readily scalable and can accommodate a large number of components, as well as more complicated/sophisticated arrangements and configurations. Accordingly, the examples provided should not limit the scope or inhibit the broad teachings of network environment 10 as potentially applied to a myriad of other architectures. Additionally, although described with reference to particular scenarios, where a particular module, such as policy module 85, is provided within a network element, these modules can be provided externally, or consolidated and/or combined in any suitable fashion. In certain instances, such modules may be provided in a single proprietary unit.

[0073] It is also important to note that the steps in the appended diagrams illustrate only some of the possible scenarios and patterns that may be executed by, or within, network environment 10. Some of these steps may be deleted or removed where appropriate, or these steps may be modified or changed considerably without departing from the scope of teachings provided herein. In addition, a number of these operations

have been described as being executed concurrently with, or in parallel to, one or more additional operations. However, the timing of these operations may be altered considerably. The preceding operational flows have been offered for purposes of example and discussion. Substantial flexibility is provided by network environment 10 in that any suitable arrangements, chronologies, configurations, and timing mechanisms may be provided without departing from the teachings provided herein.

[0074] Numerous other changes, substitutions, variations, alterations, and modifications may be ascertained to one skilled in the art and it is intended that the present disclosure encompass all such changes, substitutions, variations, alterations, and modifications as falling within the scope of the appended claims. In order to assist the United States Patent and Trademark Office (USPTO) and, additionally, any readers of any patent issued on this application in interpreting the claims appended hereto, Applicant wishes to note that the Applicant: (a) does not intend any of the appended claims to invoke paragraph six (6) of 35 U.S.C. section 112 as it exists on the date of the filing hereof unless the words "means for" or "step for" are specifically used in the particular claims; and (b) does not intend, by any statement in the specification, to limit this disclosure in any way that is not otherwise reflected in the appended claims.

WHAT IS CLAIMED IS:

1. A method, comprising:

intercepting a network flow to a destination node having a network address;

sending a discovery query based on a discovery action associated with the network address in a firewall cache;

receiving a discovery result from a firewall;

sending metadata associated with the network flow to the firewall; and

releasing the network flow.

2. The method of Claim 1, wherein the discovery query is sent in an Internet Control Message Protocol packet.

3. The method of Claim 1, wherein the discovery query is sent in an Internet Control Message Protocol echo message.

4. The method of Claim 1, wherein the discovery query is sent in a User Datagram Protocol.

5. The method of Claim 1, wherein the metadata is sent using a Datagram Transport Layer Security protocol.

6. The method of Claim 1, wherein the discovery query comprises a message authentication code.

7. The method of Claim 1, wherein the discovery query comprises a hash-based message authentication code.

8. The method of Claim 1, further comprising authenticating the discovery result with a message authentication code.

9. The method of Claim 1, wherein the discovery query comprises a hash-based message authentication code with a shared secret.

10. The method of Claim 1, wherein the discovery query comprises a public key encryption of a hash of the discovery query.

11. The method of Claim 1, further comprising authenticating the discovery result with a public key decryption of a hash of the discovery result.

12. The method of Claim 1, further comprising correlating the metadata with the network flow to apply a network policy to the network flow.

13. A method, comprising:

receiving a discovery query from a source node;

sending a discovery result to the source node, wherein the discovery result identifies a firewall for managing a route to a destination node;

receiving metadata from the source node over a metadata channel;

intercepting a network flow from the source node to the destination node; and

correlating the metadata with the network flow to apply a network policy to the network flow.

14. The method of Claim 13, wherein the metadata channel uses a Datagram Transport Layer Security protocol.

15. The method of Claim 13, further comprising authenticating the discovery query with a message authentication code.

16. The method of Claim 13, further comprising authenticating the discovery query with a hash-based message authentication code.

17. The method of Claim 13, further comprising authenticating the discovery query with a private key decryption of a public key encryption of a hash of the discovery query.

18. The method of Claim 13, further comprising authenticating the discovery result with a message authentication code.

19. The method of Claim 13, further comprising authenticating the discovery result with a private key encryption of a hash of the discovery result.

20. The method of Claim 13, wherein the discovery result is sent in an Internet Control Message Protocol packet.

21. The method of Claim 13, wherein the discovery result is sent in an Internet Control Message Protocol packet with a message authentication code.

22. The method of Claim 13, wherein the discovery result is sent in an Internet Control Message Protocol echo reply message.

23. The method of Claim 13, wherein the discovery result is sent in a User Datagram Protocol.

24. The method of Claim 13, wherein the discovery result is sent in a User Datagram Protocol with a message authentication code.

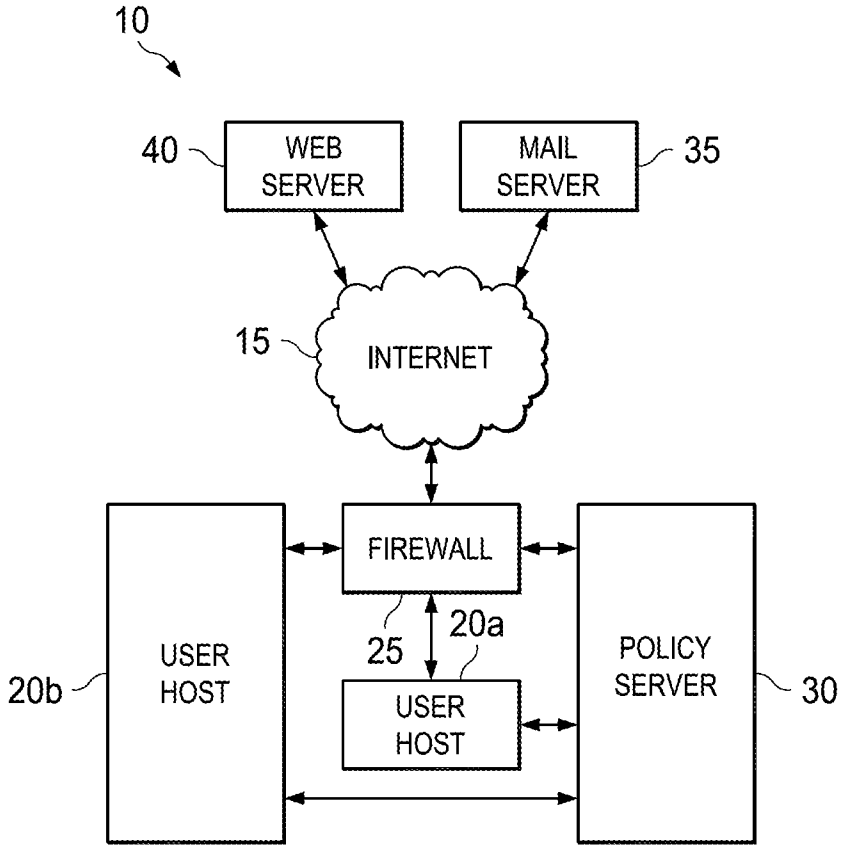

FIG. 1

300

305 PREFIX	310 ACTION
315 10.0.0.0/8	EXEMPT
320 192.168.0.0/16	EXEMPT
325 172.17.0.0/12	DISCOVER FIREWALL
330 example.com/32	DISCOVER DEFAULT INTERNET FIREWALL
335 0.0.0.0/0	USE DEFAULT INTERNET FIREWALL

FIG. 3

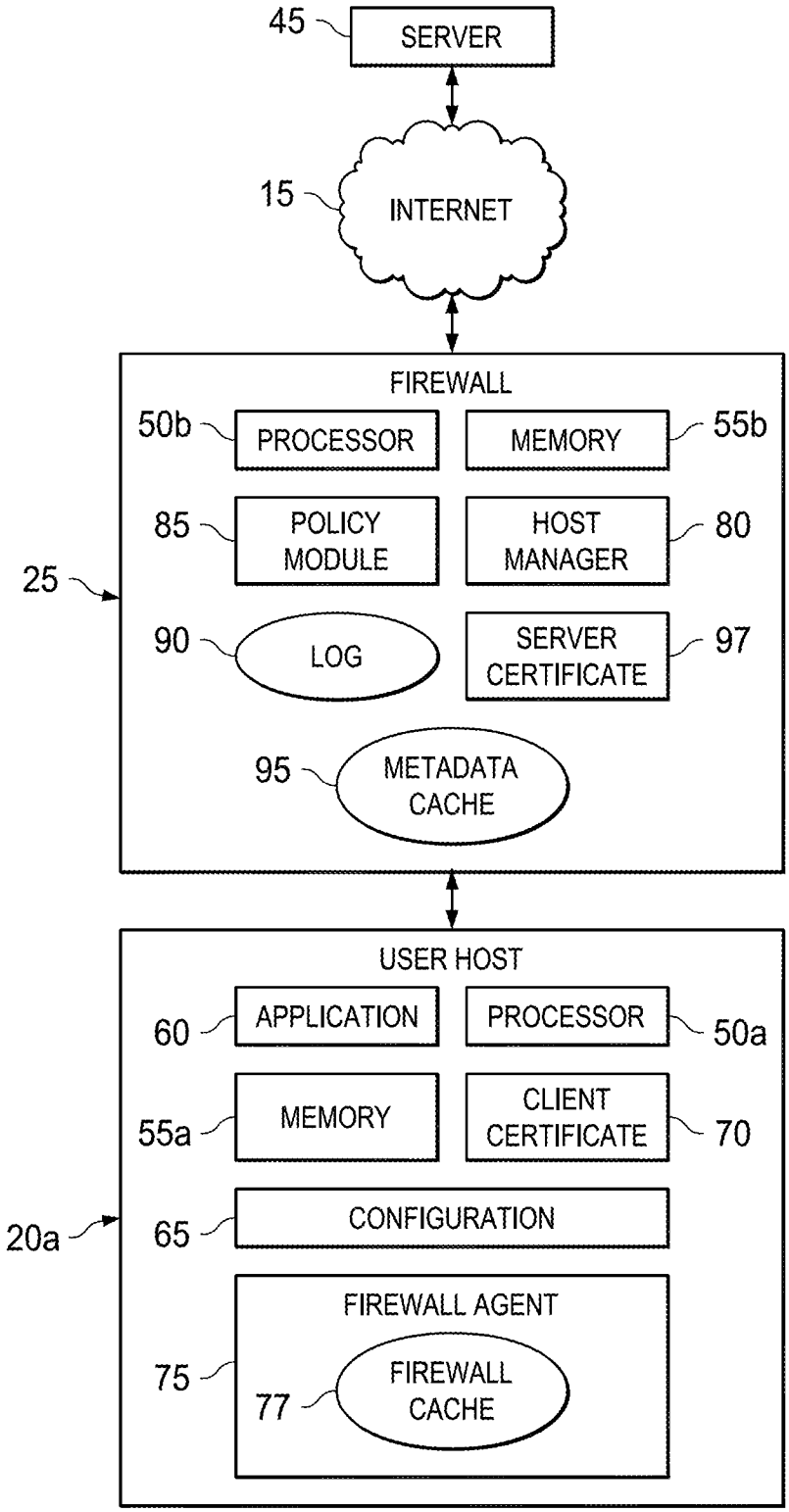
45
SERVER
15
INTERNET
FIREWALL
50b
PROCESSOR
MEMORY
55b
85
POLICY MODULE
HOST MANAGER
80
25
90
LOG
SERVER CERTIFICATE
97
95
METADATA CACHE
USER HOST
60
APPLICATION
PROCESSOR
50a
55a
MEMORY
CLIENT CERTIFICATE
70
20a
65
CONFIGURATION
FIREWALL AGENT
75
77
FIREWALL CACHE

FIG. 2

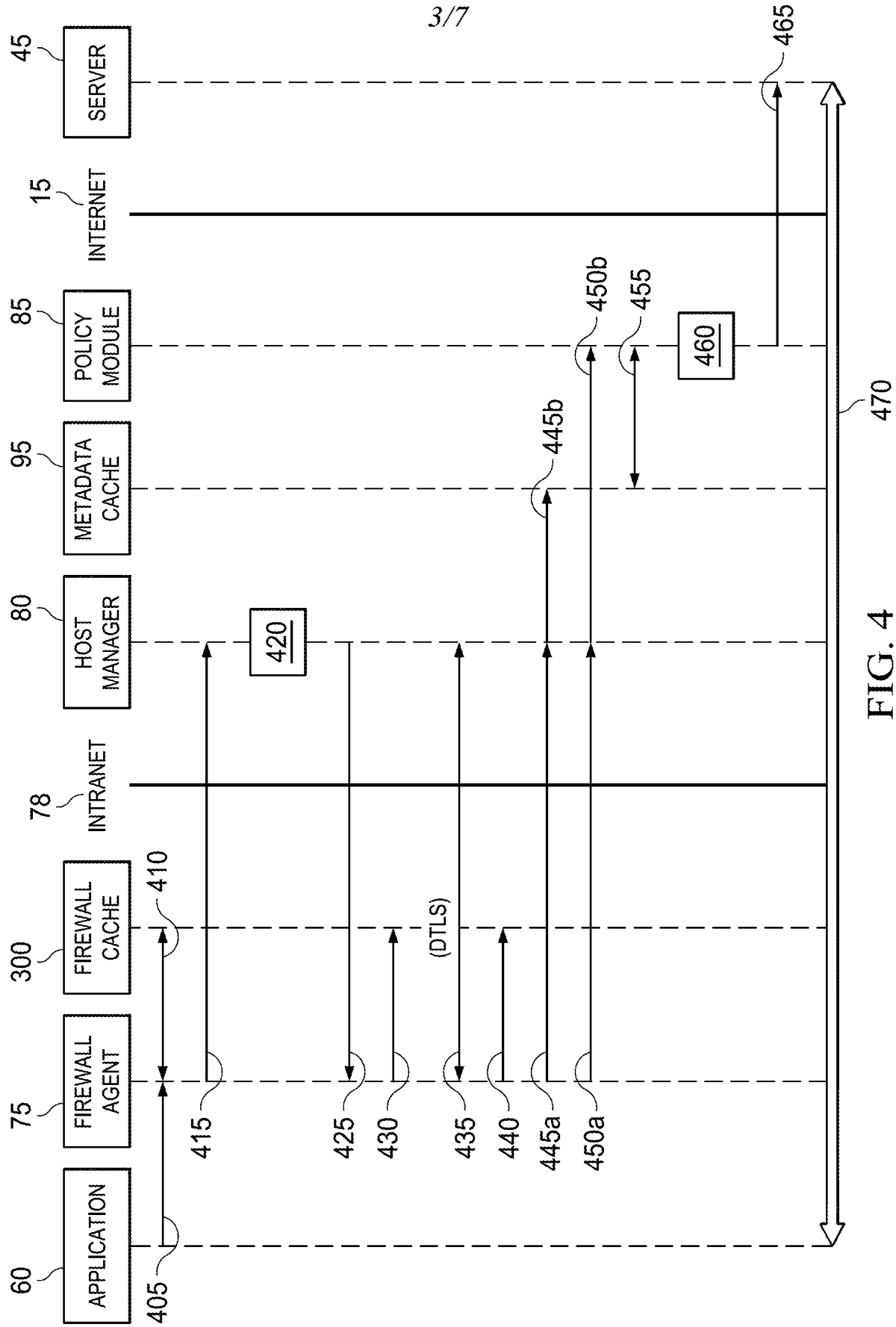
60
APPLICATION
75
FIREWALL AGENT
300
FIREWALL CACHE
78
INTRANET
80
HOST MANAGER
95
METADATA CACHE
85
POLICY MODULE
15
INTERNET
45
SERVER
405
410
415
420
425
430
435
(DTLS)
440
445a
445b
450a
450b
455
460
465
470

FIG. 4

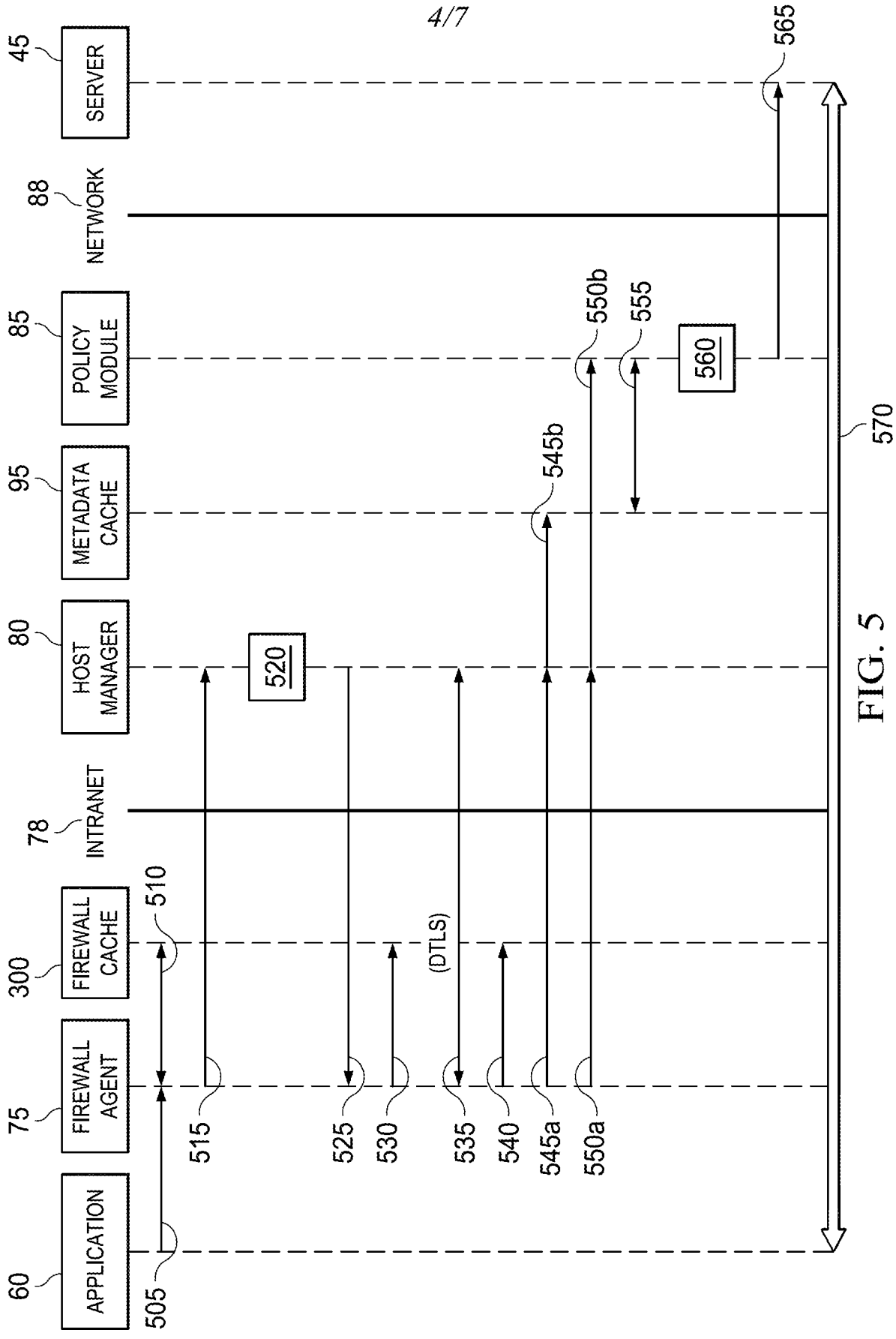
60
APPLICATION
75
FIREWALL AGENT
300
FIREWALL CACHE
78
INTRANET
80
HOST MANAGER
95
METADATA CACHE
85
POLICY MODULE
88
NETWORK
45
SERVER
505
510
515
520
525
530
(DTLS)
535
540
545a
545b
550a
550b
555
560
565
570

FIG. 5

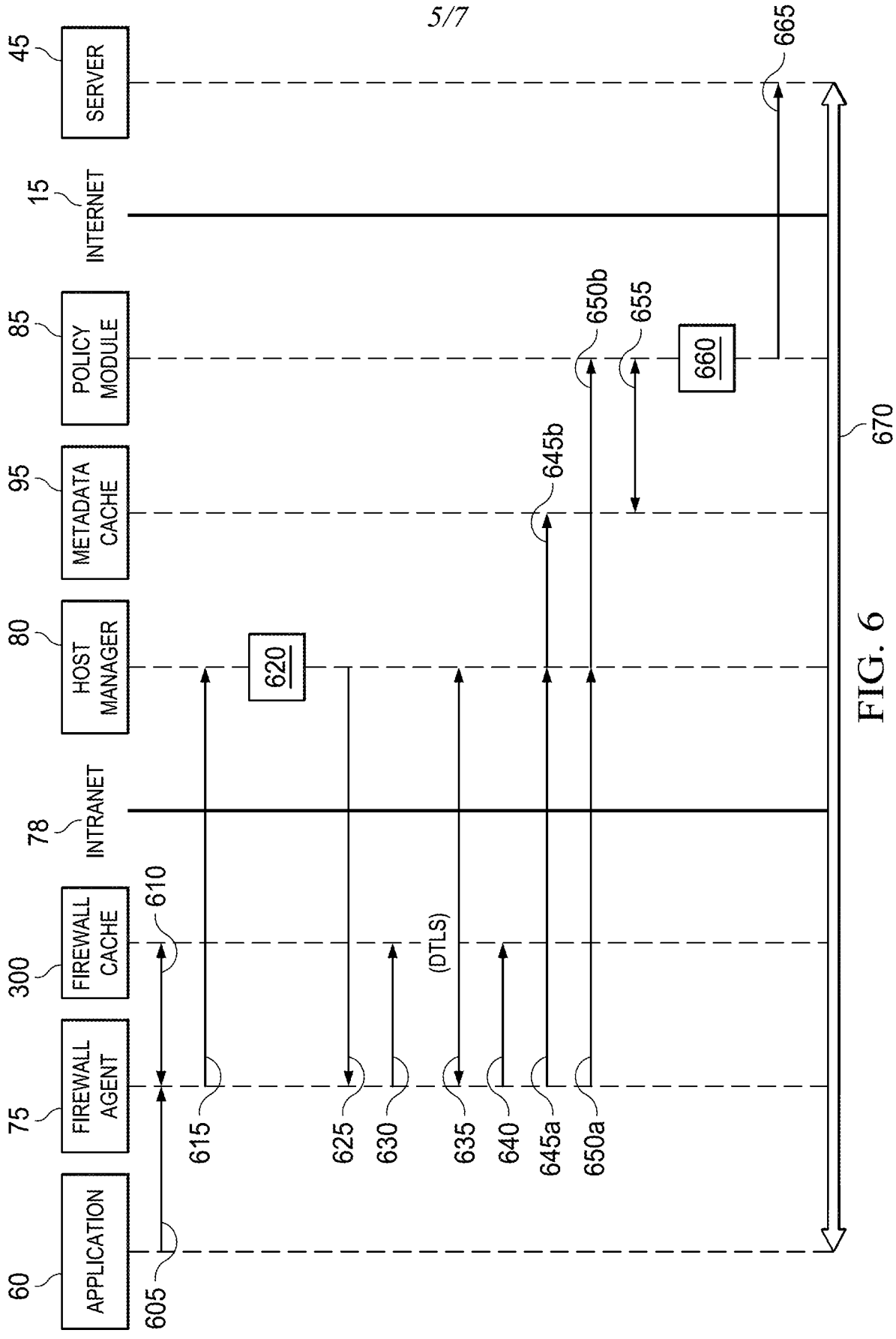
60
APPLICATION
75
FIREWALL AGENT
300
FIREWALL CACHE
78
INTRANET
80
HOST MANAGER
95
METADATA CACHE
85
POLICY MODULE
15
INTERNET
45
SERVER
605
610
615
620
625
630
635
640
645a
645b
650a
650b
655
660
665
670
(DTLS)

FIG. 6

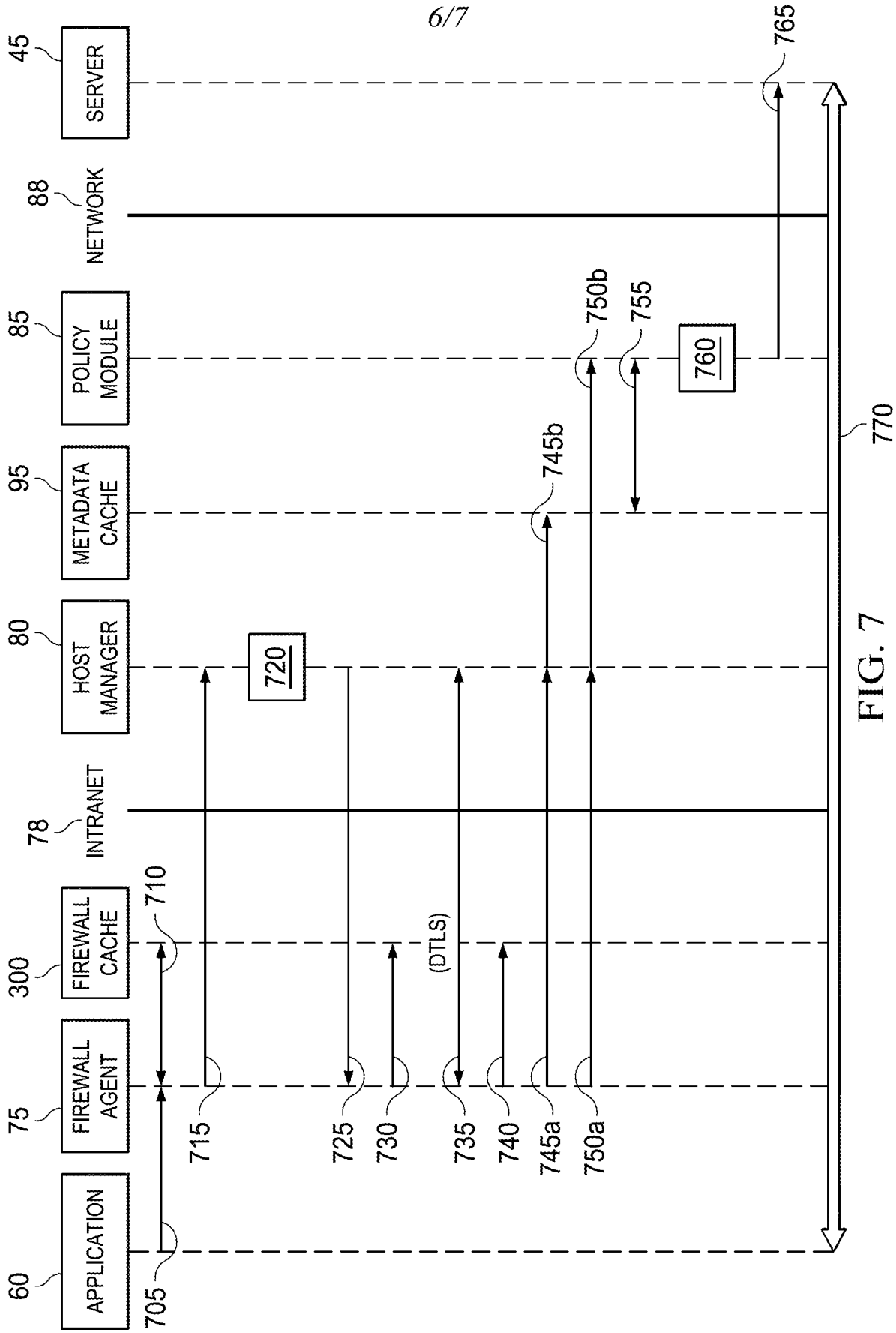
60
APPLICATION
75
FIREWALL AGENT
300
FIREWALL CACHE
78
INTRANET
80
HOST MANAGER
95
METADATA CACHE
85
POLICY MODULE
88
NETWORK
45
SERVER
705
710
715
720
725
730
735
740
745a
745b
750a
750b
755
760
765
770
(DTLS)

FIG. 7

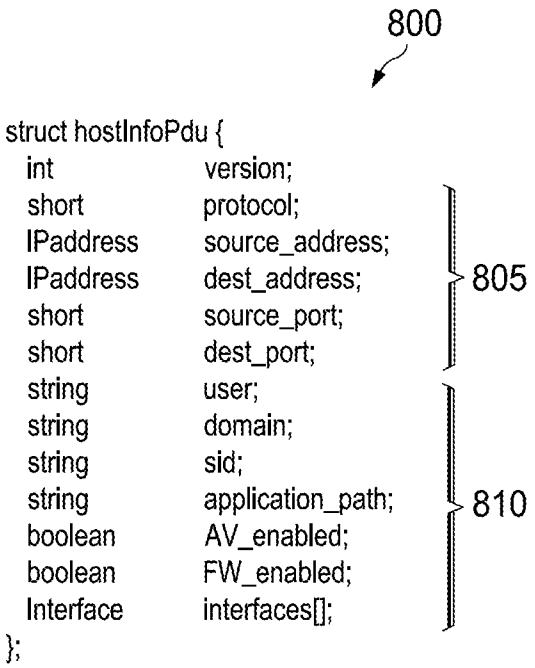
800
struct hostInfoPdu {
int version;
short protocol;
IPaddress source_address;
IPaddress dest_address;
short source_port;
short dest_port;
805
string user;
string domain;
string sid;
string application_path;
boolean AV_enabled;
boolean FW_enabled;
Interface interfaces[];
810
};

FIG. 8

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2012/057153

A. CLASSIFICATION OF SUBJECT MATTER

H04L 12/56(2006.01)i, H04L 9/32(2006.01)i, H04L 9/30(2006.01)i, H04L 29/06(2006.01)i

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

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
H04L 12/56; G06F 11/30; G06F 15/173; G06F 21/20; H04L 9/32; G06F 15/177

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
Korean utility models and applications for utility models
Japanese utility models and applications for utility models

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
eKOMPASS(KIPO internal) & Keywords: firewall cache, network flow, source node, destination node

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2011-0302647 A1 (BHATTACHARYA KAMAL et al.) 08 December 2011 See astract; paragraphs [31]-[52]; claims 1-10 and figures 2, 3.	1-24
A	US 7054930 B1 (Cisco Technology, Inc.) 30 May 2006 See abstract; column 4, line 64-column 6, line 53, claim 1 and figures 1, 3.	1-24
A	US 2003-0065945 A1 (CHARLES STEVEN LINGAFELT et al.) 03 April 2003 See abstract; paragraphs [9]-[11] and claims 1-24.	1-24
A	Zhen Chen et al. 'Application Level Network Access Control System Based on TNC Architecture for Enterprise Network', In: Wireless Communications Networking and Information Security (WCNIS), 2010 IEEE International Conference, 25-27 JUNE 2010, pp. 667-671 See abstract and sections 1-3.	1-24

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

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents,such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
24 DECEMBER 2012 (24.12.2012)	**26 DECEMBER 2012 (26.12.2012)**

Name and mailing address of the ISA/KR
Korean Intellectual Property Office
189 Cheongsa-ro, Seo-gu, Daejeon Metropolitan City, 302-701, Republic of Korea
Facsimile No. 82-42-472-7140

Authorized officer
HONG, Kyoung Ah
Telephone No. 82-42-481-5668

Form PCT/ISA/210 (second sheet) (July 2009)

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/US2012/057153

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2011-0302647 A1	08.12.2011	KR 10-2011-0132973 A	09.12.2011
US 7054930 B1	30.05.2006	US 7779126 B1	17.08.2010
US 2003-0065945 A1	03.04.2003	US 2007-0245421 A1	18.10.2007
		US 7278161 B2	02.10.2007
		US 7793348 B2	07.09.2010