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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
7 November 2013 (07.11.2013)

WIPO | PCT

(10) International Publication Number
WO 2013/165802 A1

(51) **International Patent Classification**:
H04L 12/24 (2006.01) ***H04W 28/02*** (2009.01)
H04L 12/801 (2013.01)

(21) **International Application Number**:
PCT/US2013/038168

(22) **International Filing Date**:
25 April 2013 (25.04.2013)

(25) **Filing Language**: English

(26) **Publication Language**: English

(30) **Priority Data**:
61/641,251 1 May 2012 (01.05.2012) US
13/802,254 13 March 2013 (13.03.2013) US

(71) **Applicant**: **F5 NETWORKS, INC.** [US/US]; 401 Elliott Ave. W., Seattle, WA 98119 (US).

(72) **Inventors**: **SZABO, Paul, Imre**; 401 Elliott Ave. W., Seattle, WA 98119 (US). **THORNEWELL, Peter, Michael**; 401 Elliott Ave. W., Seattle, WA 98119 (US). **MICHELS, Timothy, Scott**; 401 Elliott Ave. W., Seattle, WA 98119 (US).

(74) **Agent**: **BRANCH, John, W.**; Frommer Lawrence & Haug LLP, 745 5th Ave., New York, NY 10151 (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, 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,

[Continued on next page]

(54) **Title**: DATA FLOW SEGMENT OPTIMIZED FOR HOT FLOWS

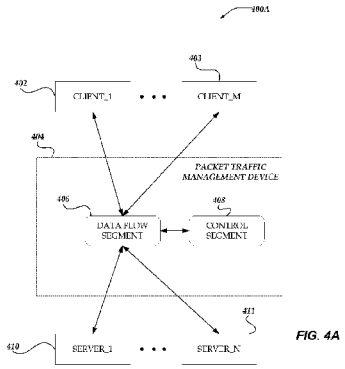


FIG. 4A

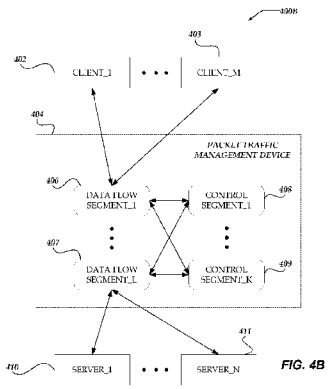


FIG. 4B

(57) **Abstract**: Embodiments are directed towards improving the performance of network traffic management devices by optimizing the management of hot connection flows. A packet traffic management device ("PTMD") may employ a data flow segment ("DFS") and control segment ("CS"). The CS may perform high-level control functions and per-flow policy enforcement for connection flows maintained at the DFS, while the DFS may perform statistics gathering, per-packet policy enforcement (e.g., packet address translations), or the like, on connection flows maintained at the DFS. The DFS may include high-speed flow caches and other high-speed components that may be comprised of high- performance computer memory. Making efficient use of the high speed flow cache capacity may be improved by maximizing the number of hot connection flows and minimizing the number of malicious and/or in-operative connections flows (e.g., non-genuine flows) that may have flow control data stored in the high-speed flow cache.

WO 2013/165802 A1

TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

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

DATA FLOW SEGMENT OPTIMIZED FOR HOT FLOWS

RELATED APPLICATIONS

This application claims priority to U.S. Patent Application Serial No. 13/802,254 filed on March 13, 2013, which claims the benefit of U.S. Provisional Patent Application Serial No. 61/641,251 filed on May 1, 2012, both of which are incorporated herein by reference.

TECHNICAL FIELD

The present invention relates generally to packet traffic management and, more particularly, but not exclusively to determining if network connection flow control data should be off-loaded to data flow segment stored in a high-speed cache.

BACKGROUND

The expanded use of the Internet has increased communication connections between client devices and server devices. Often, a client device establishes a network connection with a server device by using well-known protocols, such as Transmission Control Protocol/Internet Protocol ("TCP/IP"), User Datagram Protocol ("UDP"), and the like. This network connection may be identified by one characteristic or a combination of characteristics, such as a source port, a destination port, a source address, a destination address, a protocol, and the like. Typically, the source address, destination address, destination port, and protocol are relatively fixed for a network connection between a client device and a server device. Thus, the source port may be utilized to uniquely identify a connection between the client device and the server device. Additionally, the expansion of the Internet has led to improvements in packet traffic management. One such advancement is to split operations between a control segment and a data flow segment as described in more detail in U.S. Patent No. 7,343,413, filed March 21, 2001, and entitled "Method and System for Optimizing a Network by Independently Scaling Control Segments and Data Flow," which is hereby incorporated by reference in its entirety into this patent application. Thus, it is with respect to these considerations and others that the invention has been made.

BRIEF DESCRIPTION OF THE DRAWINGS

Non-limiting and non-exhaustive embodiments of the present invention are described with reference to the following drawings. In the drawings, like reference numerals refer to like parts throughout the various figures unless otherwise specified. For a better understanding of the present invention, reference will be made to the following Detailed Description, which is to be read in association with the accompanying drawings, wherein:

FIGURE 1 is a system diagram of an environment in which embodiments of the invention may be implemented;

FIGURE 2 shows an embodiment of a client device that may be included in a system such as that shown in FIGURE 1;

FIGURE 3 shows an embodiment of a network device that may be included in a system such as that shown in FIGURE 1;

FIGURE 4A and 4B illustrate overview system diagrams generally showing embodiments of a packet traffic management device disposed between client devices and server devices in accordance with the embodiments;

FIGURE 5 illustrates a sequence diagram generally showing one embodiment of a sequence for terminating a connection flow at a data flow segment and establishing a new connection flow at the data flow segment in accordance with the embodiments;

FIGURE 6 shows a flowchart showing a process for packet traffic management in accordance with at least one of the various embodiments;

FIGURE 7 shows a flowchart of a process for handling new connection flows at a data flow segment in accordance with at least one of the various embodiments;

FIGURE 8 shows a flowchart of a process for handling eviction messages at a control segment in accordance with at least one of the various embodiments;

FIGURE 9 shows a flowchart of a process for determining if connection flows may be candidates for off-loading to the data flow segment in accordance with at least one of the various embodiments; and

FIGURES 10 and 11 show flowcharts of processes for identifying hot connection flows in accordance with at least one of the various embodiments.

DETAILED DESCRIPTION

Throughout the specification and claims, the following terms take the meanings explicitly associated herein, unless the context clearly dictates otherwise. The phrase "in one embodiment" as used herein does not necessarily refer to the same embodiment, though it may. Furthermore, the phrase "in another embodiment" as used herein does not necessarily refer to a different embodiment, although it may. Thus, as described below, various embodiments of the invention may be readily combined, without departing from the scope or spirit of the invention.

In addition, as used herein, the term "or" is an inclusive "or" operator, and is equivalent to the term "and/or," unless the context clearly dictates otherwise. The term "based on" is not exclusive and allows for being based on additional factors not described, unless the context clearly dictates otherwise. In addition, throughout the specification, the meaning of "a," "an," and "the" include plural references. The meaning of "in" includes "in" and "on."

As used herein, the term "SYN" refers to a packet transmitted utilizing TCP that includes a set synchronize control flag in a TCP header of the packet.

As used herein, the term "ACK" refers to a packet transmitted utilizing TCP that includes a set acknowledgment flag in a TCP header of the packet.

As used herein, the term "SYN_ACK" refers to a packet transmitted utilizing TCP that includes a set synchronize control flag and a set acknowledgment flag in a TCP header of the packet.

As used herein, the term "FIN" refers to a packet transmitted utilizing TCP that includes a set no more data from sender flag in a TCP header of the packet.

As used herein, the term "FIN_ACK" refers to a packet transmitted utilizing TCP that includes a set no more data from sender flag and a set acknowledgment flag in a TCP header of the packet. FIN_ACK compress a FIN and ACK into one TCP packet.

As used herein, the term "tuple" refers to a set of values that identify a source and destination of a connection. In one embodiment, a 5 tuple may include a source address, a destination address, a source port, a destination port, and a protocol identifier. In at least one of the various embodiments, tuples may be used to identify network flows (e.g., connection flows).

As used herein, the terms "network flow," "connection flow,", "flow" refer to a network session that may be established between two endpoints. In at least one of the various embodiments, a tuple may describe the flow. In at least one of the various embodiments, flow control data associated with connection flows may be used to ensure that the network packets sent between the endpoints of a connection flow may be routed along the same path. In at least one of the various embodiments, the performance of connection oriented network protocols such as TCP/IP may impaired if network packets may be routed using varying paths and/or directed different endpoints.

As used herein, the term "genuine connection flow," refers to a connection flow that may have been determined to be associated with an operative client-server communication session. In contrast, a non-genuine connection flow may be associated with a malicious attack such as a SYN flood attack. In at least one of the various embodiments, characteristics a genuine connection flows may include, TCP/IP handshaking complete, evidence of bi-directional network packet exchange, or the like. Likewise, evidence that a connection flow may be non-genuine may include, half-open connections (incomplete handshaking and connection setup), few if any network packets exchanged, or the like.

As used herein, the term "hot connection flow," refers to a connection flow that may have been determined to be a candidate for off loading to a data flow segment. Hot flow connections may have characteristics such high-bandwidth utilization, quality of service priority, or the like.

As used herein, the term "high speed flow cache" refers to memory based cache used for storing flow control data that corresponds to connection flows. The cache may be accessible using, dedicated busses that may provide very fast performance based on a combination of factors that may include, wide-busses, fast clock speeds, dedicated channels, specialized read and/or write buffer, hardware proximity, temperature control, or the like. Also, the high speed flow cache may be comprised of very fast random access memory (RAM) components such as, static random access memory (SRAM), asynchronous SRAM, burst SRAM, extended data output dynamic RAM (EDO DRAM), or the like. In most cases, the high performance components comprising the high speed flow cache often are relatively expensive. Thus, the high speed flow cache may comprise valuable "real estate" within a traffic management device.

The following briefly describes the various embodiments to provide a basic understanding of some aspects of the invention. This brief description is not intended as an extensive overview. It is not intended to identify key or critical elements, or to delineate or otherwise narrow the scope. Its purpose is merely to present some concepts in a simplified form as a prelude to the more detailed description that is presented later.

Briefly stated, embodiments are directed towards improving the performance of network traffic management devices by optimizing the management of hot connection flows. In at least one of the various embodiments, a packet traffic management device ("PTMD") may employ a data flow segment ("DFS") component and control segment ("CS") component. In at least one of the various embodiments, the CS may perform high-level control functions and per-flow policy enforcement for connection flows maintained at the DFS , while the DFS may perform statistics gathering, per-packet policy enforcement (e.g., packet address translations), or the like, on connection flows maintained at the DFS.

The CS may be utilized to generate flow control data for connection flows that may be offloaded to the DFS based on connection flow requests received at the packet traffic management device. In one embodiment, the CS may receive a new connection flow request, such as a SYN packet, sent by a client device. The CS may generate and cache a connection flow identifier for the connection flow request. In at least one of the various embodiments, the DFS may include high-speed flow caches and other high-speed components. In at least one of the various embodiments, the high-speed flow cache may be enabled to store a defined amount of flow control data that may limit the number of connection flows that may be offloaded to the DFS for handling. In at least one of the various embodiments, making efficient use of the high speed flow cache capacity may be improved by maximizing the number of hot connection flows and minimizing the number of malicious and/or in-operative connections flows (e.g., non-genuine flows) that may be have flow control data stored in the high-speed flow cache.

In at least one of the various embodiments, if a new network connection flow may be received it may be forwarded to a control segment (CS). In at least one of the various embodiments, the CS may generate the flow control data for the new network connection flow. In one embodiment, if the CS determines that the new network connection flow should be offloaded to the DFS, the CS may send a control message that may include the

flow control data to the DFS. In at least one of the various embodiments, the DFS may store the received flow control data in the high-speed flow cache that may correspond to the DFS.

In at least one of the various embodiments, the CS may receive connection flows that may be evicted from the DFS. In at least one of the various embodiments, if the evicted connection may remain valid and/or active the CS may begin handling the network packets for the transferred connection flow (e.g., the CS may take over the packet level control in addition to providing the flow level control and policy enforcement).

In at least one of the various embodiments, in conjunction with managing the connection flows the CS may analyze flow statistics and application to identify hot connection flows. In at least one of the various embodiments, if hot connection flows may be identified, the CS may determine if any should be handled by the DFS for improved performance.

In at least one of the various embodiments, offloading a connection flow to the DFS for handling enables the DFS to manage packet translation using flow control data that may have generated by the CS. In at least one of the various embodiments, connection flows offloaded to the DFS may benefit from performance improvements that arising from the high-performance hardware that may comprise the DFS. In at least one of the various embodiments, storing the flow control data for connection flows in the high-speed flow cache that may correspond to the DFS may occur if the connection flow may be offloaded to the DFS for handling.

Illustrative Operating Environment

FIGURE 1 shows components of one embodiment of an environment in which the invention may be practiced. Not all of the components may be required to practice the invention, and variations in the arrangement and type of the components may be made without departing from the spirit or scope of the invention.

As shown, system 100 of FIGURE 1 includes local area networks ("LANs")/ wide area networks ("WANs") – (network) 108, wireless network 107, client devices 102-105, packet traffic management device ("PTMD") 109, and server devices 110-111. Network 108 is in communication with and enables communication between client devices 102-105, wireless network 107, and PTMD 109. Carrier network 107 further enables communication

with wireless devices, such as client devices 103-105. PTMD 109 is in communication with network 108 and server devices 110-111.

One embodiment of client devices 102-105 is described in more detail below in conjunction with FIGURE 2. In one embodiment, at least some of client devices 102-105 may operate over a wired and/or a wireless network, such as networks 107 and/or 108. Generally, client devices 102-105 may include virtually any computing device capable of communicating over a network to send and receive information, including instant messages, performing various online activities, or the like. It should be recognized that more or less client devices may be included within a system such as described herein, and embodiments are therefore not constrained by the number or type of client devices employed.

Devices that may operate as client device 102 may include devices that typically connect using a wired or wireless communications medium, such as personal computers, servers, multiprocessor systems, microprocessor-based or programmable consumer electronics, network PCs, or the like. In some embodiments, client devices 102-105 may include virtually any portable computing device capable of connecting to another computing device and receiving information, such as laptop computer 103, smart phone 104, tablet computer 105, or the like. However, portable computer devices are not so limited and may also include other portable devices, such as cellular telephones, display pagers, radio frequency ("RF") devices, infrared ("IR") devices, Personal Digital Assistants ("PDAs"), handheld computers, wearable computers, integrated devices combining one or more of the preceding devices, and the like. As such, client devices 102-105 typically range widely in terms of capabilities and features. Moreover, client devices 102-105 may provide access to various computing applications, including a browser, or other web-based applications.

A web-enabled client device may include a browser application that is configured to receive and to send web pages, web-based messages, and the like. The browser application may be configured to receive and display graphics, text, multimedia, and the like, employing virtually any web-based language, including a wireless application protocol messages ("WAP"), and the like. In one embodiment, the browser application is enabled to employ Handheld Device Markup Language ("HDML"), Wireless Markup Language ("WML"), WMLScript, JavaScript, Standard Generalized Markup Language ("SGML"), HyperText Markup Language ("HTML"), eXtensible Markup Language ("XML"), and the like, to display and send a message. In one embodiment, a user of the client device may

employ the browser application to perform various activities over a network (online). However, another application may also be used to perform various online activities.

Client devices 102-105 also may include at least one other client application that is configured to receive and/or send data between another computing device. The client application may include a capability to send and/or receive content, or the like. The client application may further provide information that identifies itself, including a type, capability, name, or the like. In one embodiment, client devices 102-105 may uniquely identify themselves through any of a variety of mechanisms, including a phone number, Mobile Identification Number ("MIN"), an electronic serial number ("ESN"), or other mobile device identifier. The information may also indicate a content format that the mobile device is enabled to employ. Such information may be provided in a network packet, or the like, sent between other client devices, PTMD 109, server devices 110-111, or other computing devices.

Client devices 102-105 may further be configured to include a client application that enables an end-user to log into an end-user account that may be managed by another computing device, such as server devices 110-111, or the like. Such end-user accounts, in one non-limiting example, may be configured to enable the end-user to manage one or more online activities, including in one non-limiting example, search activities, social networking activities, browse various websites, communicate with other users, participate in gaming, interact with various applications, or the like. However, participation in online activities may also be performed without logging into the end-user account.

Wireless carrier network 107 is configured to couple client devices 103-105 and its components with network 108. Wireless carrier network 107 may include any of a variety of wireless sub-networks that may further overlay stand-alone ad-hoc networks, and the like, to provide an infrastructure-oriented connection for client devices 102-105. Such sub-networks may include mesh networks, Wireless LAN ("WLAN") networks, cellular networks, and the like. In one embodiment, the system may include more than one wireless network.

Wireless carrier network 107 may further include an autonomous system of terminals, gateways, routers, and the like connected by wireless radio links, and the like. These connectors may be configured to move freely and randomly and organize themselves arbitrarily, such that the topology of wireless carrier network 107 may change rapidly.

Wireless carrier network 107 may further employ a plurality of access technologies including 2nd (2G), 3rd (3G), 4th (4G) 5th (5G) generation radio access for cellular systems, WLAN, Wireless Router ("WR") mesh, and the like. Access technologies such as 2G, 3G, 4G, 5G, and future access networks may enable wide area coverage for mobile devices, such as client devices 103-105 with various degrees of mobility. In one non-limiting example, carrier network 107 may enable a radio connection through a radio network access such as Global System for Mobil communication ("GSM"), General Packet Radio Services ("GPRS"), Enhanced Data GSM Environment ("EDGE"), code division multiple access ("CDMA"), time division multiple access ("TDMA"), Wideband Code Division Multiple Access ("WCDMA"), High Speed Downlink Packet Access ("HSDPA"), Long Term Evolution ("LTE"), and the like. In essence, carrier network 107 may include virtually any wireless communication mechanism by which information may travel between client devices 103-105 and another computing device, network, and the like.

Network 108 is configured to couple network devices with other computing devices, including, server devices 110-111 through PTMD 109, client device 102, and client devices 103-105 through wireless carrier network 107. Network 108 is enabled to employ any form of computer readable media for communicating information from one electronic device to another. Also, network 108 can include the Internet in addition to LANs, WANs, direct connections, such as through a universal serial bus ("USB") port, other forms of computer readable media, or any combination thereof. On an interconnected set of LANs, including those based on differing architectures and protocols, a router acts as a link between LANs, enabling messages to be sent from one to another. In addition, communication links within LANs typically include twisted wire pair or coaxial cable, while communication links between networks may utilize analog telephone lines, full or fractional dedicated digital lines including T1, T2, T3, and T4, and/or other carrier mechanisms including, for example, E-carriers, Integrated Services Digital Networks ("ISDNs"), Digital Subscriber Lines ("DSLs"), wireless links including satellite links, or other communications links known to those skilled in the art. Moreover, communication links may further employ any of a variety of digital signaling technologies, including without limit, for example, DS-0, DS-1, DS-2, DS-3, DS-4, OC-3, OC-12, OC-48, or the like. Furthermore, remote computers and other related electronic devices could be remotely connected to either LANs or WANs via a modem and temporary telephone link. In one embodiment, network 108 may be configured

to transport information of an Internet Protocol ("IP"). In essence, network 108 includes any communication method by which information may travel between computing devices.

Additionally, communication media typically embodies computer readable instructions, data structures, program modules, or other transport mechanism and includes any information delivery media. By way of example, communication media includes wired media such as twisted pair, coaxial cable, fiber optics, wave guides, and other wired media and wireless media such as acoustic, RF, infrared, and other wireless media.

One embodiment of PTMD 109 is described in more detail below in conjunction with FIGURE 3. Briefly, however, PTMD 109 may include virtually any network device capable of managing network traffic between client devices 102-105 and server devices 110-111. Such devices include, for example, routers, proxies, firewalls, load balancers, cache devices, devices that perform network address translation, or the like, or any combination thereof. PTMD 109 may perform the operations of routing, translating, switching packets, or the like. In one embodiment, PTMD 109 may inspect incoming network packets, and may perform an address translation, port translation, a packet sequence translation, and the like, and route the network packets based, at least in part, on the packet inspection. In some embodiments, PTMD 109 may perform load balancing operations to determine a server device to direct a request. Such load balancing operations may be based on network traffic, network topology, capacity of a server, content requested, or a host of other traffic distribution mechanisms.

PTMD 109 may include a control segment and a separate data flow segment. The control segment may include software-optimized operations that perform high-level control functions and per-flow policy enforcement for packet traffic management. In at least one of the various embodiments, the control segment may be configured to manage connection flows maintained at the data flow segment. In one embodiments, the control segment may provide instructions, such as, for example, a packet translation instruction, to the data flow segment to enable the data flow segment to route received packets to a server device, such as server device 110-111. The data flow segment may include hardware-optimized operations that perform statistics gathering, per-packet policy enforcement (e.g., packet address translations), high-speed flow caches, or the like, on connection flows maintained at DFS between client devices, such as client devices 102-105, and server devices, such as server devices 110-111.

Server devices 110-111 may include virtually any network device that may operate as a website server. However, server devices 110-111 are not limited to website servers, and may also operate as messaging server, a File Transfer Protocol (FTP) server, a database server, content server, or the like. Additionally, each of server devices 110-111 may be configured to perform a different operation. Devices that may operate as server devices 110-111 include various network devices, including, but not limited to personal computers, desktop computers, multiprocessor systems, microprocessor-based or programmable consumer electronics, network PCs, server devices, network appliances, and the like.

Although FIGURE 1 illustrates server devices 110-111 as single computing devices, the invention is not so limited. For example, one or more functions of each of server devices 110-111 may be distributed across one or more distinct network devices. Moreover, server devices 110-111 are not limited to a particular configuration. Thus, in one embodiment, server devices 110-111 may contain a plurality of network devices that operate using a master/slave approach, where one of the plurality of network devices of server devices 110-111 operate to manage and/or otherwise coordinate operations of the other network devices. In other embodiments, the server devices 110-111 may operate as a plurality of network devices within a cluster architecture, a peer-to-peer architecture, and/or even within a cloud architecture. Thus, the invention is not to be construed as being limited to a single environment, and other configurations, and architectures are also envisaged.

Illustrative Client Device

FIGURE 2 shows one embodiment of client device 200 that may be included in a system implementing embodiments of the invention. Client device 200 may include many more or less components than those shown in FIGURE 2. However, the components shown are sufficient to disclose an illustrative embodiment for practicing the present invention. Client device 200 may represent, for example, one embodiment of at least one of client devices 102-105 of FIGURE 1.

As shown in the figure, client device 200 includes a processor 202 in communication with memory 226 via a bus 234. Client device 200 also includes a power supply 228, one or more network interfaces 236, an audio interface 238, a display 240, a keypad 242, and an input/output interface 248.

Power supply 228 provides power to client device 200. A rechargeable or non-rechargeable battery may be used to provide power. The power may also be provided by an external power source, such as an AC adapter or a powered docking cradle that supplements and/or recharges a battery.

Client device 200 may optionally communicate with a base station (not shown), or directly with another computing device. Network interface 236 includes circuitry for coupling client device 200 to one or more networks, and is constructed for use with one or more communication protocols and technologies including, but not limited to, global system for mobile communication ("GSM"), code division multiple access ("CDMA"), time division multiple access ("TDMA"), High Speed Downlink Packet Access ("HSDPA"), Long Term Evolution ("LTE"), user datagram protocol ("UDP"), transmission control protocol/Internet protocol ("TCP/IP"), short message service ("SMS"), general packet radio service ("GPRS"), WAP, ultra wide band ("UWB"), IEEE 802.16 Worldwide Interoperability for Microwave Access ("WiMax"), session initiated protocol/real-time transport protocol ("SIP/RTP"), or any of a variety of other wireless communication protocols. Network interface 236 is sometimes known as a transceiver, transceiving device, or network interface card ("NIC").

Audio interface 238 is arranged to produce and receive audio signals such as the sound of a human voice. For example, audio interface 238 may be coupled to a speaker and microphone (not shown) to enable telecommunication with others and/or generate an audio acknowledgement for some action.

Display 240 may be a liquid crystal display ("LCD"), gas plasma, light emitting diode ("LED"), or any other type of display used with a computing device. Display 240 may also include a touch sensitive screen arranged to receive input from an object such as a stylus or a digit from a human hand.

Keypad 242 may comprise any input device arranged to receive input from a user. For example, keypad 242 may include a push button numeric dial, or a keyboard. Keypad 242 may also include command buttons that are associated with selecting and sending images.

Client device 200 also comprises input/output interface 248 for communicating with external devices, such as a headset, or other input or output devices not shown in FIGURE

2. Input/output interface 248 can utilize one or more communication technologies, such as USB, infrared, Bluetooth™, or the like.

Client device 200 may also include a GPS transceiver (not shown) to determine the physical coordinates of client device 200 on the surface of the Earth. A GPS transceiver typically outputs a location as latitude and longitude values. However, the GPS transceiver can also employ other geo-positioning mechanisms, including, but not limited to, triangulation, assisted GPS ("AGPS"), Enhanced Observed Time Difference ("E-OTD"), Cell Identifier ("CI"), Service Area Identifier ("SAI"), Enhanced Timing Advance ("ETA"), Base Station Subsystem ("BSS"), or the like, to further determine the physical location of client device 200 on the surface of the Earth. It is understood that under different conditions, a GPS transceiver can determine a physical location within millimeters for client device 200; and in other cases, the determined physical location may be less precise, such as within a meter or significantly greater distances. In one embodiment, however, mobile device 200 may through other components, provide other information that may be employed to determine a physical location of the device, including for example, a Media Access Control ("MAC") address, IP address, or the like.

Memory 226 includes a Random Access Memory ("RAM") 204, a Read-only Memory ("ROM") 222, and other storage means. Mass memory 226 illustrates an example of computer readable storage media (devices) for storage of information such as computer readable instructions, data structures, program modules or other data. Mass memory 226 stores a basic input/output system ("BIOS") 224 for controlling low-level operation of client device 200. The mass memory also stores an operating system 206 for controlling the operation of client device 200. It will be appreciated that this component may include a general-purpose operating system such as a version of UNIX, or LINUX™, or a specialized client communication operating system such as Windows Mobile™, or the Symbian® operating system. The operating system may include, or interface with a Java virtual machine module that enables control of hardware components and/or operating system operations via Java application programs.

Mass memory 226 further includes one or more data storage 208, which can be utilized by client device 200 to store, among other things, applications 214 and/or other data. For example, data storage 208 may also be employed to store information that describes various capabilities of client device 200. The information may then be provided

to another device based on any of a variety of events, including being sent as part of a header during a communication, sent upon request, or the like. Data storage 208 may also be employed to store social networking information including address books, buddy lists, aliases, user profile information, or the like. Further, data storage 208 may also store message, we page content, or any of a variety of user generated content. At least a portion of the information may also be stored on another component of network device 200, including, but not limited to processor readable storage device 230, a disk drive or other computer readable storage medias (not shown) within client device 200.

Processor readable storage device 230 may include volatile, nonvolatile, removable, and non-removable media implemented in any method or technology for storage of information, such as computer- or processor-readable instructions, data structures, program modules, or other data. Examples of computer readable storage media include RAM, ROM, Electrically Erasable Programmable Read-only Memory ("EEPROM"), flash memory or other memory technology, Compact Disc Read-only Memory ("CD-ROM"), digital versatile disks ("DVD") or other optical storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices, or any other physical medium which can be used to store the desired information and which can be accessed by a computing device. Processor readable storage device 230 may also be referred to herein as computer readable storage media.

Applications 214 may include computer executable instructions which, when executed by client device 200, transmit, receive, and/or otherwise process network data. Network data may include, but is not limited to, messages (e.g., SMS, Multimedia Message Service ("MMS"), instant message ("IM"), email, and/or other messages), audio, video, and enable telecommunication with another user of another client device. Applications 214 may include, for example, browser 218. Applications 214 may include other applications, which may include, but are not limited to, calendars, search programs, email clients, IM applications, SMS applications, voice over Internet Protocol ("VOIP") applications, contact managers, task managers, transcoders, database programs, word processing programs, security applications, spreadsheet programs, games, search programs, and so forth.

Browser 218 may include virtually any application configured to receive and display graphics, text, multimedia, and the like, employing virtually any web based language. In one embodiment, the browser application is enabled to employ HDML, WML, WMLScript,

JavaScript, SGML, HTML, XML, and the like, to display and send a message. However, any of a variety of other web-based programming languages may be employed. In one embodiment, browser 218 may enable a user of client device 200 to communicate with another network device, such as PTMD 109 and/or with server devices 110-111.

Illustrative Network Device

FIGURE 3 shows one embodiment of a network device 300, according to one embodiment of the invention. Network device 300 may include many more or less components than those shown. The components shown, however, are sufficient to disclose an illustrative embodiment for practicing the invention. Network device 300 may be configured to operate as a server, client, peer, a host, or any other device. Network device 300 may represent, for example PTMD 109 of FIGURE 1, server devices 110-111 of FIGURE 1, and/or other network devices.

Network device 300 includes processor 302, processor readable storage device 328, network interface unit 330, an input/output interface 332, hard disk drive 334, video display adapter 336, data flow segment ("DFS") 338 and a mass memory, all in communication with each other via bus 326. The mass memory generally includes RAM 304, ROM 322 and one or more permanent mass storage devices, such as hard disk drive 334, tape drive, optical drive, and/or floppy disk drive. The mass memory stores operating system 306 for controlling the operation of network device 300. Any general-purpose operating system may be employed. Basic input/output system ("BIOS") 324 is also provided for controlling the low-level operation of network device 300. As illustrated in FIGURE 3, network device 300 also can communicate with the Internet, or some other communications network, via network interface unit 330, which is constructed for use with various communication protocols including the TCP/IP protocol. Network interface unit 330 is sometimes known as a transceiver, transceiving device, or network interface card ("NIC").

Network device 300 also comprises input/output interface 332 for communicating with external devices, such as a keyboard, or other input or output devices not shown in FIGURE 3. Input/output interface 332 can utilize one or more communication technologies, such as USB, infrared, Bluetooth™, or the like.

The mass memory as described above illustrates another type of computer readable media, namely computer readable storage media and/or processor readable storage media,

including processor readable storage device 328. Processor readable storage device 328 may include volatile, nonvolatile, removable, and non-removable media implemented in any method or technology for storage of information, such as computer readable instructions, data structures, program modules, or other data. Examples of processor readable storage media include RAM, ROM, EEPROM, flash memory or other memory technology, CD-ROM, digital versatile disks (DVD) or other optical storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices, or any other media which can be used to store the desired information and which can be accessed by a computing device.

Data storage 308 may include a database, text, spreadsheet, folder, file, or the like, that may be configured to maintain and store user account identifiers, user profiles, email addresses, IM addresses, and/or other network addresses; or the like. Data stores 308 may further include program code, data, algorithms, and the like, for use by a processor, such as central processing unit 302 to execute and perform actions. In one embodiment, at least some of data store 308 might also be stored on another component of network device 300, including, but not limited to processor-readable storage device 328, hard disk drive 334, or the like.

The mass memory may also stores program code and data. One or more applications 314 may be loaded into mass memory and run on operating system 306. Examples of application programs may include transcoders, schedulers, calendars, database programs, word processing programs, Hypertext Transfer Protocol ("HTTP") programs, customizable user interface programs, IPSec applications, encryption programs, security programs, SMS message servers, IM message servers, email servers, account managers, and so forth. Web server 316 and control segment ("CS") 318 may also be included as application programs within applications 314.

Web server 316 represent any of a variety of services that are configured to provide content, including messages, over a network to another computing device. Thus, web server 316 includes, for example, a web server, a File Transfer Protocol ("FTP") server, a database server, a content server, or the like. Web server 316 may provide the content including messages over the network using any of a variety of formats including, but not limited to WAP, HDML, WML, SGML, HTML, XML, Compact HTML ("cHTML"), Extensible HTML ("xHTML"), or the like. Web server 316 may also be configured to enable a user of

a client device, such as client devices 102-105 of FIGURE 1, to browse websites, upload user data, or the like.

Network device 300 may also include DFS 338 for maintaining connection flows between client devices, such as client devices 102-105 of FIGURE 1, and server devices, such as server devices 110-111 of FIGURE 1. In one embodiment, DFS 338 may include hardware-optimized operations for packet traffic management, such as repetitive operations associated with packet traffic management. For example, DFS 338 may perform statistics gathering, per-packet policy enforcement (e.g., packet address translations), or the like, on connection flows maintained at DFS 338. In some embodiments, DFS 338 may route, switch, forward, direct and/or otherwise handle packets based on rules for a particular connection flow signature (e.g., a 5 tuple of a received packet). Thus, DFS 338 may include capabilities and perform tasks such as that of a router, a switch, a routing switch, or the like. In some embodiments, the rules for a particular connection flow signature may be based on instructions received from CS 318. In one embodiment, DFS 338 may store the instructions received from CS 318 in a local memory as a table or some other data structure. In some other embodiments, DFS 338 may also store a flow state table to indicate a state of current connection flows maintained at DFS 338. In at least one of the various embodiments, components of DFS 338 may comprise and/or work in combination to provide high-speed flow caches for optimizing packet traffic management.

In some embodiments, DFS 338 may provide connection flow status updates to CS 318. In one embodiment, a connection flow status update may include a status of the connection flow, a current state of the connection flow, other statistical information regarding the connection flow, or the like. The connection flow update may also include an identifier that corresponds to the connection flow. The identifier may be generated and provided by CS 318 when a connection flow is established at DFS 338. In some embodiments, the connection flow update may be a connection flow delete update provided to CS 318 after the connection flow is terminated at DFS 338. The connection flow status update and/or the connection flow delete update may be provided to CS 318 periodically, at predefined time intervals, or the like. In some embodiments, DFS 338 may stagger a time when a plurality of connection flow status updates are provided to CS.

In some other embodiments, DFS 338 may include a plurality of data flow segments. In one non-limiting example, a first data flow segment within DFS 338 may forward packets

received from a client device to a server device, while a second data flow segment within DFS 338 may forward and/or route packets received from a server device to a client device. In at least one of the various embodiments, DFS 338 may also be implemented in software.

CS 318 may include a control segment that may include software-optimized operations to perform high-level control functions and per-flow policy enforcement for packet traffic management. CS 318 may be configured to manage connection flows maintained at DFS 338. In one embodiments, CS 318 may provide instructions, such as, for example, a packet address translation instructions, to DFS 338 to enable DFS 338 to forward received packets to a server device, such as server device 110-111 of FIGURE 1. In some other embodiments, CS 318 may forward and/or route packets between a client device and a server device independent of DFS 338.

In at least one of the various embodiments, CS 318 may include a plurality of control segments. In some embodiments, a plurality of control segments may access and/or manage connection flows at a single data flow segments and/or a plurality of data flow segments. In some other embodiments, CS 318 may include an internal data flow segment. In one such embodiment, the internal data flow segment of CS 318 may be distributed and/or separate from CS 318. For example, in one embodiment, CS 318 may be employed in software, while the internal data flow segment may be employed in hardware. In some other embodiments, CS 318 may identify if connection flows are split between different data flow segments and/or between a DFS 338 and CS 318. In at least one embodiment, CS 318 may also be implemented in hardware.

In at least one of the various embodiments, CS 318 may be configured to generate an identifier for each connection flow established at DFS 338. In some embodiments, CS 318 may utilize a sequence number of a SYN to generate an identifier for a corresponding connection flow. In one embodiment, the identifier may be based on a hash of the sequence number. In another embodiment, the identifier may be based on an exclusive OR byte operation of the sequence number. CS 318 may cache the identifier at CS 318 and may provide the identifier to DFS 338. In some embodiments, CS 318 may cache an identifier for each connection flow it establishes at DFS 338.

FIGURE 4A illustrates a system diagram generally showing one embodiment of a system with a packet traffic management device disposed between client devices and server devices. System 400A may include packet traffic management device ("PTMD") 404

disposed between client devices 402-403 and server devices 410-411. Client devices 402-403 may include Client_1 through Client_M, which may include one or more client devices, such as client devices 200 of FIGURE 2. Server devices 410-411 may include Server_1 through Server_N, which may include one or more server devices, such as server devices 110-111 of FIGURE 1.

In one embodiment, PTMD 404 may be an embodiment of PTMD 109 of FIGURE 1. PTMD 404 may include data flow segment ("DFS") 406 in communication with control segment ("CS") 408. In at least one of the various embodiments, DFS 406 may be an embodiment of DFS 338 of FIGURE 3, and CS 408 may be an embodiment of CS 318 of FIGURE 3.

CS 408 may be configured to communicate with DFS 406, client devices 402-403 and/or server devices 410-411 independent of DFS 406, and/or any combination thereof. CS 408 may establish connection flows at DFS 406. In some embodiments, CS 408 may establish a connection flow at DFS 406 by providing instructions including flow control data to DFS 406 that enables DFS 406 to forward packets received at PTMD 404. In one embodiment, CS 408 may perform a load balancing operation to select a server device of server devices 410-411 to receive packets sent from a client device, such as client device 402. In some other embodiments, CS 408 may generate and cache a connection flow identifier to be provided to DFS 406 when the connection flow is established.

DFS 406 may be configured to facilitate communications between client devices 402-403 and server devices 410-411. DFS 406 may process and forward packets received at PTMD 404 based on the instructions and flow control data received from CS 408. For example, in one embodiment, DFS 406 utilizes the instructions and/or flow control data to forward packets received from client device 402 to server device 410 and to forward packets received from server device 410 to client device 402. In some embodiments, DFS 406 may forward predetermined packets to CS 408, such as, but not limited to, new connection flow requests (e.g., associated with a SYN). In yet other embodiments, DFS 406 may notify CS 408 that a packet was received and forwarded. In one non-limiting, non-exhaustive example, DFS 406 may notify CS 408 that an ACK was received from client device 402 and forwarded to server device 410. In at least one of the various embodiments, DFS 406 may also provide connection flow updates and a corresponding connection flow identifier to CS

408. CS 408 may compare the corresponding connection flow identifier with the cached identifier to determine if the connection flow update is valid.

In at least one of the various embodiments, DFS 406 may send evict messages to CS 408 if connection flow are evicted from the DFS 406. In at least one of the various embodiments, DFS 406 may evict a connection flow if new flows arrive and the capacity of the DFS to handle new connection flow may be exceeded. In at least one of the various embodiments, evictions from DFS 406 may occur if the high speed flow cache for storing flow control data exhausts its ability to store the flow control data for new connection flows. In at least one of the various embodiments, evict messages sent from DFS 406 to CS 408 may contain enough information to fully identify the connection flow (e.g., endpoints, ports, sequent numbers, flow state, or the like).

In at least one of the various embodiments, CS 408 may receive and route packets associated with evicted connection flows, thereby taking on some of the duties of DFS 406. In at least one of the various embodiments, some new connection flow may not be offloads to DFS 406 if CS 408 determines that the connection flows may be management on the CS or if the CS determines that more information may be required to determine if the connection flow should be offloaded to DFS 406.

Although PTMD 404 illustrates DFS 406 and CS 408 as two partitions within a single PTMD 404, the invention is not so limited. Rather, in some embodiments, DFS 406 and CS 408 may be functional blocks in a same PTMD 404 (i.e., a same chassis/computing device). In other embodiments, DFS 406 may be implemented by one or more chassis/computing devices separate from one or more other chassis/computing devices that may be utilized to implement CS 408. In yet other embodiments, CS 408 may be a module that plugs into DFS 406. Additionally, it is envisaged that the functionality of either DFS 406 and/or CS 408 may be separately implemented in software and/or hardware.

FIGURE 4B illustrates a system diagram generally showing one embodiment of a system with a packet traffic management device disposed between client devices and server devices. System 400B may include packet traffic management device ("PTMD") 404 disposed between client devices 402-403 and server devices 410-411. Client devices 402-403 may include Client_1 through Client_M, which may include one or more client devices, such as client devices 102-105 of FIGURE 1. Server devices 410-411 may include

Server_1 through Server_N, which may include one or more server devices, such as server devices 110-111 of FIGURE 1.

In one embodiment, PTMD 404 may be an embodiment of PTMD 404 of FIGURE 4. PTMD 404 may include data flow segments ("DFS") 406-407 and control segments ("CS") 408-409. DFS 406-407 may include a plurality of data flow segments, each of which may be an embodiment of DFS 406 of FIGURE 4A. CS 408-409 may include a plurality of control flow segments, each of which may be an embodiment of CS 408 of FIGURE 4.

In some embodiments, data communicated between client devices 402-403 and server devices 410-411 may flow through one or more data flow segments 406-407. In one embodiment, data from client devices 402-403 may flow through a first DFS, such as DFS 406 and data from server devices 410-411 may flow through a second DFS, such as DFS 407.

In at least one of the various embodiments, one or more data flow segments of DFS 406-407 may communicate with one or more control segments of CS 408-409. Similarly, one or more control segments of CS 408-409 may communicate with one or more data flow segments of DFS 406-407. In some embodiments, each control segment of CS 408-409 may communicate (not shown) with other control segments of CS 408-409. In other embodiments, each data flow segment of DFS 406-407 may communicate (not shown) with other data flow segments of DFS 406-407.

Also, in at least one of the various embodiments, connection flows may be split into flow portions based on the direction of network packet travel. In at least one of the various embodiments, the network packets coming from the client may treated as a separate connection flow and the network packets coming from a server and directed towards a client may be treated as a separate connection flow. In at least one of the various embodiments, this enables optimizations based on the amount of network packet traffic of a particular split connection flows. In at least one of the various embodiments, this may enable the upload and download direction portion of connection flows to be split across CS 408-409 and DFS 406-407 based on the characteristics of the upload and download portions of the connection flows. For example, in at least one of the various embodiments, if downloading streaming video may be a very asymmetric operation having many network packets download to the client and few uploaded. In at least one of the various embodiments, the upload and

download portions of connection flow in the download direction may be optimized independent with one portion using the DFS and a high-speed flow cache and the other portion may be handled on the CS using lower performing (e.g., less expensive) resources.

FIGURE 5 illustrates an embodiment of a sequence for establishing a connection flow and offloading the new connection flow to the data flow segment (DFS). Sequence 500 may show an embodiment using TCP/IP networking protocol but one of ordinary skill the art will appreciate that the sequence diagram (or similar sequences) may generally apply to other networking protocols that may have other handshaking sequences as well. Also, even though sequence 500 depicts a sequence including one client, one DFS, one CS, and one application server, in at least one of the various embodiments, one or more, data flow segments, control segments, clients, and servers, may be participate in handshaking and in the connection flow offloading. Also, in at least one of the various embodiments, the connection flows may be split into upload and download portions of a connection flow, with each portion representing one direction of the connection flow.

In at least one of the various embodiments, sequence 500 begins at step 502 if a client initiates a connection with a network resource that may be managed by a PTMD, such as PTMD 109. If client may be initiating the connection using TCP/IP, a SYN packet may be sent to the PTMD.

At step 504 a SYN packet may be received at a DFS that may be part of a PTMD. In at least one of the various embodiments, at step 506, because the DFS may determine that the incoming connection represents a new connection flow, the DFS may forward the SYN packet to a CS. At step 506 a CS may examine the connection flow and may determine the appropriate flow control data for the new flow and send it to the DFS. In at least one of the various embodiments, CS may apply one or more stored rules that may be used to determine the flow control data for the new network connection flow. In at least one of the various embodiments, the stored rules may implement network traffic management services such as load balancing, application access control, or the like.

In at least one of the various embodiments, at step 508 the DFS may receive the flow control data from the CS and store it in a high speed flow cache. In at least one of the various embodiments, the flow control data may be used by the DFS to forward the SYN packet to an appropriate server and/or network resource as directed by the flow control data that may be provided by the CS.

In at least one of the various embodiments, at step 510 a server and/or network resource may receive the SYN packet and may respond by sending a SYN-ACK packet to the DFS. In at least one of the various embodiments, at step 512 the DFS may again use the flow control data stored in the high speed flow cache to map and/or translate the SYN_ACK from a server to the appropriate client.

In at least one of the various embodiments, at step 514 the client device that sent the initial SYN packet may receive the corresponding SYN_ACK and subsequently may respond with an ACK packet. In at least one of the various embodiments, at step 516 the DFS, using the stored flow control data to determine the network path the to server, may forward the ACK packet to the server.

In at least one of the various embodiments, at step 518 the server may receive the ACK packet corresponding to the client device. After the ACK may have been received, the network connection flow may be in an established state. In at least one of the various embodiments, during steps 520-524, using the established network connection flow, the server may begin exchanging application data with client. In at least one of the various embodiments, at this point, for each exchange of data, the DFS may use the flow control data that may be stored in the high speed flow cache to map between the application servers and the client to route the packets on the correct path to maintain the connection flow.

General Operation

FIGURE 6 shows a flowchart showing at least one of the various embodiments of a process for packet traffic management. In process 600, after a start block, at block 602 a network packet may be received by a DFS. In at least one of the various embodiments, the network packets may be received from network 108, and/or may have been forwarded through multiple networks, switches, routers, other PTMDs or the like.

At decision block 604, in at least one of the various embodiments, if the received network packet may be associated with a new connection flow, control may move to block 606. Otherwise, in at least one of the various embodiments, control may move to decision block 608.

In at least one of the various embodiments, a DFS may examine the connection flow and compare it the flow control data that may be stored in a high-speed cache. In at least one of the various embodiments, a tuple corresponding to the network packet may be examined

to determine if the network packet is part of a new connection flow. If a tuple corresponding to the incoming network packet may not be found in the high-speed flow cache the DFS may determine that the network packet may be part of a new connection flow.

At block 606, in at least one of the various embodiments, the incoming network packet that may be associated with a new connection flow may be forwarded to a CS for further processing. In at least one of the various embodiments, the incoming network packet may be sent to a CS using a command bus that may enable DFS and CS components to exchange data and messages. Next, control may move decision block 614.

At decision block 608, in at least one of the various embodiments, if flow control data may be available for the connection flow associated with network packet, control may move to block 710. Otherwise, in at least one of the various embodiments, control may move to block 612.

At block 610, in at least one of the various embodiments, the DFS may forward the network packet to its next destination based on the flow control data and/or information associated with the network packet's corresponding connection flow that may be stored in the high speed flow cache that corresponds to the DFS. Next, in at least one of the various embodiments, control may move to decision block 614.

At block 612, in at least one of the various embodiments, the network packet having a previously seen tuple may be stored in a buffer on the DFS until flow control data may be provided by the CS.

In at least one of the various embodiments, a received network packet may be associated with a connection flow that has been previously been observed. However, in at least one of the various embodiments, if the flow control data from the CS may not be available, the DFS may store the network packets associated with the connection flow in a buffer until the relevant flow control data may be received from the CS.

Also, in at least one of the various embodiments, incoming network packets associated with unknown and/or new connection flows may be forwarded to the CS for buffering, rather than buffering on the DFS, until a flow control data determination may be made by the CS.

At decision block 614, in at least one of the various embodiments, if there may be more incoming network packets, control may loop back to block 602. Otherwise, in at least one of the various embodiments, control may be returned to a calling process.

FIGURE 7 shows a flowchart of process 700, in at least one of the various embodiments, for handling new connection flows at a DFS. After a start block, at block 702 a DFS component may receive new flow control data from a CS. In at least one of the various embodiments, if new flow control data may be received, the DFS may store the flow data into a high-speed flow cache.

In at least one of the various embodiments, the new flow control data may be sent to the DFS as part of a "new flow" control message sent from the CS to the DFS.

At decision block 704, in at least one of the various embodiments, if the DFS high-speed flow cache may be full, control may move to block 706. Otherwise, in at least one of the various embodiments, control may move block 708.

In at least one of the various embodiments, the high-speed flow cache may be implemented as a hash such that the a hash key may be generated for each new connection flow based on properties of the connection flow such as the tuple, CS generated connection identifier, SYN cookie, or the like. In at least one of the various embodiments, if the range (number of unique values) of the hash key may be more than the number of slots in the high speed flow cache, the hash key may be truncated so the number of hash key value possibilities may be equal or similar to the number of slots in the high-speed flow cache. In at least one of the various embodiments, truncation of the hash key may increase the number of hash key collisions. If, in at least one of the various embodiments, a new connection flow hash key may cause hash key collision, the connection flow currently in the cache may get evicted (e.g., its flow control data is removed from the high speed cache and the responsibility for managing the flow may be transferred to the CS) to make room for the new connection flow.

At block 706, in at least one of the various embodiments, to make room for the new flow control data received from the CS, flow control data for a different, previously cached connection flow may be removed (e.g., evicted) from the DFS high-speed flow cache. In at least one of the various embodiments, the DFS may send the CS a control message indicating that a connection flow may have been evicted from the DFS requiring the

associated flow control data to be removed from the DFS high-speed flow cache. In at least one of the various embodiments, the eviction message may include information, such as, number of packets sent or received over this network flow, age of the network flow, tuple information, or the like. In at least one of the various embodiments, the control message sent to the CS may contain enough information to enable the CS to identify the network flow that may be evicted from the DFS.

At block 708, in at least one of the various embodiments, the flow control data associated with the new connection flow may be stored in the DFS high-speed flow cache. In at least one of the various embodiments, flow control data may be stored in one or more components of the DFS that may operate singly or in combination as a high-speed flow cache.

At block 710, in at least one of the various embodiments, the DFS may begin processing received network packets associated with known connection flows using the flow control data that may be associated with the connection flow and stored in the high-speed flow cache.

FIGURE 8 shows a flowchart of process 800, in at least one of the various embodiments, for handling eviction (EVICT) messages at a CS. After a start block, at block 802, the CS may receive an EVICT message from a DFS.

At decision block 804, in at least one of the various embodiments, if the eviction message corresponds to a closed and/or terminated connection flow control may move to block 806. Otherwise, in at least one of the various embodiments, control may move to block 808.

At block 806, in at least one of the various embodiments, the closed and/or terminated flow and associated flow control data may be discarded.

At block 808, in at least one of the various embodiments, the responsibility for managing the evicted connection flow may be transferred to the CS. In at least one of the various embodiments, network packets received over the transferred connection flow may be handled by the CS.

In at least one of the various embodiments, the CS may store the flow control data for the evicted connection flow in a local flow cache. In at least one of the various embodiments, the flow cache in the CS may be arranged to include at least the same

information that may be stored regarding connection flows using the high speed flow cache on the DFS.

At decision block 810, in at least one of the various embodiments, if there may be more eviction messages to process, control may loop back to block 802. Otherwise, in at least one of the various embodiments, control may be returned to a calling process.

In at least one of the various embodiments, depending on the circumstances, a connection flow may be handled on one or more DFSs, on one or more CSs, or partially on one or more CSs and partially on one or more DFSs. In at least one of the various embodiments, if a connection flow may be being handled by the CS it may not receive a new flow network message from the DFS. Likewise, if a DFS may be handling a connection flow it may not send a new flow network message to the CS component if the DFS can associate the incoming network traffic with a known connection flow. However, in at least one of the various embodiments, the CS may analyze each connection flow to determine the connection flows may be evicted from the DFS.

FIGURE 9 shows a flowchart for process 900 that in at least one of the various embodiments determines if connection flows may be candidates for off-loading to the DFS for handling. After a start block, at block 902, in at least one of the various embodiments, the CS may receive a network packet associated with a connection flow that may be managed by the CS.

In at least one of the various embodiments, network packets received by the CS may be associated connection flows that may have their packet level processing and management processing handled on the CS rather the DFS. In at least one of the various embodiments, as the CS handles the received packets at least in accordance with the stored flow control data it may perform additional action to identify hot connection flows.

At block 904, in at least one of the various embodiments, the CS may receive a flow status update (FSU) from a DFS. In at least one of the various embodiments, the FSU may be received asynchronously with respect to the network packets that may be received by the CS. In at least one of the various embodiments, if a FSU may be not be available control may move to block 906.

At block 906, in at least one of the various embodiments, the CS may update the statistics being maintained for the connection flows. In at least one of the various

embodiments, statistics may be tracked for the connection flows being managed by the CS directly as well as the connection flows that may be managed by the DFS (e.g., off-loaded connection flows).

In at least one of the various embodiments, the updating of connection flow metrics may use a combination of information from one or more FSUs and metrics that may be collected on the CS, including, bit-rate, data sent over a time interval, data received over a time interval, or the like. In at least one of the various embodiments, the connection flow metrics collected may be based, low level network information derived from L1-L4 as well as higher level network information derived from L5-L7 (as per the Open Systems Interconnection (OSI) model).

At block 908, in at least one of the various embodiments, the CS may analyze the collected connection flow statistics and may apply relevant rules to identify hot connection flows. (See, FIGURES 10 and 11.) In at least one of the various embodiments, the CS may employ at least one connection flow metric to determine each hot connection flow out of the plurality of managed connection flows

In at least one of the various embodiments, rules may be defined that declare that one or more specific sources, endpoints, data types, or the like may indicated to be hot connection flows. Or, in at least one of the various embodiments, rules may be defined to adjust the priority of certain connection flows based on flow patterns, sources, endpoints, data types, or like.

In at least one of the various embodiments, generally the same type of flow control policies rulemaking may be extended to influence the identification and determination of how connection flows may be designated as hot connection flows.

At decision block 910, if connection flows may be identified for moving from the CS to the DFS and/or from the DFS to the CS for handling, control may move block 912. Otherwise, in at least one of the various embodiments, control may move to decision block 914.

In at least one of the various embodiments, the CS may employ at least one connection flow metric to determine each hot connection flow in the plurality of managed connection flows.

At block 912, in at least one of the various embodiments, if connection flows may be identified for moving, the CS may generate the relevant commands and/or messages to and send to the appropriate CS and/or DFS for handling. In at least one of the various embodiments, some connection flows may be moved from a DFS to the CS for handling. In at least one of the various embodiments, the DFS may be employed to handle each determined hot connection flow.

Also, in at least one of the various embodiments, some of the connection flows that may have been identified as hot connection flows may be moved and/or off-loaded to a DFS for handling to benefit from at least higher performance/processing speeds the may be associated with a DFS. Next, in at least one of the various embodiments, control may move to decision block 914.

At decision block 914, in at least one of the various embodiments, if there may be more network packets available control may loop back to block 902. Otherwise, in at least one of the various embodiments, control may be returned to a calling process.

FIGURES 10 and 11 describe various embodiments for identifying if a connection flow may be a hot connection flow. One of ordinary skill in the art will appreciate that the techniques, parameters, and thresholds used to identify "hot connection flows" may vary depending on the applications being managed by a PTMD and the goals and priorities of the operators and users of the PTMD at a particular time. In at least one of the various embodiments, generally, the criteria for identifying a hot connection flow may be defined based on the application and user goals and if connection flow properties meet the criteria, a connection flow may be deemed a hot connection flow.

In at least one of the various embodiments, as part of determining if a connection flow may be a candidate for offloading to the DFS for handling (e.g., hot connection flow), the content of received network packets may be examined. In at least one of the various embodiments, the network packets may be examined to identify data patterns and meta-data that may indicate that the connection flow may be a hot connection flow that may be a good candidate for offloading to the DFS component.

In at least one of the various embodiments, if examining the network packets, the CS may identify application level protocol data, messages, or meta-data for determining if the associated connection flow may be a hot connection flow. For example, if a CS may identify

that a connection flow may be using HTTP, the CS may examine HTTP headers such as, Content-Type, Content-Length, Cache-Control, or the like, as part of determining if a connection flow may be a hot connection flow.

In at least one of the various embodiments, if a network packet may be determined to be a first packet of a HTTP response, a content length value provided by the server sending the HTTP response may be available. In at least one of the various embodiments, the HTTP content length value may indicate the number of network packets that may be likely to be used to transmit the complete HTTP response from the server. For example, in at least one of the various embodiments, if the content length value may indicate that the response may use a single network packet, the associated connection flow may not be a candidate for offloading to the DFS because additional packets may not be expected for this response. On the other hand, in at least one of the various embodiments, if the content length value indicates that more network packets may be on the way for the same response, the connection flow may be determined to be a candidate for offloading to the DFS component. In at least one of the various embodiments, the content length value may correlate to the likelihood of offloading a connection flow to a DFS (e.g., an increase in the content length value leads to an increase in the chance of offloading the connection flow to the DFS).

In at least one of the various embodiments, in some cases, the operating characteristics of a connection flow may have significant variance. For example, in at least one of the various embodiments, the bit-rate for a connection may be prone to spikes if the content/communication may be uneven. Thus, in at least one of the various embodiments, a connection flow once determined to be a good offload candidate (leading to likely offloading to the DFS) may soon be determined to be a poor offload candidate (leading to likely removal from the DFS) depending on the immediate condition and/or characteristics of the underlying communication session.

In at least one of the various embodiments, a connection flow may repeatedly cycled back and forth from being handled on the DFS to being handled on CS, or back again. In at least one of the various embodiments, the cycling may occur based on at least the variance of the operating characteristics of the connection flow. In at least one of the various embodiments, this at least enables the performance of the connection flow and the usage of

the DFS to be continually optimized to take advantage of the variance in the connection flow operation.

For example, in at least one of the various embodiments, as the network traffic over the connection flow slows, the connection flow may be moved to the CS for handling. Likewise, in at least one of the various embodiments, as the network traffic over the connection flow increases the connection flow may be moved to the DFS for handling.

In at least one of the various embodiments, the cycling of the connection flow between the CS and the DFS components may occur one or more times during a communication session. Also, in at least one of the various embodiments, the cycling operations performed by the CS and the DFS components may be seamless and unseen/opaque to both ends of the communication session.

FIGURE 10 shows a flowchart for at least one of the various embodiments of process 1000 for identifying hot connection flows. After a start block, at decision block 1002, in at least one of the various embodiments, if the number of connection flows being handled on the PTMD may less than the capacity of the DFS control may be returned to the calling process. Otherwise, in at least one of the various embodiments, control may move to block 1004. In at least one of the various embodiments, if the high speed flow cache on the DFS has unused capacity both hot connection flows and "normal" connection flows may be processed on the DFS.

At block 1004, in at least one of the various embodiments, connection flows may be sorted in rank order based on the amount of data traffic passed through, exchanged, or communicated through the connection flow in a given time interval.

In at least one of the various embodiments, well known data structures and sorting algorithms may be employed to generate a tabular data structure wherein the connection flows may be logically order from based on amount of the data traffic passing through the connection flow over a time interval.

At block 1006, in at least one of the various embodiments, hot flow candidates may be determined and identified based on the top N flows based on the rank order.

In at least one of the various embodiments, the rules associated with determining/defining hot connection flows may include parameters such as "N" (e.g., how many of the top connection flows may be designated as hot connection flows). In at least

one of the various embodiments, "N" may be based on a formula that may include additional parameters including having different values based on the type of connection flow.

Next, control may be returned to a calling process.

FIGURE 11 shows a flowchart for at least one of the various embodiments of process 1100 for identifying hot connection flows. After a start block, at decision block 1102, in at least one of the various embodiments, if the number of connection flows being handled on the PTMD may be less than the capacity of the DFS, control may be returned to the calling process. Otherwise, in at least one of the various embodiments, control may move to block 1104. In at least one of the various embodiments, if the high speed flow cache on the DFS has unused capacity both hot connection flows and "normal" connection flows may be processed on the DFS.

At block 1104, in at least one of the various embodiments, the median bit-rate of connection flows being handled on the CS may be determined for use in predicting a maximum number of connection flows that may be processed by the CS. For example, in at least one of the various embodiments, if the median bit-rate of connection flows currently being handled on the CS may be 1 million bits per second and the total bandwidth of the CS for handling connection flows may be 2000 million bits per second , the maximum number of connection flows that may be processed may be estimated as 2000 connection flows (2000 million bits/sec / 1 million bits/sec).

At block 1106, in at least one of the various embodiments, hot connection flow candidates may be identified based on the top N-tile of connection flows based on the maximum number of flows the CS may be expected to handle. For example, in at least one of the various embodiments, if a CS may be expected to handle 2000 connection flows, the top 25% of connection flows based on bit-rate (for a count of 500 flows) may identified as hot connection flows. Next, in at least one of the various embodiments, control may be returned to a calling process.

It will be understood that figures, and combinations of actions in the flowchart-like illustrations, can be implemented by computer program instructions. These program instructions may be provided to a processor to produce a machine, such that the instructions executing on the processor create a means for implementing the actions specified in the

flowchart blocks. The computer program instructions may be executed by a processor to cause a series of operational actions to be performed by the processor to produce a computer implemented process for implementing the actions specified in the flowchart block or blocks. These program instructions may be stored on some type of machine readable storage media, such as processor readable non-transitive storage media, or the like.

CLAIMS

What is claimed is:

1. A method for managing communication over a network with a traffic management device (TMD) that includes a plurality of components and is operative to perform actions, comprising:

employing at least one data flow segment (DFS) component to provide packet level flow handling for a portion of a plurality of connection flows;

employing at least one control segment (CS) component to perform actions, including:

managing the plurality of connection flows and handling a remainder portion of the plurality of connection flows;

generating at least one connection flow metric based on at least one received network packet for at least one of the plurality of managed connection flows;

employing the at least one connection flow metric to determine each hot connection flow in the plurality of managed connection flows;

determining each hot connection flow to be handled by the DFS component; and

employing the DFS component to handle each determined hot connection flow.

2. The method of Claim 1, wherein employing the at least one connection flow metric further comprises determining each hot connection flow handled by the DFS component if the plurality of managed connection flows exceeds a capacity of the DFS component.

3. The method of Claim 1, wherein employing the at least one connection flow metric further comprises sorting the plurality of managed connection flows based on at least a total amount of data exchanged over a time interval.

4. The method of Claim 1, wherein employing the at least one connection flow metric further comprises:

determining a median bit-rate of data communicated for at least one connection flow being handled by the CS component; and

employing the median bit-rate of data communicated for at least one connection flow and at least a bit-rate capacity of the CS component to estimate the maximum number of connection flows the CS component can handle.

5. The method of Claim 1, wherein determining each hot connection flow to be handled by the DFS component further comprises, identifying each hot connection flow to be handled by the DFS component based at least on a total amount of data communicated over a time interval, wherein a top N number of connection flows are identified as hot connection flows.

6. The method of Claim 1, wherein determining each hot connection flow to be handled by the DFS component further comprises, identifying each hot connection flow to be handled by the DFS component based at least on a predicted connection flow capacity of the CS component, wherein a percentile of connection flows are identified as hot connection flows.

7. The method of Claim 1, wherein generating the at least one connection flow metric further comprises, examining contents of the at least one received network packet to identify at least one of a data pattern, or a meta data which indicates that the at least one of the plurality of connection flows is a hot connection flow.

8. The method of Claim 1, wherein the CS component performs further actions, comprising:

dividing each connection flow into an upload portion and a download portion;

generating separate connection flow metrics for each upload portion and each download portion of each of the plurality of managed connection flows;

employing each connection flow metric to determine each hot download portion and each hot upload portion of the plurality of managed connection flows;

determining each hot upload portion and each download portion of the plurality of managed connection flows to be handled by the DFS component; and

employing the DFS component to handle each determined hot upload portion and each download portion of the plurality of connection flows.

9. A traffic management device (TMD) that includes a plurality of components for managing communication over a network and is operative to perform actions, comprising:

a transceiver that is operative to communicate data over the network;

a memory that is operative to store instructions; and

a processor that is operative to execute instructions that enable actions, including:

employing at least one data flow segment (DFS) component to provide packet level flow handling for a portion of a plurality of connection flows; and

employing at least one control segment (CS) component to perform actions, comprising:

managing the plurality of connection flows and handling a remainder portion of the plurality of connection flows;

generating at least one connection flow metric based on at least one received network packet for at least one of the plurality of managed connection flows;

employing the at least one connection flow metric to determine each hot connection flow in the plurality of managed connection flows;

determining each hot connection flow to be handled by the DFS component; and

employing the DFS component to handle each determined hot connection flow.

10. The TMD of Claim 9, wherein employing the at least one connection flow metric further comprises determining each hot connection flow handled by the DFS component if the plurality of managed connection flows exceeds a capacity of the DFS component.

11. The TMD of Claim 9, wherein employing the at least one connection flow metric further comprises sorting the plurality of managed connection flows based on at least a total amount of data exchanged over a time interval.

12. The TMD of Claim 9, wherein employing the at least one connection flow metric further comprises:

determining a median bit-rate of data communicated for at least one connection flow being handled by the CS component; and

employing the median bit-rate of data communicated for at least one connection flow and at least a bit-rate capacity of the CS component to estimate the maximum number of connection flows the CS component can handle.

13. The TMD of Claim 9, wherein determining each hot connection flow to be handled by the DFS component further comprises, identifying each hot connection flow to be handled by the DFS component based at least on a total amount of data communicated over a time interval, wherein a top N number of connection flows are identified as hot connection flows.

14. The TMD of Claim 9, wherein determining each hot connection flow to be handled by the DFS component further comprises, identifying each hot connection flow to be handled by the DFS component based at least on a predicted connection flow capacity of the CS component, wherein a percentile of connection flows are identified as hot connection flows.

15. The TMD of Claim 9, wherein generating the at least one connection flow metric further comprises, examining contents of the at least one received network packet to identify at least one of a data pattern, or a meta data which indicates that the at least one of the plurality of connection flows is a hot connection flow.

16. The TMD of Claim 9, wherein the CS component performs further actions, comprising:

dividing each connection flow into an upload portion and a download portion;

generating separate connection flow metrics for each upload portion and each download portion of each of the plurality of managed connection flows;

employing each connection flow metric to determine each hot download portion and each hot upload portion of the plurality of managed connection flows;

determining each hot upload portion and each download portion of the plurality of managed connection flows to be handled by the DFS component; and

employing the DFS component to handle each determined hot upload portion and each download portion of the plurality of connection flows.

17. A processor readable non-transitive storage media that is operative to store processor executable instructions for managing communication over a network with a traffic management device (TMD) having a plurality of components, wherein execution of the instructions by a processor enables the TMD to perform actions, comprising:

employing at least one data flow segment (DFS) component to provide packet level flow handling for a portion of a plurality of connection flows;

employing at least one control segment (CS) component to perform actions, including:

managing the plurality of connection flows and handling a remainder portion of the plurality of connection flows;

generating at least one connection flow metric based on at least one received network packet for at least one of the plurality of managed connection flows;

employing the at least one connection flow metric to determine each hot connection flow in the plurality of managed connection flows;

determining each hot connection flow to be handled by the DFS component; and

employing the DFS component to handle each determined hot connection flow.

18. The media of Claim 17, wherein employing the at least one connection flow metric further comprises determining each hot connection flow handled by the DFS component if the plurality of managed connection flows exceeds a capacity of the DFS component.

19. The media of Claim 17, wherein employing the at least one connection flow metric further comprises sorting the plurality of managed connection flows based on at least a total amount of data exchanged over a time interval.

20. The media of Claim 17, wherein employing the at least one connection flow metric further comprises:

determining a median bit-rate of data communicated for at least one connection flow being handled by the CS component; and

employing the median bit-rate of data communicated for at least one connection flow and at least a bit-rate capacity of the CS component to estimate the maximum number of connection flows the CS component can handle.

21. The media of Claim 17, wherein determining each hot connection flow to be handled by the DFS component further comprises, identifying each hot connection flow to be handled by the DFS component based at least on a total amount of data communicated over a time interval, wherein a top N number of connection flows are identified as hot connection flows.

22. The media of Claim 17, wherein determining each hot connection flow to be handled by the DFS component further comprises, identifying each hot connection flow to be handled by the DFS component based at least on a predicted connection flow capacity of the CS component, wherein a percentile of connection flows are identified as hot connection flows.

23. The media of Claim 17, wherein generating the at least one connection flow metric further comprises, examining contents of the at least one received network packet to identify at least one of a data pattern, or a meta data which indicates that the at least one of the plurality of connection flows is a hot connection flow.

24. The media of Claim 17, wherein the CS component performs further actions, comprising:

dividing each connection flow into an upload portion and a download portion;

generating separate connection flow metrics for each upload portion and each download portion of each of the plurality of managed connection flows;

employing each connection flow metric to determine each hot download portion and each hot upload portion of the plurality of managed connection flows;

determining each hot upload portion and each download portion of the plurality of managed connection flows to be handled by the DFS component; and

employing the DFS component to handle each determined hot upload portion and each download portion of the plurality of connection flows.

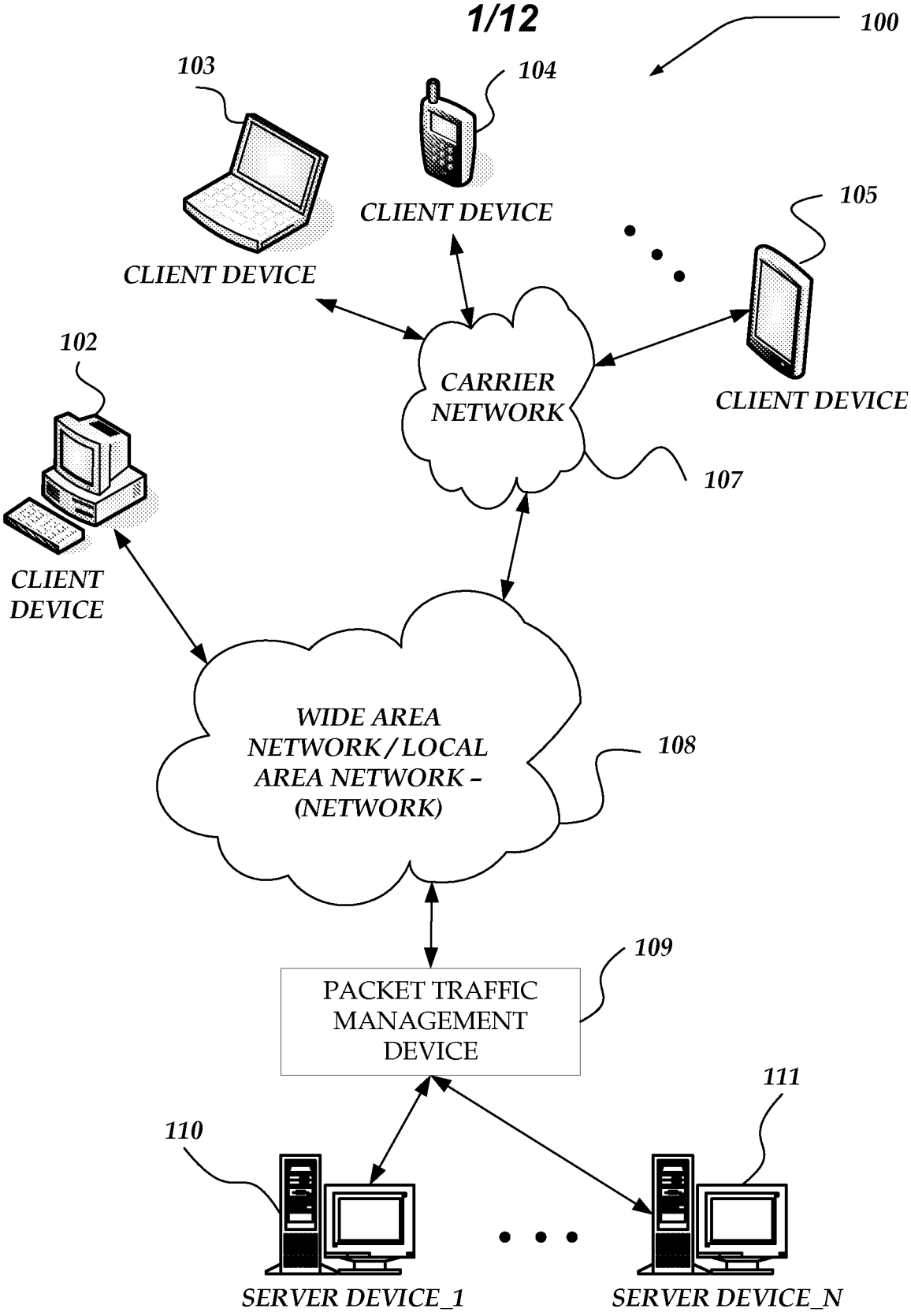

100
103
104
CLIENT DEVICE
CLIENT DEVICE
105
CLIENT DEVICE
102
CARRIER NETWORK
107
CLIENT DEVICE
WIDE AREA NETWORK / LOCAL AREA NETWORK – (NETWORK)
108
109
PACKET TRAFFIC MANAGEMENT DEVICE
111
110
SERVER DEVICE_1
SERVER DEVICE_N

FIG. 1

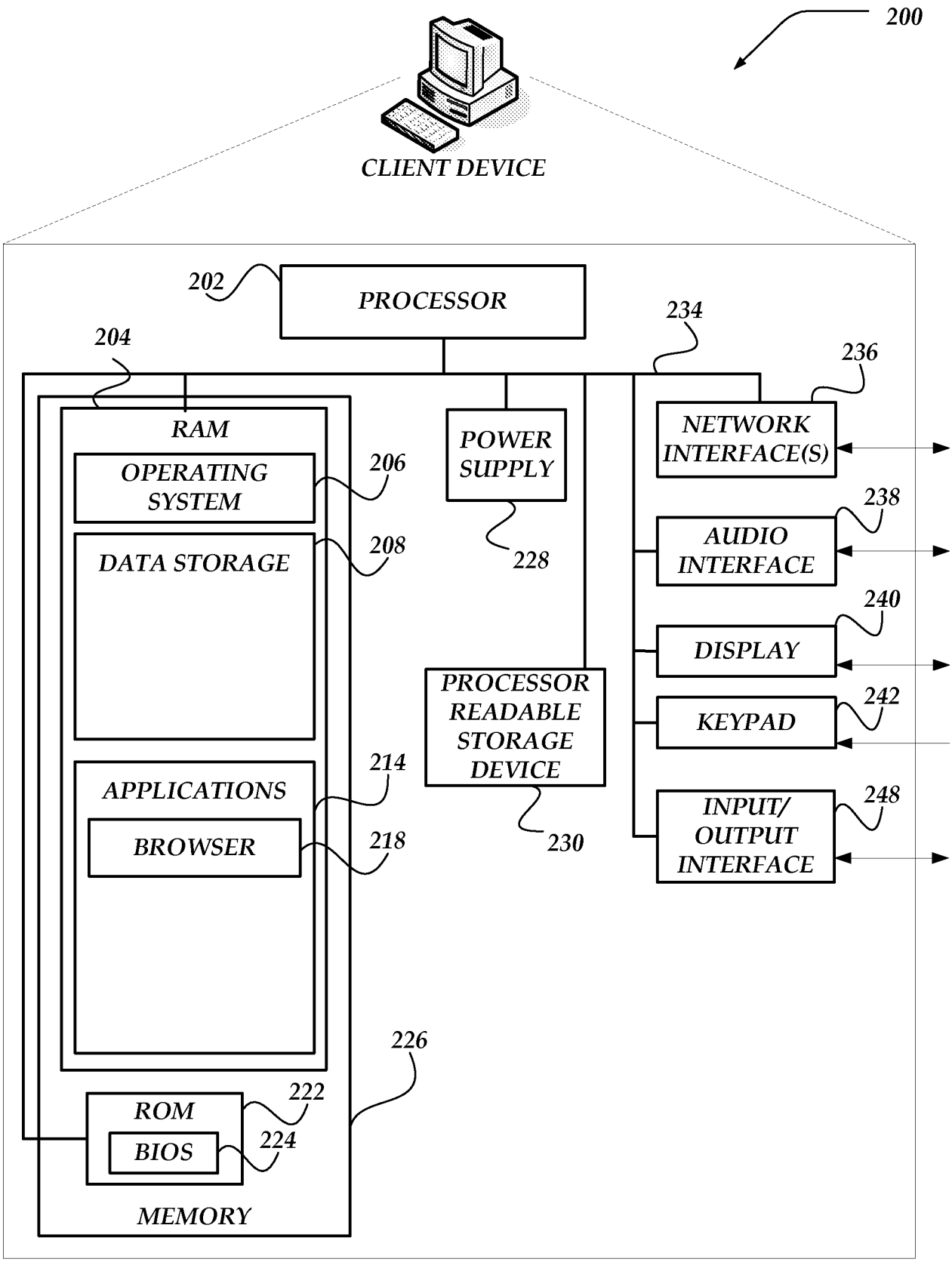

200
CLIENT DEVICE
202
PROCESSOR
204
234
236
RAM
OPERATING SYSTEM
206
DATA STORAGE
208
POWER SUPPLY
228
NETWORK INTERFACE(S)
238
AUDIO INTERFACE
240
DISPLAY
242
KEYPAD
PROCESSOR READABLE STORAGE DEVICE
230
248
INPUT/ OUTPUT INTERFACE
214
APPLICATIONS
BROWSER
218
226
222
ROM
BIOS
224
MEMORY

FIG. 2

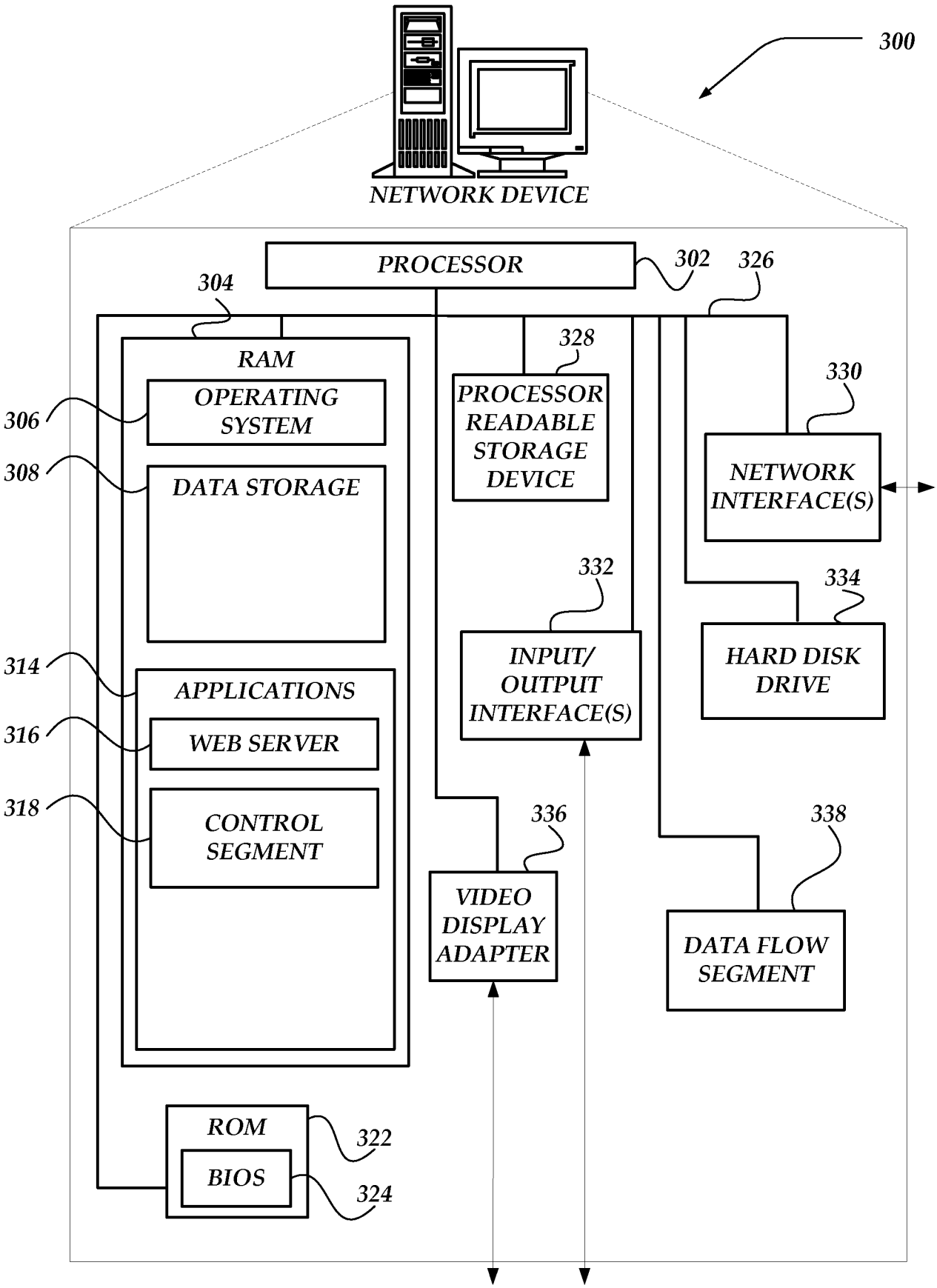

300
NETWORK DEVICE
PROCESSOR
302
326
304
RAM
328
330
306
OPERATING SYSTEM
PROCESSOR READABLE STORAGE DEVICE
NETWORK INTERFACE(S)
308
DATA STORAGE
332
334
INPUT/ OUTPUT INTERFACE(S)
HARD DISK DRIVE
314
APPLICATIONS
316
WEB SERVER
318
CONTROL SEGMENT
336
338
VIDEO DISPLAY ADAPTER
DATA FLOW SEGMENT
ROM
322
BIOS
324

FIG. 3

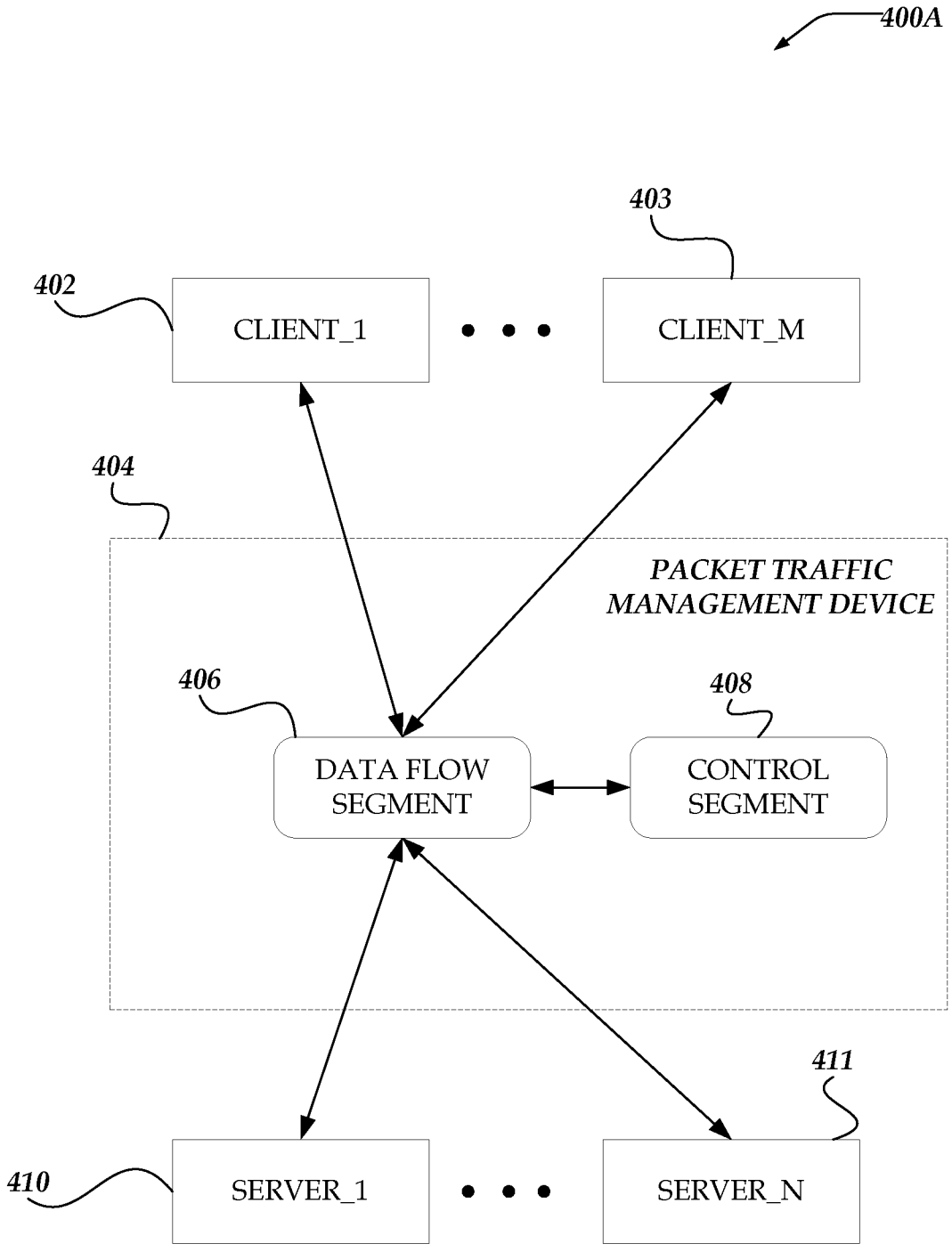

400A
403
402
CLIENT_1
CLIENT_M
404
PACKET TRAFFIC MANAGEMENT DEVICE
406
408
DATA FLOW SEGMENT
CONTROL SEGMENT
411
410
SERVER_1
SERVER_N

FIG. 4A

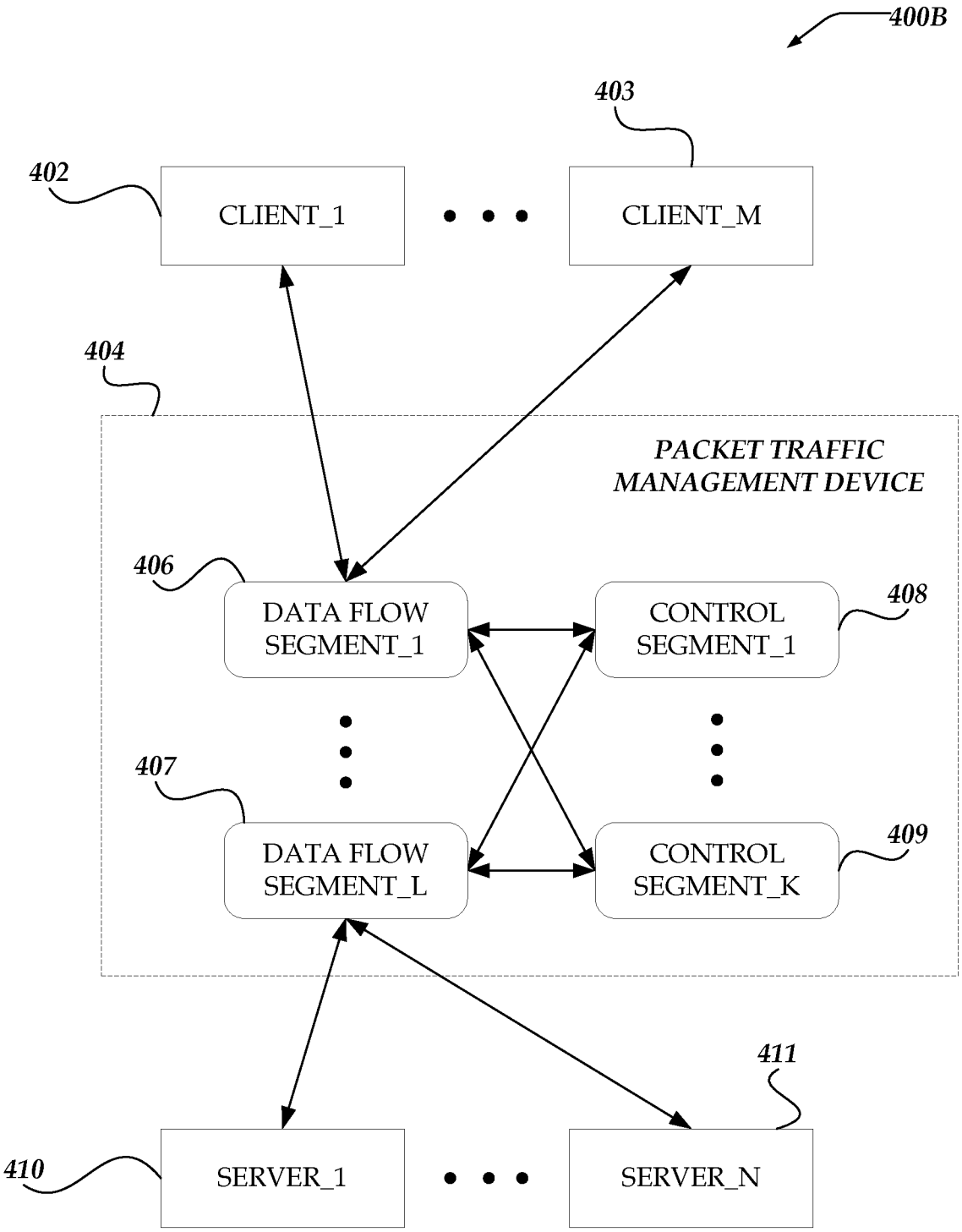

400B
403
402
CLIENT_1
CLIENT_M
404
PACKET TRAFFIC MANAGEMENT DEVICE
406
DATA FLOW SEGMENT_1
CONTROL SEGMENT_1
408
407
DATA FLOW SEGMENT_L
CONTROL SEGMENT_K
409
411
410
SERVER_1
SERVER_N

FIG. 4B

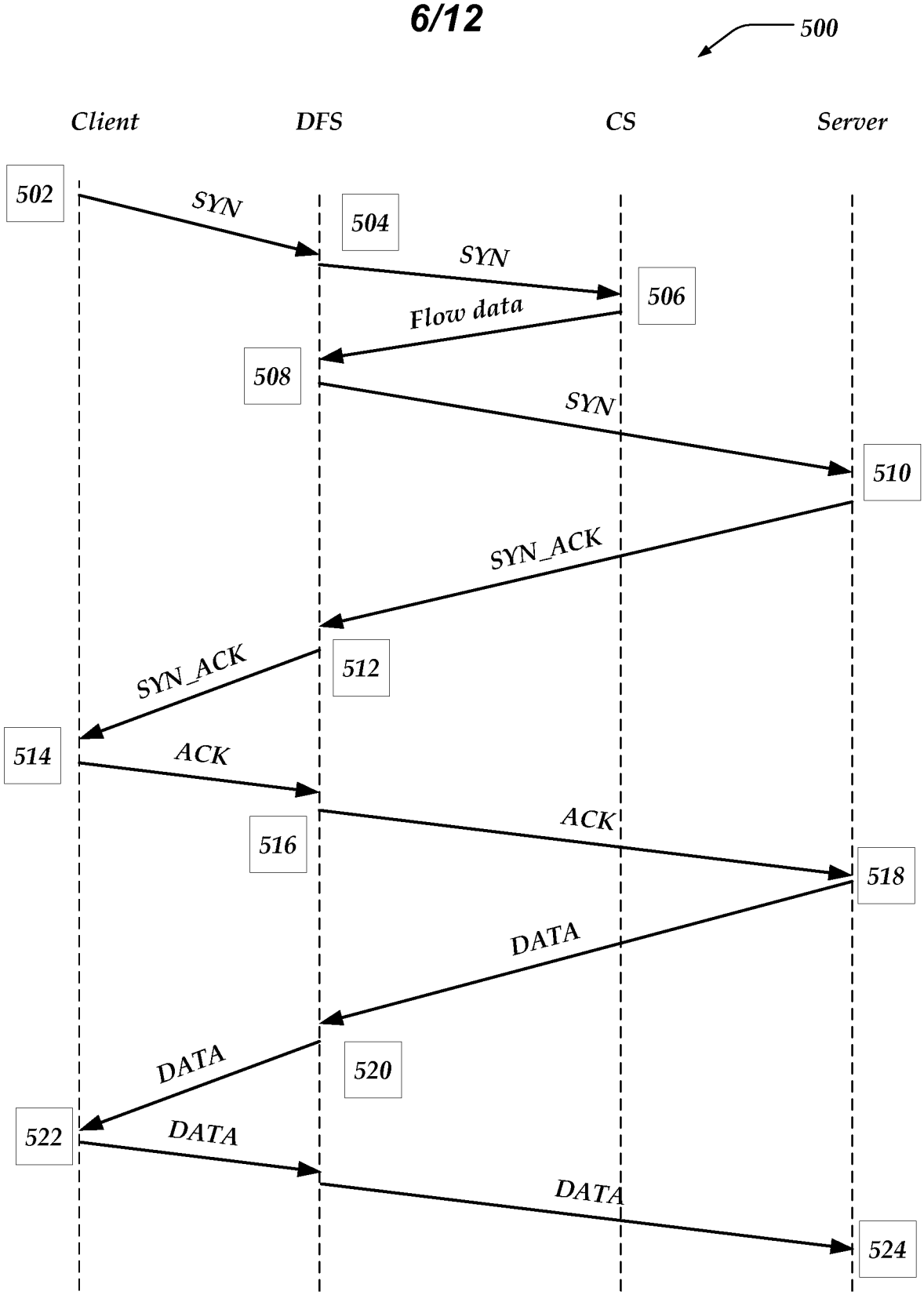

500
Client
DFS
CS
Server
502
SYN
504
SYN
506
Flow data
508
SYN
510
SYN_ACK
SYN_ACK
512
514
ACK
ACK
516
518
DATA
DATA
520
522
DATA
DATA
524

FIG. 5

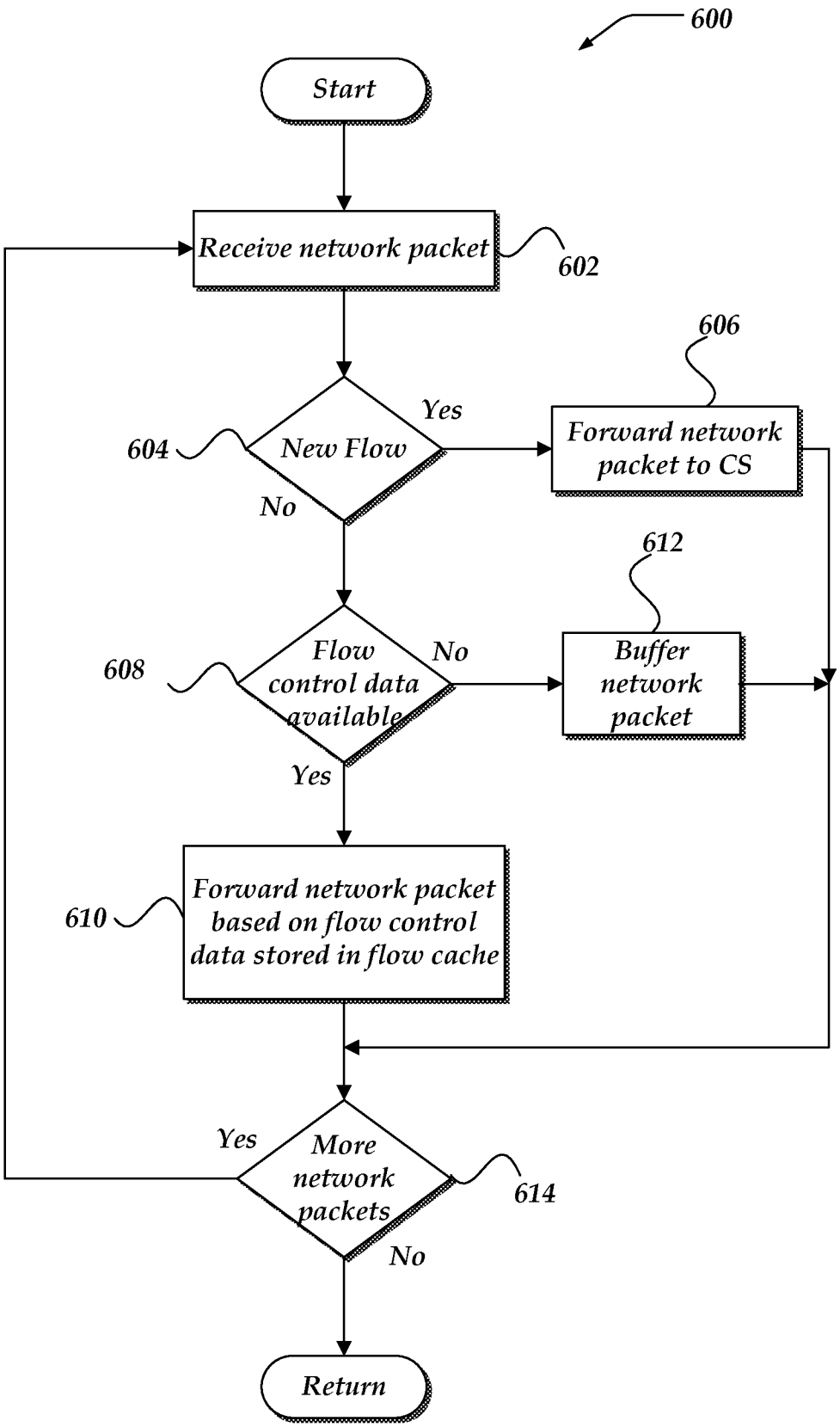

600
Start
Receive network packet
602
606
Yes
604
New Flow
Forward network packet to CS
No
612
608
Flow control data available
No
Buffer network packet
Yes
610
Forward network packet based on flow control data stored in flow cache
Yes
More network packets
614
No
Return

FIG. 6

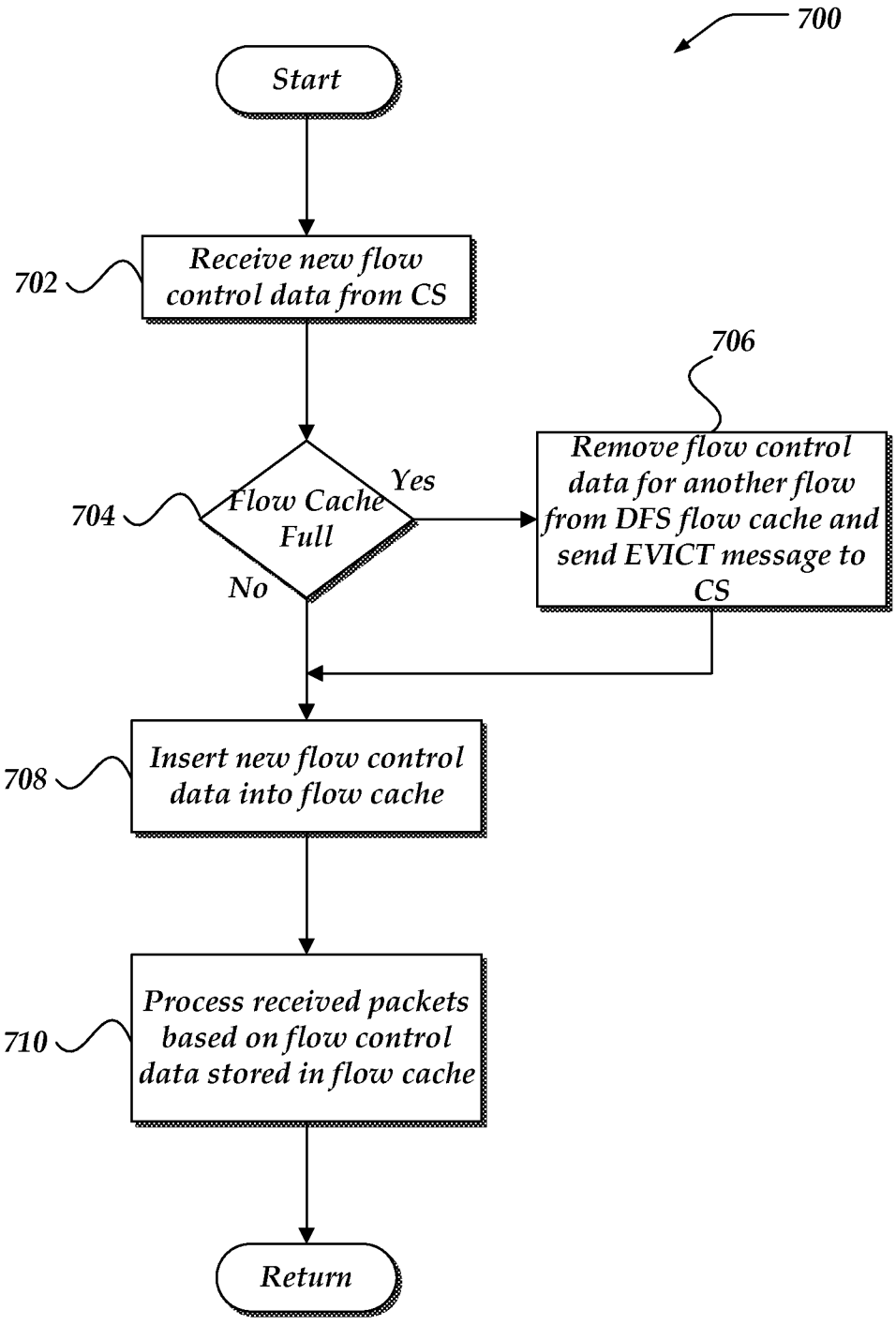

700
Start
Receive new flow control data from CS
702
706
Remove flow control data for another flow from DFS flow cache and send EVICT message to CS
Flow Cache Full
704
Yes
No
Insert new flow control data into flow cache
708
Process received packets based on flow control data stored in flow cache
710
Return

FIG. 7

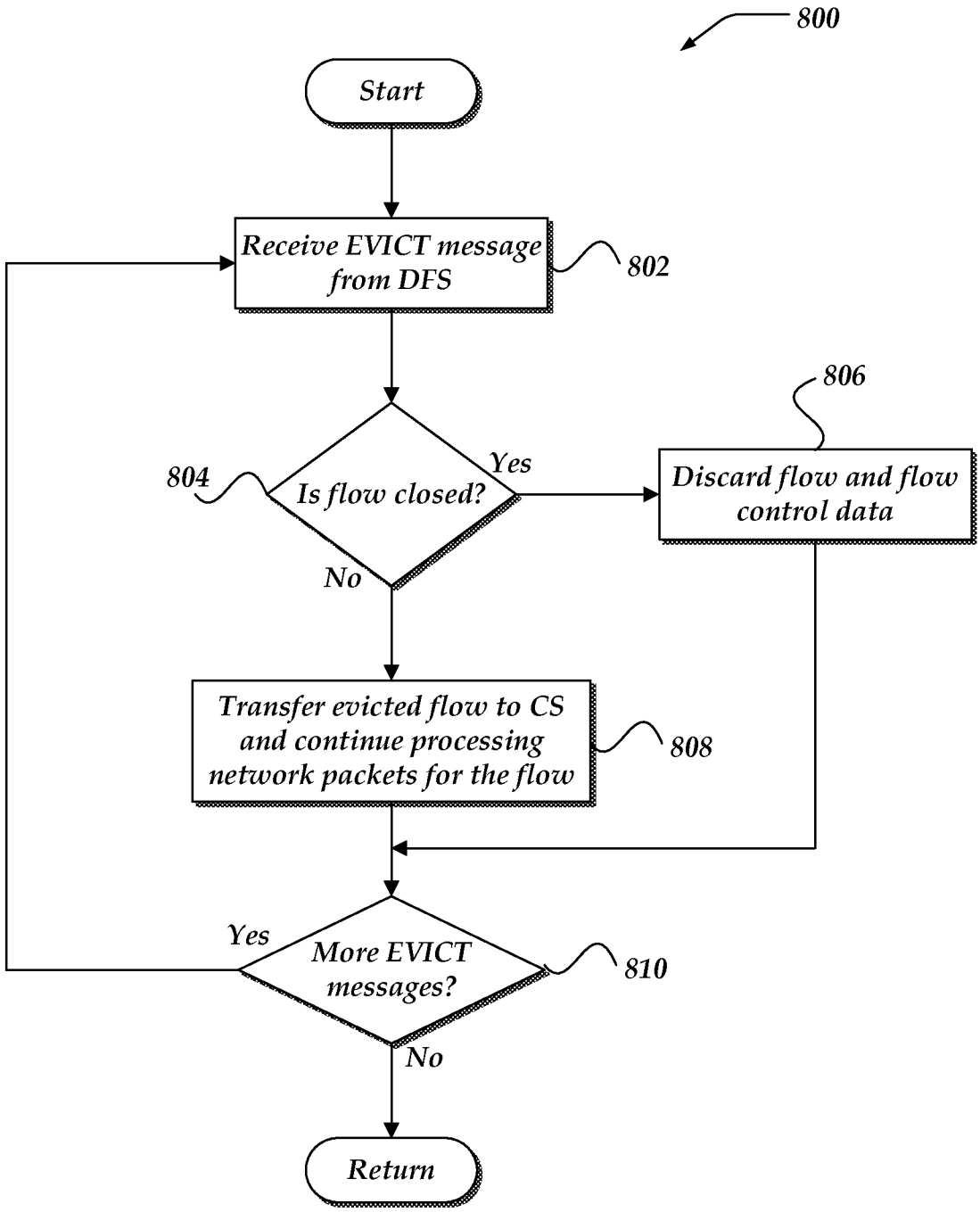

800
Start
Receive EVICT message from DFS
802
806
804
Is flow closed?
Yes
Discard flow and flow control data
No
Transfer evicted flow to CS and continue processing network packets for the flow
808
Yes
More EVICT messages?
810
No
Return

FIG. 8

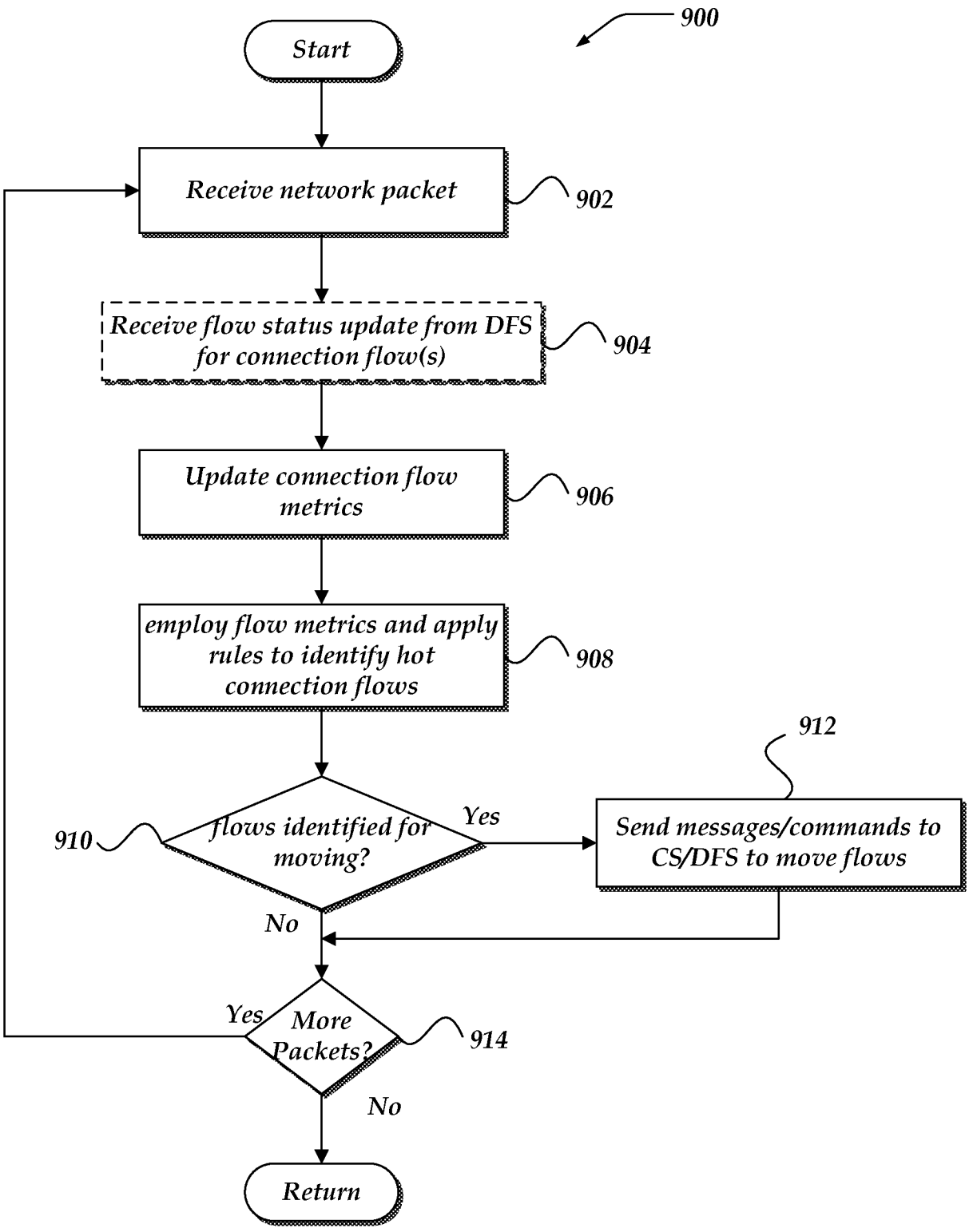

900
Start
Receive network packet
902
Receive flow status update from DFS for connection flow(s)
904
Update connection flow metrics
906
employ flow metrics and apply rules to identify hot connection flows
908
912
910
flows identified for moving?
Yes
Send messages/commands to CS/DFS to move flows
No
Yes
More Packets?
914
No
Return

FIG. 9

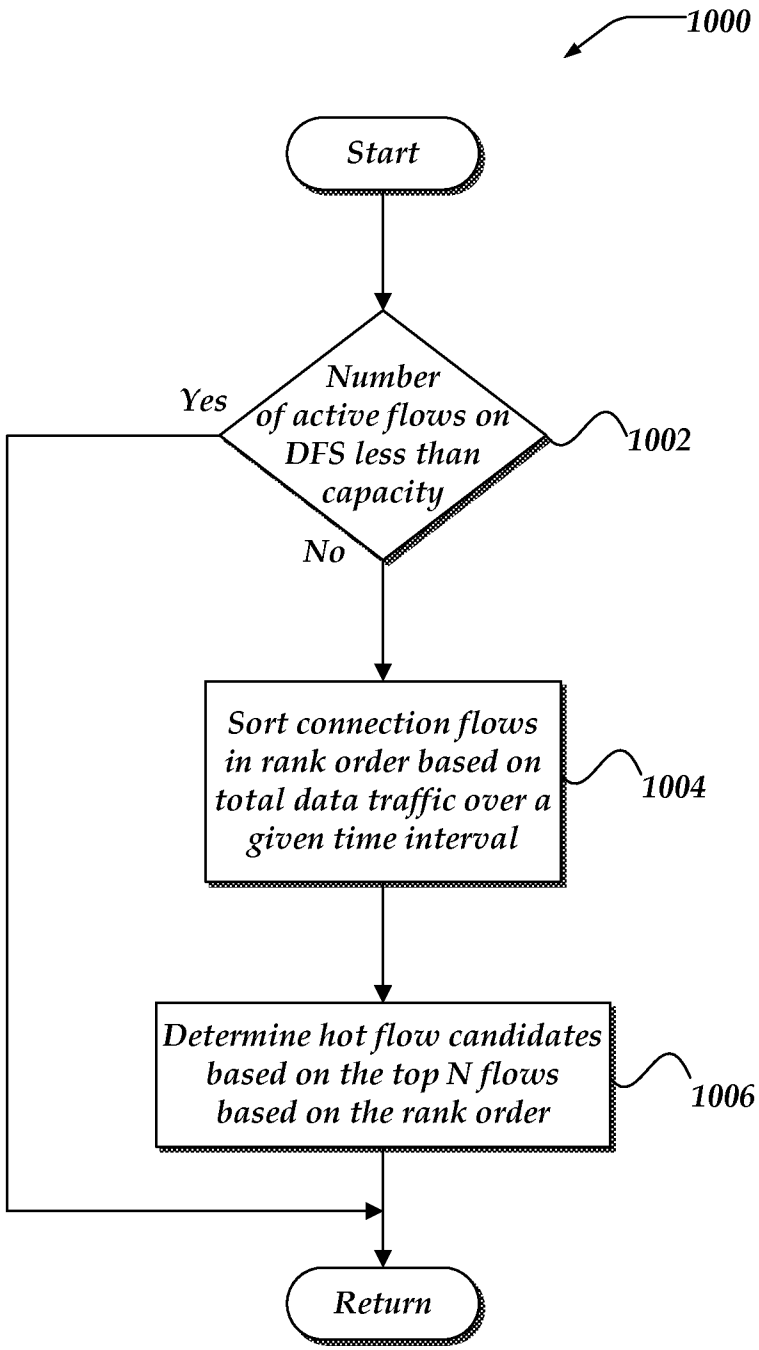

1000
Start
Number of active flows on DFS less than capacity
1002
Yes
No
Sort connection flows in rank order based on total data traffic over a given time interval
1004
Determine hot flow candidates based on the top N flows based on the rank order
1006
Return

FIG. 10

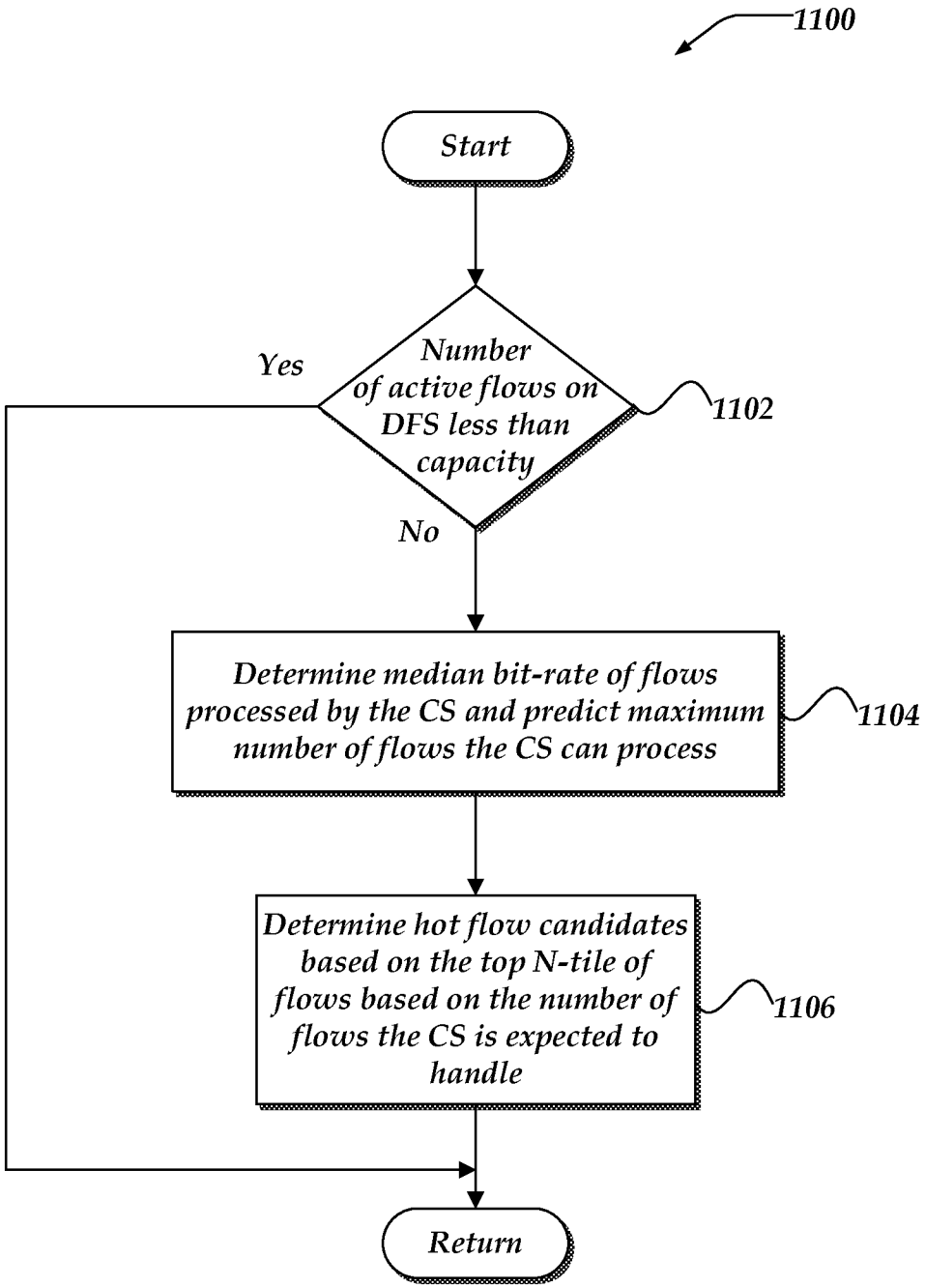

1100
Start
Number of active flows on DFS less than capacity
1102
Yes
No
Determine median bit-rate of flows processed by the CS and predict maximum number of flows the CS can process
1104
Determine hot flow candidates based on the top N-tile of flows based on the number of flows the CS is expected to handle
1106
Return

FIG. 11

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2013/038168

A. CLASSIFICATION OF SUBJECT MATTER

H04L 12/24(2006.01)i, H04L 12/801(2013.01)i, H04W 28/02(2009.01)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 12/24; G06F 15/16; G06F 15/173; H04Q 7/20; G01R 31/08; G06F 15/18; H04L 12/801; H04W 28/02

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: traffic, management, flow, control, metric, determination, hot connection, segment

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 7561517 B2 (KLINKER, JAMES ERIC et al.) 14 July 2009 See abstract; column 4 lines 16-56, column 8, line 63 - column, 9 line 8, column 18, line 66 - column 19, line 60; claims 1-5, 7; and figures 7-8, 19-22.	1-3,5,9-11,13,17-19,21
A		4,6-8,12,14-16,20,22-24
A	US 8024483 B1 (ROTHSTEIN, JESSE ARBRAHAM et al.) 20 September 2011 See column 9, line 47 - column 11, line 29; claims 1-2; and figures 4-5.	1-24
A	US 2009-0209262 A1 (STAMOULIS, ANASTASIOS et al.) 20 August 2009 See paragraphs [0061]-[0068]; and figure 5.	1-24
A	US 2008-0162390 A1 (KAPOOR, HARSH et al.) 3 July 2008 See paragraphs [0013]-[0017], [0138]-[0141]; claim 1; and figure 1.	1-24
A	US 2008-0256239 A1 (GILDE, ROBERT GEORGE et al.) 16 October 2008 See paragraphs [0048]-[0055], [0100]-[0135]; claims 1-10; and figures 17-22.	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
12 August 2013 (12.08.2013)	**14 August 2013 (14.08.2013)**

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
LEE Dong Yun
Telephone No. +82-42-481-8734

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.
PCT/US2013/038168

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 7561517 B2	14/07/2009	AU 2002-353974 A8	19/05/2003
		AU 2002-363519 A8	19/05/2003
		AU 2003-287262 A1	25/05/2004
		EP 1446917 A1	18/08/2004
		EP 1446921 A2	18/08/2004
		EP 1449106 A1	25/08/2004
		EP 1579629 A2	28/09/2005
		EP 1579629 A4	18/11/2009
		JP 2005-508593 A	31/03/2005
		JP 2005-508596 A	31/03/2005
		JP 2005-509369 A	07/04/2005
		US 2003-0086422 A1	08/05/2003
		US 2003-0088529 A1	08/05/2003
		US 2003-0088671 A1	08/05/2003
		US 2003-0133443 A1	17/07/2003
		US 2007-0140128 A1	21/06/2007
		US 7133365 B2	07/11/2006
		US 7222190 B2	22/05/2007
		US 7606160 B2	20/10/2009
		US 7668966 B2	23/02/2010
		WO 03-040874 A3	17/07/2003
		WO 03-040947 A1	15/05/2003
		WO 03-041342 A1	15/05/2003
		WO 2004-040423 A2	13/05/2004
		WO 2004-040423 A3	29/07/2004
US 8024483 B1	20/09/2011	US 8010668 B1	30/08/2011
		US 8326984 B1	04/12/2012
US 2009-0209262 A1	20/08/2009	CN 101946540 A	12/01/2011
		EP 2091282 A1	19/08/2009
		EP 2192803 A1	02/06/2010
		EP 2192803 B1	01/05/2013
		JP 2011-512759 A	21/04/2011
		JP 2013-070387 A	18/04/2013
		KR 10-1178372 B1	29/08/2012
		KR 10-2010-0114933 A	26/10/2010
		TW 201004406 A	16/01/2010
		WO 2009-103013 A1	20/08/2009
US 2008-0162390 A1	03/07/2008	AU 2001-91227 A1	08/04/2002
		AU 2006-265624 A1	11/01/2007
		AU 2006-265624 B2	17/11/2011
		CA 2423475 A1	04/04/2002
		CA 2614328 A1	11/01/2007
		CN 1518694 A	04/08/2004
		CN 1518694 C0	16/05/2007
		EP 1381937 A2	21/01/2004
		EP 1898822 A2	19/03/2008

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.
PCT/US2013/038168

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
		EP 1960867 A2	27/08/2008
		EP 2432188 A1	21/03/2012
		EP 2442525 A1	18/04/2012
		JP 2004-524598 A	12/08/2004
		JP 2009-504201 A	05/02/2009
		KR 10-2004-0005824 A	16/01/2004
		US 2002-0059424 A1	16/05/2002
		US 2002-0165947 A1	07/11/2002
		US 2006-0010207 A1	12/01/2006
		US 2006-0143499 A1	29/06/2006
		US 2007-0006215 A1	04/01/2007
		US 2007-0016183 A1	18/01/2007
		US 2007-0192863 A1	16/08/2007
		US 2008-0133517 A1	05/06/2008
		US 2008-0133518 A1	05/06/2008
		US 2008-0134330 A1	05/06/2008
		US 2008-0229415 A1	18/09/2008
		US 2008-0262990 A1	23/10/2008
		US 2008-0262991 A1	23/10/2008
		US 2010-0042565 A1	18/02/2010
		US 7836443 B2	16/11/2010
		US 7979368 B2	12/07/2011
		US 8010469 B2	30/08/2011
		US 8046465 B2	25/10/2011
		US 8080009 B2	20/12/2011
		US 8402540 B2	19/03/2013
		WO 02-027469 A2	04/04/2002
		WO 02-027469 A3	06/11/2003
		WO 2007-005830 A2	11/01/2007
		WO 2007-005830 A3	18/12/2008
		WO 2007-070838 A2	21/06/2007
		WO 2007-070838 A3	03/07/2008
US 2008-0256239 A1	16/10/2008	US 2001-0037387 A1	01/11/2001
		US 2002-0138618 A1	26/09/2002
		US 7343413 B2	11/03/2008
		US 8380854 B2	19/02/2013