(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) **EP 3 616 075 B1**

(12) **EUROPEAN PATENT SPECIFICATION**

(45) Date of publication and mention of the grant of the patent:
16.08.2023 Bulletin 2023/33

(21) Application number: **18790188.9**

(22) Date of filing: **27.04.2018**

(51) International Patent Classification (IPC):
H04L 61/4511 *(2022.01)* ***H04L 47/2475*** *(2022.01)*
H04L 45/7453 *(2022.01)* ***H04L 47/2441*** *(2022.01)*
H04L 47/80 *(2022.01)*

(52) Cooperative Patent Classification (CPC):
H04L 47/2441; H04L 45/7453; H04L 47/2475; H04L 47/80; H04L 61/4511

(86) International application number:
PCT/US2018/030014

(87) International publication number:
WO 2018/201084 (01.11.2018 Gazette 2018/44)

(54) **SYSTEM AND METHOD FOR TRACKING DOMAIN NAMES FOR THE PURPOSES OF NETWORK MANAGEMENT**

SYSTEM UND VERFAHREN ZUR VERFOLGUNG VON DOMÄNENNAMEN FÜR NETZWERKVERWALTUNGSZWECKE

SYSTÈME ET PROCÉDÉ DE SUIVI DE NOMS DE DOMAINE À DES FINS DE GESTION DE RÉSEAU

(84) Designated Contracting States:
AL AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HR HU IE IS IT LI LT LU LV MC MK MT NL NO PL PT RO RS SE SI SK SM TR

(30) Priority: **28.04.2017 US 201762491581 P**

(43) Date of publication of application:
04.03.2020 Bulletin 2020/10

(73) Proprietor: **Opanga Networks, Inc.**
Seattle, WA 98101 (US)

(72) Inventors:
- **BROWN, Sean**
  Seattle, Washington 98101 (US)
- **BURNETTE, John**
  Seattle, Washington 98101 (US)
- **HADORN, Ben**
  Seattle, Washington 98101 (US)
- **GARZA, Hugo**
  Seattle, Washington 98101 (US)
- **NORDNESS, Ethan**
  Seattle, Washington 98101 (US)

(74) Representative: **Herrero & Asociados, S.L.**
Cedaceros, 1
28014 Madrid (ES)

(56) References cited:
US-A1- 2006 011 720
US-A1- 2010 138 910
US-A1- 2012 327 778
US-A1- 2012 327 779
US-A1- 2016 219 024
US-A1- 2016 261 510
US-A1- 2016 323 186
US-B1- 9 253 068
US-B2- 8 509 787

EP 3 616 075 B1

Description

CROSS-REFERENCE TO RELATED APPLICATION

[0001] This patent document claims the benefit of U.S. Provisional Patent Application No. 62/491,581, filed on April 28, 2017.

BACKGROUND

[0002] Data can traverse a network in the form of network traffic. Network traffic may include one or more encapsulated packets that are transmitted across a network between two endpoints. For example, a data packet traverses a data network from a content server to a user equipment.

[0003] Identifying network traffic is important for various network management and analysis applications. Historically, network traffic has been used by network service providers to understand traffic patterns and consumer behavior on their networks.

[0004] Network traffic can be identified by an identification string that defines a realm of administrative autonomy, authority, and/or control within the Internet. For example, network traffic can be identified by hostnames that identify endpoints of the network traffic.

[0005] Network identification information, such as hostnames, can be derived by performing deep packet inspection of application layer data of data packets in individual data flows. However, identifying network traffic using traditional means has become difficult, timeconsuming, and otherwise troublesome with the adoption of encryption by Transport Layer Security (TLS) or Secure Sockets Layer (SSL). Payloads of data packets are now encrypted. As a result, past implementations that have relied on deep packet inspection techniques (e.g., reading into application layer data) are no longer viable due to encryption, which now hides most that data.

[0006] Deep packet inspection has other drawbacks. For example, deep packet inspection has also historically been an expensive computational operation with negative side effects for networks.

[0007] Accordingly, a new strategy for identifying network traffic carrying encrypted data packets, and which does not require computationally expensive deep packet inspection processes, would be advantageous for performing network analysis and management techniques. One prior art solution is disclosed by document US2016/0323186.

BRIEF SUMMARY

[0008] The invention is defined by the subject-matter of the independent claims. Further embodiments are defined by the subject-matter of the dependent claims.

BRIEF DESCRIPTION OF THE DRAWINGS

[0009]

Fig. 1 illustrates a system architecture according to the invention.
Fig. 2 illustrates a system architecture according to the invention.
Fig. 3A illustrates a client device according to the invention.
Fig. 3B illustrates a domain name system (DNS) spy according to the invention.
Fig. 3C illustrates a DNS server according to the invention.
Fig. 3D illustrates a transport manager according to the invention.
Fig. 3E illustrates a content server according to the invention.
Fig. 4 illustrates a data packet according to the invention.
Fig. 5 illustrates a DNS response according to the invention.
Fig. 6 illustrates an identification table according to the invention.
Fig. 7 illustrates a method of extracting a hostname and internet protocol (IP) address from a DNS response according to the invention.
Fig. 8 illustrates a method of adding a new entry to an identification table according to the invention.
Fig. 9 illustrates a method of determining a length of a non-DNS packet according to the invention.
Fig. 10 illustrates a method of determining a timestamp of a non-DNS packet according to the invention.
Fig. 11 illustrates a method of updating an entry of an identification table according to the invention.
Fig. 12 illustrates a method of managing a data flow using an identification table according to the invention.
Fig. 13 illustrates a method of identifying and pacing an elephant flow using an identification table according to the invention.
Fig. 14 illustrates a method of controlling a size of an identification table according to the invention.
Fig. 15 illustrates a method of generating, using, and updating an identification table according to the invention.

DETAILED DESCRIPTION

[0010] The invention relates to a system and method for identifying domain name service (DNS) responses transmitted from a server, parsing the DNS responses to create a mapping between a content delivery network (CDN) server's internet protocol (IP) address(es) and the domain name based on the DNS responses, further identifying a specific data flow from a client to a server by this domain name, applying a traffic management policy or

statistics collection based on the mapped domain name / server IP address, cleaning/expiring entries from the map based on a significance of data transferred and activity, collecting/storing statistical data on previously mentioned data flows as a whole or as determined by the domain name, and determining the most significant domain name(s) on a given network by analyzing the whole of the statistical data collected.

[0011] According to an embodiment, a device, a system, a method, or a combination thereof, can be used to capture and create a mapping, or cache, of hostname and corresponding server IP addresses of a variety of data flows, even when application level packet data in the data flows is encrypted.

[0012] As used herein, the term "hostname" refers to a name of a specific host, content provider, content, or a combination thereof. A hostname is, for example, a nickname, a nodename, a domain name, a web address, or a combination thereof.

[0013] In the invention, a reverse DNS lookup table is generated. A device accesses the table, and uses entries within the table to identify one or both endpoints of data flows traversing a network. The device implements network management policies based on the identified endpoints. For example, a transport manager accesses the table, uses the table to identify that an endpoint of a specific data flow has been historically associated with relatively burdensome data flows, and paces the data flow.

[0014] Fig. 1 illustrates a system 100 architecture according to an embodiment of the present disclosure.

[0015] The system 100 includes a client device 110, a first network 120, a DNS spy 130, a second network 140, and a DNS server 150.

[0016] The client device 110 is configured to receive one or more inputs from one or more users, and to communicate with the DNS server via the first and second networks 120 and 140. The client device 110 is, for example, a desktop computer, a laptop computer, a tablet device, a smartphone, an e-reader, a smart watch, a smart television, or a combination thereof.

[0017] The first network 120 is a wired or wireless network that connects the client device 110 and the DNS spy 130 to each other. According to various embodiments, the first network 120 may be any network that enables communication between or among machines, databases, and devices. Accordingly, the first network 120 may be a wide access network (WAN), wired network, a fiber network, a wireless network (e.g., a mobile or cellular network), a cellular or telecommunications network (e.g., WiFi, Global System for Mobile Communications (GSM), Universal Mobile Telecommunications System (UMTS), Long Term Evolution (LTE) network), or any suitable combination thereof. The network 130 may include one or more portions of a private network, a public network (e.g., the Internet), or any suitable combination thereof. In a specific embodiment, the first network 120 includes a core collection of subnodes linking to a radio access network (RAN).

[0018] The DNS spy 130 is connected between the client device 110 and the DNS server 150 via the first network 120 and the second network 140. In various embodiments, the DNS spy 130 is configured to intercept DNS responses sent from the DNS server 150 to the client device 110, and to generate and/or update an identification table based on hostname/IP address pairs in the DNS responses. The DNS spy 130 may also intercept other types of packets traversing the first network 120, the second network 140, or both, (e.g., non-DNS data packets) and update the identification table based on characteristics of the intercepted packets. For example, the DNS spy 130 may update an existing entry including a hostname/IP address pair with a size, a timestamp, or both, derived from an intercepted non-DNS packet.

[0019] The second network 140 is a wired or wireless network that connects the DNS spy 130 and the DNS server 150 to each other. In a specific embodiment, the second network 140 is the Internet.

[0020] The DNS server 150 is a server device configured to generate DNS responses in response to DNS requests from various client devices, including the client device 110. Upon receiving a DNS request that specifies a specific hostname, the DNS server 150 looks up an IP address associated with the specific hostname. The DNS server 150 then transmits the IP address to the requesting device, so that the requesting device can access a content server associated with the hostname.

[0021] In the invention, the client device 110 transmits a DNS request to the DNS server 150 via one or more networks, such as the first and second networks 120 and 140. The DNS response includes a hostname, for example, a name of a video-hosting website, such as "YOUTUBE" or "YOUTUBE.COM."

[0022] When the DNS server 150 receives the DNS request, the DNS server 150 extracts the hostname from the DNS request, and looks up an IP address associated with the hostname. The DNS server 150 generates a DNS response that includes the hostname and the IP address associated with the hostname. The DNS server 150 transmits the DNS response back to the client device 110 over the first and second networks 120 and 140.

[0023] The DNS spy 130 intercepts the DNS response. Because the DNS response is not encrypted, the DNS spy 130 extracts the hostname and the IP address from the DNS response without performing decryption. The DNS spy 130 then stores the hostname/IP address pair in an entry of an identification table. The DNS spy 130 maps multiple hostname/IP address pairs to entries of the identification table by intercepting and inspecting multiple DNS responses.

[0024] According to an embodiment, the DNS spy 130 controls a size of the identification table by pruning the identification table when a triggering event occurs. For example, the DNS spy 130 deletes one or more entries in the identification table when the identification table reaches a predetermined size.

[0025] In various embodiments, the DNS server 150

is unable to return a hostname based on a given IP address. However, a device accessing the identification table (e.g., a transport manager) can determine a hostname associated with a specific IP address by identifying an entry including the specific IP address. The identification table generated by the DNS spy 130 can therefore be used to perform a reverse DNS lookup operation, according to various embodiments.

[0026] Fig. 2 illustrates a system 200 architecture according to an embodiment of the present disclosure.

[0027] The system 200 includes a client device 210, a first network 220, a DNS spy 230, a second network 240, a DNS server 250, a third network 260, a transport manager 270, a fourth network 280, and a content server 290. Although Fig. 2 only illustrates one of each of the client device 210, the first network 220, the DNS spy 230, the second network 240, the DNS server 250, the third network 260, the transport manager 270, the fourth network 280, and the content server 290, the system 200 can include a plurality of each of the client device 210, the first network 220, the DNS spy 230, the second network 240, the DNS server 250, the third network 260, the transport manager 270, the fourth network 280, and the content server 290 according to various embodiments. For example, multiple client devices 210 can be connected to the first network 220, the third network 260, or both, multiple DNS servers 250 can be connected to the second network 240, and multiple content servers 290 can be connected to the fourth network 280.

[0028] In various embodiments, the client device 210, the first network 220, the DNS spy 230, the second network 240, and the DNS server 250 are equivalent to the client device 110, the first network 120, the DNS spy 130, the second network 140, and the DNS server 150 described above with respect to Fig. 1. In certain embodiments, the first network 220 is the same as the third network 260, and the second network 240 is the same as the fourth network 280. Accordingly, the DNS spy 230, the transport manager 270, or both, are between the client device 210 and the DNS server 250, the client device 210 and the content server 290, or both.

[0029] When the client device 210 receives a DNS response including a requested IP address from the DNS server 250, the client device 210 generates a request for content including the IP address. The client device 210 transmits the request for content to the content server 290 over the third and fourth networks 260 and 280.

[0030] Equipment associated with the third and fourth networks 260 and 280 route the request for content to the content server 290 using the IP address. In a specific embodiment, the third network 260 is the first network 220, and the fourth network 280 is the second network 240.

[0031] When the content server 290 receives the request for content, the content server 290 transmits the requested content to the client device 210.

[0032] In various embodiments, the request for content from the client device 210, the content from the content server 290, or both, are each transmitted in the form of one or more data packets. The data packets are non-DNS data packets, for example.

[0033] The transport manager 270 is configured to intercept the request for content transmitted from the client device 210 to the content server 290, the one or more data packets transmitted from the content server 290 to the client device 210, or both. According to various embodiments, the transport manager 270 intercepts data packets transported over the third network 260 between one or more client devices and one or more content servers.

[0034] The transport manager 270 is located between the client device 210 and the content server 290. In an embodiment, the transport manager 270 is at a border traffic aggregation point connecting the third network 260 to the fourth network 280. In an example in which the third network 260 is a 3rd Generation Partnership Project (3GPP) standard mobile network, the aggregate point is part of the sGi-LAN connecting to the Packet Data Network (PDN)-Gateway core element and outwards to the Internet. In an example in which the third network 260 is a 4G network, the aggregate point is part of a Gi-LAN connecting to a gateway GPRS support node (GGSN)-Gateway and outwards to the Internet. However, in other embodiments, the transport manager 270 is located elsewhere.

[0035] The transport manager 270 manages data traffic transmitted over the third network 260. In certain embodiments, the transport manager 270 is configured to optimize network resources, alleviate congestion, perform other data management operations, or a combination thereof, in the third network 260 by managing data traffic through the third network 260. For example, the transport manager 270 paces a data flow between the client device 210 and the content server 290 based on one or more policies stored at the transport manager 270. The transport manager 270 paces the data flow after identifying that the data flow is relatively burdensome to the third network 260, determining that the third network 260 currently congested, determining that the data flow is relatively unimportant compared to other data flows traversing the third network 260, or a combination thereof.

[0036] The transport manager 270 paces the data flow by throttling the data flow, temporarily storing data packets in the data flow, requiring the data flow to traverse a network other than the third network 260, or a combination thereof. For example, the transport manager 270 may pace the data flow by requiring data packets within the data flow to traverse a local WIFI network, rather than the third network 260, when the WIFI network connects the client device 210 to the content server 290.

[0037] Further details regarding the pacing of data flows may be found in commonly-assigned U.S. Application No. 15/060,486, entitled "SYSTEMS AND METHODS FOR PACING DATA FLOWS," which was filed on March 3, 2016, and hereby incorporated by reference in

its entirety.

[0038] In various embodiments, the transport manager 270 selectively paces elephant flows traversing the third network 260. An elephant flow is a data flow that is relatively burdensome to the third network 260. For example, an elephant flow is a data flow including an amount of transferred data that is greater than a threshold, a data flow with a duration that exceeds a threshold duration, a data flow including data packets with a particular file type, or a combination thereof.

[0039] In some embodiments, a data flow can be identified as an elephant flow by identifying a hostname associated with the data flow. In an embodiment, when a data flow is identified as being to or from a host that has been previously known to be likely to generate elephant flows, the transport manager 270 identifies the data flow as an elephant flow. For example, when the transport manager 270 identifies that the content server 290 is associated with a host that transmits data packet that require a significant amount of network resources, e.g., a video streaming service (e.g., YOUTUBE.COM, NETFLIX.COM, etc.), the transport manager 270 will automatically identify a data flow between the content server 290 and the client device 210 as an elephant flow and pace the data flow. In another example, when the transport manager 270 identifies that the content server 290 is associated with a gaming service (e.g., POKEMON GO, etc.), the transport manager 270 will automatically identify the data flow between the content server 290 and the client device 210 as an elephant flow and pace the data flow. Further details regarding the management of elephant flows may be found in commonly-assigned U.S. Application No. 15/703,908, entitled "DIRECTED HANDOVER OF ELEPHANT FLOWS," which was filed on September 13, 2017, and hereby incorporated by reference in its entirety.

[0040] In the invention, the transport manager 270 identifies a data flow between the content server 290 and the client device 210 for management by identifying a hostname associated with the content server 290, according to various embodiments of the present disclosure. The transport manager 270 identifies the hostname by intercepting data packets transmitted between the content server 290 and the client device 210 that traverse the third network 260.

[0041] While the data packets include information identifying the hostname, this information is encrypted. For example, the hostname is included in an encrypted payload of each of the data packets.

[0042] Rather than decrypting the data packets, in an embodiment, the transport manager 270 identifies the hostname of the content server 290 by extracting the IP address of the content server 290 from one of the data packets, accessing an information table stored in a storage 232, identifying an entry in the information table including the extracted IP address, and determining the hostname by reading the identified entry. The information table is generated by the DNS spy 230, according to various embodiments.

[0043] Accordingly, the transport manager 270 manages data flows traversing the third network 260 by accessing the information table generated by the DNS spy, rather than decrypting individual data packets within the data flows.

[0044] In some embodiments, the DNS spy 230, the transport manager 270, and the storage 232 are separate and interconnected devices. In other embodiments, the DNS spy 230, the transport manager 270, and the storage 232, are the same device.

[0045] Figs. 3A to 3E illustrate devices according to various embodiments of the present disclosure. Any of the devices shown in Figs. 3A to 3E may be implemented in a generalpurpose computer modified (e.g., configured or programmed) by software to be a specialpurpose computer to perform the functions described herein for that machine, database, or device. Moreover, any two or more of the machines, databases, or devices illustrated in Figs. 3A to 3E may be combined into a single machine, and the functions described herein for any single machine, database, or device may be subdivided among multiple machines, databases, or devices.

[0046] Fig. 3A illustrates a client device 310 according to an embodiment of the present disclosure. The client device 310 may include various types of user devices, such as mobile devices (e.g., laptops, smart phones, tablet computers, and so on), computing devices, set-top boxes, vehicle computing devices, gaming devices, and so on. The client device 310 may support and run various different operating systems, such as Microsoft® Windows®, Mac OS®, iOS®, Google® Chrome®, Linux®, Unix®, or any other mobile operating system, including Symbian®, Palm®, Windows Mobile®, Google® Android®, Mobile Linux®, and so on.

[0047] The client device 310 includes an interface 312, a processor 314, a storage 316, and one or more apps 316.

[0048] The interface 312 includes, for example, a touch screen, a keyboard, a camera, one or more sensors, or a combination thereof. In an embodiment, the client device 310 receives an input from a user via the interface 312. In a specific embodiment, the input specifies a hostname of a source of content. For example, the input specifies a universal resource locator (URL) address of a specific website on the internet.

[0049] The processor 314 executes program commands. The storage 316, for example, stores the program commands that are executed by the processor 312. In an embodiment, the storage 316 is a local memory.

[0050] The client device 310 runs the one or more apps 316. In an embodiment, the one or more apps 316 includes an internet browser application, a video streaming application, a video game app, etc.

[0051] In the invention, the client device 316 is configured to request an IP address of a content server associated with a known hostname and receive a DNS response identifying the requested IP address. The client

device 316 further requests content by transmitting a request to the content server using the IP address, and receives the requested content from the content server in the form of one or more data packets.

[0052] Fig. 3B illustrates a DNS spy 330 according to an embodiment of the present disclosure. The DNS spy 330 includes an interface 332, a processor 334, a first storage 336, and a second storage 390.

[0053] The second storage 390 stores an identification table 392 generated by the DNS spy 330. The identification table 392 includes a plurality of entries identifying a plurality of IP address/hostname pairs, respectively, according to an embodiment. Although the second storage 390 is illustrated inside of the DNS spy 330, the second storage 390 may be a storage device that is separate from the DNS spy 330, in an embodiment.

[0054] In various embodiments, the DNS spy 330 is located between one or more DNS servers and one or more client devices. DNS response transmitted from the one or more DNS servers and the one or more client devices pass through, or are intercepted by, the DNS spy 220. The DNS spy 220 reads the hostnames and the IP address from resource records (RRs) in the DNS responses, for example. The hostname and the IP address in the RRs of the DNS response are not encrypted.

[0055] The DNS spy 330 identifies a plurality of hostname/IP address pairs from the DNS responses. The DNS spy 220 stores the hostname/IP address pairs in the identification table 392 in the second storage 390. In an embodiment, the hostname/IP address pairs are included in respective entries in the identification table 392.

[0056] In various embodiments, the identification table 392 is accessible by other devices, such as a transport manager. The transport manager, for example, can manage network traffic by reading entries in the identification table 392.

[0057] Fig. 3C illustrates a DNS server 350 according to an embodiment of the present disclosure. The DNS server 350 includes a processor 352, a storage 354, and DNS records 358.

[0058] The processor 352 executes one or more policies 356 stored in the storage 354. For example, the processor 352 executes program commands stored in the storage 354.

[0059] When the DNS server 350 receives a DNS request including a hostname, the DNS server 350 searches the DNS records 358 for an IP address associated with the hostname. The IP address is an IP address for a content server associated with the hostname, for example. The DNS records 358 are structured such that the DNS server 350 can search for an IP address associated with a given hostname, but cannot search for a hostname associated with a given IP address.

[0060] The DNS server 350 then generates a DNS response including a plurality of RRs that include the hostname and the IP address. The DNS server 350 transmits the DNS response to the source of the DNS request. For example, when the DNS request is transmitted from a client device, the DNS server 350 transmits the DNS response to the client device.

[0061] Fig. 3D illustrates a transport manager 370 according to an embodiment of the present disclosure.

[0062] The transport manager 270 is configured to manage data traffic traversing a network when one or more conditions are satisfied. In some embodiments, the transport manager 370 is a delivery manager that directs or manages the delivery of content via a delivery policy that utilizes or uses surplus network bandwidth or surplus network capacity. A surplus of network bandwidth or network capacity may be network bandwidth or network capacity that is determined to be available (e.g., idle or free) in a network in view of the total capacity of the network and/or and the total usage of the network. In some embodiments, a network provider determines the amount of surplus network capacity available in a network in view of the total capacity of the network and/or and the total usage of the network. The surplus network capacity may be determined statically or dynamically, and, therefore, a determined surplus network capacity for a network may vary substantially and/or randomly over time (e.g., during peak use periods), for long or short time scales, and/or between one service provider to another.

[0063] The surplus capacity, therefore, may be the free bandwidth or capacity between an actual and/or current usage of the bandwidth a total capacity (or, a predetermined percentage of the total capacity). Therefore, the transport manager 370 may direct or manage the delivery of content between content providers (e.g., content servers), network edge caches, and client devices over various selected delivery policies or protocols that utilize free, available, idle, or otherwise surplus bandwidths or capacities of networks, such as paths or protocols that deliver data over currently underused networks that would not otherwise be in use, and/or without substantially impacting or altering the transport performance associated with other data traffic sharing the network.

[0064] Further details regarding the delivery of content using surplus network capacity may be found in commonly-assigned U.S. Pat. No. 7,500,010, issued on Mar. 3, 2009, entitled ADAPTIVE FILE DELIVERY SYSTEM AND METHOD, U.S. Pat. No. 8,589,585, issued on Nov. 19, 2013, entitled ADAPTIVE FILE DELIVERY SYSTEM AND METHOD, U.S. Published Patent Application No. 2010/0198943, filed on Apr. 15, 2010, entitled SYSTEM AND METHOD FOR PROGRESSIVE DOWNLOAD USING SURPLUS NETWORK CAPACITY, and U.S. Published Patent Application No. 2013/0124679, filed on Jan. 3, 2013, entitled SYSTEM AND METHOD FOR PROGRESSIVE DOWNLOAD WITH MINIMAL PLAY LATENCY.

[0065] The transport manager 370 includes an interface 372, a processor 374, a queue 376, a manager 378, and a storage 380.

[0066] The processor 374 executes one or more policies 382 stored in the storage 380. For example, the processor 374 executes program commands stored in the

storage 380. Various functions of the transport manager 370 are executed by the processor 374.

[0067] The transport manager 370 identifies characteristics of data flows traversing a network, and characteristics of the network itself, and manages the data flows based on the characteristics. In an embodiment, the transport manager 370 identifies a data flow for management by intercepting a data packet in the data flow, reading an IP address of a source of the data packet from the data packet, and identifying a hostname of the source of the data packet by accessing an identification table.

[0068] For example, when the transport manager 370 determines that a data flow traversing a network is an elephant flow based on a hostname of a source of the data flow, the transport manager 370 paces the data flow. In a specific example, the transport manager 370 temporarily stores data including packets of the elephant flow in the queue 376, and selectively releases the packets to their destination over surplus network capacity of a network, when the surplus network capacity is available.

[0069] Fig. 3E illustrates a content server 390 according to an embodiment of the present disclosure. The content server 390 includes an interface 392, a processor 394, and a storage 396.

[0070] The processor 394 executes one or more policies 392 stored in the storage 396. For example, the processor 394 executes program commands stored in the storage 396. Various functions of the content server 390 are executed by the processor 394.

[0071] The content server 390 receives a request for content from a source and transmits one or more of files 398 stored in the storage 396 in response to the request. The content server 390 transmits the one or more files 398 in the form of a plurality of data packets, for example.

[0072] The content server 390 may provide a variety of different media and other content types, such as video content (e.g., movies, television shows, news programming, video clips), image content (e.g., image or picture slideshows), audio content (e.g., radio programming, music, podcasts), and so on. The content server 390 may deliver, transfer, transport, and/or otherwise provide media files and other content to network edge caches (not shown), which may deliver, transfer, transport, and/or otherwise provide the content to requesting devices (e.g., user equipment 110 a-c) via various media transfer protocols (e.g., Hypertext Transfer Protocol (HTTP), File Transfer Protocol (FTP), HTTP Live Streaming (HLS), HTTP Dynamic Streaming (HDS), HTTP Smooth Streaming (HSS), Dynamic Adaptive Streaming over HTTP (DASH), Real Time Streaming Protocol (RTSP), and so on).

[0073] Fig. 4 illustrates a data packet 400 according to an embodiment of the present disclosure. The data packet 400 includes a header 410 and a payload 420.

[0074] The header 410 includes a source IP address 412 and a destination IP address 414. The source IP address 412 is an IP address of a source of the data packet 400, and the destination IP address 414 is an IP address of a destination of the data packet 400. The header 410 is not encrypted, according to an embodiment.

[0075] The payload 420 includes one or more files 422. The payload 420 includes, for example, application layer data. The payload 420 is encrypted.

[0076] In the invention, the data packet 400 is transmitted between a content server and a client device, and is intercepted by a transport manager.

[0077] Fig. 5 illustrates a DNS response 500 according to the invention.

[0078] The DNS response 500 includes a plurality of RRs 510. For example, the RRs 510 include an RDATA field 512 and a NAME field 514. The RDATA field 512 includes an IP address, and the NAME field 514 includes a hostname associated with the IP address.

[0079] In an embodiment, the DNS response 500 is transmitted from a DNS server to a client device. Furthermore, the DNS response 500 is not encrypted. A DNS spy, for example, intercepts the DNS response 500 transmitted from the DNS server to the client device, and reads the IP address from the RDATA field 512 and the hostname from the NAME field 514 without performing decryption.

[0080] Fig. 6 illustrates an identification table 600 according to the invention.

[0081] The identification table 600 includes a plurality of entries 610_1 to 610_N, where N is a size of the identification table 600. Each of the entries 610_1 to 610_N corresponds to a specific hostname/IP address pair. The identification table 600 is a cache table indexed by the IP addresses of the hostname/IP address pairs, in an embodiment.

[0082] Each of the entries 610_1 to 610_N includes an IP address, a hostname, a last active time of a data packet from the IP address, and an amount of cumulative bytes associated with data traffic transmitted from the IP address.

[0083] In this disclosure, fields like the last active time and the amount of cumulative bytes may be termed "characteristics" of the corresponding hostname/IP address pairs. In an embodiment, the characteristics are used by a DNS spy to identify one or more least significant entries in the identification table 600. The DNS spy may then prune the least significant entries from the identification table 600 in a pruning operation.

[0084] In some embodiments, the characteristics are used by a transport manager to identify hostname/IP address pairs associated with relatively burdensome data flows. For example, the transport manager identifies entries including relatively large amounts of cumulative bytes as likely to be associated with elephant flows.

[0085] According to various embodiments, each of the entries 610_1 to 610_N is generated and updated by a DNS spy and stored in a storage.

[0086] Fig. 7 illustrates a method 700 of extracting a hostname and IP address from a DNS response according to the invention.

[0087] The method 700 may be performed by a DNS spy, for example.

[0088] At S710, a packet is received. In an embodiment, the packet is intercepted as it is being transmitted from a DNS server to a client device.

[0089] The packet is identified as a DNS response at S720.

[0090] After the packet is identified as a DNS response, a hostname and an IP address is extracted from the DNS response at S730. The hostname and the IP address are read from RRs in the DNS response. For example, the hostname is read from a NAME field in the resource records, and the IP address is read from an RDATA field in the resource records

[0091] Fig. 8 illustrates a method 800 of adding a new entry to an identification table according to an embodiment of the present disclosure. In an embodiment, the method 800 is performed by a DNS spy.

[0092] A hostname and IP address pair are determined at S810. For example, the hostname and the IP address are read from a DNS response.

[0093] Fig. 9 illustrates a method 900 of determining a length of a non-DNS packet according to an embodiment of the present disclosure. The method 900 is performed by a DNS spy, for example.

[0094] At S910, a data packet is received. In an embodiment, the data packet is intercepted while being transmitted between a content server and a client device.

[0095] The data packet is identified as a non-DNS packet at S920. The data packet is not a DNS request or a DNS response. The non-DNS packet is, for example, a video streaming packet.

[0096] At S930, a size of the data packet is determined. For example, a number of bytes contained in the packet is determined.

[0097] Fig. 10 illustrates a method 1000 of determining a timestamp of a non-DNS packet according to an embodiment of the present disclosure. The method 1000 is performed by a DNS spy, for example.

[0098] At S1010, a data packet is received. In an embodiment, the data packet is intercepted while being transmitted between a content server and a client device.

[0099] The data packet is identified as a non-DNS packet at S1020. The data packet is not a DNS request or a DNS response. The non-DNS packet is, for example, a video streaming packet.

[0100] Next, at S1030, a timestamp associated with the packet is determined. For example, the timestamp is a time that the data packet is received. In another example, the timestamp is derived from the non-DNS packet directly.

[0101] Fig. 11 illustrates a method 1100 of updating an entry of an identification table according to an embodiment of the present disclosure. The method 1100 is performed by a DNS spy, for example.

[0102] At S1110, an IP address and a characteristic of a non-DNS packet are determined. In an embodiment, the non-DNS packet is intercepted between a content server and a client device. In an example, the IP address is an IP address of a source of the data packet, and is determined by reading the IP address from a non-encrypted header of the non-DNS packet. According to an embodiment, the characteristic is a size of the data packet, a timestamp of the data packet, or a combination thereof.

[0103] At S1120, an entry in an identification table is identified. The entry includes the IP address and a hostname associated with the IP address, for example.

[0104] The entry is updated based on the characteristic at S1130. For example, when the entry includes an IP address field, a hostname field, a last active time field, and a number of cumulative bytes field, the last active time field is updated based on the timestamp of the data packet, and the number of cumulative bytes field is updated based on the size of the data packet.

[0105] Fig. 12 illustrates a method 1200 of managing a data flow using an identification table according to an embodiment of the present disclosure. The method 1200 is performed by a transport manager, for example.

[0106] At S1210, a packet in a data flow is received. The packet is, for example, a data packet including a header and a payload. The header is unencrypted. The payload is encrypted, and includes application layer data.

[0107] At S1220, an IP address is identified from the header of the packet. The IP address indicates a destination of the packet. In an embodiment, an IP address of a source of the packet is derived by reading the header of the packet. The IP address is an IP address of a content server, for example.

[0108] At S1230, a hostname associated with the identified IP address is determined by accessing an identification table. The identification table includes a plurality of IP address/hostname pairs in a plurality of entries. In an embodiment, the identification table is a cache table that is indexed by the IP addresses. Accordingly, the hostname associated with the identified IP address can be determined by identifying an entry in the identification table including the identified IP address, and reading the corresponding hostname in the identified entry.

[0109] The data flow is managed based on the identified hostname at S1240. For example, the data flow is paced when the identified hostname is determined to correspond to a host that has been historically associated with data traffic that is burdensome to a network. In a specific example, the data flow is identified as an elephant flow based on the identified hostname, and is paced.

[0110] Fig. 13 illustrates a method 1300 of identifying and pacing an elephant flow using an identification table according to an embodiment of the present disclosure. The method 1300 is performed by a transport manager, for example.

[0111] At S 1310, a packet in a data flow is received. The packet is a non-DNS data packet, for example. In an embodiment, a copy of the packet is temporarily cached, and the packet itself is released, so that it can

reach its destination without any significant delays.

[0112] At S1320, an IP address is identified from the header of the packet. In an embodiment, the IP address indicates a destination of the packet or a source of the packet. In various embodiments, an IP address indicating the destination of the packet and an IP address indicating the source of the packet are both determined. The IP address is, for example, an IP address of a content server.

[0113] A hostname associated with the identified IP address is determined at S1330 by accessing an identification table. The identification table stores a plurality of previously identified hostname/IP address pairs, and is indexed by IP address. In an embodiment, the hostname is retrieved from an entry including the identified IP address.

[0114] At S 1340, the data flow is determined to be an elephant flow based on the hostname. For example, the data flow is identified as an elephant flow when the hostname indicates a known video streaming host.

[0115] After the data flow is determined to be an elephant flow, the data flow is paced at S1350. In an embodiment, a plurality of data packets in the data flow traversing a network are managed after the data flow is determined to be an elephant flow. For example, the data packets of the data flow may be selectively routed over surplus network capacity. In a specific embodiment, the data packets of that data flow is selectively routed over a different network when the different network is available, such as a WIFI network that is connected to a client device receiving data packets in the data flow.

[0116] Fig. 14 illustrates a method 1400 of controlling a size of an identification table according to an embodiment of the present disclosure. The method 1400 is performed by a DNS spy, for example.

[0117] At S1410, a new entry is added to an identification table. The new entry includes a hostname/IP address pair. For example, the new entry includes a hostname/IP address pair that was not previously included in the identification table.

[0118] At S1420, a size of the identification table is determined to exceed a predetermined size. In various embodiments, the size is defined as a number of indices, an amount of memory required to store the identification table, or a combination thereof.

[0119] At S1430, one or more entries of the identification table are pruned from the identification table. For example, the one or more entries are deleted from the identification table.

[0120] The one or more pruned entries can be identified in various ways. In some embodiments, the one or more entries are the least significant entries of the identification table. The least significant entries can be identified based on characteristics fields of the identification table. For example, if each of the entries includes a timestamp indicating the most recent observed data packet corresponding to the hostname/IP address pairs, the entries with the oldest timestamps are pruned from the identification table. In another example, if each of the entries includes a number of transferred bytes observed to correspond to hostname/IP address pairs, the entries with the lowest number of transferred bytes are pruned from the identification table. In some embodiments, the one or more entries are identified based on a combination of multiple criteria.

[0121] A number of the one or more pruned entries can also be identified in various ways. For example, when the identification table reaches its maximum size, a predetermined percentage of entries are deleted from the identification table.

[0122] Fig. 15 illustrates a method 1500 of generating, using, and updating an identification table according to an embodiment of the present disclosure.

[0123] At S1510, a packet is received. For example, the packet is intercepted as it is being transmitted to a client device.

[0124] At S1512 and S1520, the packet is inspected and identified as either a DNS response, a TCP or UDP payload packet, or neither.

[0125] When the packet is a DNS response, the packet is then parsed and a hostname and one or more IP addresses are extracted from the DNS response at S1530. According to an embodiment, the extracted data is enqueued for an independent process to analyze after the packet is released, to limit the amount of time the packet is held by the system and eliminate any impact on client/server communication latency.

[0126] When the packet is not a DNS response, the packet is examined to determine whether the packet is a non-DNS packet, such as a transmission control protocol (TCP) or user datagram protocol (UDP) packet, at S1520.

[0127] When the packet is a TCP/UDP packet, a length of a payload of the packet is examined at S1524. The packet is enqueued and released, so that the payload length is examined in an independent process to limit the amount of time the packet is held by the system and eliminate any impact on client/server communication latency. For example, the packet is enqueued and treated similarly as the independent process performed at S1330, so that the payload length is determined and analyzed after the packet is released.

[0128] If the packet is neither a DNS response or a TCP/UDP packet, the packet is released at S1522.

[0129] At S1532, the presence of the extracted IP address in an identification table is determined. For example, each of the enqueued items are read and compared to a plurality of entries in the identification table.

[0130] In the case when no existing table entry including the extracted IP address exists, identification table is determined to be full or not at S1540.

[0131] When the identification table is not full, a new table entry is created at S1542. According to various embodiments, the creation of new table entries effectively builds the identification table as a DNS cache, in which various hostname/IP address pairs are mapped to each

other.

[0132] In the case where an entry does exist, the entry is updated based on the packet at S1534. The table entry, in some examples, includes the following fields: IP address, domain name, last active time, and cumulative bytes. The entries are stored in the identification table. The identification table is, for example, a hash table, indexed by the IP address.

[0133] When the identification table is full, one or more least significant entries in the identification table are pruned at S1544. That is, the hash table has a configurable maximum length, so that the identification table does not exceed a certain size. When the identification table has the maximum length, the identification table may be pruned to a percentage of its size, in an embodiment. In another embodiment, a predetermined number of entries are pruned. The entries are pruned by being deleted from the identification table, for example.

[0134] In various embodiments, the least significant entries are pruned from the identification table. In an embodiment, the least significant entries are entries including the lowest number of cumulative bytes recorded, the earliest last active time recorded, or a combination thereof. For example, a percentage of the entries corresponding to the lowest number of cumulative bytes are removed first, and in the case of a tie, the entries corresponding to the earliest Last Active Time are removed.

[0135] When it is determined that one or more entries are removed at S1546, the method 1500 returns to S 1542, such that a new entry is generated in the identification table based on the packet without exceeding the maximum size of the identification table.

[0136] In various embodiments, data traffic through a network is managed by accessing the identification table. The identification table is accessed by querying by IP address, which will allow another system (or system with DNS Spy integration) to obtain the domain name of said IP address. In one version, this data is made available to a statistics collection system, which activates the collection of granular statistics on all traffic with a specific domain. In turn, these statistics can be used to determine the top n domains on a specified network.

[0137] Some examples of the granular statistical data that can be collected on a specific domain include (but are not limited to): Total Upstream/Downstream Bytes, Latency, Throughput, Goodput, Radio access type, Congestion Ratio, Bytes Transferred in flows that met a specified cumulative transferred byte threshold (standard flows, elephant flows, etc.), Stats broken down by time of day and or by location, or a combination thereof.

[0138] Embodiments provide a system and method for delivering packet data content across shared access networks in a manner that more evenly distributes aggregate user traffic in time, for example by moving traffic from times of bottleneck network congestion into following adjacent moments of surplus network capacity. The net effect of this redistribution of traffic may reduce intervals of peak usage and congestion (where the network is unable to supply enough throughput capacity for all users), which can result in higher allowed aggregate network utilization before network service quality degrades for the users.

[0139] The term "surplus network capacity" (aka "idle capacity") is understood in some embodiments to mean shared network capacity (e.g. network bandwidth, network resources) that may be used by embodiments of the invention for transferring portions or all of the data over a network, but is otherwise unused. In other words, if the network capacity is X and the current aggregate network traffic load is Y then the available surplus capacity is X-Y where Y cannot be larger than X. In an embodiment, one of the goals of using surplus network capacity is to use some or all the capacity Y to transfer data which implies that if Y is zero that the transfer slows or stops and yields the channel to other users' traffic sharing the network. In some scenarios, surplus network capacity in shared multi-user data networks is transient and can fluctuate randomly from moment to moment. Further, use of surplus as defined may be distinct from a fair-share or similar competitive shared use of network capacity (e.g. when the aggregate traffic load exceeds the network capacity limit X then each of N users sharing the network receives a X/N share of the network capacity).

[0140] In a specific embodiment, the system may uniquely identify a data traffic flow based on its associated client IP address and destination server IP address. The system may then characterize the traffic as a large flow based on parameters, such as throughput, bytes delivered, other characteristics, or a combination thereof. The system is a transport manager, for example.

[0141] The system may query the identification table generated by the DNS Spy, and determine that the hostname that the server IP address is associated with belongs to a known video provider (e.g., YOUTUBE). Thus, one conclusion could be made that the data flow is in fact a video due to its size and known destination.

[0142] This information may be made available to a transport manager, which enables specified management rules to be applied to a specified subset of traffic (such as a video in this example) that may or may not belong to a specific domain (such as YOUTUBE in this example). Rules may include policies that ensure that the managed traffic yields shared bandwidth to other traffic (i.e. uses surplus network capacity), is otherwise prioritized, or is marked to not apply any specific management.

[0143] According to various embodiments, the DNS Spy operates with a number of characteristics, such as 1) the separation of the DNS hostname/IP address resolution process from traffic characterization/management/statistics processes, 2) its ability to perform granular statistics collection on encrypted traffic flows, 3) the creation of a dynamic DNS cache that adapts to changing hostname/IP mappings in real time without requiring external manipulation, and 4) the application of the hostname/IP address mapping toward traffic management to

improve network efficiencies.

[0144] According to an embodiment, a system builds a virtual DNS cache for its own internal use, but does not act as a virtual DNS server. The system intercepts DNS response packets sent across a network, extracts information from the DNS response packets, builds a table based on the extracted information, but does not modify or manipulate the DNS response packets sent across the network.

[0145] Embodiments of the present disclosure could be used to identify and track traffic to and from specific hostnames. In one version, the collected domain name information could be provided to a system that collects statistics on subset of traffic, single flows of traffic, or both. In another version, the information could be provided to a system that manages content delivery to and from UE terminals, which would apply specified management policies based on the domain name.

[0146] The system and methods comprising various embodiments of the present disclosure are used to analyze and manage data traffic between various client devices and content servers, but do not require modification to a user app installed on the client devices that requests content, or to a content server providing content, which enables rapid deployment into commercial networks, such as mobile cellular networks.

[0147] Various embodiments of the present disclosure address multiple technological problems associated with content delivery, wireless networks, security, and other technological fields.

[0148] For example, by generating an identification table that maps IP addresses to hostnames, network traffic can be effectively analyzed and data flows can be identified for management without performing computationally expensive decryption operations.

[0149] For example, by selectively managing data flows traversing a network based on hostnames associated with the data flows, network resources can be efficiently conserved and network congestion can be efficiently prevented.

[0150] For example, by selectively pruning the least significant entries of the identification table, the size of the identification table does not become unmanageable, but the identification table nevertheless maintains records of IP addresses and hostnames that are likely to be relevant for managing new data flows traversing the network.

Claims

1. A method, comprising:

intercepting a first data packet being transmitted from a domain name system, DNS, server to a first client device, the first data packet including a DNS response (S1512);
extracting (S1530) a first internet protocol (IP) address and a first hostname from the DNS response;
creating a first entry in an identification table (S 1542), each entry of the identification table includes an IP address, a hostname, a last active time field, and a number of cumulative bytes field (600);
storing the first IP address and the first hostname in the first entry of the identification table (S1542);
intercepting a second data packet being transmitted in a data flow from a content server to a second client device (S1210);
identifying a second IP address in a header of the second data packet, the second IP address being a source of the second data packet (S1220);
determining that the second IP address is in the first entry in response to the first IP address stored in the first entry being identical to the second IP address (S 1230);
in response to determining that the second IP address is in the first entry:

determining, based on the first hostname of the first entry, whether a traffic management policy should be applied to the data flow (S1240), and
in response to determining that the data traffic management policy should be applied, applying the traffic management policy to the data flow to deliver the second data packet to the second client device (S1240);
characterized by:

when the identification table exceeds a predetermined size (S1420), identifying a second entry of the identification table having the number of cumulative bytes field with a lowest value, and removing the second entry from the identification table (S1430),
in response to determining that the second IP address is in the first entry, and

determining a characteristic of the second data packet, the characteristic being an amount of bytes in the second data packet and a timestamp indicating a time of the second data packet (S1110) being received,
updating a last active time field of the first entry according to the timestamp of the characteristic (S1130), and
updating a number of cumulative bytes field of the first entry according the amount of bytes of the characteristic (S1130).

2. The method of claim 1, wherein identifying the second entry of the identification table includes when a

plurality of entries in the identification table have respective number of cumulative bytes fields with the lowest value, identifying as the second entry an entry having the earliest last active time recorded among the plurality of entries having respective number of cumulative bytes fields with the lowest value (S1544).

3. The method of claim 1, wherein managing the data flow includes causing the data flow to be transferred to the second user device over surplus network capacity of a network, the surplus network capacity being an available capacity of the network in view of a total capacity of the network and a total usage of the network.

4. The method of claim 1, wherein a payload of the second data is encrypted, and
wherein the second IP address is extracted from the header without performing decryption.

5. The method of claim 1, wherein extracting the IP address and the hostname from the DNS response includes reading the IP address and the hostname from resource records (RRs) in the DNS response.

6. The method of claim 5, wherein reading the IP address and the hostname from RRs in the DNS response includes reading the IP address in an 'RDATA' field of the DNS response and reading the hostname in a 'NAME' field of the DNS response.

7. A system, comprising:
a processor; and
a memory storing program commands that, when executed by the processor, cause the first processor to carry out the method of claim 1.

8. A computer-readable storage medium comprising instructions which, when executed by a computer, cause the computer to carry out the method of claim 1.

Patentansprüche

1. Ein Verfahren, umfassend:

Abfangen eines ersten Datenpakets, das von einem Domain Name System, DNS, Server zu einem ersten Client-Gerät übertragen wird, wobei das erste Datenpaket eine DNS-Antwort (S1512) enthält;
Extrahieren (S1530) einer ersten Internetprotokoll (IP)-Adresse und eines ersten Hostnamen aus der DNS-Antwort;
Erzeugen eines ersten Eintrags in einer Identifikationstabelle (S1542), wobei jeder Eintrag der Identifikationstabelle eine IP-Adresse, einen Hostnamen, ein Feld für die letzte aktive Zeit und ein Feld für die Anzahl der kumulativen Bytes (600) enthält;
Speichern der ersten IP-Adresse und des ersten Hostnamens in dem ersten Eintrag der Identifikationstabelle (S1542);
Abfangen eines zweiten Datenpakets, das in einem Datenfluss von einem Inhaltsserver zu einem zweiten Client-Gerät (S1210) übertragen wird;
Identifizieren einer zweiten IP-Adresse in einem Header des zweiten Datenpakets, wobei die zweite IP-Adresse eine Quelle des zweiten Datenpakets (S1220) ist;
Bestimmen, dass die zweite IP-Adresse in dem ersten Eintrag ist, in Reaktion darauf, dass die erste IP-Adresse, die in dem ersten Eintrag gespeichert ist, mit der zweiten IP-Adresse (S1230) identisch ist;
als Reaktion auf die Feststellung, dass die zweite IP-Adresse in dem ersten Eintrag ist:

Bestimmen, basierend auf dem ersten Hostnamen des ersten Eintrags, ob eine Verkehrsverwaltungsrichtlinie auf den Datenfluss (S1240) angewendet werden soll, und als Reaktion auf die Feststellung, dass die Datenverkehrsverwaltungsrichtlinie angewendet werden soll,
Anwenden der Verkehrsverwaltungsrichtlinie auf den Datenfluss, um das zweite Datenpaket an das zweiten Client-Gerät (S1240) zu übertragen;
gekennzeichnet durch:

wenn die Identifikationstabelle eine vorbestimmte Größe (S1420) überschreitet, Identifizieren eines zweiten Eintrags der Identifikationstabelle, aufweisend das Feld für die Anzahl der kumulativen Bytes mit einem niedrigsten Wert, und
Entfernen des zweiten Eintrags aus der Identifikationstabelle (S1430), als Reaktion auf die Feststellung, dass die zweite IP-Adresse in dem ersten Eintrag ist, und
Bestimmen einer Eigenschaft des zweiten Datenpakets, wobei die Eigenschaft eine Menge von Bytes in dem zweiten Datenpaket und ein Zeitstempel ist, der eine Zeit des Empfangs des zweiten Datenpakets (S1110) anzeigt,
Aktualisieren eines Felds für die letzte Aktivität des ersten Eintrags gemäß dem Zeitstempel der Eigenschaft (S1130), und

Aktualisieren eines Feldes für die Anzahl der kumulativen Bytes des ersten Eintrags entsprechend der Menge der Bytes der Eigenschaft (S1130).

2. Verfahren nach Anspruch 1, wobei, wenn eine Vielzahl von Einträgen in der Identifikationstabelle jeweils Felder für die Anzahl der kumulativen Bytes mit einem niedrigsten Wert aufweisen, das Identifizieren des zweiten Eintrags der Identifikationstabelle das Identifizieren des zweiten Eintrags als den Eintrag, der den frühesten aufgezeichneten Zeitpunkt der letzten Aktivität unter der Vielzahl von Einträgen, die jeweils Felder für die Anzahl der kumulativen Bytes mit dem niedrigsten Wert (S1544) aufweisen, aufweist, umfasst.

3. Verfahren nach Anspruch 1, wobei das Verwalten des Datenflusses das Veranlassen des Datenflusses umfasst, über eine überschüssige Netzwerkkapazität eines Netzwerks an das zweite Benutzergerät übertragen zu werden, wobei die überschüssige Netzkapazität eine verfügbare Kapazität des Netzes im Hinblick auf eine Gesamtkapazität des Netzes und einer Gesamtnutzung des Netzes ist.

4. Verfahren nach Anspruch 1, wobei eine Nutzlast der zweiten Daten verschlüsselt ist und wobei die zweite IP-Adresse aus dem Header extrahiert wird, ohne eine Entschlüsselung durchzuführen.

5. Verfahren nach Anspruch 1, wobei das Extrahieren der IP-Adresse und des Hostnamens aus der DNS-Antwort das Auslesen der IP-Adresse und des Hostnamens aus Resource Einträgen (RRs) in der DNS-Antwort umfasst.

6. Verfahren nach Anspruch 5, wobei das Auslesen der IP-Adresse und des Hostnamens aus RRs aus der DNS-Antwort das Auslesen der IP-Adresse aus einem 'RDATA'-Feld der DNS-Antwort und das Auslesen des Hostnamens aus einem "NAME"-Feld der DNS-Antwort umfasst.

7. Ein System, das Folgendes umfasst: einen Prozessor; und einen Speicher, der Programmbefehle speichert, die, wenn sie vom Prozessor ausgeführt werden, den ersten Prozessor veranlassen, das Verfahren nach Anspruch 1 auszuführen.

8. Computerlesbares Speichermedium mit Befehlen, die bei Ausführung durch einen Computer den Computer veranlassen, das Verfahren nach Anspruch 1 auszuführen.

Revendications

1. Procédé, comprenant les étapes ci-dessous consistant à :

intercepter un premier paquet de données transmis d'un serveur de système de noms de domaine (DNS) à un premier dispositif client, le premier paquet de données incluant une réponse de système DNS (S1512) ;
extraire (S1530) une première adresse de protocole Internet (IP) et un premier nom d'hôte à partir de la réponse de système DNS ;
créer une première entrée dans une table d'identification (S1542), chaque entrée de la table d'identification inclut une adresse de protocole IP, un nom d'hôte, un champ de dernière heure d'activité et un champ de nombre d'octets cumulés (600) ;
stocker la première adresse de protocole IP et le premier nom d'hôte dans la première entrée de la table d'identification (S1542) ;
intercepter un second paquet de données transmis dans un flux de données d'un serveur de contenu à un second dispositif client (S1210) ;
identifier une seconde adresse de protocole IP dans un en-tête du second paquet de données, la seconde adresse de protocole IP étant une source du second paquet de données (S1220) ;
déterminer que la seconde adresse de protocole IP se situe dans la première entrée en réponse au fait que la première adresse de protocole IP stockée dans la première entrée est identique à la seconde adresse de protocole IP (S1230) ;
en réponse à une détermination selon laquelle la seconde adresse de protocole IP se situe dans la première entrée :

déterminer, sur la base du premier nom d'hôte de la première entrée, si une politique de gestion de trafic doit être appliquée au flux de données (S1240) ; et
en réponse à une détermination selon laquelle la politique de gestion de trafic doit être appliquée, appliquer la politique de gestion de trafic au flux de données pour livrer le second paquet de données au second dispositif client (S1240) ; **caractérisé par** les étapes ci-dessous consistant à :

lorsque la table d'identification dépasse une taille prédéterminée (S1420), identifier une seconde entrée de la table d'identification présentant le champ de nombre d'octets cumulés avec une valeur la plus faible, et supprimer la seconde entrée de la table d'identification (S1430) ;

en réponse à une détermination selon laquelle la seconde adresse de protocole IP se situe dans la première entrée :

déterminer une caractéristique du second paquet de données, la caractéristique étant une quantité d'octets dans le second paquet de données, et une estampille temporelle indiquant une heure de réception du second paquet de données (S1110) ;
mettre à jour un champ de dernière heure d'activité de la première entrée selon l'estampille temporelle de la caractéristique (S1130) ; et
mettre à jour un champ de nombre d'octets cumulés de la première entrée selon la quantité d'octets de la caractéristique (S1130).

2. Procédé selon la revendication 1, dans lequel l'étape d'identification de la seconde entrée de la table d'identification inclut l'étape consistant à, lorsqu'une pluralité d'entrées dans la table d'identification présente des champs respectifs de nombre d'octets cumulés présentant la valeur la plus faible, identifier, en tant que la seconde entrée, une entrée présentant la dernière heure d'activité la plus ancienne enregistrée parmi la pluralité d'entrées présentant des champs respectifs de nombre d'octets cumulés présentant la valeur la plus faible (S1544).

3. Procédé selon la revendication 1, dans lequel l'étape de gestion du flux de données inclut l'étape consistant à amener le flux de données à être transféré au second dispositif d'utilisateur sur une capacité de réseau excédentaire d'un réseau, la capacité de réseau excédentaire étant une capacité disponible du réseau compte tenu d'une capacité totale du réseau et d'une utilisation totale du réseau.

4. Procédé selon la revendication 1, dans lequel une charge utile des secondes données est chiffrée ; et dans lequel la seconde adresse de protocole IP est extraite de l'en-tête sans mettre en oeuvre un déchiffrement.

5. Procédé selon la revendication 1, dans lequel l'étape d'extraction de l'adresse de protocole IP et du nom d'hôte de la réponse de système DNS inclut l'étape consistant à lire l'adresse de protocole IP et le nom d'hôte à partir d'enregistrements de ressources (RR) dans la réponse de système DNS.

6. Procédé selon la revendication 5, dans lequel l'étape de lecture de l'adresse de protocole IP et du nom d'hôte à partir d'enregistrements RR dans la réponse de système DNS consiste à lire l'adresse de protocole IP dans un champ « RDATA » de la réponse de système DNS et à lire le nom d'hôte dans un champ « NAME » de la réponse de système DNS.

7. Système, comprenant :

un processeur ; et
une mémoire stockant des instructions de programme qui, lorsqu'elles sont exécutées par le processeur, amènent le premier processeur à mettre en oeuvre le procédé selon la revendication 1.

8. Support de stockage lisible par ordinateur comprenant des instructions qui, lorsqu'elles sont exécutées par un ordinateur, amènent l'ordinateur à mettre en oeuvre le procédé selon la revendication 1.

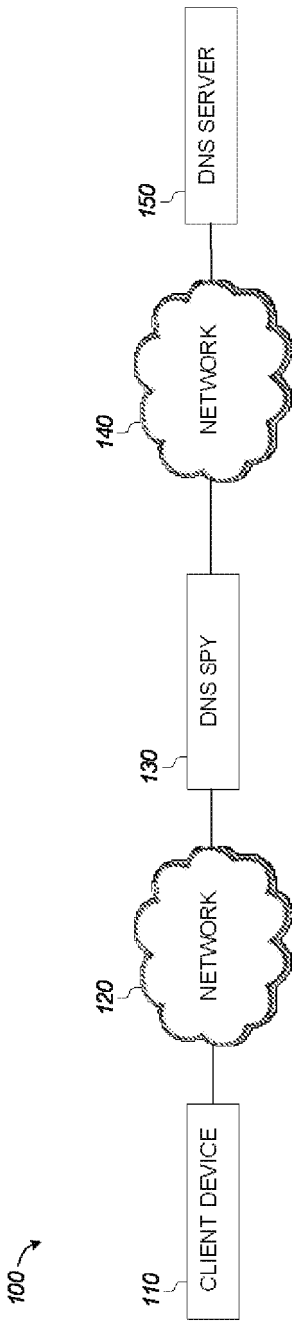
100
110
CLIENT DEVICE
120
NETWORK
130
DNS SPY
140
NETWORK
150
DNS SERVER

Fig. 1

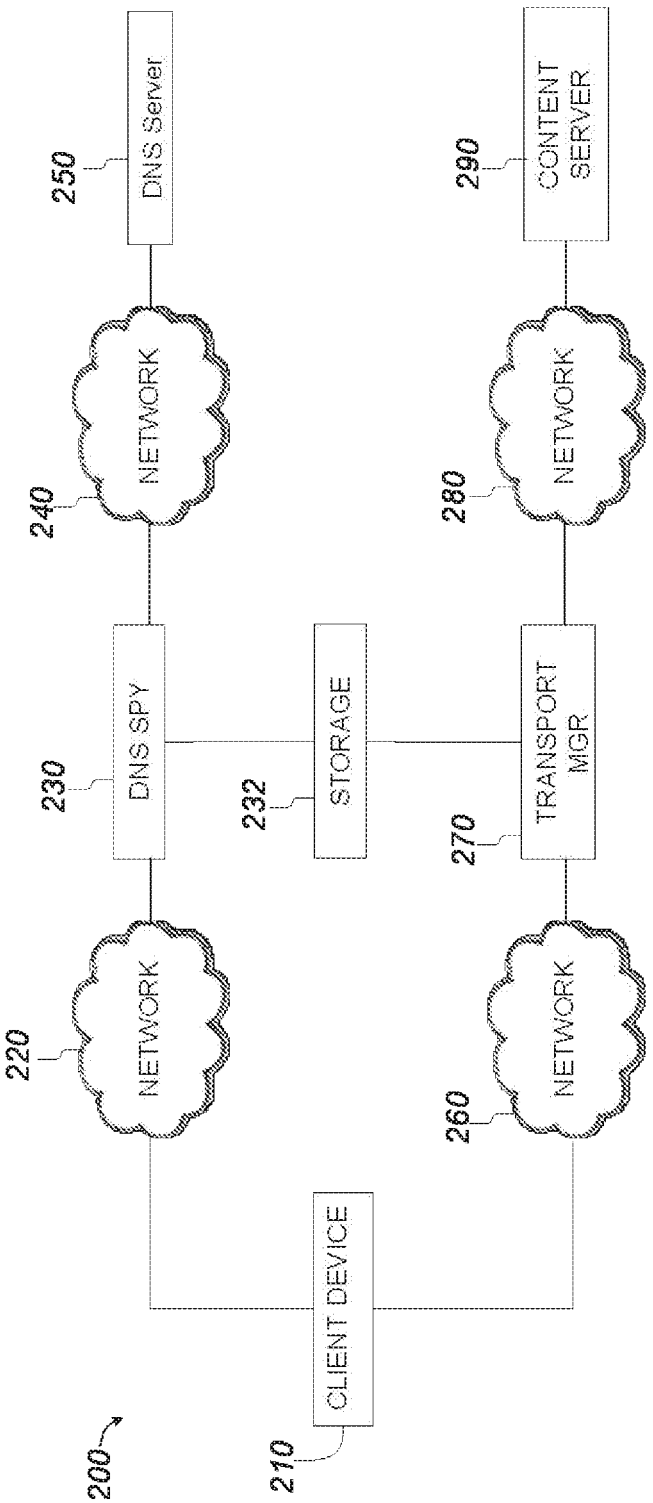
200
210
CLIENT DEVICE
220
NETWORK
230
DNS SPY
232
STORAGE
240
NETWORK
250
DNS Server
260
NETWORK
270
TRANSPORT MGR
280
NETWORK
290
CONTENT SERVER

Fig. 2

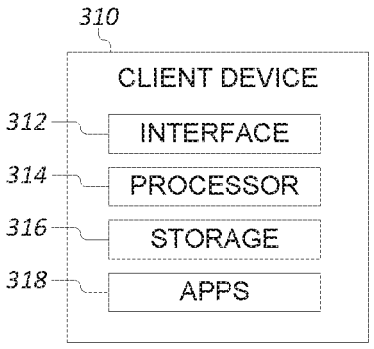

Fig. 3A

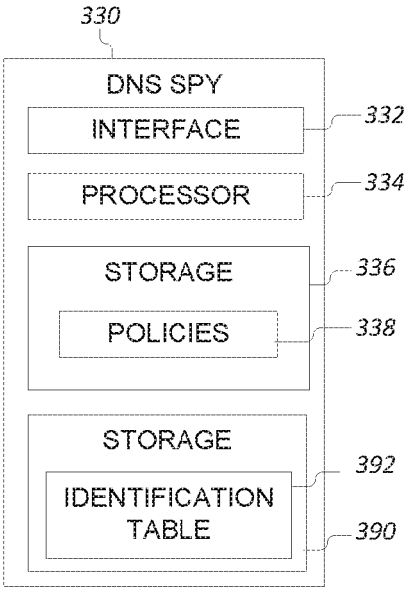

Fig. 3B

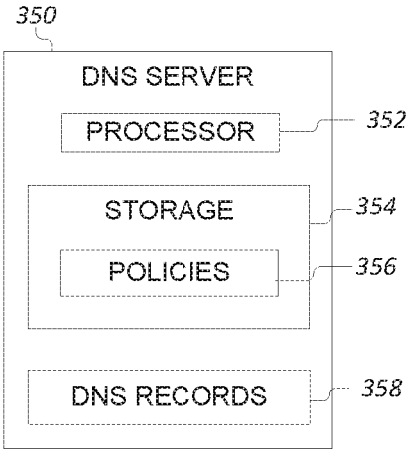

Fig. 3C

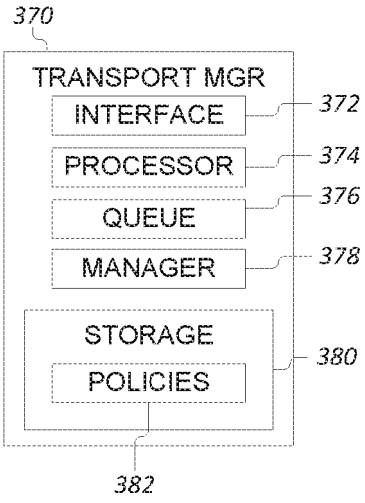
370
TRANSPORT MGR
INTERFACE
372
PROCESSOR
374
QUEUE
376
MANAGER
378
STORAGE
POLICIES
380
382

Fig. 3D

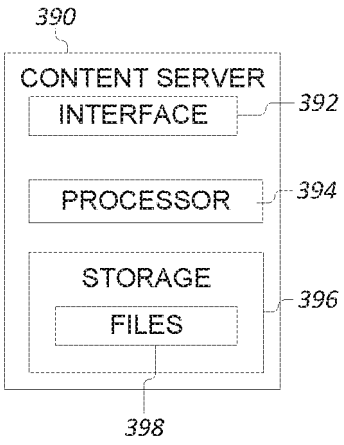
390
CONTENT SERVER
INTERFACE
392
PROCESSOR
394
STORAGE
FILES
396
398

Fig. 3E

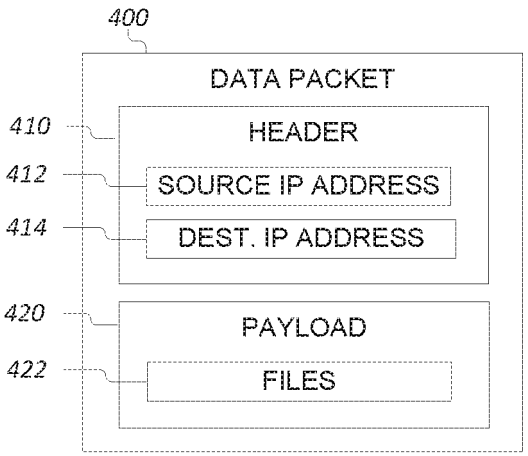
400
DATA PACKET
410
HEADER
412
SOURCE IP ADDRESS
414
DEST. IP ADDRESS
420
PAYLOAD
422
FILES

Fig. 4

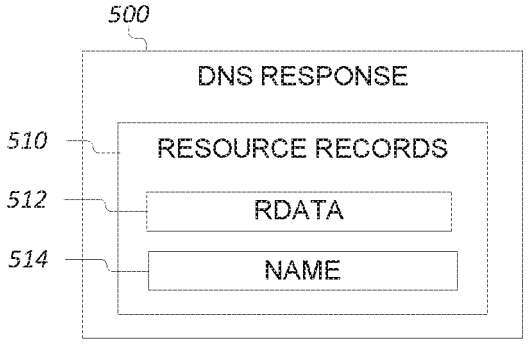
500
DNS RESPONSE
510
RESOURCE RECORDS
512
RDATA
514
NAME

Fig. 5

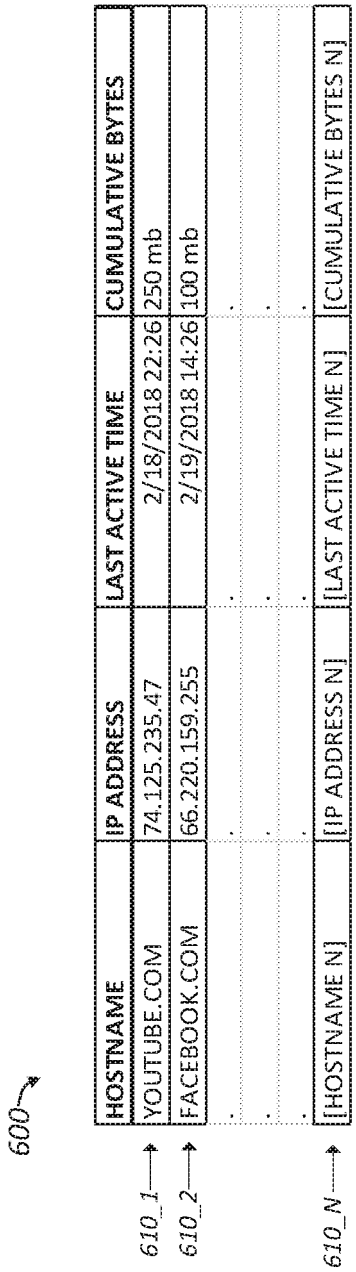
600
610_1
610_2
610_N
HOSTNAME
IP ADDRESS
LAST ACTIVE TIME
CUMULATIVE BYTES
YOUTUBE.COM
74.125.235.47
2/18/2018 22:26
250 mb
FACEBOOK.COM
66.220.159.255
2/19/2018 14:26
100 mb
[HOSTNAME N]
[IP ADDRESS N]
[LAST ACTIVE TIME N]
[CUMULATIVE BYTES N]

Fig. 6

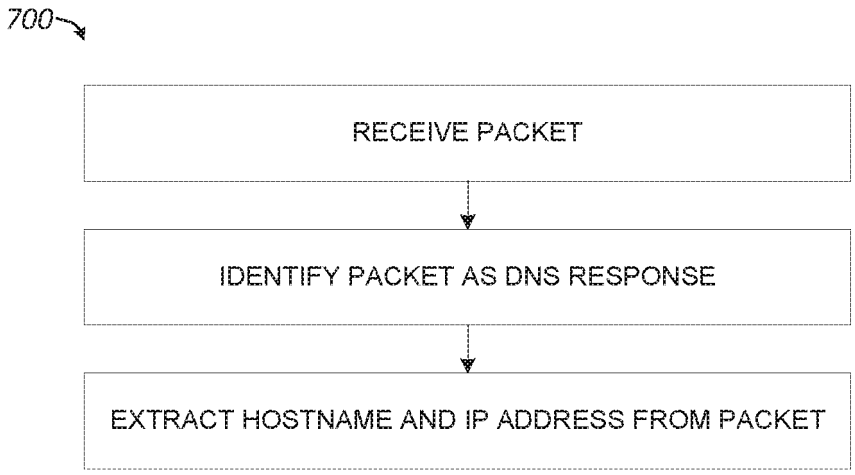
700
RECEIVE PACKET
IDENTIFY PACKET AS DNS RESPONSE
EXTRACT HOSTNAME AND IP ADDRESS FROM PACKET

Fig. 7

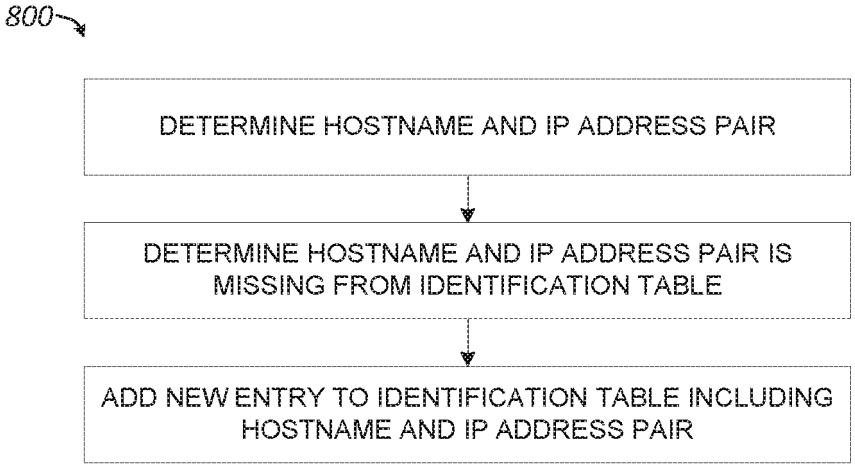
800
DETERMINE HOSTNAME AND IP ADDRESS PAIR
DETERMINE HOSTNAME AND IP ADDRESS PAIR IS MISSING FROM IDENTIFICATION TABLE
ADD NEW ENTRY TO IDENTIFICATION TABLE INCLUDING HOSTNAME AND IP ADDRESS PAIR

Fig. 8

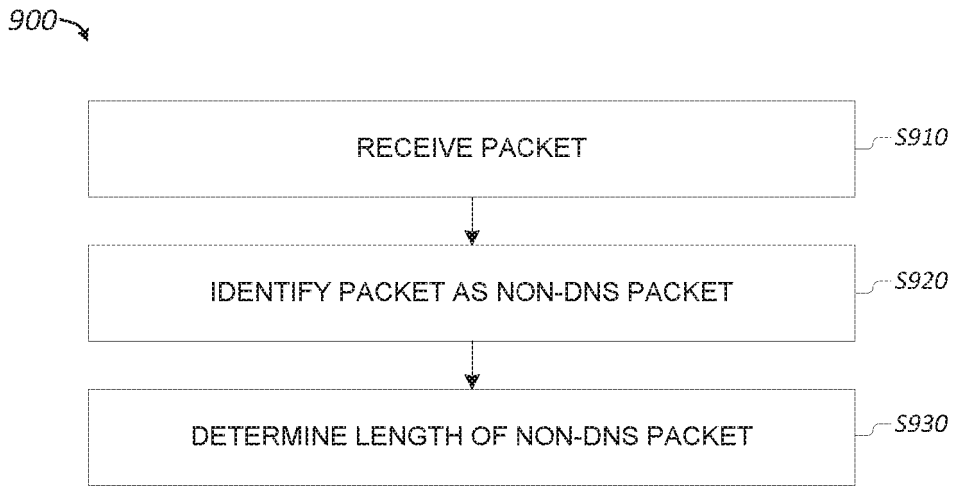
900
RECEIVE PACKET
S910
IDENTIFY PACKET AS NON-DNS PACKET
S920
DETERMINE LENGTH OF NON-DNS PACKET
S930

Fig. 9

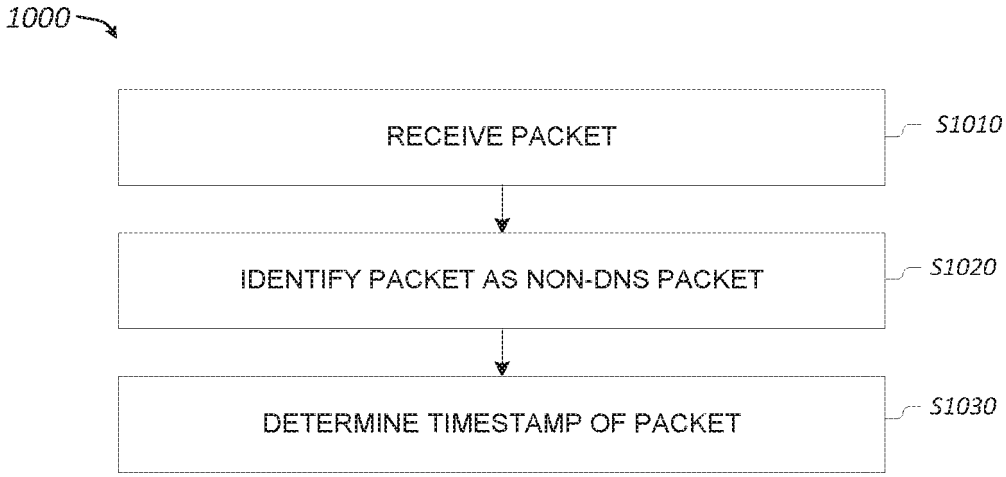
1000
RECEIVE PACKET
S1010
IDENTIFY PACKET AS NON-DNS PACKET
S1020
DETERMINE TIMESTAMP OF PACKET
S1030

Fig. 10

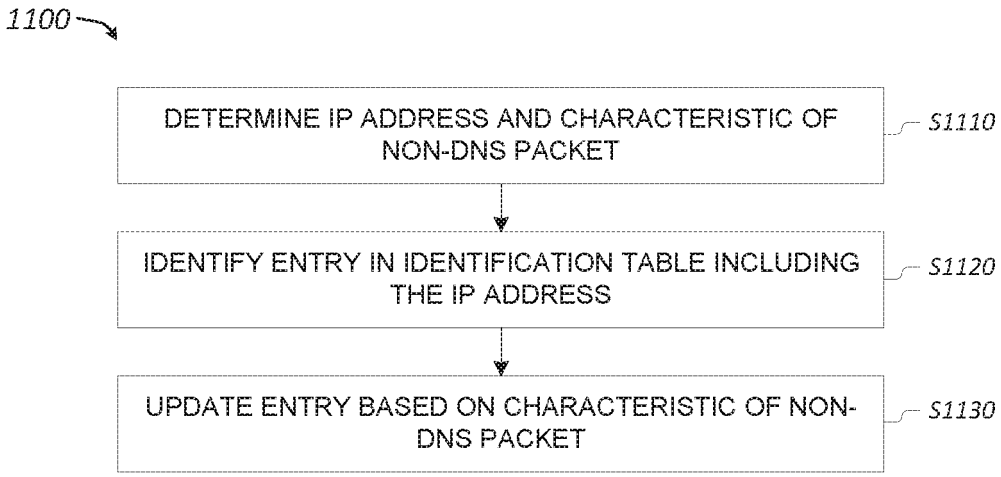
1100
DETERMINE IP ADDRESS AND CHARACTERISTIC OF NON-DNS PACKET
S1110
IDENTIFY ENTRY IN IDENTIFICATION TABLE INCLUDING THE IP ADDRESS
S1120
UPDATE ENTRY BASED ON CHARACTERISTIC OF NON-DNS PACKET
S1130

Fig. 11

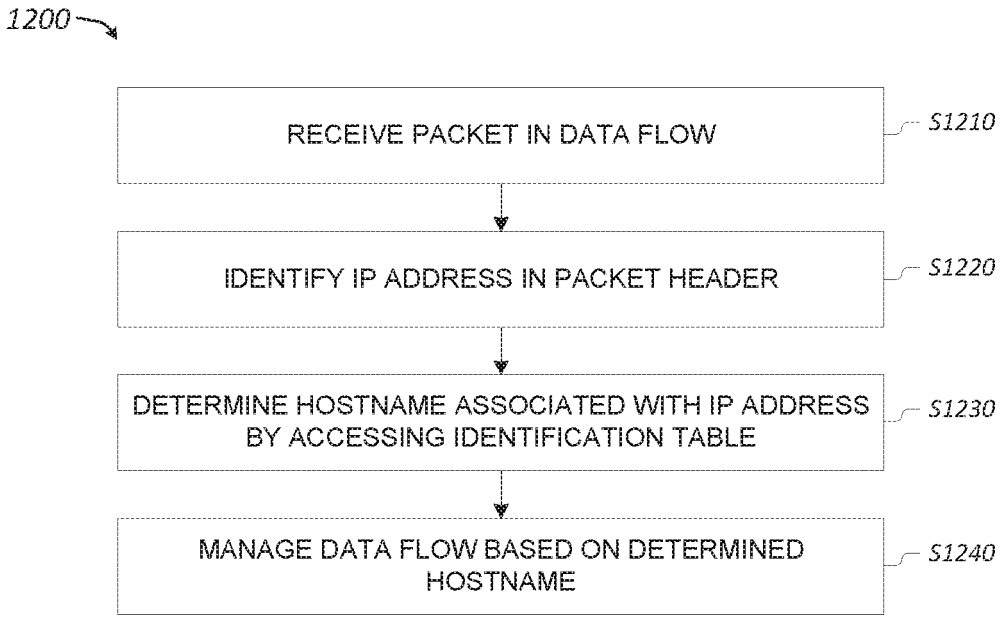
1200
RECEIVE PACKET IN DATA FLOW
S1210
IDENTIFY IP ADDRESS IN PACKET HEADER
S1220
DETERMINE HOSTNAME ASSOCIATED WITH IP ADDRESS BY ACCESSING IDENTIFICATION TABLE
S1230
MANAGE DATA FLOW BASED ON DETERMINED HOSTNAME
S1240

Fig. 12

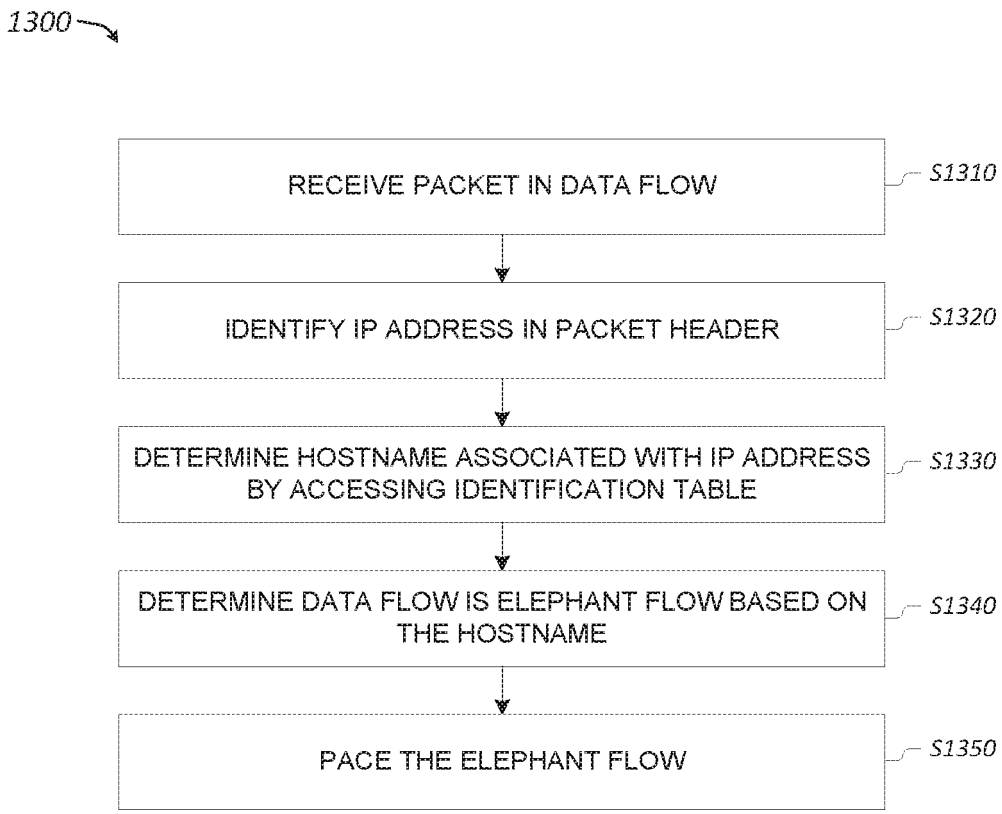
1300
RECEIVE PACKET IN DATA FLOW
S1310
IDENTIFY IP ADDRESS IN PACKET HEADER
S1320
DETERMINE HOSTNAME ASSOCIATED WITH IP ADDRESS BY ACCESSING IDENTIFICATION TABLE
S1330
DETERMINE DATA FLOW IS ELEPHANT FLOW BASED ON THE HOSTNAME
S1340
PACE THE ELEPHANT FLOW
S1350

Fig. 13

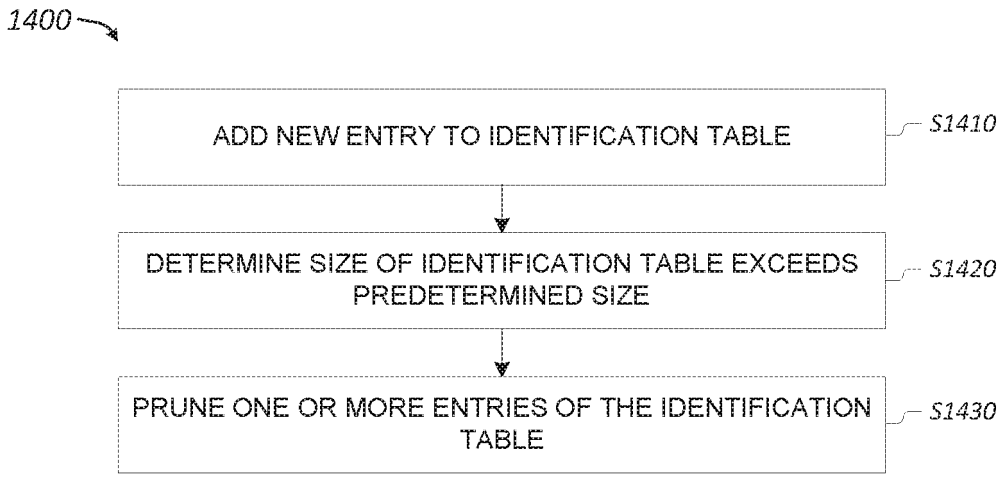
1400
ADD NEW ENTRY TO IDENTIFICATION TABLE
S1410
DETERMINE SIZE OF IDENTIFICATION TABLE EXCEEDS PREDETERMINED SIZE
S1420
PRUNE ONE OR MORE ENTRIES OF THE IDENTIFICATION TABLE
S1430

Fig. 14

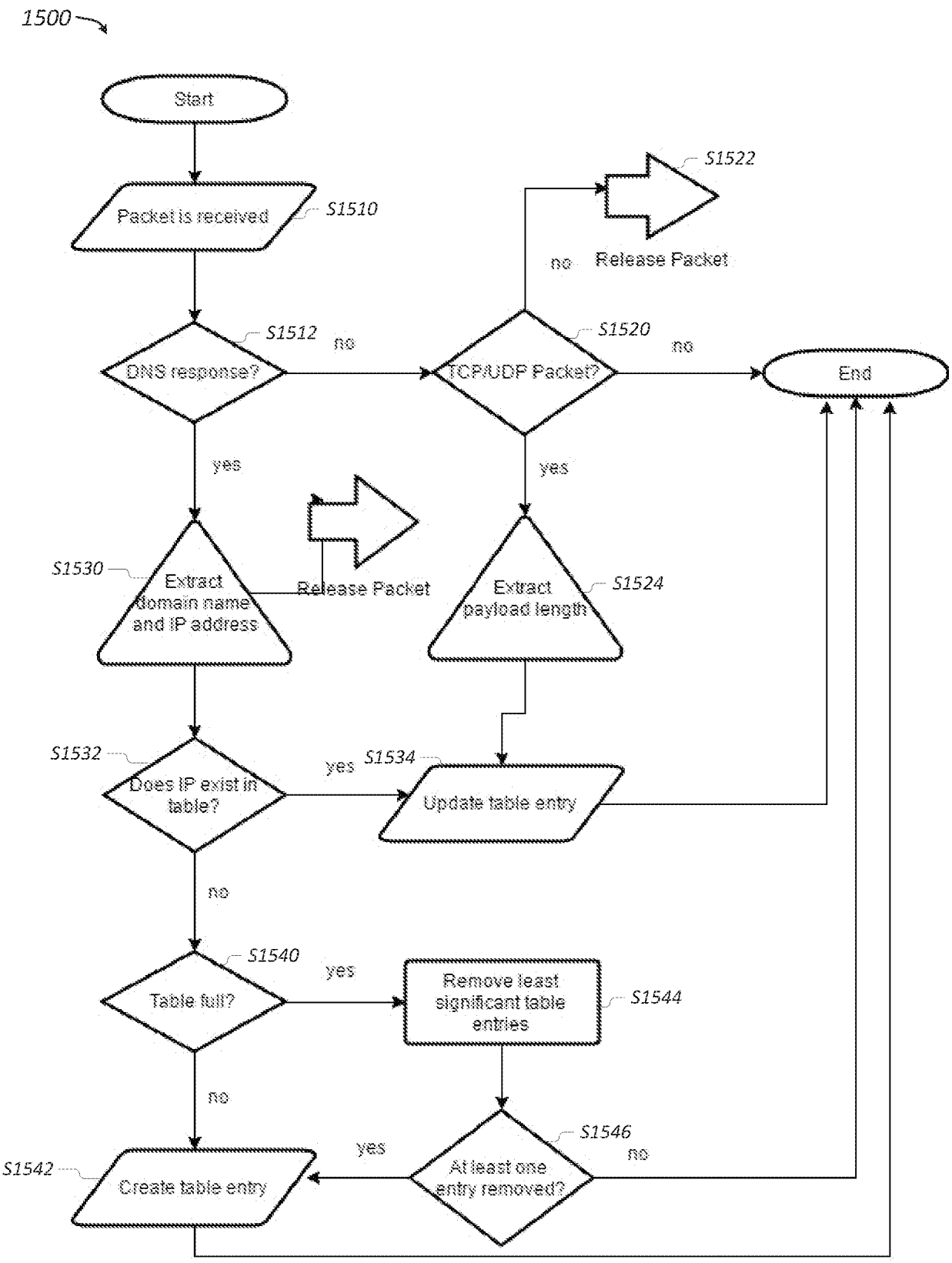
1500
Start
Packet is received
S1510
DNS response?
S1512
no
TCP/UDP Packet?
S1520
no
S1522
no
Release Packet
End
yes
yes
S1530
Extract domain name and IP address
Release Packet
Extract payload length
S1524
S1532
Does IP exist in table?
yes
S1534
Update table entry
no
Table full?
S1540
yes
Remove least significant table entries
S1544
no
S1546
yes
At least one entry removed?
no
S1542
Create table entry

Fig. 15

REFERENCES CITED IN THE DESCRIPTION

This list of references cited by the applicant is for the reader's convenience only. It does not form part of the European patent document. Even though great care has been taken in compiling the references, errors or omissions cannot be excluded and the EPO disclaims all liability in this regard.

Patent documents cited in the description

- US 62491581 **[0001]**
- US 20160323186 A **[0007]**
- US 06048616 **[0037]**
- US 70390817 **[0039]**
- US 7500010 B **[0064]**
- US 8589585 B **[0064]**
- US 20100198943 A **[0064]**
- US 20130124679 A **[0064]**