

(51) International Patent Classification:
H04L 12/801 (2013.01)(21) International Application Number:
PCT/US2014/050541(22) International Filing Date:
11 August 2014 (11.08.2014)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
2973/CHE/2014 18 June 2014 (18.06.2014) IN(71) Applicant: **HEWLETT-PACKARD DEVELOPMENT COMPANY, L.P.** [US/US]; 11445 Compaq Center Drive W., Houston, Texas 77070 (US).(72) Inventors: **MATHEW, Subin Cyriac**; Sy.No.192, Whitefield Road, Mahadevapura Post, Bangalore, Karnataka 560048 (IN). **CHANDRAN, Sugesh**; Sy.No.192, Whitefield Road, Mahadevapura Post, Bangalore, Karnataka 560048 (IN). **KANIAMPADY SEBASTIAN, Celestian**; Sy.No.192, Whitefield Road, Mahadevapura Post, Bangalore, Karnataka 560048 (IN).(74) Agents: **FERGUSON, Christopher W.** et al.; Hewlett-Packard Company, Intellectual Property Administration, 3404 E. Harmony Road, Mail Stop 35, Fort Collins, Colorado 80528 (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, IR, IS, JP, KE, KG, 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, SA, 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, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

[Continued on next page]

(54) Title: PREVENTING FLOW TABLE MISSES IN A SOFTWARE DEFINED NETWORK

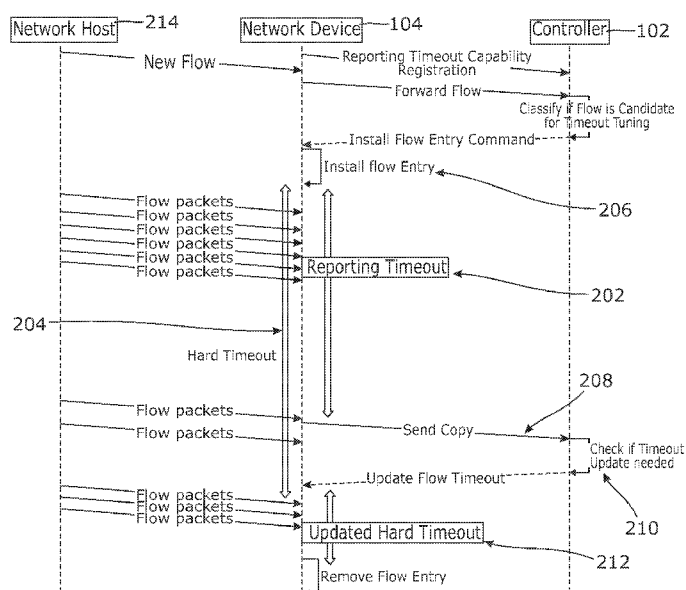


Fig. 2

(57) Abstract: Examples disclosed herein relate to preventing flow table misses in a software defined network. A first timeout period may be associated with a flow entry in a software defined network (SDN) enabled device, wherein upon expiry of the first timeout period the flow entry is removed from the SDN enabled device. A second timeout period may be associated with the flow entry in the SDN enabled device, wherein the second timeout period is shorter than the first timeout period. A copy of a packet from the first matching the flow entry may be sent from the SDN enabled device to a SDN controller upon expiry of the second timeout period. The SDN controller may update the first timeout period associated with the flow entry in the SDN enabled device.

**Declarations under Rule 4.17:**

- *as to the identity of the inventor (Rule 4.17(i))*
- *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

Published:

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

PREVENTING FLOW TABLE MISSES IN A SOFTWARE DEFINED NETWORK

Background

[001] A software defined network (SDN) is based on a network architecture that decouples the control plane from the data plane. The control plane is implemented in an SDN controller and the data plane is implemented in the networking infrastructure (e.g., switches and routers). In SDN, data forwarding on a network device is controlled through flow table entries populated by the SDN controller that manages the control plane for that network. A network device that receives packets on its interfaces looks up its flow table to check the actions that need to be taken on a received packet. OpenFlow is a leading protocol for SDN architecture. A SDN controller may communicate with network devices via OpenFlow or a suitable API.

Brief Description of the Drawings

[002] For a better understanding of the solution, embodiments will now be described, purely by way of example, with reference to the accompanying drawings, in which:

[003] FIG. 1 is a diagram of an example network system to prevent flow table misses in a software defined network;

[004] FIG. 2 is a diagram illustrating association of a second time period with a flow entry in a network device, according to an example;

[005] FIG. 3 is a flow chart of an example method for preventing flow table misses in a software defined network;

[006] FIG. 4 is a flow chart of an example method for preventing flow table misses in a software defined network; and

[007] FIG. 5 is a diagram of an example system to prevent flow table misses in a software defined network.

Detailed Description

[008] Software defined networking (SDN) is an approach to networking in which control is decoupled from networking equipment and given to a device called a controller (or SDN controller). The controller may be aware of all the devices and their points of interconnection in a SDN network and performs various functions such as routing, policy implementation, etc. Each new or missed flow through the network is routed via the controller that decides the network path for a flow, and adds an entry for that flow in a flow table in each of the network devices along the path. A SDN enabled device consults a flow table(s) for forwarding packets in the data plane. Each forwarding rule

(flow entry) includes an action that dictates how traffic that matches the rule is to be handled. A SDN controller may communicate with network devices via a standardized protocol (such as OpenFlow) or API.

[009] Apart from installing new flow rules, an SDN controller may be responsible for removing the flow rules as well. Towards this task, the controller may assign a timeout to each flow entry. When a timeout expires on an SDN enabled device (e.g., switches), the device may evict the rule from the flow table and, optionally, notify the controller regarding the removal. However, often, when a new flow arrives at the controller for flow programming, the controller may not know the exact period of time during which the flow may remain active. In an instance, the controller may determine an expected duration of a flow session based on the flow signature, and heuristics from prior traffic patterns. Based on these, the controller may program a flow entry with a hard timeout. When a hard timeout expires the flow entry is evicted by the switch irrespective of whether or not packets are matching the flow entry. In other words, the flow entry is removed from the network device regardless of the presence of a matching flow.

[0010] It is useful to have a flow replacement mechanism to prevent a switch flow table from overflowing as the number of flows increases. There may be many reasons why a session may last more than what was expected by the controller. These may include network retransmission necessitated by network congestion, delays due to host application bottlenecks, etc. However, an issue with having a flow replacement mechanism is that a switch may remove the flow entry when the hard timeout expires even if a session is still ongoing. Needless to say such situation is not desirable since any succeeding packets of the same flow may hit a flow miss, and may need to be forwarded to the controller to make a forwarding decision. The situation may become more challenging, for instance, in case of a high speed data transfer session, multimedia data streaming etc., when a large number of

packets may need to be forwarded to the controller for further action. This unnecessarily burdens the controller which may not only have to receive these packets, buffer them, identify their future network, but may also need to install the flow entries again in all the network devices along the path so that the session may continue.

[0011] Further, since all devices in a network may not necessarily be time synchronized, any device in a network path may timeout the hard timeout rule first. For example, in the event it is the last device in the path where the hard timeout occurs first, the condition may spiral to all previous devices in the network path thereby leading to jitter due to increased network latency and packet re-ordering. The issue may get compounded further if the host application starts retransmitting the packets because of lack of acknowledgement from the destination.

[0012] To address these issues, the present disclosure describes various examples for preventing flow table misses in a software defined network. A hard timeout period may be associated with a flow entry in a software defined network (SDN) enabled switch, wherein the hard timeout period represents maximum amount of time before the flow entry is removed from the SDN enabled switch irrespective of whether or not packets are arriving at the flow entry in the SDN enabled switch. A reporting timeout period may be assigned to the flow entry in the SDN enabled switch, wherein the reporting timeout period is less than the hard timeout period. Upon expiry of the reporting timeout period, a copy of a packet from a flow matching the flow entry may be received from the SDN enabled switch, and the hard timeout period associated with the flow entry in the SDN enabled switch may be extended by an SDN controller.

[0013] The present disclosure may reduce additional load on a controller which may occur subsequent to a flow table miss whereupon the controller may need to take a forwarding decision for all packets that it may receive. The

present disclosure thus reduces network latency and reordering of packets in a flow thereby preventing jitter caused by flow table misses in an ongoing session.

[0014] FIG. 1 is a diagram of an example network system 100 to prevent flow table misses in a software defined network. Network system 100 may include an SDN controller 102 and a network device 104. Although only one SDN controller 102 and one network device is shown in FIG. 1, other examples of this disclosure may include more than one SDN controller 102 and more than one network device 104. In an example, network system 100 may include one or more computer systems or an end user device(s) (not shown) that may be the source or destination of packet flows in network system 100. In an example, network system 100 may be based on software-defined networking (SDN) architecture.

[0015] SDN controller 102 may be any server, computing device, or the like. In an example, SDN controller 102 may be a computer application (machine-executable instructions). SDN controller 102 may define the data flow that occurs in network system 100. In other words, SDN controller 102 may determine how packets should flow through the network device 104 of network system 100. SDN controller 102 may communicate with network device 104 via a standardized protocol (example, OpenFlow) or a suitable API.

[0016] SDN controller 102 may maintain all network rules and provide appropriate instructions (such as forwarding instructions) to network device 104. SDN controller 102 may centralize the network intelligence, while network maintains a distributed forwarding plane through network device 104. In other words, SDN controller 102 may become aware of the network topology prior to computing forwarding paths in network system 100. SDN controller 102 then programs rules on each network device (for example, 104) which may be used by network device to forward packets to another

device in the network 100. In an example, aforesaid programming rules may take the form of a flow entry in one or more flow tables in network device (for example, 104). Each flow entry may be associated with an action (example, forward, redirect, drop, etc.) that describes what process may be followed with respect to a flow that corresponds to a flow entry. SDN controller 102 thus controls the way data packets are routed in network 100.

[0017] SDN controller 102 may communicate with network device 104 over a computer network 112. Computer network 112 may be a wireless or wired network. Computer network 106 may include, for example, a Local Area Network (LAN), a Wireless Local Area Network (WAN), a Metropolitan Area Network (MAN), a Storage Area Network (SAN), a Campus Area Network (CAN), or the like. Further, computer network 112 may be a public network (for example, the Internet) or a private network (for example, an intranet).

[0018] Network device 104 may include, by way of non-limiting examples, a network switch, a network router, a virtual switch, and a virtual router. In an example, network device 104 may be an SDN enabled device or an Open-Flow enabled device.

[0019] Network device 104 may include one or more flow tables (not shown). Each flow table in network device 104 may contain a flow entry (or flow entries) 106. SDN controller 102 may add, update, and delete flow entries 106 in flow tables both reactively (in response to packets) and proactively. Network device 104 may communicate with SDN controller 102 and the controller 102 may manage the device via a standardized protocol such as OpenFlow. For instance, network device 104 may forward the first packet of a flow to SDN controller 102, enabling the controller 102 to decide whether the flow should be added to a flow table in the network device 104. Network device 104 thus may accept directions from an SDN controller 102 to change values in a flow table.

[0020] A flow table matches an incoming packet to a particular flow and specifies the function that may be performed on the packet. If a flow entry 106 matching with a flow is found in a flow table, instructions associated with the specific flow entry may be executed. A packet matches a flow table entry 106 if the values in the packet match fields used for the lookup match those defined in the flow table entry 106. If no match is found in a flow table (such cases may be termed as “flow table misses”), the outcome may depend on configuration of the table-miss flow entry. For example, the packet may be forwarded to SDN controller 102, dropped, etc.

[0021] Each flow entry 106 may consist of match fields, counters, and a set of instructions to apply to matching packets. In an example, each flow table entry 106 may include a number of components, for example, match fields, priority, counters, instructions, timeouts, and cookie.

[0022] A timeout component may define the maximum amount of time before a flow may be expired by a network device (for example, 104). A first timeout period 108 may be associated with a flow entry 106 in network device (for example, 104), wherein upon expiry of the first timeout period 108 the flow entry 106 is removed from the network device (for example, 104). In an example, the first timeout period 108 may represent a hard timeout period. The hard timeout period represents maximum amount of time before the flow entry is removed from the network device (for example, 104) irrespective of whether or not packets are arriving at the flow entry in the network device (for example, 104).

[0023] A second timeout period 110 may be associated with a flow entry in a network device (for example, 104). FIG. 2 shows a diagram illustrating association of a second time period 202 with a flow entry in a network device (for example, 104), according to an example. A network host 214 may send a new flow to a network device (example, 104). A second timeout period 202 may be associated with a flow entry by an SDN controller 102 when the

controller 102 is setting up the flow entry 206 for a new flow in a network device (for example, 104). A second timeout period 202 may be shorter than a first timeout period (or hard timeout period) 204. A second timeout period 202 may also be called a reporting timeout period.

[0024] In an example, upon expiry of the second timeout period 202, a network device (for example, 104) may send a copy 208 of a packet from a flow matching the flow entry to an SDN controller 102. In an instance, such packet is the first packet or most recent packet received by the network device (for example, 104) after expiry of the second timeout period 202. The SDN controller 102 may inspect the copy of the packet and, based on one or more parameters, may decide whether a first timeout period 204 associated with a flow entry may need to be updated 210. In an instance, the SDN controller 102 may extend (update) 212 the first timeout period 204. Extension of the first time period 204 may allow a flow session to continue without jitter. Along with the extension of the first time period 204, the SDN controller 102 may extend (update) the second timeout or reporting timeout period 202 as well.

[0025] In an example, an SDN controller 102 may associate a hard timeout duration and a reporting timeout duration with a flow entry during addition of said flow entry to an SDN enabled device (for example, 104). After expiry of the reporting timeout duration, the SDN controller 102 may receive a copy of a packet from a flow matching the flow entry from the SDN enabled device (for example, 104). Upon receipt of said packet, the SDN controller 102 may extend the hard timeout duration associated with the flow entry in the SDN enabled device (for example, 104) based on one or more parameters. In an example, an SDN controller 102 may extend the hard timeout duration based on type or contents of the packet. For instance, an SDN controller 102 may perform a Deep Packet Inspection (DPI) on the received packet. Using Deep Packet Inspection (DPI), an SDN controller 102 may examine the data part (and also the header) of a packet for a defined criteria to decide whether

the packet may pass or if it needs to be routed through a different network path. The use of DPI makes it possible to identify, reroute or block packets with specific data. For example, an SDN controller 102 may use DPI to look at the type or content of a flow packet (example, streaming video) to decide whether the hard timeout duration associated with a flow entry in an SDN enabled device (for example, 104) may be extended. In another example, the SDN controller 102 may extend the hard timeout duration based on a pre-defined policy which may be applied for certain type of flows. For instance, flows with better Quality of Service (QoS) requirements may be given higher priority over other flows. In an example, subsequent to the extension of the hard timeout duration, the SDN controller may reprogram the same path or configure a new network path for the flow.

[0026] FIG. 3 is a flow chart of an example method for preventing flow table misses in a software defined network. The method 300, which is described below, may be executed on a computing device such as SDN controller 102 of FIG. 1. However, other suitable computing devices may execute method 300 as well. In particular, method may execute, at least partially, in a network device (for example, 104). At block 302, an SDN controller 102 may associate a first timeout period with a flow entry in a software defined network (SDN) enabled device (for example, 104). The first timeout period defines the maximum amount of time before the flow entry may be removed from the SDN enabled device (for example, 104). In example, the first timeout period represents a hard timeout period, wherein the hard timeout period represents maximum amount of time before the flow entry is removed from the SDN enabled device (for example, 104) irrespective of whether or not packets are arriving at the flow entry in the SDN enabled device (for example, 104). At block 304, the SDN controller 102 may associate a second timeout period with the flow entry in the SDN enabled device (for example, 104). The second timeout period is shorter than the first timeout period. In other words, the second timeout period may occur prior to the first timeout period. At block 306, upon expiry of the second timeout period, the

SDN enabled device (for example, 104) may send a copy of a packet from a flow matching the flow entry to the SDN controller 102. In an example, a copy of the most recent packet (i.e. first packet) received by the SDN enabled device (for example, 104) upon expiry of the second timeout period is sent to the SDN controller 102. At block 308, the SDN controller 102 may update the first timeout period associated with the flow entry in the SDN enabled device (for example, 104). In an example, such update may include extending the first timeout period associated with the flow entry in the SDN enabled device (for example, 104). The SDN controller 102 may update the first timeout period associated with the flow entry based on one or more parameters. In an instance, the SDN controller 102 may also extend the second timeout period associated with the flow entry in the SDN enabled device (for example, 104).

[0027] FIG. 4 is a flow chart of an example method for preventing flow table misses in a software defined network. The method 400, which is described below, may be executed on a computing device such as SDN controller 102 of FIG. 1. However, other suitable computing devices may execute method 400 as well. In particular, method may execute, at least partially, in a network device (for example, 104). At block 402, an SDN controller 102 may receive a flow miss packet from an SDN enabled device (for example, 104). A flow miss packet may be generated when a new flow reaches the SDN enabled device (for example, 104). Since there is no flow entry in the SDN enabled device (for example, 104) for a new flow, it results in a flow miss, and a packet from the new flow may be forwarded to the SDN controller 102. At block 404, the SDN controller 102 may determine that a flow miss needs to be avoided for this flow, and may associate a hard timeout duration and a reporting timeout duration with a flow entry during addition of the flow entry to the SDN enabled device (for example, 104). In other words, while setting-up a flow entry in the SDN enabled device (for example, 104) for the new flow, the SDN controller 102 may associate a hard timeout duration and a reporting timeout duration with a flow entry. The hard timeout duration may

represent the maximum amount of time before the flow entry is removed from the SDN enabled device (for example, 104). The reporting timeout duration is less than the hard timeout duration. The reporting timeout duration may be selected such that sufficient time may be provided for the SDN controller 102 to process the packet sample which may arrive upon expiry of said timeout, decide that the flow entry may need to be updated and configure a previous or new flow before the hard timeout period expires. At block 406, upon expiry of the reporting timeout duration, the SDN enabled device (for example, 104) may send a copy of a packet from a flow matching the flow entry to the SDN controller 102. In an example, a copy of the most recent packet (i.e. first packet) received by the SDN enabled device (for example, 104) upon expiry of the second timeout period is sent to the SDN controller 102. At block 408, the SDN controller 102 may update the hard timeout duration associated with the flow entry in the SDN enabled device. In an example, such update may include extending the hard timeout duration associated with the flow entry in the SDN enabled device (for example, 104). The SDN controller 102 may update the hard timeout duration associated with the flow entry based on one or more parameters. For instance, the SDN controller 102 may carry out an L3-L7 inspection of the received packet to decide whether or not to extend the hard timeout period. For example, the absence of certain Transmission Control Protocol (TCP) flags like FIN, FIN/ACK, RST, URG, and the like, may indicate that a flow session is still ongoing. On the other hand, a TCP PUSH flag may indicate that the sender SDN enabled device (for example, 104) may not contain any new packets for sending to the SDN controller 102. In an instance, the SDN controller 102 may also extend the reporting timeout duration associated with the flow entry in the SDN enabled device (for example, 104).

[0028] FIG. 5 is a diagram of an example system to prevent flow table misses in a software defined network. System 500 includes a processor 502 and a machine-readable storage medium 504 communicatively coupled through a system bus. In an example, system 500 may be analogous to network

device of FIG. 1. Processor 502 may be any type of Central Processing Unit (CPU), microprocessor, or processing logic that interprets and executes machine-readable instructions stored in machine-readable storage medium 504. Machine-readable storage medium 504 may be a random access memory (RAM) or another type of dynamic storage device that may store information and machine-readable instructions that may be executed by processor 502. For example, machine-readable storage medium 504 may be Synchronous DRAM (SDRAM), Double Data Rate (DDR), Rambus DRAM (RDRAM), Rambus RAM, etc. or storage memory media such as a floppy disk, a hard disk, a CD-ROM, a DVD, a pen drive, and the like. In an example, machine-readable storage medium may be a non-transitory machine-readable medium. Machine-readable storage medium 504 may store 506, 508, 510, and 512. In an example, instructions 506 may be executed by processor 502 to assign a hard timeout period to a flow entry in a flow table of a software defined network (SDN) enabled switch (for example, 104), wherein the hard timeout period represents maximum amount of time before the flow entry is removed from the SDN enabled switch (for example, 104) irrespective of whether or not packets are arriving at the flow entry in the SDN enabled switch. Instructions 508 may be executed by processor 502 to assign a reporting timeout period to the flow entry in the SDN enabled switch (for example, 104), wherein the reporting timeout period is less than the hard timeout period. Instructions 510 may be executed by processor 502 to send a copy of a packet from a flow matching the flow entry to an SDN controller 102 upon expiry of the reporting timeout period. Instructions 512 may be executed by processor 502 to extend the hard timeout period associated with the flow entry in the SDN enabled switch (for example, 104) based on a pre-defined factor (for example, contents of the packet). In an example, machine-readable storage medium 504 may store further instructions to update the reporting timeout period assigned to the flow entry in the SDN enabled switch (for example, 104). The storage medium 504 may also store instructions to configure, in the SDN enabled switch (for example, 104), the most recent network path used by the flow

subsequent to extension of the hard timeout period associated with the flow entry. The storage medium 504 may also include instructions to configure, in the SDN enabled switch (for example, 104), a new network path for the flow subsequent to extension of the hard timeout period associated with the flow entry.

[0029] For the purpose of simplicity of explanation, the example methods of FIGS. 3 and 4 are shown as executing serially, however it is to be understood and appreciated that the present and other examples are not limited by the illustrated order. The example systems of FIGS. 1 and 5, and methods of FIGS. 3 and 4 may be implemented in the form of a computer program product including computer-executable instructions, such as program code, which may be run on any suitable computing device in conjunction with a suitable operating system (for example, Microsoft Windows, Linux, UNIX, and the like). Embodiments within the scope of the present solution may also include program products comprising non-transitory computer-readable media for carrying or having computer-executable instructions or data structures stored thereon. Such computer-readable media can be any available media that can be accessed by a general purpose or special purpose computer. By way of example, such computer-readable media can comprise RAM, ROM, EPROM, EEPROM, CD-ROM, magnetic disk storage or other storage devices, or any other medium which can be used to carry or store desired program code in the form of computer-executable instructions and which can be accessed by a general purpose or special purpose computer. The computer readable instructions can also be accessed from memory and executed by a processor.

[0030] It may be noted that the above-described examples of the present solution is for the purpose of illustration only. Although the solution has been described in conjunction with a specific embodiment thereof, numerous modifications may be possible without materially departing from the

teachings and advantages of the subject matter described herein. Other substitutions, modifications and changes may be made without departing from the spirit of the present solution. All of the features disclosed in this specification (including any accompanying claims, abstract and drawings), and/or all of the steps of any method or process so disclosed, may be combined in any combination, except combinations where at least some of such features and/or steps are mutually exclusive.

Claims:

1. A method for preventing flow table misses in a software defined network, comprising:

associating a first timeout period with a flow entry in a software defined network (SDN) enabled device, wherein upon expiry of the first timeout period the flow entry is removed from the SDN enabled device;

associating a second timeout period with the flow entry in the SDN enabled device, wherein the second timeout period is shorter than the first timeout period;

sending a copy of a packet from a flow matching the flow entry, from the SDN enabled device to a SDN controller, upon expiry of the second timeout period; and

updating, by the SDN controller, the first timeout period associated with the flow entry in the SDN enabled device.

2. The method of claim 1, further comprising updating the second timeout period associated with the flow entry in the SDN enabled device.

3. The method of claim 1, wherein the first timeout period represents a hard timeout period, wherein the hard timeout period represents maximum amount of time before the flow entry is removed from the SDN enabled device irrespective of whether or not packets are arriving at the flow entry in the SDN enabled device.

4. The method of claim 1, wherein the packet is most recent packet received by the SDN enabled device upon expiry of the second timeout period.

5. The method of claim 1, wherein the second timeout period is associated with the flow entry during setting up of the flow entry in the SDN enabled device by the SDN controller.

6. A system to prevent flow table misses in a software defined network, comprising:

a software defined network (SDN) enabled device; and
an SDN controller to:

associate a hard timeout duration and a reporting timeout duration with a flow entry during addition of the flow entry to the SDN enabled device, wherein the hard timeout duration represents maximum amount of time before the flow entry is removed from the SDN enabled device, and wherein the reporting timeout duration is less than the hard timeout duration;

receive a copy of a packet from a flow matching the flow entry, from the SDN enabled device, upon expiry of the reporting timeout duration; and

extend the hard timeout duration associated with the flow entry in the SDN enabled device.

7. The system of claim 6, wherein the SDN controller is to extend the hard timeout duration based on Deep Packet Inspection (DPI) of the received packet.

8. The system of claim 6, wherein the SDN controller is to extend the hard timeout duration based on a pre-defined policy.

9. The system of claim 6, wherein the SDN enabled device is a network switch or a router.

10. The system of claim 6, wherein the SDN controller is to configure a previous or new network path for the flow prior to expiry of the hard timeout duration.

11. A non-transitory machine-readable storage medium comprising instructions for flow table misses in a software defined network, the instructions executable by a processor to:

assign a hard timeout period to a flow entry in a flow table of a software defined network (SDN) enabled switch, wherein the hard timeout period represents maximum amount of time before the flow entry is removed from the SDN enabled switch irrespective of whether or not packets are arriving at the flow entry in the SDN enabled switch;

assign a reporting timeout period to the flow entry in the SDN enabled switch, wherein the reporting timeout period is less than the hard timeout period;

send a copy of a packet from a flow matching the flow entry to an SDN controller upon expiry of the reporting timeout period; and

extend the hard timeout period associated with the flow entry in the SDN enabled switch based on a pre-defined factor.

12. The storage medium of claim 11, further comprising instructions to update the reporting timeout period assigned to the flow entry in the SDN enabled switch.

13. The storage medium of claim 11, further comprising instructions to configure in the SDN enabled switch most recent network path used by the flow subsequent to extension of the hard timeout period associated with the flow entry.

14. The storage medium of claim 11, further comprising instructions to configure in the SDN enabled switch a new network path for the flow subsequent to extension of the hard timeout period associated with the flow entry.

15. The storage medium of claim 11, wherein the pre-defined factor includes content of the packet.

1/5

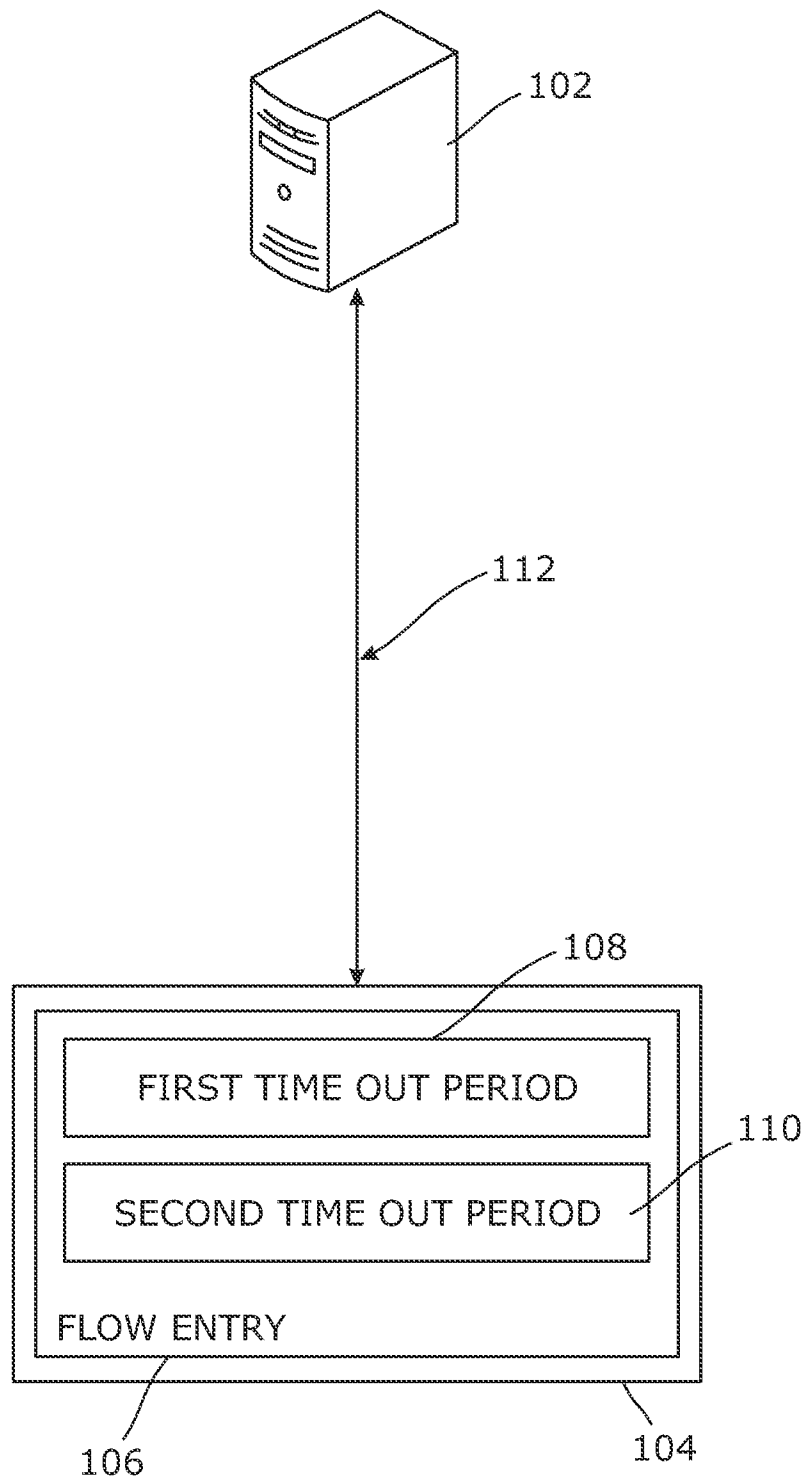
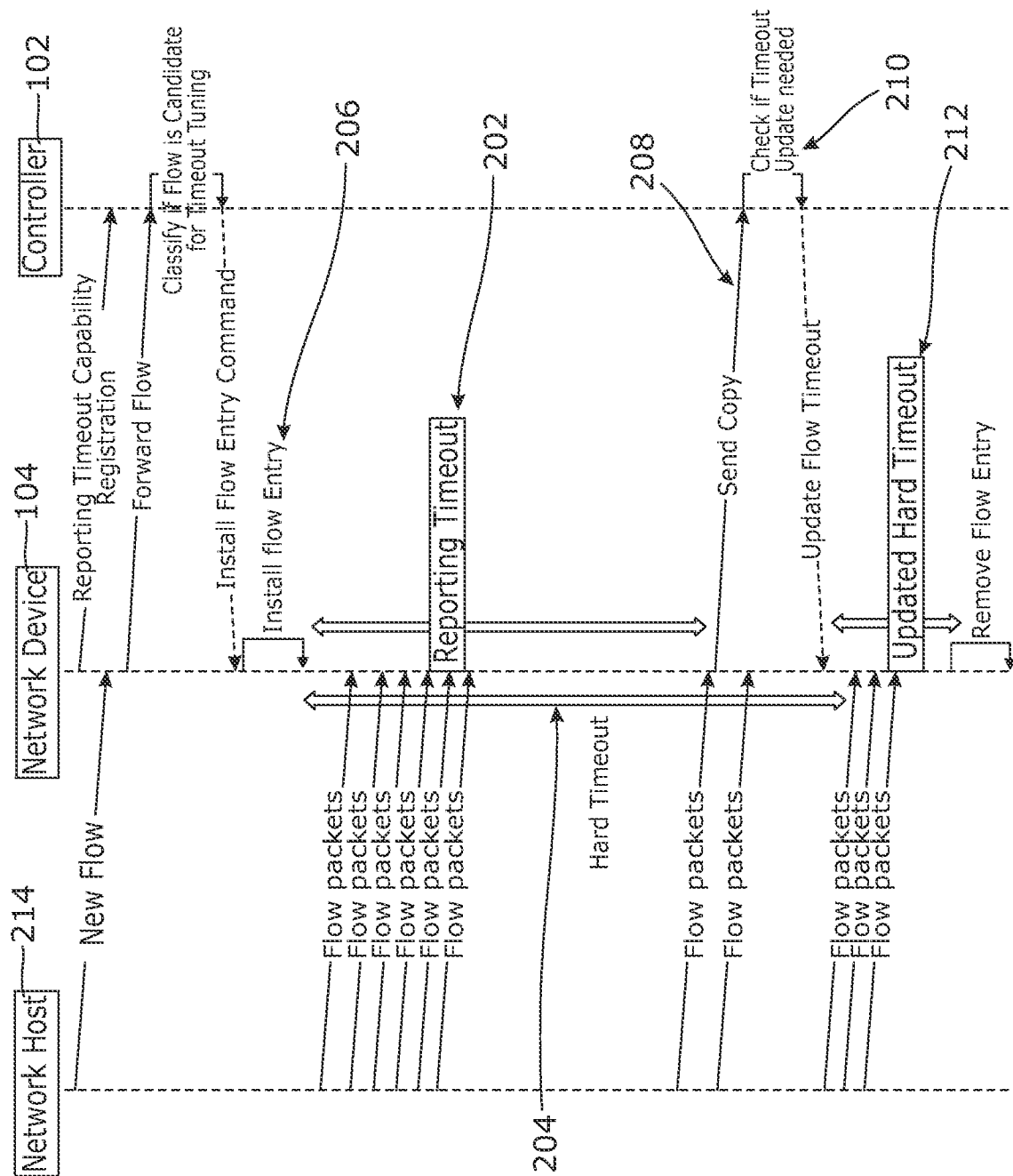


Fig. 1



25

3/5

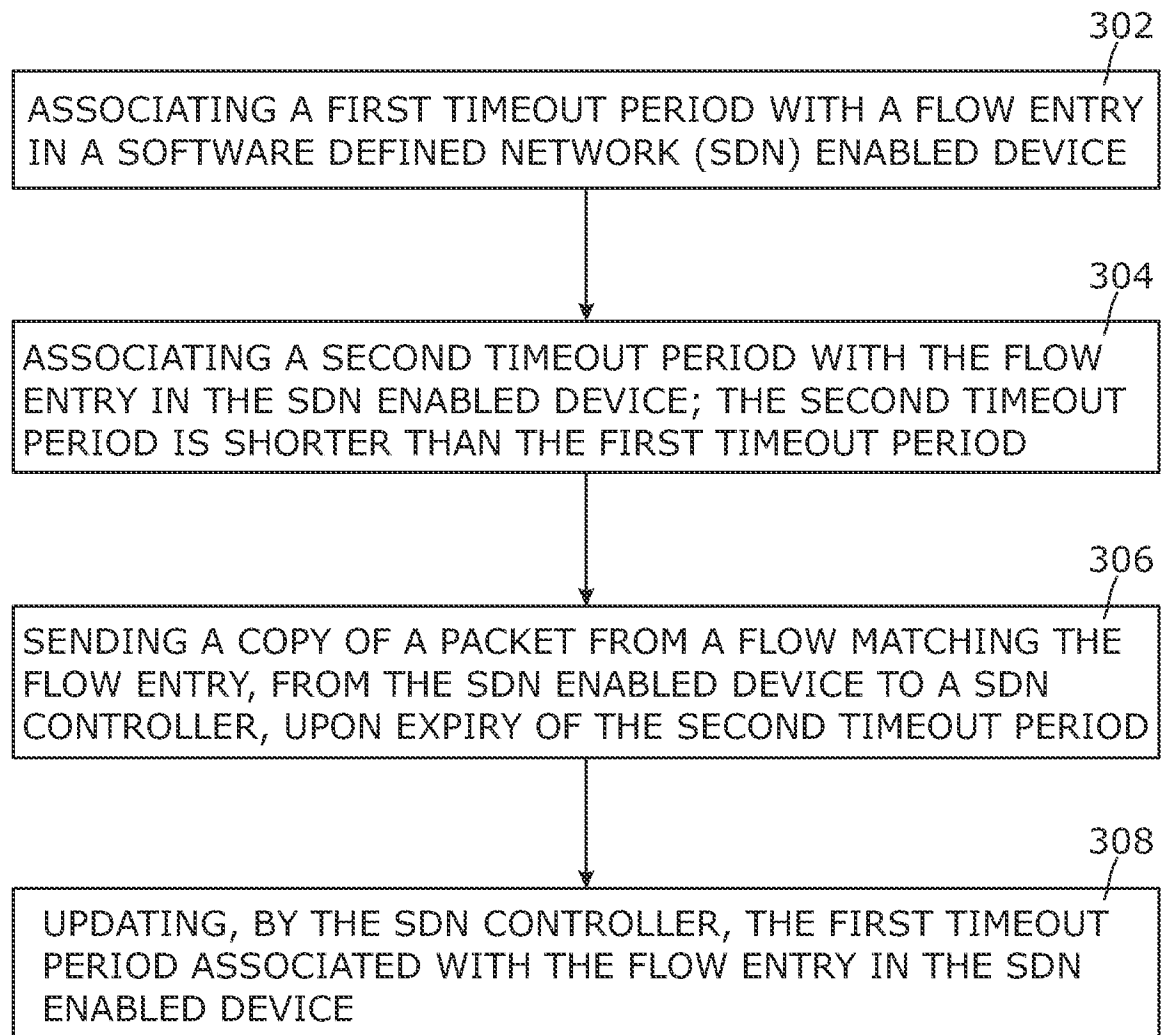
300

Fig. 3

4/5

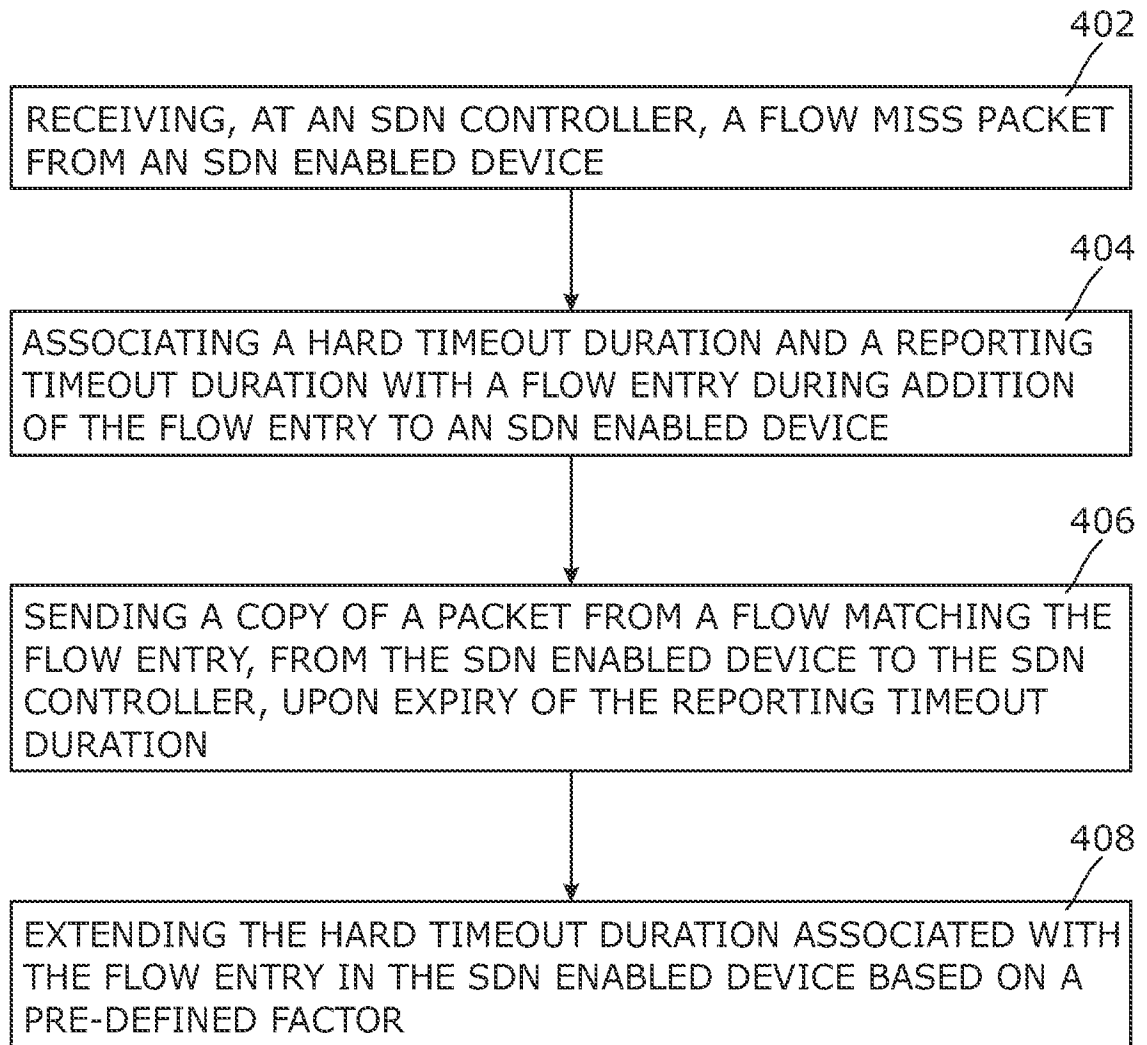
400

Fig. 4

5/5

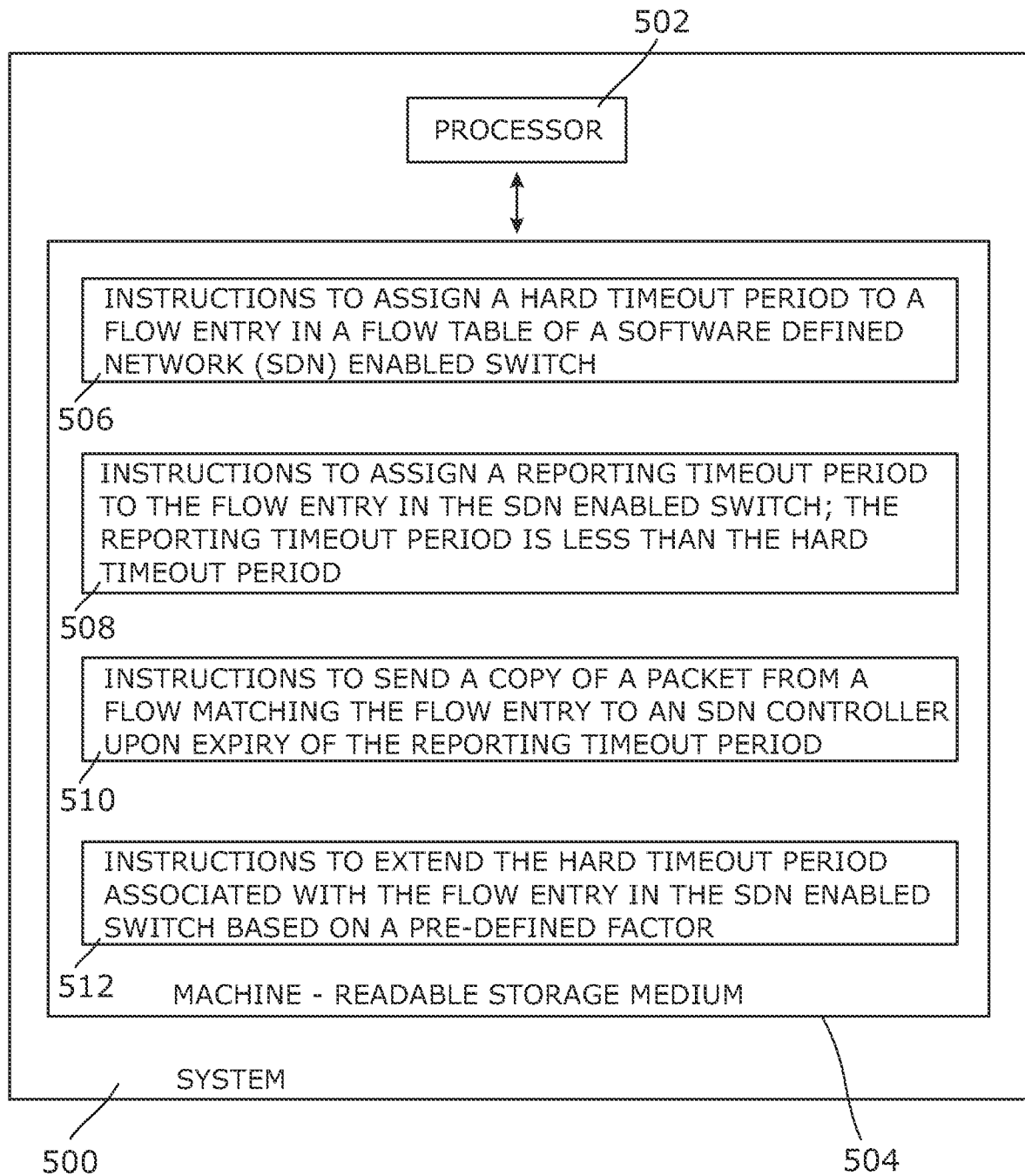


Fig. 5

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2014/050541**A. CLASSIFICATION OF SUBJECT MATTER****H04L 12/801(2013.01)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 12/801; none ; H04L 12/717; H04L 12/26; H04L 12/721

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: flow table, miss, entry, SDN(Software Defined Network), timeout, removing

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	WO 2014-057375 A2 (TELEFONAKTIEBOLAGET L M ERICSSON (PUBL)) 17 April 2014 See page 4, lines 25-27; page 11, lines 2-3; page 19, line 23 - page 21, line 5; claim 1; and figures 1, 3.	1-15
A	ADAM ZAREK, 'OpenFlow Timeouts Demystified', Department of Computer Science University of Toronto, 2012 See page 2, lines 2-3; page 8, lines 2-5; page 9, lines 1-10; page 19, line 27 - page 20, line 9; page 31, lines 10-11; and figure 3.	1-15
A	OPEN NETWORKING FOUNDATION, 'OpenFlow Switch Specification' version 1.3.0 (Wire protocol 0x04) 25 June 2012 See page 14, lines 21-24; and figure 3.	1-15
A	US 2013-0128746 A1 (KIRAN YEDAVALLI) 23 May 2013 See paragraphs [0035]-[0041]; and figure 6.	1-15
A	WO 2013-132395 A1 (TELEFONAKTIEBOLAGET L M ERICSSON (PUBL)) 12 September 2013 See page 4, line 13 - page 5, line 4; and figure 4.	1-15

☒ 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 another 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

11 February 2015 (11.02.2015)

Date of mailing of the international search report

24 February 2015 (24.02.2015)

Name and mailing address of the ISA/KR

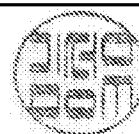
International Application Division
Korean Intellectual Property Office
189 Cheongsa-ro, Seo-gu, Daejeon Metropolitan City, 302-701,
Republic of Korea

Facsimile No. ++82 42 472 3473

Authorized officer

KIM, Seong Woo

Telephone No. +82-42-481-3348



INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/US2014/050541

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 2014-057375 A2	17/04/2014	US 2014-0098669 A1 WO 2014-057375 A3	10/04/2014 05/06/2014
US 2013-0128746 A1	23/05/2013	EP 2597831 A1 EP 2597831 B1 US 8644149 B2	29/05/2013 03/12/2014 04/02/2014
WO 2013-132395 A1	12/09/2013	CN 104145457 A US 2013-0230047 A1 US 8705536 B2	12/11/2014 05/09/2013 22/04/2014