



- (51) **International Patent Classification:** Not classified
- (21) **International Application Number:** PCT/IB2013/002156
- (22) **International Filing Date:** 2 August 2013 (02.08.2013)
- (25) **Filing Language:** English
- (26) **Publication Language:** English
- (30) **Priority Data:** 61/679,361 3 August 2012 (03.08.2012) US
- (71) **Applicant:** MARVELL WORLD TRADE LTD. [BB/BB]; L'horizon, Gunsite Road, Brittons Hill, St. Michael, BB14027 (BB).
- (72) **Inventors:** HUTT, Gad; 1476 Kingfisher Way, Sunnyvale, CA 94087 (US). ORR, Michael; 1082 Susan Way, Sunnyvale, CA 94087 (US). TAUSI, Ronen; 38 Hayovel St., 43401 Raanana (IL). VOVNOBOY, Vitaly; 15 Hanevel Street, 44855 Karney Shomron (IL).
- (74) **Agents:** SAUER, Joseph, M. et al.; Jones Day, 901 Lakeside Avenue, Cleveland, OH 44114 (US).

- (81) **Designated States** (unless otherwise indicated, for every kind of national protection available): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.
- (84) **Designated States** (unless otherwise indicated, for every kind of regional protection available): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:

- without international search report and to be republished upon receipt of that report (Rule 48.2(g))

(54) **Title:** SYSTEMS AND METHODS FOR DEEP PACKET INSPECTION WITH A VIRTUAL MACHINE

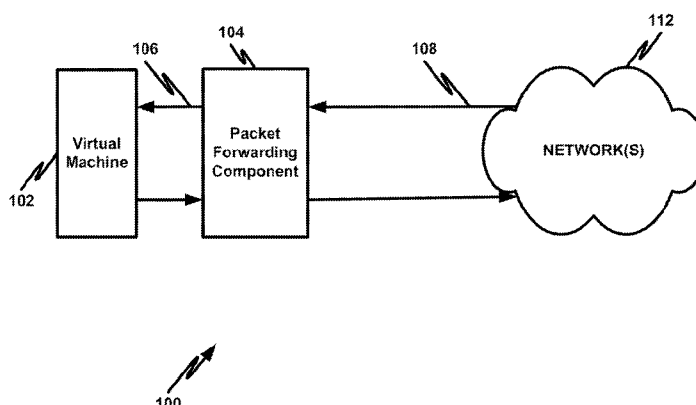


FIG. 1

(57) **Abstract:** System and methods are provided for performing deep packet inspection of data packets. An example system includes a packet forwarding component and a virtual machine component. The packet forwarding component is configured to receive data packets for transmission and to select one or more of the data packets based at least in part on a first set of rules for deep packet inspection. The virtual machine component is configured to perform deep packet inspection on the selected data packets according to a second set of rules to determine whether the selected data packets are allowed for transmission. The packet forwarding component is further configured to transmit the selected data packets when the selected data packets are allowed for transmission after the deep packet inspection.

Systems and Methods for Deep Packet Inspection With a Virtual Machine

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This disclosure claims priority to and benefit from U.S. Provisional Patent Application No. 61/679,361, filed on August 3, 2012, the entirety of which is incorporated herein by reference.

FIELD

[0002] The technology described in this patent document relates generally to communication systems and more particularly to deep packet inspection in communication systems.

BACKGROUND

[0003] Deep packet inspection (DPI) has been implemented in many communication systems for detecting protocol non-compliance, viruses, spam, intrusions, or for determining, based on defined criteria, whether certain data packets in network traffic may pass or if the data packets need to be routed to a different destination, *etc.* Network traffic usually includes data packets of certain internet protocols transmitted between different network entities. Usually, DPI not only inspects the header portion of a data packet which often includes information related to the protocol, source and destination IP addresses and ports, but also scans the payload portion of the data packet which often includes user data to be transmitted.

[0004] Oftentimes, a DPI system may identify data packets from different applications/network entities using certain search algorithms, such as signature analyses. For example, a signature of a particular application/network entity includes a unique pattern (*e.g.*, bytes/characters/string). A reference database may be created based on analyses of signatures of

various applications/network entities. A classification engine of the DPI system may then compare data packets from the network traffic against this reference database to identify the exact applications/network entities. The reference database may be updated periodically to keep current with new applications/network entities as well as new developments of protocols associated with the existing applications/network entities.

[0005] Hardware implementation of DPI is often adopted to achieve good processing speed. For example, content addressable memories (CAM) are used in DPI systems. A CAM may make parallel comparisons between entries stored in the CAM and certain input values of the data packets in the network traffic and return the memory address of the matched entry. For example, a binary CAM is a simple type of CAM which often stores search words including two matching states, "1" and "0." A ternary CAM (TCAM) allows a third matching state of "X" or "Don't Care" for one or more bits in the stored search words, thus adding flexibility to the search. As an example, a ternary CAM might have a stored search word of "1XX10" which can match any of the four words "10010," "10110," "11010," or "11110." This added search flexibility comes at an additional cost over a binary CAM as the internal memory cell needs to encode three states instead of two.

SUMMARY

[0006] In accordance with the teachings described herein, system and methods are provided for performing deep packet inspection of data packets. An example system includes a packet forwarding component and a virtual machine component. The packet forwarding component is configured to receive data packets for transmission and to select one or more of the data packets based at least in part on a first set of rules for deep packet inspection. The virtual machine component is configured to perform deep packet inspection on the selected data packets

according to a second set of rules to determine whether the selected data packets are allowed for transmission. The packet forwarding component is further configured to transmit the selected data packets when the selected data packets are allowed for transmission after the deep packet inspection.

[0007] In one embodiment, a method for performing deep packet inspection of data packets includes, receiving data packets for transmission, selecting one or more of the data packets based at least in part on a first set of rules for deep packet inspection, performing deep packet inspection on the selected data packets according to a second set of rules to determine whether the selected data packets are allowed for transmission, and transmitting the selected data packets when the selected data packets are allowed for transmission after the deep packet inspection.

[0008] In another embodiment, a processing system for performing deep packet inspection of data packets includes one or more data processors configured to implement a virtual machine. The virtual machine is configured to receive one or more data packets from a switch core and perform deep packet inspection on the received data packets according to a first set of rules to determine whether the received data packets are allowed for transmission. The virtual machine is further configured to cause the received data packets to be transmitted by the switch core when the data packets are allowed for transmission after the deep packet inspection.

BRIEF DESCRIPTION OF THE DRAWINGS

[0009] FIG. 1 depicts an example diagram for performing deep packet inspection on data packets from network traffic.

[0010] FIG. 2 depicts an example diagram for performing deep packet inspection on data packets using a virtual machine implemented in a DPI core.

[0011] FIG. 3 depicts another example diagram for performing deep packet inspection on data packets using a virtual machine implemented in a DPI core.

[0012] FIG. 4 depicts an example flow diagram for deep packet inspection using a virtual machine.

DETAILED DESCRIPTION

[0013] A hardware implementation of DPI, such as a CAM (*e.g.*, a binary CAM or a TCAM), may have some disadvantages under certain circumstances. For example, data packets of real-time transport protocol (RTP) each include a header, optional header extensions, and a payload portion. Usually, RTP data packets from different applications/network entities vary in length and structure. Thus, in order to discern the RTP data packets from different sources, a lot of entries may be stored in the CAM (*e.g.*, a binary CAM or a TCAM), which may result in significantly high system costs.

[0014] FIG. 1 depicts an example diagram for performing deep packet inspection on data packets from network traffic. As shown in FIG. 1, a packet forwarding component 104 (*e.g.*, implemented in a switch or other gateway devices) performs preliminary inspection of data packets 108 received from a network 112, and selects one or more data packets for deep packet inspection by a virtual machine 102 (*e.g.*, implemented in one or more data processors). The virtual machine 102 has the flexibility to perform deep packet inspection at any depth of the data packets according to one or more predetermined rules. In some embodiments, the virtual machine 102 corresponds to a virtual implementation of one or more processor elements which are dedicated to performing deep packet inspection. The virtual machine 102 is “virtual” because, to an observer, it appears to be part of the underlying hardware and the way it is implemented is not apparent.

[0015] Specifically, the packet forwarding component 104 inspects the data packets 108 and selects one or more of the data packets 108 for deep packet inspection based on a set of rules. For example, the packet forwarding component 104 inspects a predetermined sequence of bits at a predetermined offset of each of the data packets 108, and identifies one or more of the data packets 108 in need of deep packet inspection. The virtual machine 102 may be programmed to inspect the selected data packets 106 according to another set of rules to determine whether the selected data packets 106 are allowed for transmission. For example, the virtual machine 102 may inspect any predetermined sequence of bits at any particular depth for each of the selected data packets 106.

[0016] When the virtual machine 102 determines the selected data packets 106 are suitable for transmission, the packet forwarding component 104 may be configured to transmit the selected data packets 106 to their respective destinations, *e.g.*, through the network 112. If the selected data packets 106 are not allowed to be transmitted, then the packet forwarding component 104 may take proper measures, such as discarding/quarantining/rerouting these data packets.

[0017] FIG. 2 depicts an example diagram for performing deep packet inspection on data packets using a virtual machine implemented in a DPI core. As shown in FIG. 2, a virtual machine 206 within a DPI core 204 is used for performing deep packet inspection on data packets forwarded from a TCAM 210 within a switch core 208.

[0018] Specifically, the TCAM 210 inspects a predetermined sequence of bits at a predetermined offset of each of the data packets 212 received from a network 230. For example, an RTP data packet may usually be identified by inspecting the first 128 bits which typically includes a header portion or any optional header extensions. The TCAM 210 may be programmed to inspect the first 128 bits of each of the data packets 212 to identify the RTP data

packets, and to send all identified RTP data packets to the virtual machine 206 for deep packet inspection.

[0019] The DPI core 204 is part of a central-processing-unit (CPU) core 202 which implements one or more data processors. The virtual machine 206 may inspect data packets 214 forwarded from the TCAM 210, and determine whether the data packets 214 are suitable for transmission. Further, the virtual machine 206 may use a direct matching algorithm (*e.g.*, identical string matching) or a pattern matching algorithm (*e.g.*, approximate string matching). When the virtual machine 206 determines that the data packets 214 are allowed for transmission, the data packets 214 may be assigned to a particular process queue of the switch core 208 for transmission.

[0020] FIG. 3 depicts another example diagram for performing deep packet inspection on data packets using a virtual machine implemented in a DPI core. As shown in FIG. 3, a TCAM 304 within a switch core 306 forwards one or more data packets to a virtual machine 308 within a DPI core 312 for deep packet inspection.

[0021] The DPI core 312 resides in a CPU core 302, and further includes a DPI control plane 310 which provides configuration information (*e.g.*, the rules for deep packet inspection) to the virtual machine 308. In addition, the DPI control plane 310 may receive information related to deep-packet-inspection events from the virtual machine 308. For example, when the virtual machine 308 determines that the data packets 318 from the TCAM 304 is suitable for transmission, the DPI control plane 310 may receive a command from the virtual machine 308 to perform certain actions in order to cause the switch core 306 to transmit the data packets 318.

[0022] The CPU core 302 further includes an application processor 314 which provides configuration information (*e.g.*, process queues and associated queue numbers) to the switch core

306 and receives information related to hardware events (*e.g.*, system calls) from the switch core 306. In addition, the application processor 314 may process data packets which are not forwarded to the DPI core 312. For example, the application processor 314 may inspect the header portion or a header extension of the received data packets.

[0023] The virtual machine 308 may perform deep packet inspection using different methods. For example, the virtual machine 308 may inspect the payload of data packets to locate certain embedded patterns (*e.g.*, bytes/characters/strings) even though these embedded patterns may not be located at a specific deterministic offset. As another example, the virtual machine 308 may look into the numerical characteristics of data packets, such as the payload size, the number of response packets, and offsets, to determine whether the data packets are suitable for transmission. In addition, the virtual machine 308 may perform behavioral and statistical (heuristic) analyses of the data packets to identify viruses or analyze network traffic. Further, the virtual machine 308 may analyze a number of consecutive data packets to discern a certain sequence of steps or actions related to a particular application/network entity.

[0024] FIG. 4 depicts an example flow diagram for deep packet inspection using a virtual machine. At 402, data packets are received for transmission. At 404, one or more of the data packets are selected, *e.g.*, based at least in part on a first set of rules for deep packet inspection. At 406, deep packet inspection is performed on the selected data packets, *e.g.*, by a virtual machine, according to a second set of rules to determine whether the selected data packets are allowed for transmission. In one implementation, the second set of rules is different from the first set of rules. Then, at 408, the selected data packets are transmitted when the selected data packets are allowed for transmission after the deep packet inspection.

[0025] This written description uses examples to disclose the invention, include the best mode, and also to enable a person skilled in the art to make and use the invention. The patentable scope of the invention may include other examples that occur to those skilled in the art.

[0026] For example, the systems and methods described herein may be implemented on many different types of processing systems by program code comprising program instructions that are executable by the system processing subsystem. Other implementations may also be used, however, such as firmware or appropriately designed hardware configured to carry out the methods and systems described herein. For example, the systems and methods described herein may be implemented in a stateful operation mode or a stateless operation mode. In another example, the systems and methods described herein may be implemented for network traffic analyses using behavioral algorithms. In yet another example, the systems and methods described herein may be implemented to build flexible DPI engines fully programmable with hardware-assist capturing engines. In yet another example, the systems and methods described herein may be implemented in an independent processing engine, as a co-processor, or as a hardware accelerator. In yet another example, the systems and methods described herein may be provided on many different types of computer-readable media including computer storage mechanisms (e.g., CD-ROM, diskette, RAM, flash memory, computer's hard drive, etc.) that contain instructions (e.g., software) for use in execution by a processor to perform the methods' operations and implement the systems described herein.

CLAIMS

What is claimed is:

1. A system for performing deep packet inspection of data packets, the system comprising:

a packet forwarding component configured to

receive data packets for transmission, and

select one or more of the data packets based at least in part on a first set of rules for deep packet inspection; and

a virtual machine component configured to perform, according to a second set of rules, deep packet inspection on the selected data packets to determine whether the selected data packets are allowed for transmission,

wherein the packet forwarding component is further configured to transmit the selected data packets in response to the virtual machine component determining, based on the deep packet inspection, that the selected data packets are allowed for transmission.
2. The system of claim 1, where the packet forwarding component is configured to select the one or more of the data packets by inspecting a predetermined sequence of bits at a predetermined offset of each of the data packets.
3. The system of claim 1, wherein the virtual machine component is configured to determine whether the selected data packets are allowed for transmission by inspecting a particular sequence of bits of each of the selected data packets.

4. The system of claim 1, wherein the virtual machine component is further configured to perform a direct matching algorithm or a pattern matching algorithm for deep packet inspection.

5. The system of claim 1, wherein the virtual machine component is included in a deep-packet-inspection core, and wherein the packet forwarding component is included in a switch core.

6. The system of claim 5, wherein the deep-packet-inspection core is included in a central-processing-unit core, and wherein the central-processing-unit core further includes an application processor configured to (i) receive information related to hardware events from the switch core, and (ii) output information related to configuration of the switch core.

7. The system of claim 1, wherein the deep-packet-inspection core further includes a deep-packet-inspection control plane configured to (i) receive information related to deep-packet-inspection events from the virtual machine component, and (ii) output information related to configuration of the virtual machine component.

8. A method for performing deep packet inspection of data packets, the method comprising:

receiving data packets for transmission;

selecting, based at least in part on a first set of rules, one or more of the data packets for deep packet inspection;

performing, according to a second set of rules, deep packet inspection on the selected data packets to determine whether the selected data packets are allowed for transmission; and

in response to the selected packets being determined, based on the deep packet inspection, to be allowed for transmission, transmitting the selected data packets.

9. The method of claim 8, wherein selecting, based at least in part on a first set of rules, one or more of the data packets for deep packet inspection includes:

inspecting a predetermined sequence of bits at a predetermined offset of each of the data packets.

10. The method of claim 8, wherein performing, according to a second set of rules, deep packet inspection on the selected data packets to determine whether the selected data packets are allowed for transmission includes:

inspecting a particular sequence of bits of each of the selected data packets.

11. The method of claim 8, wherein performing, according to a second set of rules, deep packet inspection on the selected data packets includes performing a direct matching algorithm or a pattern matching algorithm for deep packet inspection.

12. The method of claim 11, further comprising:

receiving information related to hardware events from a switch core, the switch core selecting, based at least in part on the first set of rules, one or more of the data packets for deep packet inspection; and

outputting information related to configuration of the switch core.

13. The method of claim 8, further comprising:
receiving information related to deep-packet-inspection events from a virtual machine,
the virtual machine performing the deep packet inspection; and
outputting information related to configuration of the virtual machine.

14. A processing system for performing deep packet inspection of data packets, the processing system comprising:
one or more data processors configured to implement a virtual machine,
wherein the virtual machine is configured to
receive one or more data packets from a switch core, and
perform, according to a first set of rules, deep packet inspection on the received data packets to determine whether the received data packets are allowed for transmission, and
wherein the virtual machine is further configured to cause the received data packets to be transmitted by the switch core in response to the virtual machine core determining, based on the deep packet inspection, that the data packets are allowed for transmission.

15. The processing system of claim 14, wherein the virtual machine is configured to perform the deep packet inspection on the received data packets by inspecting a particular sequence of bits of each of the received data packets.

16. The processing system of claim 14, wherein the switch core is configured to select, based at least in part on a second set of rules, the one or more data packets from a data stream for deep packet inspection.

17. The processing system of claim 16, where the switch core is configured to select, based at least in part on the second set of rules, the one or more data packets by inspecting a predetermined sequence of bits at a predetermined offset of each of the data packets.

18. The processing system of claim 14, wherein the virtual machine is further configured to perform a string matching algorithm or a pattern matching algorithm for deep packet inspection.

19. The processing system of claim 14, wherein the one or more data processors are further configured to implement an application processor, and wherein the application processor is configured to (i) receive information related to hardware events from the switch core, and (ii) output information related to configuration of the switch core.

20. The processing system of claim 14, wherein the one or more data processors are further configured to (i) receive information related to deep-packet-inspection events from the virtual machine, and (ii) output information related to configuration of the virtual machine.

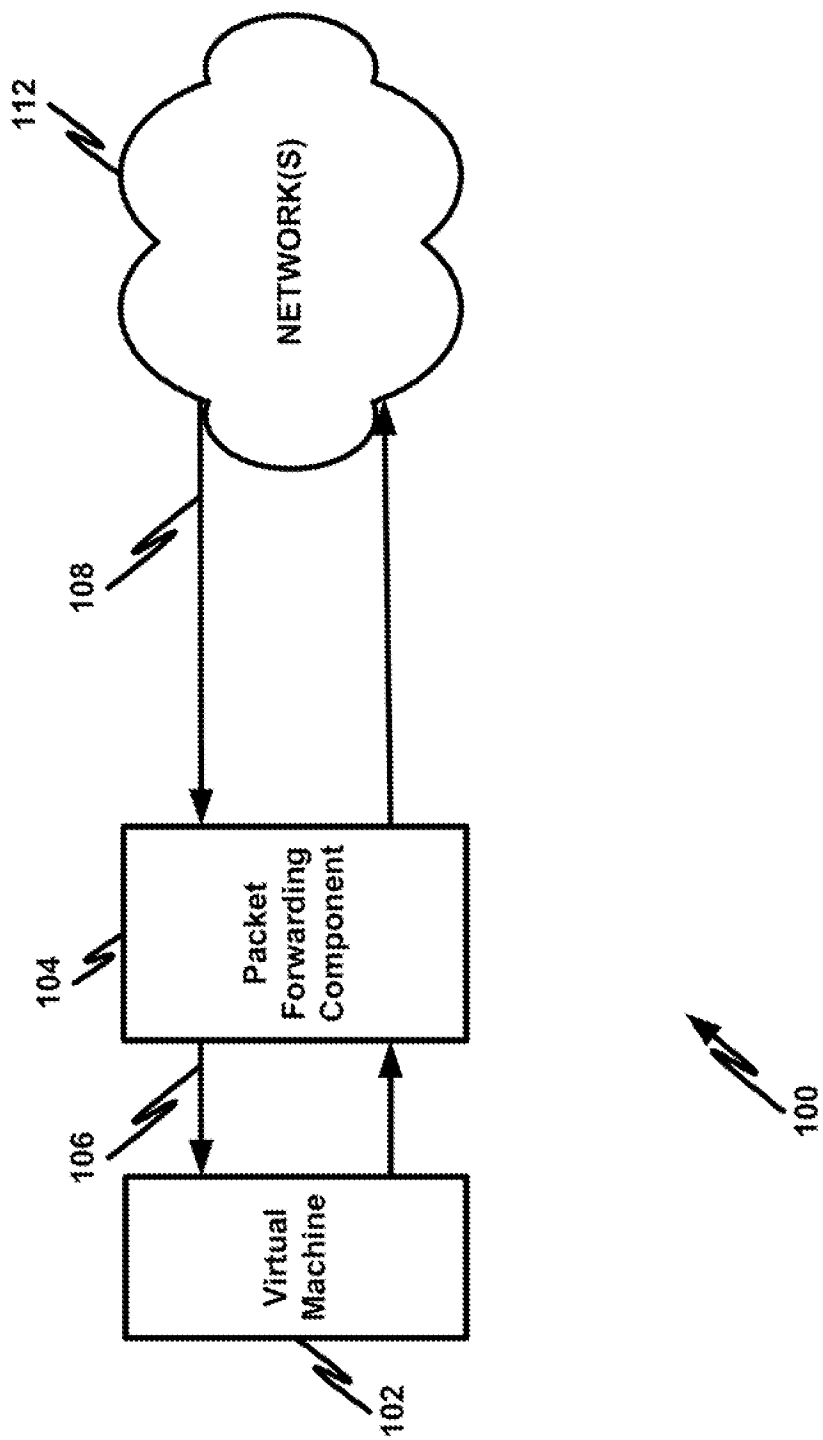


FIG. 1

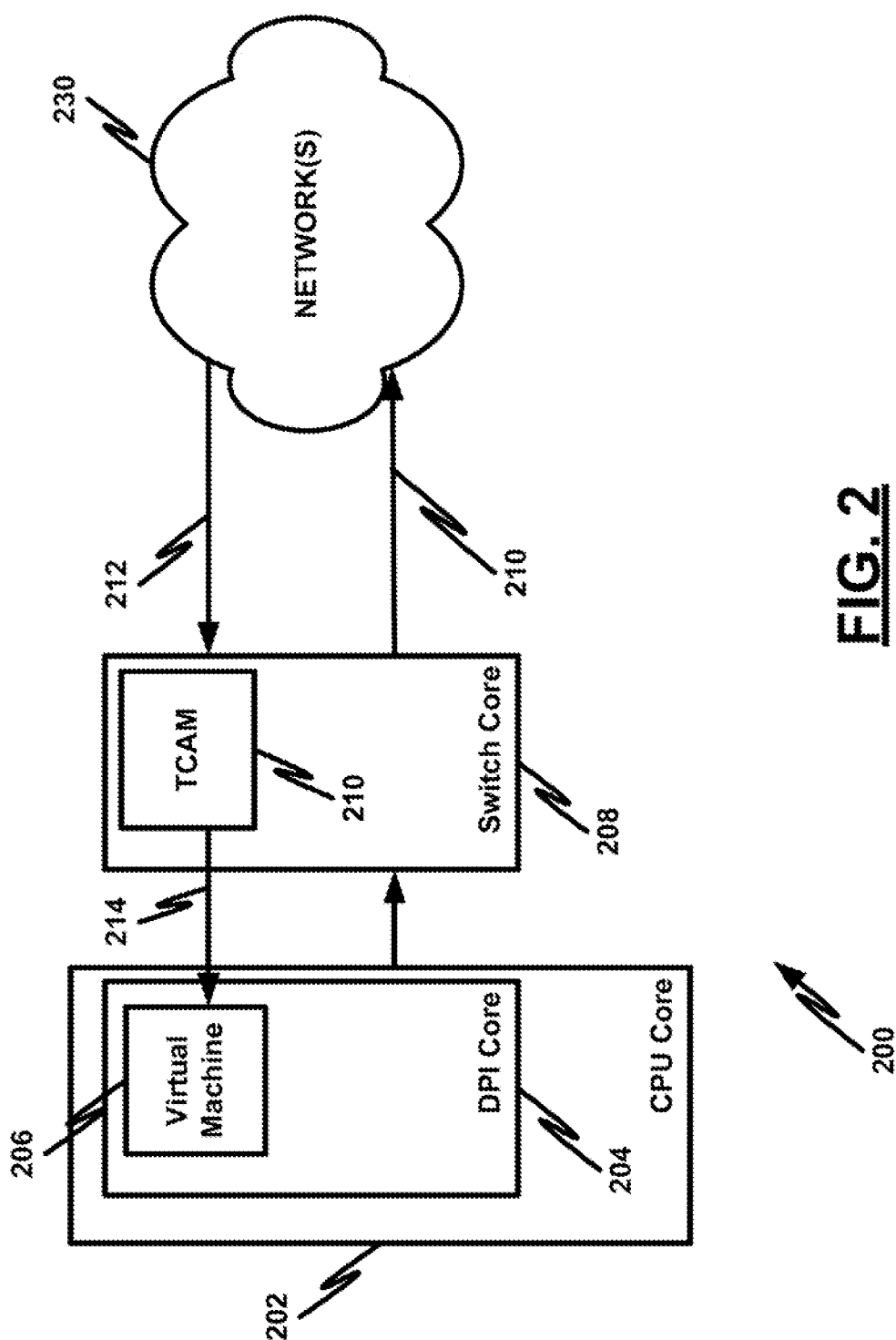


FIG. 2

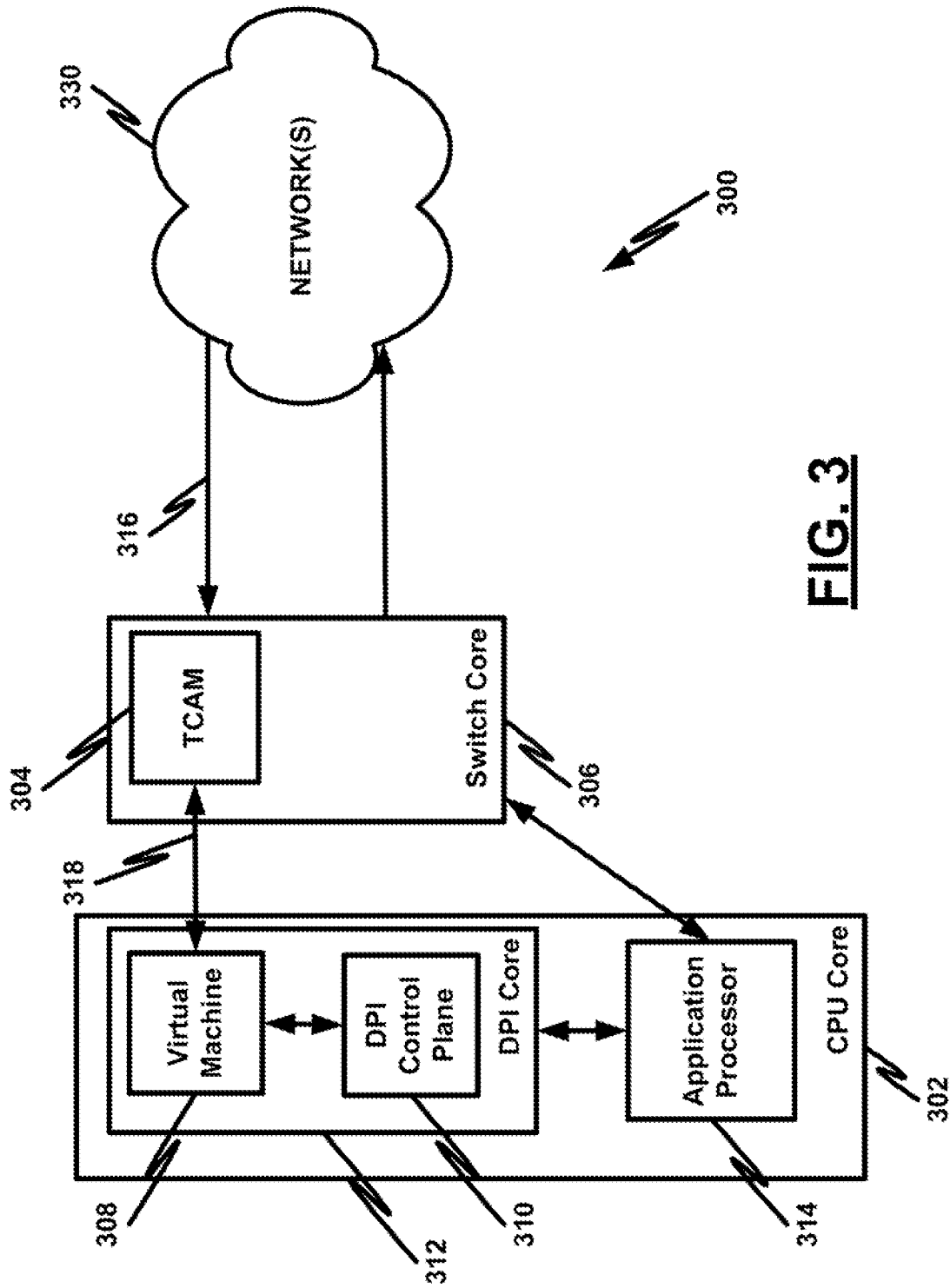
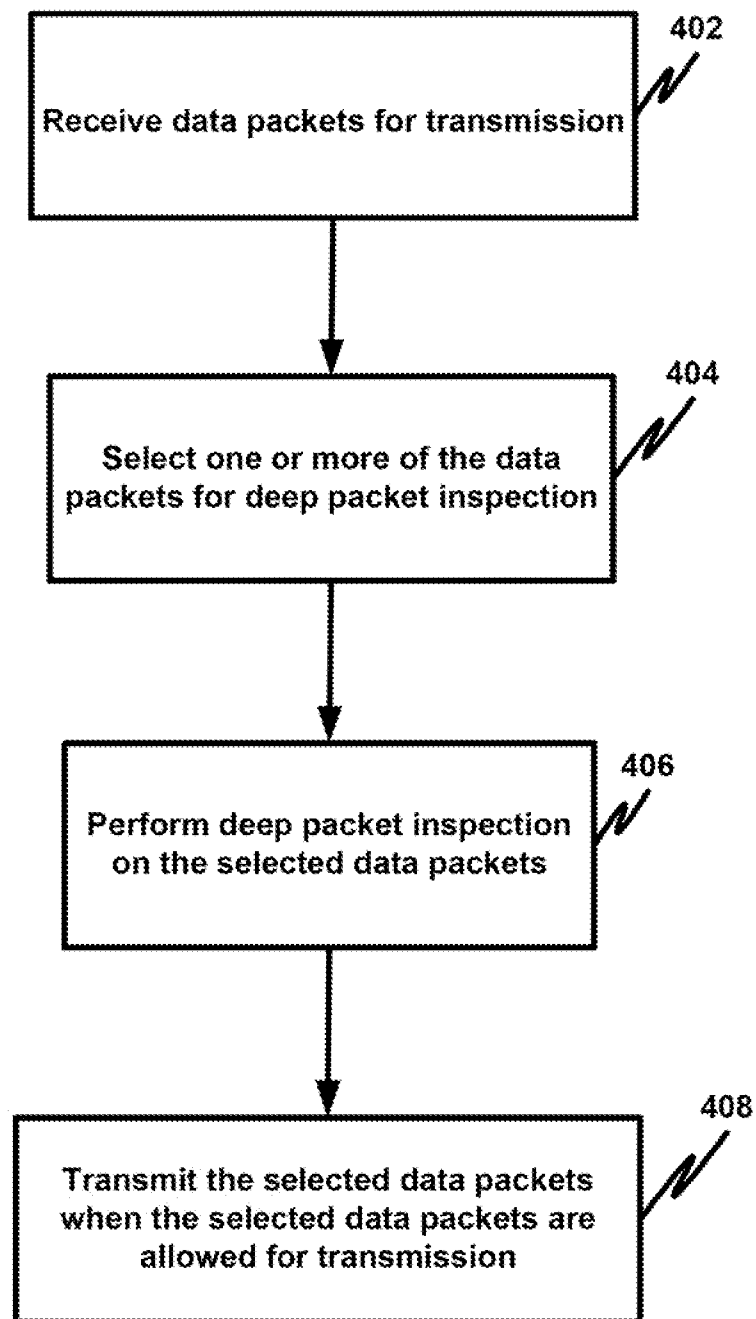


FIG. 3

**FIG. 4**