



- (51) **International Patent Classification:** Not classified
- (21) **International Application Number:** PCT/IB2013/003231
- (22) **International Filing Date:** 17 December 2013 (17.12.2013)
- (25) **Filing Language:** English
- (26) **Publication Language:** English
- (30) **Priority Data:** 61/738,280 17 December 2012 (17.12.2012) US
- (71) **Applicant:** MARVELL WORLD TRADE LTD. [BB/BB]; L'horizon, Gunsite Road, Brittons Hill, St. Michael, BB14027 (BB).
- (72) **Inventors:** KARDASHOV, Timor; Ha-maagal 48/5, Kyriat Ono (IL). KOVALENKO, Maxim; 42/10 Heizenberg St., 76286 Rehovot (IL). DANIEL, Haim; 12/13 Rosh Pina St., 77643 Ashdod (IL). PLOTNIKOV, Anatoli; 34 Balfur St., 4953501 Petach Tikva (IL).
- (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).

Published:

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

(54) **Title:** NETWORK DISCOVERY APPARATUS

(57) **Abstract:** A network device has a packet input unit, a checking unit, and a discovery unit. The packet input unit is configured to receive ingress packets. The checking unit is configured to determine whether identifying characteristics of received ingress packets match stored identifying characteristics of a packet flow that is stored in a memory, to perform a network action when the identifying characteristic of the received ingress packet matches the stored identifying characteristic. The discovery unit is configured to intercept an egress packet received from the control plane processor, the egress packet corresponding to the received ingress packet, to determine one or more differences in selected portions of a header portion of the received ingress packet resulting from processing at the control plane processor, and to store in the memory a new network action based on the one or more differences.



WO 2014/096974 A2

NETWORK DISCOVERY APPARATUS

CROSS-REFERENCE TO RELATED APPLICATION

- [1] This application claims priority to and the benefit of U.S. Provisional Patent Application No. 61/738,280 filed December 17, 2012 the disclosure of which is incorporated by reference herein in its entirety.

BACKGROUND

- [2] The present disclosure relates to a network device that processes packets.
- [3] The background description provided herein is for the purpose of generally presenting the context of the disclosure. Work of the presently named inventors, to the extent it is described in the background section, as well as aspects of the description that may not otherwise qualify as prior art at the time of filing, are neither expressly nor impliedly admitted as prior art against the present disclosure.
- [4] A network device typically includes a control plane processor which implements processing operations to make determinations related to forwarding, routing, access control lists, and the like. Control plane processors located in a network upstream of a final destination typically update the packet header information of incoming packets. A packet flow exists when a number of received packets are identified as having the same or similar characteristics, for example the packets are received on the same port or are from the same source, are to be forwarded to the same destination, are the same type of packet and are to be handled with a same quality of service. Network devices encounter situations in which multiple, different flows need to be handled concurrently.

SUMMARY

[5] One or more example embodiments of the disclosure generally relate to a network discovery apparatus that monitors packets as they ingress to and egress from a control plane of a network device. Such observations are used to detect packet flows as well as to learn the particular changes made to the packets in those flows. Packet flow information is stored together with information indicating what changes the control plane made to the packets. For newly received packets that match a particular packet flow, the network discovery apparatus intercepts the packets and makes the same kind of changes to the packets that the control plane had previously been making. Newly received packets that do not match any defined flow are simply monitored by the network discovery apparatus as they are passed to the control plane.

[6] In an example embodiment, a network device includes a packet input unit configured to receive ingress packets, the ingress packets having an ingress packet identifying characteristic; a checking unit configured to determine whether the identifying characteristic of a received ingress packet matches a stored identifying characteristic of a packet flow that is stored in a memory, to perform on the received ingress packet a network action associated with the stored identifying characteristic when the identifying characteristic of the received ingress packet matches the stored identifying characteristic, and to provide the received ingress packet to a control plane processor configured to process the received ingress packet to determine the network action to be performed on the received ingress packet when the identifying characteristic of the received ingress packet does not match the stored identifying characteristic; and a

discovery unit configured to intercept an egress packet received from the control plane processor, the egress packet corresponding to the received ingress packet, to determine one or more differences in selected portions of a header portion of the received ingress packet resulting from processing at the control plane processor, and to store in the memory a new network action based on the one or more differences.

- [7] In another example embodiment, a network device method includes receiving an ingress packet having an ingress packet identifying characteristic, detecting with a checking unit whether the identifying characteristic is a match with a stored identifying characteristic of stored first packet flow information, when the match is detected, performing on the received ingress packet a stored network action, using the checking unit, to output an egress packet, the stored network action corresponding to the stored identifying characteristic; when the match is not detected, communicating the ingress packet to a control plane processor configured to process the ingress packet; intercepting from the control plane processor an egress packet corresponding to the communicated ingress packet, with a discovery unit, detecting with the discovery unit one or more differences between at least part of a header portion of the communicated ingress packet and a header portion of the intercepted egress packet, and using a processor core to store in a memory a new network action based on the one or more differences, in association with the unmatched identifying characteristic, as second packet flow information.

BRIEF DESCRIPTION OF THE DRAWINGS

- [8] Fig. 1 – Fig. 4 are highly simplified illustrative drawings showing a network device configured and operative according to example embodiments.

[9] Fig. 5 is a state diagram of an example machine according to an example embodiment.

[10] Fig. 6 is a highly simplified illustrative drawing showing a concept of operation according to example embodiments.

[11] Fig. 7 is a flow diagram of an example method according to example embodiments.

[12] Fig. 8 is a flow diagram of an example method according to example embodiments.

[13] Fig. 9 is a flow diagram of an example method according to example embodiments.

[14] Fig. 10 is a flow diagram of an example method according to example embodiments.

[15] Fig. 11 is a flow diagram of an example method according to example embodiments.

[16] Fig. 12 is a flow diagram of an example method according to example embodiments.

[17] Fig. 13 is a flow diagram of an example method according to example embodiments.

[18] Fig. 14 - Fig. 17 are highly simplified illustrative drawings showing a concept of operation according to example embodiments.

DETAILED DESCRIPTION OF EXEMPLARY EMBODIMENTS

[19] In the following discussion, descriptions of well-known functions and constructions may be omitted for increased clarity and conciseness.

[20] Fig. 1 shows a network device 1000 which includes a control plane processor 200 a checking unit 40, and a discovery unit 50 and a processing core (not shown). The network device 1000 also includes a packet input unit 20, a packet output unit 30 and a memory 60 according to an example embodiment. The memory 60 is shared between units of the network device 1000 according to an example embodiment, however, in Fig. 16 for instance, the memory 60 is split into multiple memories with individualized unit-access. In Fig. 1, the elements of discovery apparatus 100 are separate from the control plane processor 200, but this arrangement is merely an example embodiment, and, in Fig. 15 for instance, the network device 1000 is integrated as a System on a Chip (SoC) with the control plane processor 200 in example embodiments.

[21] The packet input unit 20 is connected to a stream of ingress packets 10 at an ingress port thereof. The packet input unit 20 communicates ingress packets 10 to a checking unit 40.

[22] The checking unit 40 communicates with the control plane processor 200 and the discovery unit 50 via connection 70a and with the packet output unit 30 via connection 70c. The checking unit 40 interacts with and performs read/write operations upon the memory 60. The checking unit 40 has other components interposed between the packet output unit 30, the discovery unit 50, the memory 60, and the control plane processor 200 according to an example embodiment, for example the processing core (not shown) is interposed between the elements and operates as a mediator controlling the overall operation of the packet input unit 20, the checking unit 40, the discovery unit 50, the memory 60, the packet output unit 30, and the control plane processor 200.

[23] The discovery unit 50 and the packet output unit 30 receive communications from the control plane processor 200 via connection 70b. The discovery unit 50 is connected to the memory 60. The discovery unit 50 interacts with and performs read/write operations at the memory 60, which is shared between the checking unit 40 and the discovery unit 50 according to an example embodiment. The discovery unit 50 includes additional components interposed between the control plane processor 200, the packet output unit 30, and the memory 60 according to an example embodiment.

[24] Fig. 1 also shows a stream 10 of ingress packets 11, 12. The stream 10 includes packets each with an initial ingress packet header 11a and an initial ingress packet payload 11b. The initial ingress packet header 11a includes an initial packet characteristic α . A stream of packets which each contain the same initial packet characteristic α , for example a same source, a same destination, a same type of packet and a same quality of service designation, composes a packet flow 15. Although, Fig. 1 illustrates packets of a single flow, it is to be understood that packets of other flows exist interwoven into the stream 10, for instance see Fig. 2, where one or more other ingress packets each containing a subsequent ingress packet characteristic γ different from the initial ingress packet characteristic α is received between ingress packets 11 and 12.

[25] Returning to Fig. 1, in an example embodiment, the initial ingress packet characteristic α of the initial ingress packet header 11a includes, e.g., any of the following: ETH header information, IP header information, UDP/TCP header information, and the like. These examples of header information are non-limiting.

[26] Packets of a packet flow 15 are seen in Fig. 1. Here, a plurality of ingress packets 10 includes a corresponding initial ingress packet characteristic α . This is just an example, and it is to be noted that packets of a given flow may, but need not, arrive in immediate succession.

[27] The control plane processor 200 is communicatively coupled to the discovery unit 50 and the packet output unit 30 via connection 70b according to an example embodiment. The control plane processor 200 includes other elements interposed between the discovery unit 50 and the packet output unit 30 according to an example embodiment. In an example embodiment, the control plane processor 200 is implemented in a programmable processor and contains various network configurations and routines relating to processing for bridging, routing, NAT, PPPoE, VLAN, IPSec, Layer 7 ALG, L2TP, access control lists, filtering, firewalls, customer extensions, and QoS and the like. These examples are non-limiting.

[28] The control plane processor 200 receives an initial ingress packet 11 and outputs, after processing, an initial egress packet 13 corresponding to the initial ingress packet 11. A control plane processor processes an ingress packet based on various header information; the processing results in the forwarding of the ingress packet to an egress port as an egress packet. The various header information of the ingress packet is different than the corresponding header information of the egress packet, as the egress packet reflects processing decisions of the control plane process. According to an example embodiment, the control plane processor 200 changes an initial packet characteristic α

into an initial egress packet header 13a, with an initial egress packet characteristic β , and passes the initial ingress packet payload 11b along with the initial egress packet 13.

[29] The memory 60 stores ingress characteristics 61, egress characteristics 62, network actions 63, monitor thresholds 64, learned thresholds 65, and states 66 according to an example embodiment. The memory 60 stores more data than elements 61-65 according to some example embodiments, and the memory 60 stores less data than elements 61-65 according to other example embodiments.

[30] At the lower portion of Fig. 1, there is illustrated a more detailed view of the memory 60, according to an example embodiment. This more detailed representation of the memory 60 is shown as a table containing six columns corresponding to the previously discussed elements 61-66. The memory 60 stores ingress characteristics 61 of packet flows received by the checking unit 40. The memory 60 also stores egress characteristics 62 of packet flows output from the control plane processor 200. These egress characteristics 62 are used by the discovery unit 50 and the checking unit 40 to make processing decisions. The memory 60 also stores network actions 63 relating ingress characteristics 61 to egress characteristics 62. The memory 60 also stores monitor thresholds 64 and learned thresholds 65 in correspondence with the states 66. The flow states 66 will be discussed momentarily. The above examples are non-limiting, however, and the memory 60 according to other example embodiments contains alternative data and holds data in structures other than tables.

[31] The flow states 66 relate to whether a packet flow 15 has been determined to be in a monitor state , a learned state, or a unstable state) according to a non-limiting example,

however, the flow states 66 are merely examples and according to an example embodiment the flow states 66 include multiple other states for use with the network device 1000. A packet of a flow that is in the monitor state is to be monitored to determine changes enacted upon various characteristics of the packet header by the control plane processor. A packet of a flow that is in the learned state is made to have its header changed corresponding to the learned characteristics monitored during the monitor state. A packet of a flow that is in the unstable state is sent to the control plane processor but is not monitored to determine header change characteristics.

[32] The network device 1000 reacts differently to packet flows set to different states. The packet flow will be received by the network device 1000, analyzed by the checking unit 40, and sent to the control plane processor 200 and the discovery apparatus 50 when the packet flow is either a new flow or in the monitor state. According to an example embodiment, only the characteristic of the ingress packet is sent to the discovery apparatus 50. Analysis of the packet involves discovery of one or more packet characteristics designated by the packet header, which is compared with packet characteristics stored in the memory 60.

[33] If the checking unit 40 determines that the packet characteristic is not stored in the memory 60, then an entry corresponding to the characteristic is added to the table and the corresponding flow is set to the monitor state, however, if the packet characteristic is stored in the memory, the network device 1000 makes a determination as to whether the packet characteristic is to be monitored.

[34] When the checking unit 40 makes the determination that the packet is to be monitored, the packet is sent to the control plane processor 200 which designates some network action corresponding to the packet header. The network action is reviewed by the discovery unit 50, which intercepts the packet from the control plane processor 200 stores the action in the memory corresponding to the packet characteristic.

[35] If the checking unit 40 determines that the packet characteristic is associated with a flow set to the learned state, then the checking unit 40 reads and applies the network action, stored in the memory 60, to the packet and sends the packet to the output of the control plane processor 200 corresponding to the network action.

[36] According to an example embodiment, the network action designates that the packet is to be dropped. An ingress packet corresponding to both this network action and a flow of a learned state is correspondingly designated to be dropped in view of the network action learned from the control plane processor.

[37] According to another example embodiment, the network action is learned by the discovery unit to modify the packet header and pass the packet to the packet output unit 30. An ingress packet corresponding to both this network action and a flow of a learned state is correspondingly modified by the checking unit 40 and passed to the egress port of the control plane processor.

[38] The network device 1000 also increments a counter and the like to make determinations regarding switching states corresponding to a packet flow. The counters correspond to, for example, a number of packets which have been received by the network device 1000 corresponding to an same entry and network action or a timing

between receipt of packets of different flows, or the like. The network device 1000 is configured to count, via counters, to thresholds for switching to a learned state from a monitor state and vice versa.

[39] The network device 1000 processes ingress packets 10. Returning to Fig. 1, an ingress packet 11 is received at packet input unit 20. The initial ingress packet 11 contains both an initial ingress packet header 11a and initial ingress packet payload 11b. Fig. 1 illustrates an example wherein the initial ingress packet 11 has an initial ingress packet characteristic α . Further, the initial ingress packet 11 is of a packet flow 15 which also includes a subsequent ingress packet 12 having an initial ingress packet header 11a with a subsequent ingress packet characteristic α corresponding to that of the initial ingress packet, however, this is merely an example embodiment and the subsequent ingress packet is not of the same packet flow 15 as that of ingress packet 11 according to another example embodiment. Further the packet payload 11b of the ingress packet 11 typically is not the same as the packet payload 11b of the subsequent ingress packet 12, in an embodiment. The packet input unit 20 passes the packet into the network device 1000.

[40] The packet input unit 20 passes the initial ingress packet 11 to the checking unit 40. The checking unit 40 determines when the initial ingress packet characteristic α of the initial ingress packet 11 matches any of the ingress characteristics 61 stored in the memory 60.

[41] According to an example embodiment, the checking unit 40 determines that the initial ingress packet characteristic α matches an ingress packet characteristic that is

stored in the memory 60. The checking unit 40 further determines the state of the packet flow 15 by checking states 66. When the checking unit 40 determines that the state of the packet flow is in the monitor state, the packet is sent to the control plane processor 200 and the discovery unit 50 by the checking unit 40. When the checking unit determines that the state of the packet flow is in the learned state, the network action is applied to the packet. When the checking unit 40 determines that the state of the packet flow is in the unstable state, the packet is sent to the control plane processor 200 but is not sent to the discovery unit.

- [42] Fig. 2 illustrates a subsequent ingress packet 12 having a subsequent ingress packet characteristic γ . The subsequent ingress packet 12 is received at the packet input unit 20 and is provided to the checking unit 40 which determines that the subsequent ingress packet characteristic γ is not stored in the memory 60. The checking unit 40 stores the subsequent ingress packet characteristic γ in the memory 60. The checking unit 40 passes the subsequent ingress packet along connection 70a to the control plane processor 200 and the discovery unit 50. The control plane processor 200 passes a subsequent egress packet 14 along connection 70b to the discovery unit 50 and the packet output unit 30. The discovery unit 50 determines a network action designated to the subsequent ingress packet 12 by the control plane processor 200 with respect to the subsequent egress packet 14. The discovery unit 50 determines the network action to be a change in one or more portions the subsequent ingress packet header 12a to the subsequent egress packet header 14a, wherein the subsequent ingress packet

characteristic γ is changed to subsequent egress packet characteristic δ . The discovery unit 50 stores the network action in the memory 60.

[43] According to an example embodiment, when the checking unit 40 determines both that the initial ingress packet characteristic α is stored in the memory 60 and that the state associated with the packet flow packet flow 15 is set to "learned," the checking unit 40 then applies the network action specified by the network actions 63 of the memory 60. Fig. 3 shows that the checking unit 40 converts the initial ingress packet characteristic α into the initial egress packet characteristic β and passes the packet along connection 70c to the packet output unit 30 as the initial egress packet 13; the initial ingress packet 11 is not passed to the control plane processor 200.

[44] According to an example embodiment, when the checking unit 40 determines that the initial ingress packet characteristic α is not stored in the memory 60, the checking unit 40 writes the initial ingress packet characteristic α to the memory 60 and passes the initial ingress packet 11 along connection 70a to the control plane processor 200 and to the discovery unit discovery unit 50, however, this is merely a non-limiting embodiment and initial ingress packet 11 is not passed to the discovery unit 50 by the checking unit checking unit 40 according to an example embodiment, and the discovery unit accesses the memory 60 when comparing the initial egress packet 13 with the initial ingress packet characteristic α .

[45] According to an example embodiment, the discovery unit 50 receives a portion of the initial ingress packet 11 from the checking unit 40.

[46] According to an example embodiment, the discovery unit 50 accesses the ingress characteristics 61 of the memory 60 to determine the initial ingress packet characteristic α .

[47] The discovery unit 50 is further configured, in an embodiment, to receive the initial egress packet 13 after processing by the control plane processor 200. The initial egress packet 13 received by the discovery apparatus from the control plane processor 200 corresponds to the initial ingress packet 12 passed to the control plane processor 200 from the checking unit 40. The discovery unit 50 determines the network action from the control plane processor by comparing the initial ingress packet characteristic α to the initial egress packet characteristic β . The discovery unit 50 stores the network action in the memory 60.

[48] According to an example embodiment the control plane processor 200 passes the initial egress packet to the packet output unit 30. According to an example embodiment the discovery unit 50 passes the initial egress packet to the packet output unit 30.

[49] According to an embodiment, a processing core (reference numeral 610 in FIG. 6, discussed in more detail below) sets a flow state in the memory 60 associated with a flow of the initial ingress packet 11 to be in the monitor state 510 when the checking unit 40 determines that the subsequent ingress packet characteristic γ is not stored in the memory 60.

[50] According to an example embodiment, the processing core(not shown) sets a flow state in the memory 60 associated with a flow of the subsequent ingress packet 13 to be

in the monitor state 510 when the checking unit 40 determines that the subsequent ingress packet characteristic γ is not stored in the memory 60.

[51] According to an example embodiment, the processing core increments the monitor thresholds 64, the learned thresholds 65, and the timed thresholds 67 of the memory 60.

[52] According to an example embodiment, the processing core increments the monitor thresholds 64, when a packet flow state is set to the monitor state 510, corresponding to a packet flow 15 when the discovery unit 50 determines for a corresponding initial egress packet 13 that a change from an initial ingress packet characteristic α to an initial egress packet characteristic β matches the network action, corresponding to the packet flow 15, stored in the memory 60.

[53] Fig. 3 illustrates that the processing core (not shown) determines that the incremented monitor threshold 65 corresponding to a packet flow 15 has reached a configurable threshold N (in this example, the threshold N is 10). The processing core subsequently changes the state of the packet flow 15 from the monitor state 510 to the learned state 520.

[54] According to an example embodiment, the processing core increments the learned thresholds 65, when a packet flow state is set to the learned state 520, corresponding to a packet flow 15, when the checking unit 40 determines for an initial ingress packet 11 that its corresponding initial ingress packet characteristic α matches the ingress characteristic stored in the memory 60.

[55] According to an example embodiment, the processing core determines whether the incremented learned threshold 65 corresponding to a packet flow 15 has reached a configurable threshold M. The processing core subsequently changes the state of the packet flow 15 from the learned state to the monitor state 510.

[56] Fig. 4 illustrates that the processing core increments the timed threshold 67. The increments of the timed threshold 67 correspond to a period of time elapsed since the packet flow was changed from the monitor state to the learned state. According to an example embodiment, the processing core increments the timed threshold 67 to correspond to a period of time from when the packet flow state was set to the learned state. According to an example embodiment, the processing core 610 increments the timed threshold to correspond to a period of time from when the most recent packet of the corresponding packet flow 15 was received by the checking unit 40.

[57] According to an example embodiment, when the processing core determines that the incremented timed threshold 67 has reached a configurable threshold T, the processing core subsequently changes the state of the packet flow 15 from the learned state to the monitor state.

[58] Fig. 5 illustrates a state diagram 500 showing both states and transition states for a packet flow. The state diagram 500 includes the monitor state 510, the learned state 520, and the unstable state 530 in an example embodiment. When a flow state is set to the monitor state 510, any initial ingress packets 12 corresponding to the packet flow 15 are "monitored."

[59] According to an example embodiment, the discovery unit discovery unit 50 determines the network action of the control plane processor on a packet associated with the initial ingress packet characteristic α and initial egress packet characteristic β , as shown in Fig. 1. The discovery unit 50 subsequently stores the network action in the memory 60 as one of the network actions 63.

[60] Fig. 5 also illustrates a flow state of a packet flow 15 in the learned state 520. An initial ingress packet 12 of corresponding to the packet flow is determined to be "learned."

[61] In the learned state 520 the initial ingress packet 12 of a learned flow is not passed to the control plane processor 200, however, this is a non-limiting embodiment, and an initial ingress packet of a learned flow may be passed to the control plane processor 200 for verification purposes and the like. According to an example embodiment, when the processing core determines that the transition condition 501 is met, the N, a number matching network actions corresponding to a packet flow, is less than or equal to a first configurable threshold. When the processing core determines that the transition condition 501 is met, the packet flow state remains in the monitor state 510.

[62] According to an example embodiment, when the processing core determines that the transition condition 502 is met, the N has exceeded the threshold T1, the state of the packet flow 15 is changed to the learned state 520.

[63] According to an example embodiment, when the processing core determines that the transition condition 503 is met, the M, a counter corresponding to a packet flow in a learned state 520, is less than or equal to a second configurable threshold, T2. When

transition condition 502 is met, the corresponding packet flow remains in the learned state 520.

[64] According to an example embodiment, when the processing core determines that the transition condition 504 is met, the M has exceeded T2, the packet flow state is changed to the monitor state 510.

[65] According to an example embodiment, the counter M corresponds to a number of packets received by the checking unit 40 while the corresponding packet flow is in the learned state. According to another example embodiment, the counter M corresponds to a passage of time since the packet flow was changed to the learned state. According to another example embodiment, the counter M corresponds to a passage of time since the most recent packet of the corresponding packet flow was received by the checking unit 40. According to another example embodiment, the counter M may be any combination of the above example embodiments.

[66] According to an example embodiment, when the processing core determines that an egress characteristic of a packet does not match the egress characteristic of another packet of the same flow, the processing core increments a counter X. When X reaches a configurable threshold T0, the processing core implements transition condition 500 thereby setting the corresponding packet flow to the unstable state 530.

[67] According to an example embodiment, any threshold data corresponding to a packet flow 15 stored in the memory 60 is erased when the state of the packet flow 15 is changed. According to an example embodiment, any threshold data corresponding to a packet flow 15 in the memory 60 is not erased when the state of the packet flow 15 is

changed. According to an example embodiment the configurable thresholds are reconfigured at any time.

[68] Fig. 6 depicts a System on Chip implementation 600 wherein the components of network device 1000 correspond to each other through the processing core 610.

[69] The network device 1000 includes a checking unit 620, an input storage unit 630, a count control unit 640, a discovery unit 650, and a storage I/O 670. The network device 1000 also includes various memories such as ROM 685, RAM 686, and other memory 687. The memories store data related to network actions and are accessible to the other components of the network device 1000.

[70] The input storage unit 630 and storage I/O 670 access the memories to perform read/write operations related to at least the network device 1000, and further carry out read/write operations related count control unit 640 which performs incremented counting operations. The components of the network device 1000 communicate with each other via the processing core 610 which transmits various indicators and signals corresponding to monitored and learned packet flows.

[71] According to an example embodiment, the processing core 610 allows the network device 1000 to operate as a hardware component of the network device 1000. According to an example embodiment, the processing core 610 allows the network device 1000 to operate as a software component of the network device 1000. According to an example embodiment the checking unit 620 is a hardware unit, and according to another example embodiment the checking unit 620 is a software unit, e.g. in Linux software.

[72] Fig. 6 also illustrates the input storage unit 630. According to an example embodiment, the processing core 610 uses the input storage unit 630 to perform memory operations corresponding to the memory 687 which may be subsequently accessed by the .

[73] Fig. 6 also illustrates the count control unit 640. According to an example embodiment, the processing core 610 uses the count control unit 640 to increment counting operations corresponding to the transition condition of state diagram 500 and also the various states associated with the packet flows.

[74] Fig. 6 also illustrates the storage I/O 670. According to an example embodiment, the processing core 610 uses the storage I/O 670 to access various memories such as ROM 685, RAM 686, and other memory 687.

[75] Fig. 7 is a flow diagram of an example algorithm and method according to example embodiments when a packet is received by the checking unit. The example method of Fig. 7 may be applied to multiple example embodiments wherein the network device is utilized. At S701, the network device receives an initial ingress packet.

[76] At S702, the network device determines the ingress characteristic of the packet of the initial ingress packet header. Processing then continues to S703.

[77] At S703, if the ingress characteristic is found in the memory , processing continues to S705. On the other hand, if the ingress characteristic is not found in the memory, processing continues to S704.

[78] At S704, since the ingress characteristic is not stored, it is written in the memory as a new row. Following this, at S706, the flow state corresponding to the newly-entered

ingress characteristic is set to the monitor state. Processing then continues via connector B to that shown in Fig. 8.

[79] Returning to S705, when the ingress characteristic was found in the memory, it is determined whether the corresponding state is in the monitor state or the learned state. If the state corresponding to the found ingress characteristic is the monitor state, processing continues via connector B to that shown in Fig. 8. On the other hand, if the corresponding state is the learned state, processing continues via connector D to that shown in Fig. 10.

[80] Fig. 8 is a flow diagram of an example algorithm for a packet of a flow that is set to the monitor state and method according to example embodiments wherein the network device is utilized. The example method of Fig. 8 may be applied to multiple example embodiments. At S801, the network device passes an initial ingress packet to the control plane processor.

[81] At S802, the network device intercepts an egress packet destined to the control plane processor. The egress packet corresponding to the initial ingress packet sent to the control plane processor at S801. According to an embodiment, the egress packet is also sent to the discovery apparatus. At S803, the network device determines what network action was applied to the initial ingress packet by comparing a characteristic of the initial ingress packet with a corresponding characteristic of the egress packet and determining changes made to the packet by the control plane processor, for example changes to the packet header.

[82] At S803, according to multiple non-limiting example embodiments, the network device determines the network action based on any of the methods, or by any

combination of the methods, shown in Figs. 11-13 which illustrates multiple algorithms to each determine a specific type of change to the packet header.

[83] At S804, the network device determines if the determined network action for the corresponding flow is found stored in a memory.

[84] At S805, when the network action is not stored, the network device stores the network action, and at S808, will forward the packet based on the network action designated by the control plane processor, and continue processing via connector E as shown in Fig. 7.

[85] At S806, when the network action is stored, the network device will determine if the stored network action matches the determined network action.

[86] At S808, when the stored network action does not match the determined network action, the network device will, forward the packet based on the network action designated by the control plane processor and processing will continue via connector E as shown in Fig. 7.

[87] Returning to S806, when the stored network action matches the determined action, then the processing continues via connector C to that shown in Fig. 9.

[88] Fig. 9 is a flow diagram of an example algorithm and method according to example embodiments. The example method of Fig. 9 may be applied to multiple example embodiments wherein the network device is utilized At S901, the network device determines whether a stored network action matches a presently determined network action for a corresponding flow and subsequently increments the counter N.

[89] At S902, the network device 1000 determines if the counter N, stored in the memory 60, has reached a configurable threshold.

[90] At S903, when the network device 1000 determines that the counter N has reached the configurable threshold, the network device 1000 changes the flow state corresponding to the determined network action to the learned state 520, forwards the packet at S904, and continue processing via connector E as shown in Fig. 7.

[91] At S903, when the network device 1000 determines that the counter N has not reached the configurable threshold, at S904, the network device 1000 forwards the packet and processing continues via connector E as shown in Fig. 7.

[92] Fig. 10 is a flow diagram of an example algorithm and method according to example embodiments. The example method of Fig. 10 may be applied to multiple example embodiments. At S1001, when the checking unit determines that an ingress packet belongs to a packet flow which is classified as in the learned state 520, the checking unit applies the stored network action to the ingress packet corresponding to the flow. At S1002, the network device then performs the corresponding action and forwards the packet corresponding to the network action, and in parallel to S1002, at S1003 the network device increments the learned state threshold counter M.

[93] At S1005, the network device determined that the learned state threshold counter M reached the learned state threshold T2 and changed the flow state to the monitor state. Processing continues via connector E as shown in Fig. 7 whereby the network device is waiting for another ingress packet.

- [94] Fig. 11 is a flow diagram of an example algorithm and method according to example embodiments whereby the network device monitors VLAN tag characteristics of a packet prior to and post-control plane processing. The example method of Fig. 11 may be applied to multiple example embodiments. At S1100, the discovery apparatus network device attempts to discover the network action as discussed with respect to S803.
- [95] At S1101, the network device determines if the ingress packet has a VLAN tag.
- [96] At S1102, the network device determined that the ingress packet has a VLAN tag, and determines whether the egress packet has a VLAN tag.
- [97] At S1103, when the network device determines that the egress packet does not have a VLAN tag, it defines a new network action to remove the VLAN tag, and at S1110, stores the new action.
- [98] At S1106, when the network device determines that the egress packet has a VLAN tag, it also determines whether the ingress and egress VLAN tags are equal.
- [99] At S1107, when the network device determines that the VLAN tag are equal, it defines a new network action to do nothing, and at S1110 stores the new action.
- [100] At S1109, when the network device determines that the VLAN tags are not equal, it defines a new network action to change the VLAN tag, and at S1110, stores the new action.
- [101] Returning to S1101, when the network device determines that the ingress packet does not have a VLAN tag, at S1104, the network device determines whether the egress packet has a VLAN tag.

[102] At S1105, when the network device determines that the egress packet does not have a VLAN tag, it defines a new network action to do nothing, and at S1110, stores the new action.

[103] At S1108, when the network device determines that the egress packet does have a VLAN tag, it defines a new network action to add the VLAN tag, and at S1110, stores the new action.

[104] Fig. 12 is a flow diagram of an example algorithm and method according to example embodiments whereby the network device monitors destination IP and port characteristics of a packet prior to and post-control plane processing. The example method of Fig. 12 is applied to multiple example embodiments wherein the network device is utilized. At S1200, the discovery apparatus network device attempts to discover the network action as discussed with respect to S803.

[105] At S1201, the network device receives the destination IP and port addresses found in the header of an ingress packet.

[106] At S1202, the network device receives the destination IP and port addressed found in the header of an egress packet corresponding to the ingress packet which has been passed to the discovery apparatus from the control plane processor.

[107] At S1203, the network device determines whether the destination IP address of an ingress packet matches the destination IP address of the corresponding egress packet.

[108] At S1204, when the network device determines that the destination IP addresses do not match for corresponding ingress and egress packets, it defines a new network

action for routing based on the IP address change in the egress packet header, and at S1208, stores the network action.

[109] At S1205, when the destination IP addresses of corresponding ingress and egress packets match, the network device determines whether the destination ports also match for corresponding ingress and egress packets..

[110] At S1206, when the network device determines that the destination ports do not match for the ingress and egress packets, it defines a new network action for routing based on the port change in the egress packet header, and at S1208, stores the network action.

[111] At S1207, when the network device determines that the destination ports do match for the ingress and egress packets, it defines a new network action to do nothing, and at S1208, stores the network action.

[112] Fig. 13 is a flow diagram of an example algorithm and method according to example embodiments whereby the network device monitors PPPoE tunnel characteristics of a packet prior to and post-control plane processing. The example method of Fig. 13 may be applied to multiple example embodiments as discussed wherein the network device is utilized At S1300, the discovery apparatus network device 1000 attempts to discover the network action regarding PPPoE tunnel characteristics as discussed with respect to S803.

[113] At S1301, the network device determines whether the ingress packet has a PPPoE header.

[114] At S1302, when the network device determines whether the egress packet has a PPPoE header.

[115] At S1303, when the network device determines that the egress packet does not have a PPPoE header, because there is a different between the PPOE header of the ingress packet and the corresponding egress packet it defines a new network action to remove the PPPoE tunnel, and at S1310, stores the network action.

[116] At S1306, when the network device determines that the egress packet does have a PPPoE header it further determines whether the ingress and egress PPPoE headers match.

[117] At S1307, when the network device determines that the ingress and egress PPPoE headers do match, it defines a new network action to do nothing, and at S1310, stores the network action.

[118] At S1309, when the network device determines that the ingress and egress PPPoE headers do not match, it defines a new network action to translate the PPPoE header based on the egress packet, and at S1310, stores the network action.

[119] Returning to S1301, when the network device determines that the ingress packet does not have a PPPoE header, at S1304, the network device 1000 determines whether the egress packet has a PPPoE header.

[120] At S1305, when the network device determines that the egress packet does not have a PPPoE header, defines a new network action to do nothing, and at S1310 stores the network action the case when both the ingress and corresponding egress packet do not have a PPPoE header.

[121] At S1308, when the network device determines that the egress packet does have a PPPoE header, even though the corresponding ingress packet does not have a PPPoE header, it defines a new network action to add a PPPoE tunnel to the egress packet, and at S1310, stores the network action.

[122] Fig. 14 shows the network device 1000 in which the network device 1000 has entered into a configurable testing mode. According to an example embodiment, the processing core (not shown) causes the checking unit 40 to produce the pseudo-ingress packet 14 mimicking a packet of the packet flow 15 which is in the learned state. The pseudo-ingress packet 14 is sent to the control plane processor 200. According to an example embodiment, a subsequent initial egress packet 13 is received by the discovery unit 50 from the control plane processor 200. The processing core 610 determines the network action applied to the pseudo-ingress packet 14.

[123] According to an example embodiment, the pseudo-ingress packet 14 received the same network action from the control plane processor 200 as stored by the memory 60, and the processing core 610 returns the network device 1000 to normal operation.

[124] According to an example embodiment, the pseudo-ingress packet 14 receives a different network action from the control plane process 200 in view of the network action stored by the memory 60. The processing core 610 changes the state of the corresponding flow to the monitor state 510, and the network device 1000 returns to normal operation.

[125] It is noted that the network device 1000 allows for an increased network throughput rate in situations where packet flows are present. The network device 1000

obviates the requirement that network actions be applied to every packet by the typically slower path represented by the control plane processor 200. When a packet flow is learned, the appropriate network action is applied to subsequent packets in the flow by the checking unit, resulting in a typically much faster processing path.

[126] From the foregoing, it is noted that the network device 1000 treats the control plane processor as a unit., the network device 1000 therefore allows for discovery of control rules which are inferred from raw packet flows. The network device 1000 also allows for non-intrusive discovery of network actions independent of the particularities of the control plane processor and its implementation details. Thus, particularities such as custom extensions, kernel versions, and the like, all of which are closely related to the control plane processor, are implemented and updated in the control plane processor without any need for change at the discovery apparatus, in an embodiment. Furthermore, the memory of the network device 1000 allows for a unified representation of control plane information while reducing the complexity of data structures which describe the network topology. That is to say, the control plane processor contains complex rules for handling packets of a flow, however, the network device 1000 simplifies the complex rules of the control plane processor into

[127] Fig. 15 illustrates an example embodiment wherein the network device 1000 is integrated as a System on a Chip (SoC) 100. The SoC 100 includes substantially similar components to those discussed with respect to Fig. 1, including the packet input unit 20, packet output unit 30, checking unit 40, discovery unit 50, memory 60, and control plane processor 200. The ingress port of the packet input unit 20 is illustrated as being on the

left side of the packet input unit 20 and located at a border of the SoC 100, however, this is merely an example embodiment and the ingress port of the packet input unit 20 is located entirely within the network device 1000 in an example embodiment, and the ingress port of the packet input unit 20 is located entirely outside of the network device 1000 in an example embodiment.

[128] Fig. 16 illustrates an example embodiment wherein the network device 1000 is substantially similar to that shown in Fig. 3. A subsequent ingress packet 13 arrives at the packet input unit 20. The subsequent ingress packet characteristic γ corresponds to the characteristic stored in memory 60 in the ingress characteristic 61 column. The discovery unit 50 receives the subsequent egress packet from the control plane, however, the subsequent egress packet characteristic does not correspond to the subsequent egress packet characteristic stored in the memory in the egress characteristic column 62. The processing core 610 (not shown) is configured to set the flow state of the corresponding flow to the unstable state, as shown in state 66 column. According to an embodiment, any packets corresponding to a packet flow set to the unstable state are sent to the control plane processor by the network device 1000 and are not sent to the discovery apparatus 50. The network device 1000 implements counters similar to those for the learned state for setting a flow in a unstable state to a flow in a monitor state. Further, the network device 1000 counts to a threshold number of a plurality of subsequent egress packet with characteristics not corresponding to the characteristic stored in the memory 60.

[129] Fig. 17 illustrates an example embodiment wherein the network device 1000 operates with separate memories, memory 82 and memory 84, corresponding to checking

unit 40 and discovery unit 50 respectively. The network device 1000 of Fig. 17 further illustrates the processing core 610 communicatively coupled to the packet input unit 20, the packet output unit 30, the checking unit 40, the discovery unit 50, the memory 82, the memory 84, and the control plane processor 200.

[130] Although the inventive concept has been described above with respect to the various example embodiments, it is noted that there can be a variety of permutations and modifications of the described features by those who are familiar with this field, without departing from the technical ideas and scope of the features, which shall be defined by the appended claims.

[131] Further, while this specification contains many features, the features should not be construed as limitations on the scope of the disclosure or the appended claims. Certain features described in the context of separate embodiments can also be implemented in combination. Conversely, various features described in the context of a single embodiment can also be implemented in multiple embodiments separately or in any suitable sub-combination.

[132] Although the drawings describe operations in a specific order and/or show specific arrangements of components, one should not interpret that such specific order and/or arrangements are limited, or that all the operations performed and the components disclosed are needed to obtain a desired result. Accordingly, other implementations are within the scope of the following claims.

What is Claimed is:

1. A network device, comprising:

a packet input unit configured to receive ingress packets, the ingress packets having an ingress packet identifying characteristic;

a checking unit configured to determine whether the identifying characteristic of a received ingress packet matches a stored identifying characteristic of a packet flow that is stored in a memory, to perform on the received ingress packet a network action associated with the stored identifying characteristic when the identifying characteristic of the received ingress packet matches the stored identifying characteristic, and to provide the received ingress packet to a control plane processor configured to process the received ingress packet to determine the network action to be performed on the received ingress packet when the identifying characteristic of the received ingress packet does not match the stored identifying characteristic; and

a discovery unit configured to intercept an egress packet received from the control plane processor, the egress packet corresponding to the received ingress packet, to determine one or more differences in selected portions of a header portion of the received ingress packet resulting from processing at the control plane processor, and to store in the memory a new network action based on the one or more differences.

2. The network device of claim 1, wherein the checking unit is further configured to set the packet flow to a first state in response to an indication that the identifying characteristic of one of the received ingress packets does not match any said stored identifying characteristic.
3. The network device of claim 1, wherein the discovery unit is further configured to determine when the new network action is applied to a subsequent egress packet received from the control plane processor.
4. The network device of claim 3, wherein the processing core is further configured to determine when the new network action is applied to a number of subsequent egress packets.
5. The network device of claim 4, wherein the checking unit is further configured to set the packet flow, in response to an indication that the network action is applied to the number of subsequent egress packets, to a second state in which the discovery unit does not intercept subsequent packets of the packet flow from the control plane processor.
6. The network device of claim 1, wherein:
 - a processing core is configured to generate a pseudo-packet, having the stored identifying characteristic, for communication to the control plane processor;
 - the discovery unit is further configured to intercept the egress packet, from the control plane processor, corresponding to the pseudo-packet, and to determine the resulting network action applied to the pseudo-packet by the control plane processor; and

the processor core is further configured to verify the resulting network action with reference to the network action stored in association with the stored identifying characteristic.

7. The network device of claim 5, wherein the checking unit is further configured to reset the data flow state when a number of network actions have been applied to packets of the packet flow associated with the stored identifying characteristic.

8. The network device of claim 1, wherein the one or more differences correspond to differences in a plurality of different fields of the header subsequent to processing by the control plane processor.

9. The network device of claim 8, wherein the plurality of different fields of the header correspond to any of a VLAN tag, a destination IP, a port, and a PPPoE tunnel.

10. The network device of claim 1, wherein the checking unit is further configured to set the packet flow, in response to an indication a subsequent network action associated with a subsequent packet of the packet flow does not match the stored network action, to a third state in which the discovery unit does not intercept subsequent packets of the packet flow from the control plane processor.

11. A network device method, comprising:
receiving an ingress packet having an ingress packet identifying characteristic;

detecting with a checking unit whether the identifying characteristic matches a stored identifying characteristic of stored first packet flow information;

when the match is detected, performing on the received ingress packet a stored network action, using the checking unit, to output an egress packet, the stored network action corresponding to the stored identifying characteristic;

when the match is not detected, communicating the ingress packet to a control plane processor configured to process the ingress packet;

intercepting from the control plane processor an egress packet corresponding to the communicated ingress packet, with a discovery unit;

detecting with the discovery unit one or more differences between at least part of a header portion of the communicated ingress packet and a header portion of the intercepted egress packet; and

storing in a memory a new network action based on the one or more differences, in association with the unmatched identifying characteristic, as second packet flow information.

12. The method of claim 11, further comprising responding to an indication, that the identifying characteristic of one of the received ingress packets does not match any said stored identifying characteristic, by setting the packet flow to a first state.

13. The method of claim 11, further comprising determining when the new network action is applied to a subsequent egress packet received from the control plane processor.

14. The method of claim 13, further comprising determining when the new network action is applied to a number of subsequent egress packets.

15. The method of claim 14, further comprising responding to an indication that the network action is applied to the number of subsequent egress packets by setting the packet flow to a second state wherein the discovery unit does not intercept subsequent packets of the packet flow from the control plane processor.

16. The method of claim 14, further comprising:

with the checking unit, generating a pseudo-packet having one of the stored identifying characteristics, for communication to the control plane processor;

with the discovery unit, receiving the egress packet, from the control plane processor, corresponding to the pseudo-packet;

with the discovery unit, detecting the resulting network action applied to the pseudo-packet by the control plane processor; and

verifying the resulting network action with reference to the network action stored in association with the one of the stored identifying characteristics.

17. The method of claim 15, further comprising resetting the data flow state when a number of network actions have been applied to packets of the packet flow associated with the stored identifying characteristic when the identifying characteristic of the received ingress packet matches the stored identifying characteristic.

18. The method of claim 11, further comprising the discovery unit detecting the one or more differences corresponding to differences in a plurality of different fields of the header subsequent to processing by the control plane processor.

19. The method of claim 18, wherein the plurality of different fields of the header correspond to any of a VLAN tag, a destination IP, a port, and a PPPoE tunnel.

20. The method of claim 11, further comprising responding to an indication, that a subsequent network action associated with a subsequent packet of the packet flow does not match the stored network action, by setting the packet flow to a third state wherein the discovery unit does not intercept subsequent packets of the packet flow from the control plane processor.

FIGURE 1

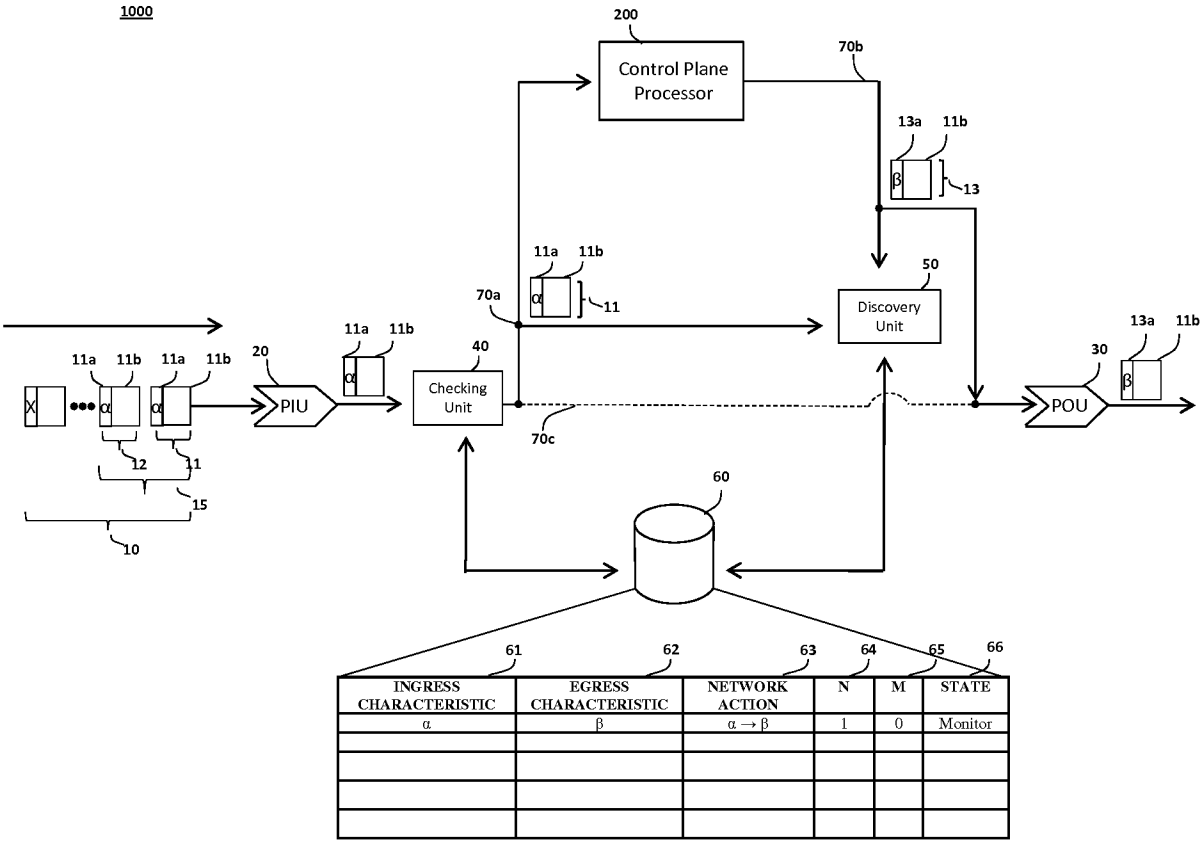


FIGURE 2

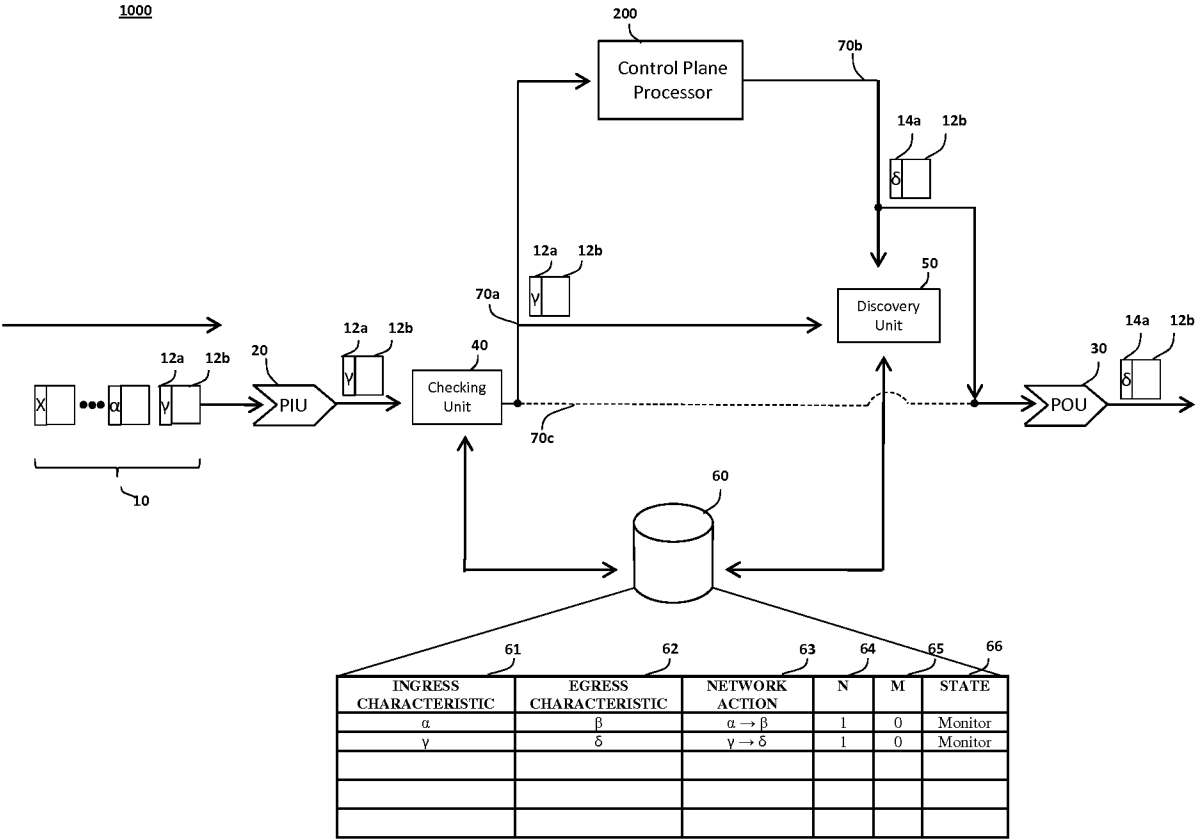


FIGURE 3

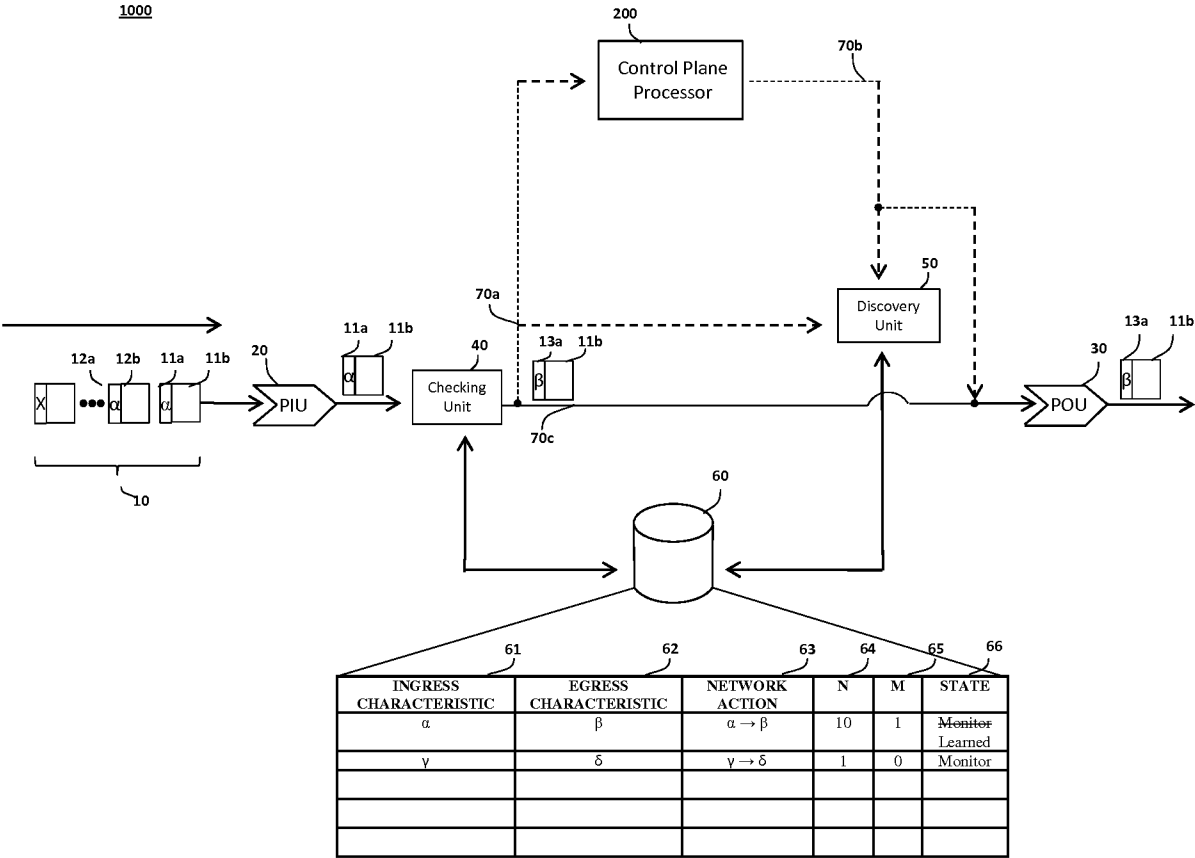
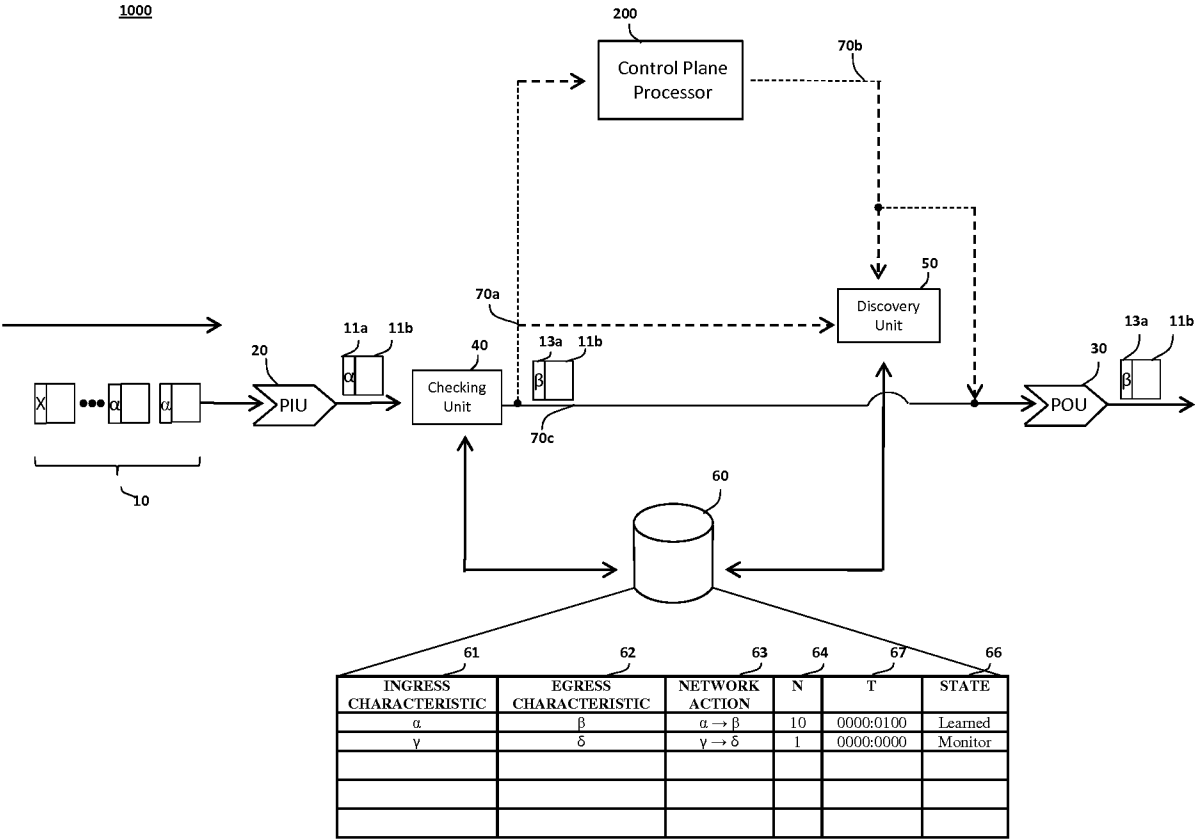
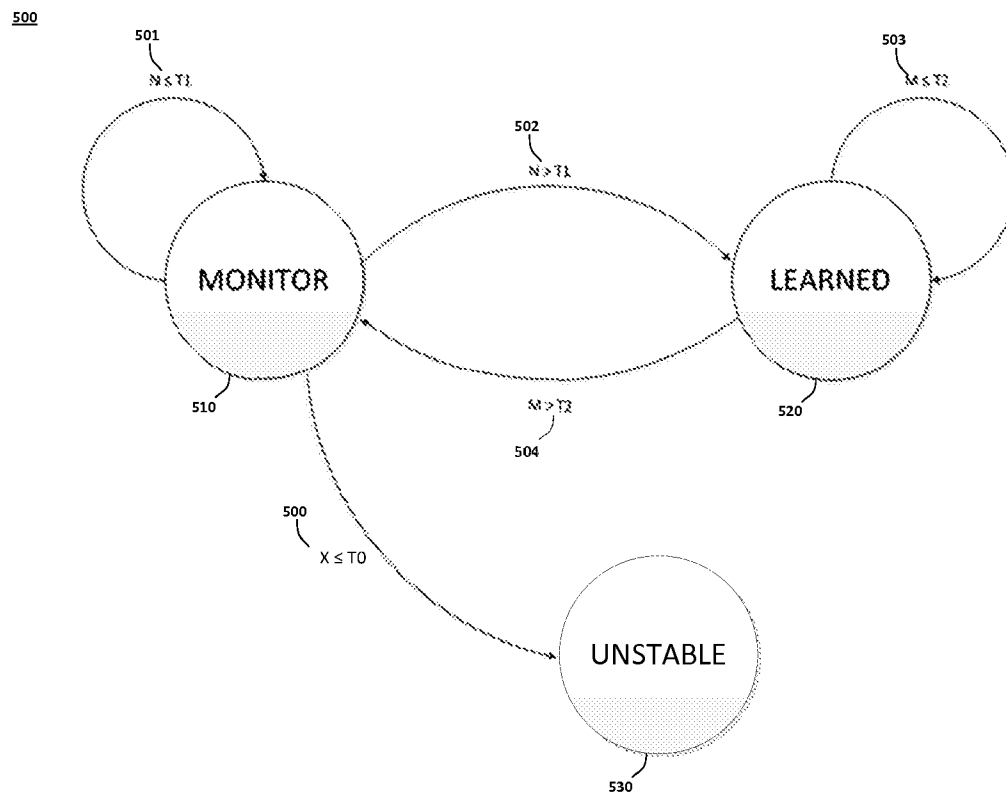


FIGURE 4



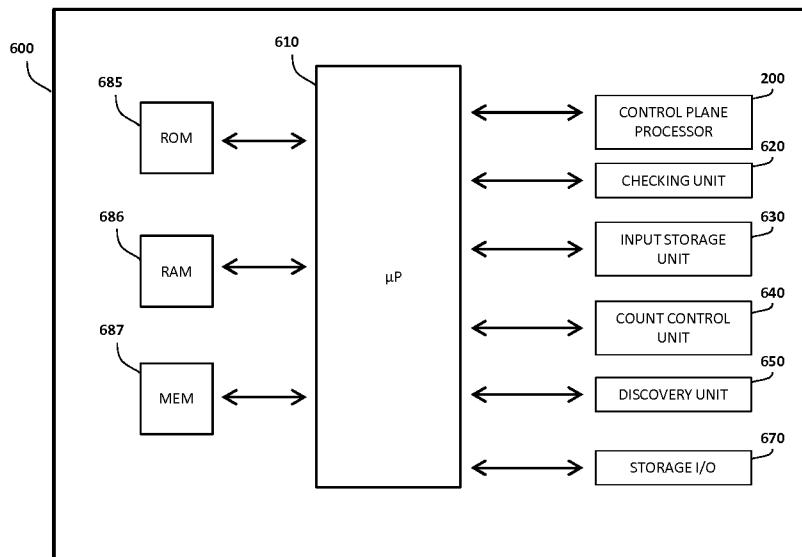
5/17

FIGURE 5



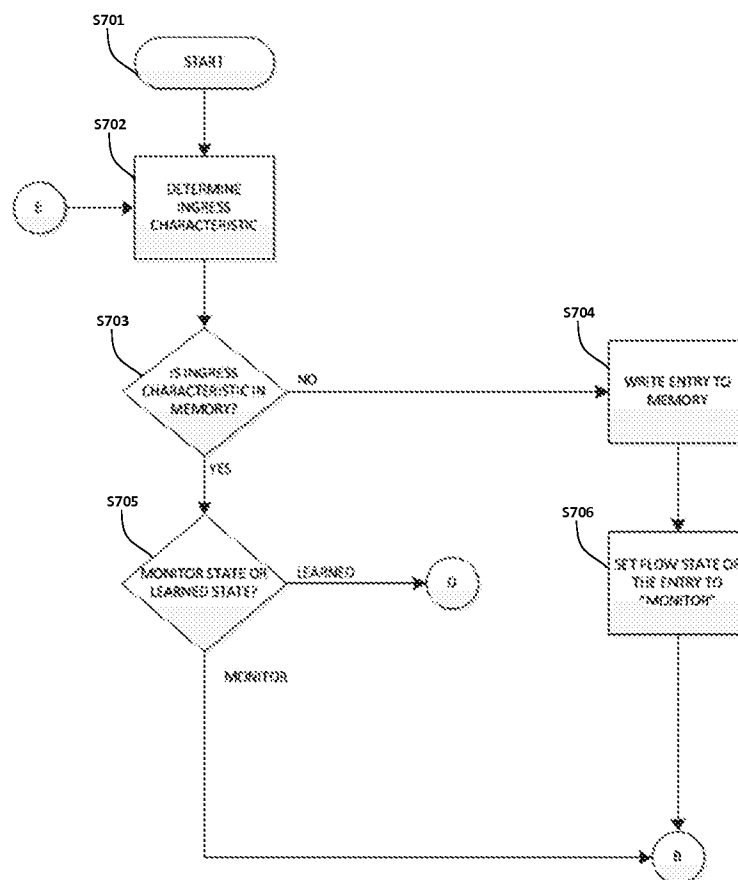
6/17

FIGURE 6

1000

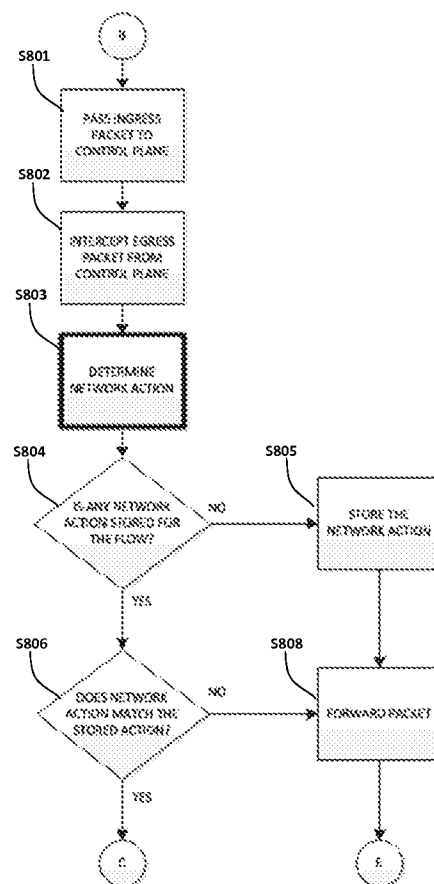
7/17

FIGURE 7



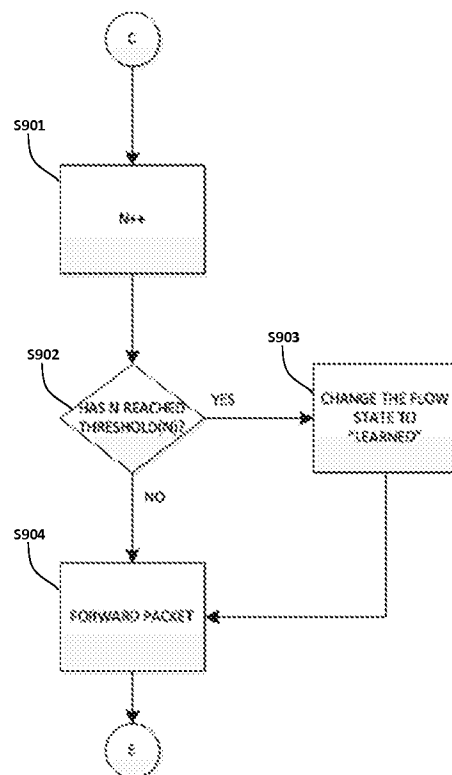
8/17

FIGURE 8



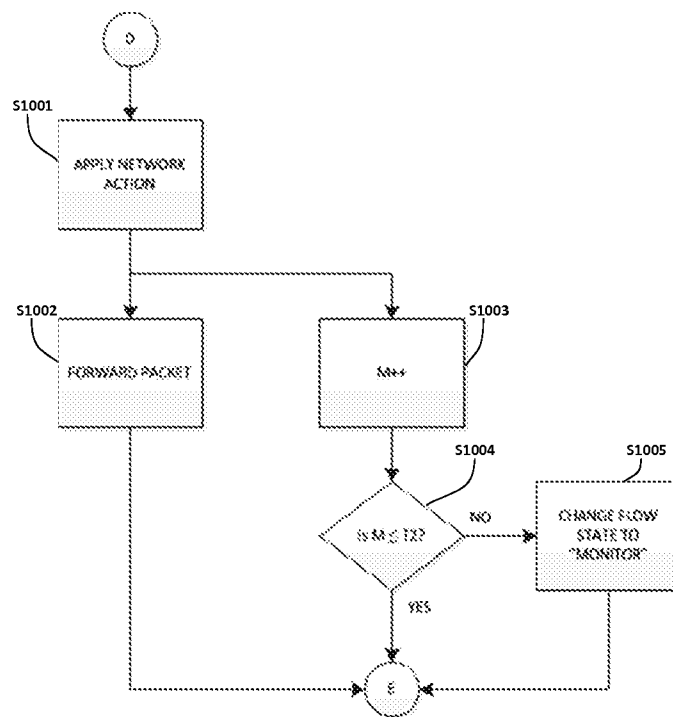
9/17

FIGURE 9



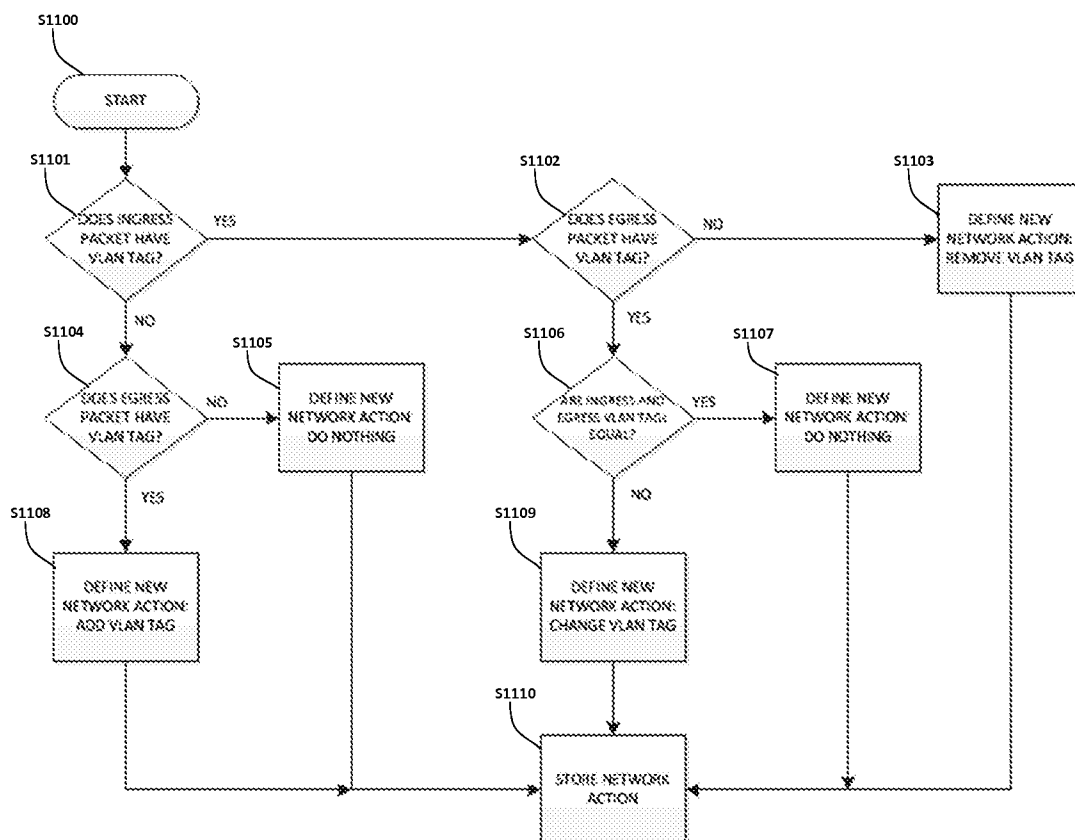
10/17

FIGURE 10



11/17

FIGURE 11



12/17

FIGURE 12

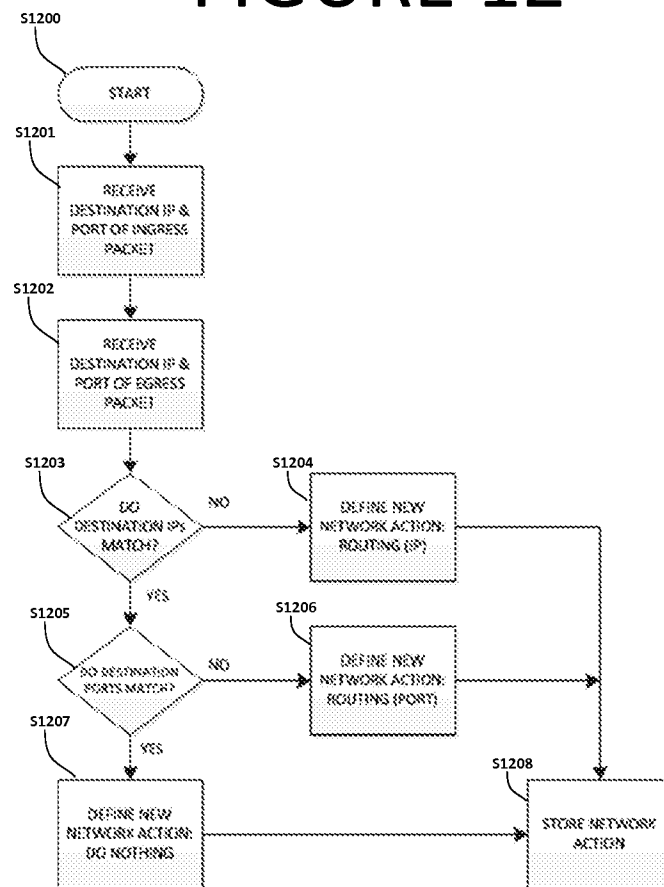


FIGURE 13

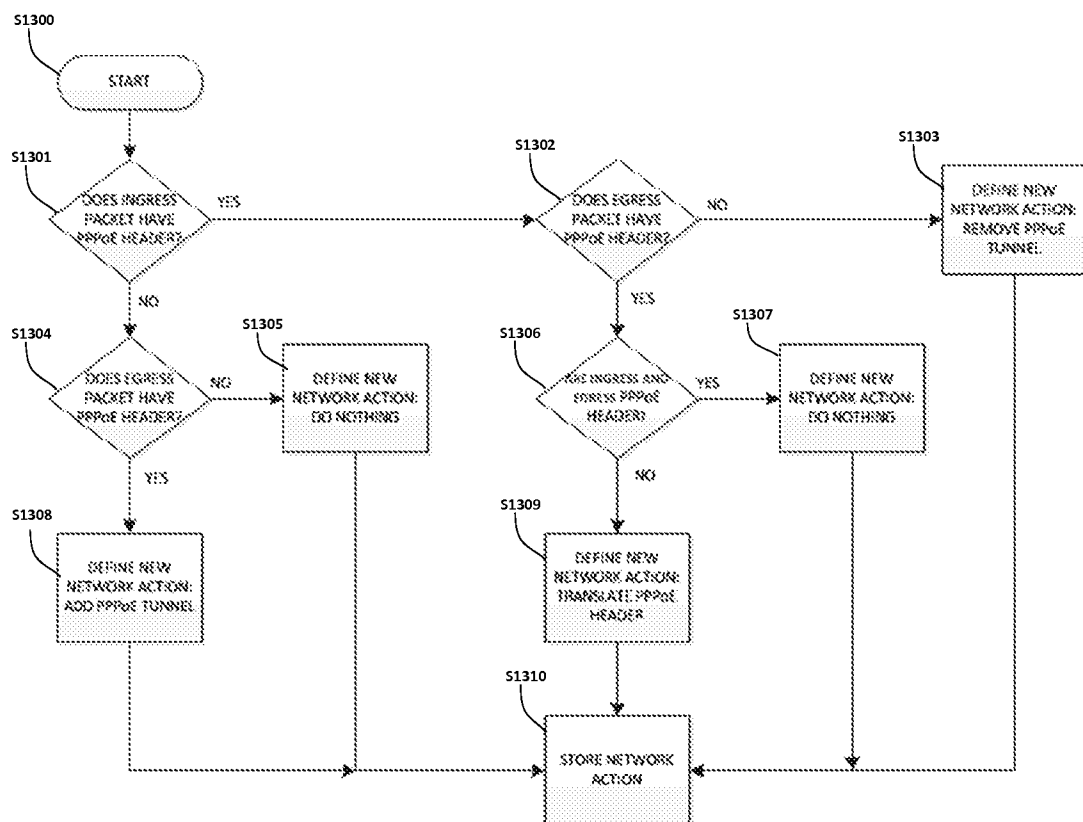


FIGURE 14

1000

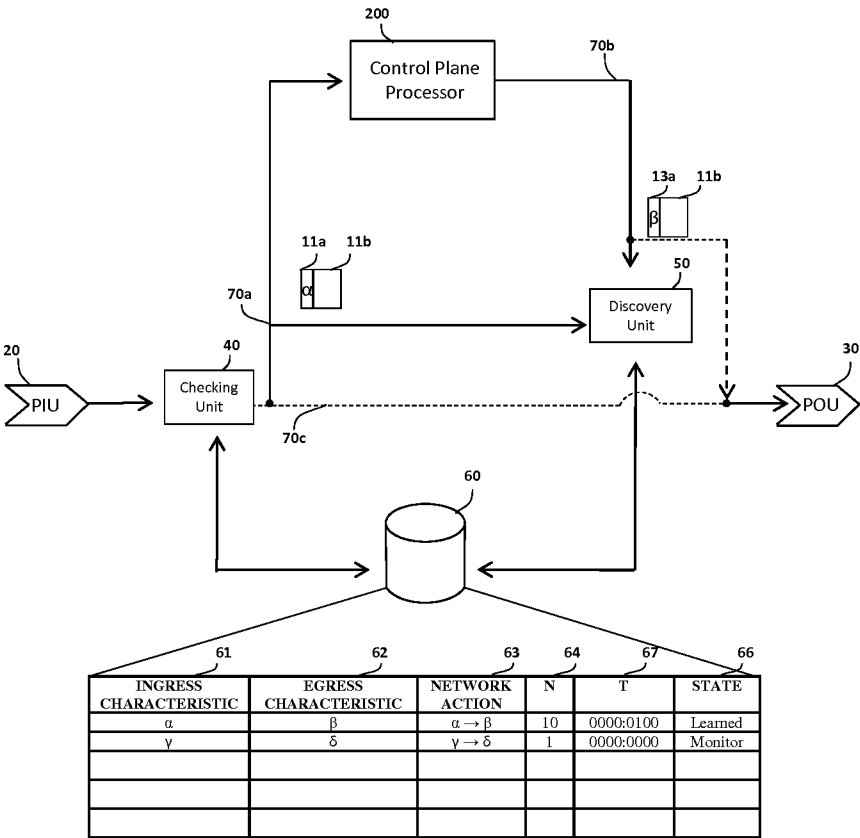


FIGURE 15

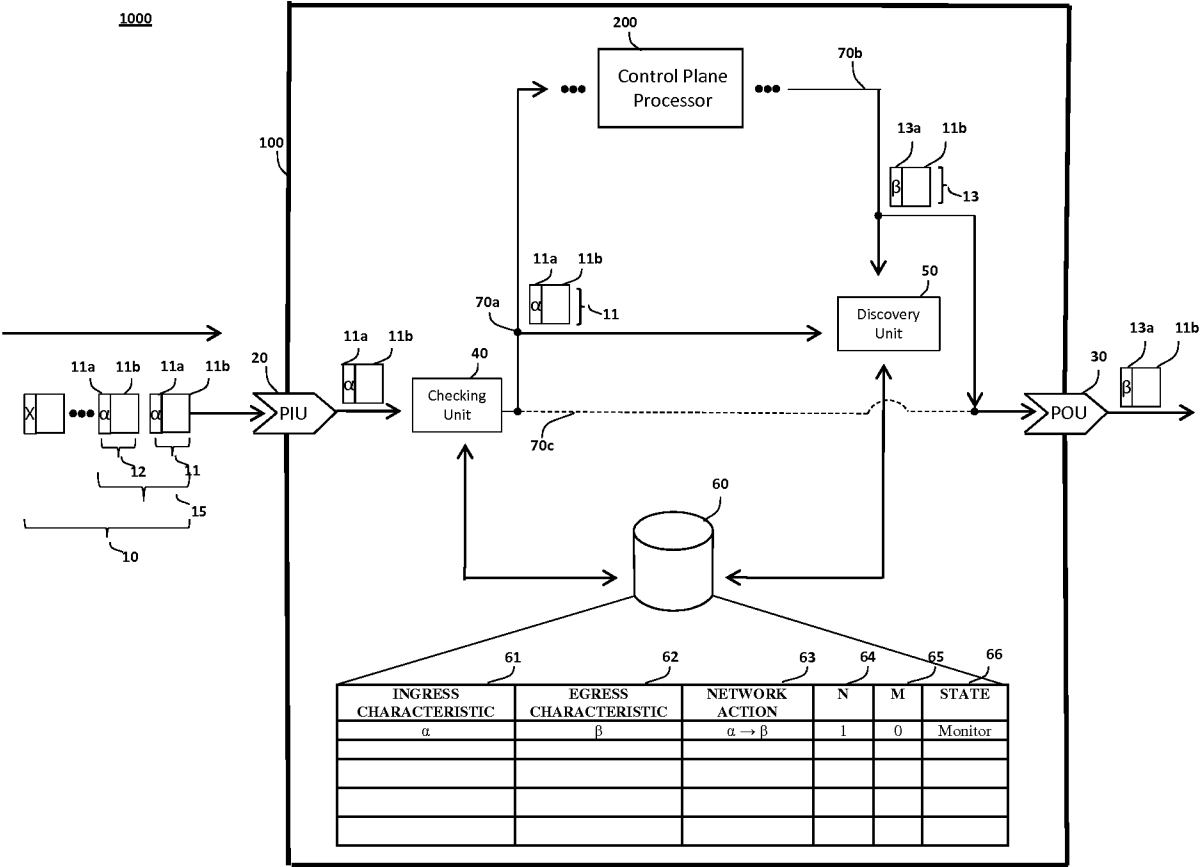


FIGURE 16

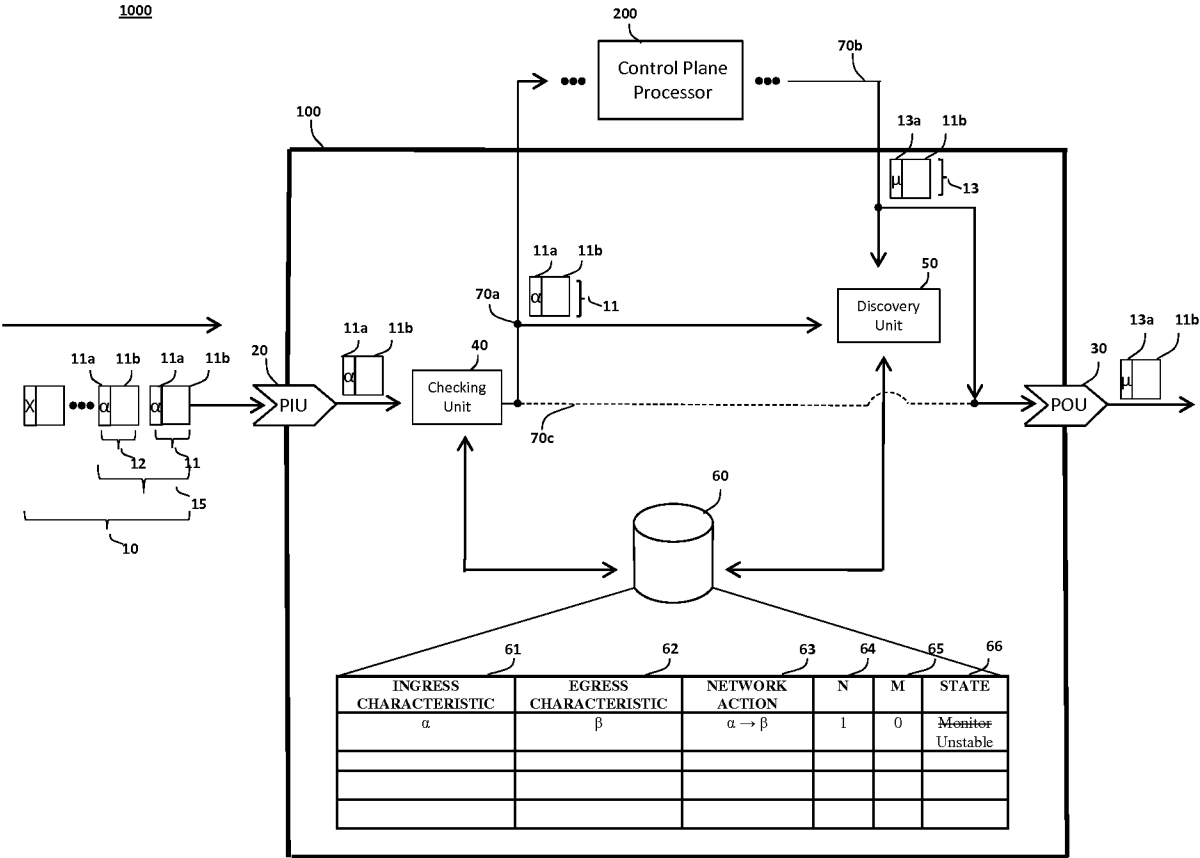


FIGURE 17

