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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
12 October 2017 (12.10.2017)

WIPO | PCT

(10) International Publication Number
WO 2017/175033 A1

(51) International Patent Classification:
H04L 12/755 (2013.01) *H04L 12/707* (2013.01)

(21) International Application Number:
PCT/IB2016/051951

(22) International Filing Date:
6 April 2016 (06.04.2016)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: **TELEFONAKTIEBOLAGET LM ERICSSON (PUBL)** [SE/SE]; 164 83 Stockholm (SE).

(72) Inventor: **RAJURE, Abhay**; 300 Holger Way, San Jose, California 95134 (US).

(74) Agents: **DE VOS, Daniel M.** et al.; Nicholson De Vos Webster & Elliott LLP, 99 Almaden Boulevard, Suite 710, San Jose, CA 95113 (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, 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.

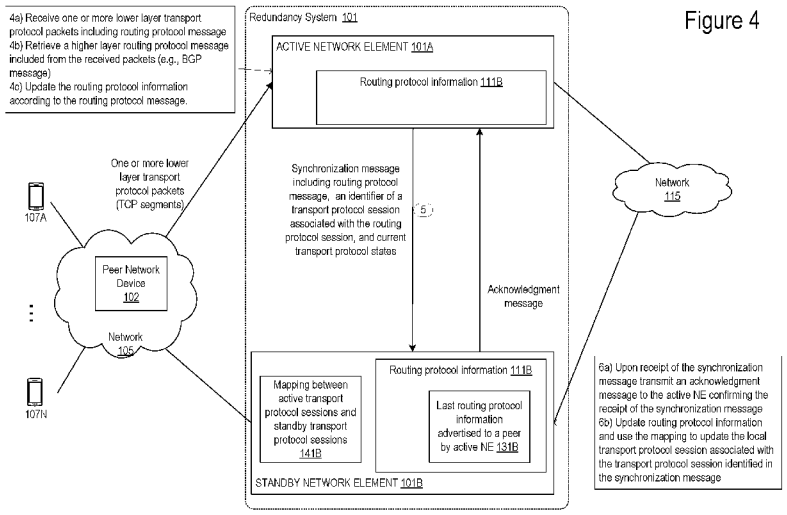
(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, 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:

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

(54) Title: METHOD AND APPARATUS FOR ENABLING NON STOP ROUTING (NSR) IN A PACKET NETWORK

Figure 4



(57) Abstract: Methods and apparatuses for enabling nonstop routing are described. One or more transport protocol packets are received (4a) at a first network element, NE, 101A from a peer network device 102, where the one or more transport protocol packets include a routing protocol message associated with a routing protocol session established between the first network element 101A and the peer network device 102. The routing protocol message is retrieved (4b) and a synchronization message is transmitted (5) to a second network element 101B, where the second network element 101B and the first network element 101A are part of a redundancy system 101. The synchronization message includes the retrieved routing protocol message, an identifier of a transport protocol session associated with the routing protocol session, and current transport protocol states associated with the transport protocol session. Flow then moves to operation (6a) at which NE 101B transmits an acknowledgment message to the active NE confirming the receipt of the synchronization message. The standby NE 101B updates (6b) routing protocol information and uses the mapping information stored at the NE to update the local transport protocol session associated with the transport protocol session of the active NE 101 A identified in the synchronization message. Thus instead of transmitting all transport protocol packets (TCP segments) received at the active NE 101A to be processed by a transport stack (TCP stack)

[Continued on next page]

WO 2017/175033 A1

of the standby NE 101B for determining current transport states and/or updated routing protocol information (transmitted within the TCP segments) as performed in prior art techniques, the embodiments of the present invention process the transport packets at the active NE 101A and transmit only relevant information that may be needed to seamlessly transition the TCP sockets from the NE 101A to the NE 101B when a switchover occurs in the redundancy system.

METHOD AND APPARATUS FOR ENABLING NON STOP ROUTING (NSR) IN A PACKET NETWORK

TECHNICAL FIELD

[0001] Embodiments of the invention relate to the field of packet networks; and more specifically, to nonstop routing in a packet network.

BACKGROUND

[0002] In packet networks it is generally desirable to prevent service outages and/or loss of network traffic. By way of example, such service outages and/or loss of network traffic may occur when a network element (NE) fails, loses power, is taken offline, is rebooted, a communication link to the network element breaks, etc. In order to help prevent such service outages and/or loss of network traffic, the communication networks may utilize redundancy systems. Redundancy systems aim at providing a high availability (HA) solution that increases the availability of network elements, and may optionally be used to provide geographical redundancy. Redundancy systems are commonly implemented through a mated pair of an active network element and a standby network element. The active network element handles current sessions using session states. The session data is synchronized or replicated from the active network element to the standby network element. The standby network element begins to handle the sessions when a switchover event occurs.

[0003] However a switchover occurring in a redundancy system causes routing protocol adjacencies to break and corresponding routing protocol sessions to fail. When a primary network element (i.e., the NE which was in an active state prior to the switchover) goes down any neighboring NE that had a routing protocol session with it sees these session fail. When the backup NE (i.e., the NE which was in a standby state prior to the switchover) becomes active it re-establishes the adjacency, but in the interim the neighbor may have advertised to its own neighbors that the NE X (i.e., the redundancy system) is no longer a valid next hop to any destinations beyond it, and the neighbors should find another path. When the backup NE comes on-line and reestablishes adjacencies its neighbors advertise the information that the NE X is again available as a next hop and everyone should again recalculate best paths. This can be highly disruptive to the network. Techniques of Nonstop routing (NSR) are used to prevent, or at least minimize, the effect of broken routing protocol sessions.

[0004] An approach for controlling broken adjacencies during NE switchovers is referred to as Graceful Restart (GR) protocol extensions. When an NE goes down its neighbors wait a predetermined amount of time (the grace period) prior to advertising to their neighbors that the

NE is no longer available. If the redundancy system converges (i.e., completes the switchover) and reestablishes the routing protocol sessions before the grace period expires, the temporarily broken sessions do not affect the network beyond the neighbors. However, with this approach, the neighbors are required to support the GR protocol extensions. Further NE switchovers are most disruptive on provider edge (PE) routers, where there are many routing protocol sessions to customer edge (CE) routers; yet CEs are likely implemented on small routers which are less likely to support GR.

[0005] In other approaches, NSR uses internal processes to keep the standby NE aware of routing protocol state and adjacency maintenance activities, so that after a switchover the standby NE can take charge of the existing routing protocol sessions rather than having to establish new ones. The switchover is then transparent to the neighbors, and because the NSR process is internal there is no need for the neighbors to support any kind of protocol extension. One technique for enabling a standby/backup NE to synchronize its routing protocol states is performed by storing a copy of the neighbor state information and transmitting route refresh messages from the newly active NE (i.e., the NE which has switched from the standby state to the active state) to the neighbors of the redundancy system based on the stored copy of neighbor state information, to obtain the best routes. Therefore, according to this technique, after switchover, all neighbors are sent a route refresh request which results in a flood of network traffic. This technique degrades the runtime performance of all the neighbors of the redundancy system due to the processing of the route refresh requests. It further deteriorates the performance of the new active NE, as it needs to process the route updates received from the neighbors. Thus, the convergence time after switchover will be significantly higher for a deployment involving large numbers of peers and routes (e.g., hundreds of peers and millions of routes).

[0006] Another technique, packet replication is used to enable both the active and standby NEs of the redundancy system to be synchronized to the same states of a routing protocol. Thus in this technique, packets are processed at both NEs of the redundancy system in parallel causing the two NEs to maintain identical routing protocol states. However, the routing protocol states of the two NEs easily result in being out of synch due to processing delays and processing priorities in the respective NEs, creating a complex synchronization problem.

[0007] Routing protocols (e.g., Border Gateway Protocol (BGP)), generally run over a transport protocol (e.g., Transmission Control Protocol (TCP)) to provide reliable data transmission. In a third technique, protocol information (e.g., TCP segments as well as BGP packets, etc.) that is processed on a first NE is similarly processed on a second NE. In this approach TCP segments are transmitted from the active NE to the standby NE to be processed along with the BGP packets. According to this approach both NEs process TCP and BGP

messages received at the active NE and achieve synchronization by enabling the standby NE to process the BGP packets only once the active NE has successfully processed the BGP packets.

SUMMARY

[0008] One general aspect includes a method in a first network element coupled with a second network element, where the first and the second network elements are part of a redundancy system, and the first network element is in an active state, of enabling nonstop routing. The method includes receiving one or more transport protocol packets from a peer network device that has a routing protocol session established with the first network element, where the one or more transport protocol packets include a routing protocol message associated with the routing protocol session; processing the transport protocol packets to retrieve the routing protocol message; and transmitting a synchronization message to the second network element, where the synchronization message includes the retrieved routing protocol message, an identifier of a transport protocol session associated with the routing protocol session, and current transport protocol states associated with the routing protocol session.

[0009] One general aspect includes a first network element to be coupled with a second network element, where the first and the second network elements are part of a redundancy system, and where the first network element is in an active state, for enabling nonstop routing. The first network element includes a non-transitory computer readable medium to store instructions; and a processor coupled with the non-transitory computer readable medium to process the stored instructions to receive one or more transport protocol packets from a peer network device that has a routing protocol session established with the first network element, where the one or more transport protocol packets include a routing protocol message associated with the routing protocol session. The first network element is further to process the transport protocol packets to retrieve the routing protocol message and transmit a synchronization message to the second network element, where the synchronization message includes the retrieved routing protocol message, an identifier of a transport protocol session associated with the routing protocol session, and current transport protocol states associated with the transport protocol session.

[0010] One general aspect includes a non-transitory computer readable storage medium that provide instructions, which when executed by a processor of a first network element to be coupled with a second network element, where the first and the second network elements are part of a redundancy system, and where the first network element is in an active state, cause said processor to perform operations including: receiving one or more transport protocol packets from a peer network device that has a routing protocol session established with the first network

element, where the one or more transport protocol packets include a routing protocol message associated with the routing protocol session; processing the transport protocol packets to retrieve the routing protocol message; and transmitting a synchronization message to the second network element, where the synchronization message includes the retrieved routing protocol message, an identifier of a transport protocol session associated with the routing protocol session, and current transport protocol states associated with the transport protocol session.

BRIEF DESCRIPTION OF THE DRAWINGS

[0011] The invention may best be understood by referring to the following description and accompanying drawings that are used to illustrate embodiments of the invention. In the drawings:

[0012] Figure 1 illustrates a block diagram of a redundancy system for enabling nonstop routing in a packet network according to some embodiments of the invention.

[0013] Figure 2 illustrates a flow diagram of operations performed for updating the standby NE upon receipt of information indicating a mapping between the routing protocol sessions and the transport protocol sessions according to some embodiments of the invention.

[0014] Figure 3 illustrates a block diagram of exemplary state transitions when performing an initialization process within the redundancy system according to some embodiments of the invention.

[0015] Figure 4 illustrates a block diagram of operations performed for synchronization of NEs of a redundancy system to enable nonstop routing according to some embodiments of the invention.

[0016] Figure 5 illustrates a flow diagram of operations performed for synchronization of NEs of a redundancy system to enable nonstop routing according to some embodiments of the invention.

[0017] Figure 6A illustrates a flow diagram of exemplary detailed operations performed at the active network device upon receipt of transport protocol packets including a routing protocol message according to some embodiments of the invention.

[0018] Figure 6B illustrates a flow diagram of exemplary operations performed at the standby network device upon receipt of a synchronization message including a BGP message according to some embodiments of the invention.

[0019] Figure 7 illustrates a block diagram of operations performed for synchronization of NEs of a redundancy system to enable nonstop routing according to some embodiments of the invention.

[0020] Figure 8A illustrates a flow diagram of exemplary detailed operations performed at the active network element for transmitting a BGP message according to some embodiments of the invention.

[0021] Figure 8B illustrates a flow diagram of exemplary operations performed at the standby network device upon receipt of a synchronization message including a BGP message according to some embodiments of the invention.

[0022] Figure 9 illustrates a flow diagram of operations performed at the NE 101B when a switchover occurs and the NE assumes an active role within the redundancy system in which a nonstop routing is enabled according to some embodiments of the invention.

[0023] Figure 10A illustrates connectivity between network devices (NDs) within an exemplary network, as well as three exemplary implementations of the NDs, according to some embodiments of the invention.

[0024] Figure 10B illustrates an exemplary way to implement a special-purpose network device according to some embodiments of the invention.

[0025] Figure 10C illustrates various exemplary ways in which virtual network elements (VNEs) may be coupled according to some embodiments of the invention.

[0026] Figure 10D illustrates a network with a single network element (NE) on each of the NDs, and within this straight forward approach contrasts a traditional distributed approach (commonly used by traditional routers) with a centralized approach for maintaining reachability and forwarding information (also called network control), according to some embodiments of the invention.

[0027] Figure 10E illustrates the simple case of where each of the NDs implements a single NE, but a centralized control plane has abstracted multiple of the NEs in different NDs into (to represent) a single NE in one of the virtual network(s), according to some embodiments of the invention.

[0028] Figure 10F illustrates a case where multiple VNEs are implemented on different NDs and are coupled to each other, and where a centralized control plane has abstracted these multiple VNEs such that they appear as a single VNE within one of the virtual networks, according to some embodiments of the invention.

[0029] Figure 11 illustrates a general purpose control plane device with centralized control plane (CCP) software 1150), according to some embodiments of the invention.

DETAILED DESCRIPTION

[0030] The following description describes methods and apparatus for enabling nonstop routing in a packet network. In the following description, numerous specific details such as

logic implementations, opcodes, means to specify operands, resource partitioning/sharing/duplication implementations, types and interrelationships of system components, and logic partitioning/integration choices are set forth in order to provide a more thorough understanding of the present invention. It will be appreciated, however, by one skilled in the art that the invention may be practiced without such specific details. In other instances, control structures, gate level circuits and full software instruction sequences have not been shown in detail in order not to obscure the invention. Those of ordinary skill in the art, with the included descriptions, will be able to implement appropriate functionality without undue experimentation.

[0031] References in the specification to "one embodiment," "an embodiment," "an example embodiment," etc., indicate that the embodiment described may include a particular feature, structure, or characteristic, but every embodiment may not necessarily include the particular feature, structure, or characteristic. Moreover, such phrases are not necessarily referring to the same embodiment. Further, when a particular feature, structure, or characteristic is described in connection with an embodiment, it is submitted that it is within the knowledge of one skilled in the art to affect such feature, structure, or characteristic in connection with other embodiments whether or not explicitly described.

[0032] Bracketed text and blocks with dashed borders (e.g., large dashes, small dashes, dot-dash, and dots) may be used herein to illustrate optional operations that add additional features to embodiments of the invention. However, such notation should not be taken to mean that these are the only options or optional operations, and/or that blocks with solid borders are not optional in certain embodiments of the invention.

[0033] In the following description and claims, the terms "coupled" and "connected," along with their derivatives, may be used. It should be understood that these terms are not intended as synonyms for each other. "Coupled" is used to indicate that two or more elements, which may or may not be in direct physical or electrical contact with each other, co-operate or interact with each other. "Connected" is used to indicate the establishment of communication between two or more elements that are coupled with each other.

[0034] Figure 1 illustrates a block diagram of an exemplary redundancy system 101 for enabling nonstop routing in a packet network 100 in accordance with some embodiments. Packet network 100 includes one or more end users' devices 107A-N. For example the end user devices may belong to subscribers to a service offered over the packet network. Examples of suitable end user devices include, but are not limited to, servers, workstations, laptops, netbooks, palm tops, mobile phones, smartphones, multimedia phones, tablets, phablets, Voice Over Internet Protocol (VOIP) phones, user equipment, terminals, portable media players, GPS units,

gaming systems, set-top boxes, and combinations thereof. End users' devices 107A-N access content/services provided over the Internet and/or content/services provided on virtual private networks (VPNs) overlaid on (e.g., tunneled through) the Internet. The content and/or services are typically provided by one or more provider end stations (e.g., application servers) belonging to a service or content provider. Examples of such content and/or services include, but are not limited to, public webpages (e.g., free content, store fronts, search services), private webpages (e.g., username/password accessed webpages providing email services), and/or corporate networks over VPNs, etc.

[0035] As illustrated, end users' devices 107A-N are communicatively coupled to network 105 (e.g., a backhaul network) which includes one or more network devices such as peer ND 102. The network devices (e.g., peer ND 102) can be communicatively coupled to redundancy system 101. For example redundancy system can implement a provider edge of a provider network. The provider edge may be communicatively coupled to one or more provider end stations or application servers (not illustrated). While in Figure 1, two end users devices are illustrated (107A and 107N), the provider edge may host on the order of thousands to millions of wire line type and/or wireless end users' devices, and the scope of the invention is not limited to any known number.

[0036] In one embodiment, network elements 101A a NE 101B form the redundancy system/cluster 101. In a redundancy system, there are typically two network elements; however, the system may include more than two network elements. During normal operation, one network element operates in an active role (herein referred to as active network element (NE) 101A) while the other operates in a standby role (herein referred to as standby NE 101B). The active NE is responsible for handling network traffic with a plurality of other network devices (e.g., end users' devices 107A-N, or neighbor network devices as identified with respect to routing protocols). During a switchover event (herein referred to simply as a "switchover"), the active and standby network elements switch roles (e.g., the active network element becomes the standby network element, and the standby network element becomes the active network element). Each of the network elements may be implemented on a separate network device, where the network devices can be any of the network devices described with respect to Figures 10A to Figure 11. In some embodiments, each of the NEs 101A and 101B is a control plane of a single network device which are separate such that a failure of a first one of the control planes does not cause a failure of the other control plane. In alternative embodiments, each of the NEs includes a control plane of a separate network device associated with a corresponding forwarding plane.

[0037] Each one of the NEs 101A and 101B are operative to run one or more higher layer routing protocols as well as a lower layer transport protocol. The embodiments will be described with respect to higher layer routing protocol BGP, however alternative embodiments could use other higher layer routing protocols, such as Multiprotocol Label Switching (MPLS) Label Distribution Protocol (LDP). While BGP messages encapsulated within TCP segments are mentioned herein for transmitting and receiving routing protocol information and updates, one of ordinary skill in the art would understand that routes can also arrive within messages of other protocols (e.g., Open Shortest Path First (OSPF)), or alternatively as a result of configuration changes (e.g., static routes). One or more peer network devices (e.g., peer network device 102 within network 105) communicate with the redundancy system 101 through routing protocol sessions to exchange routing protocol messages which will be used to dynamically update routing protocol information at the redundancy system.

[0038] Prior the standby NE joining the redundancy system 101 (where the standby NE may joining the redundancy system as a result of a first activation of the standby NE; its coupling with the active NE and/or a restart), the active NE 101A includes initial routing protocol information and transport protocol session information. For example, the NE 101A have one or more transport protocol sessions (e.g., TCP sockets) established between the redundancy system (i.e., the active NE 101A) and multiple peer network devices (e.g., exemplary peer network device 102 of network 105) and may include transport protocol session states and information associated with each transport protocol session. Further NE 101A includes routing protocol information stored in one or more databases related to a routing protocol (e.g., BGP). For example, the NE 101A may include a Routing Information Base (RIB)-In (which includes IP prefixes received locally or from a routing protocol peer), Label Information Base (LIB) (which includes a database for labels allocated/received for one or more paths on the network), one or more adjacency structures (which may be referred to as next hop database (NH database) including the next hop network device for each IP prefix), path attribute database (which includes attributes for one or more paths in the network), a RIB-out (which is includes last BGP messages sent by the NE to each peer network device for a given IP prefix's reachability advertisement or in other words the best paths advertised by the active NE for each IP prefix). The NE 101A may further include routing protocol states associated with current routing protocol sessions. For example, the NE 101A may include routing protocol instance identifier (ID) identifying an instance of the protocol running on the NE; an identifier of a peer network device; the peer's BGP Finite State Machine (FSM) state; and any additional dynamic information that is generated by the active NE 101A while processing BGP messages prior to the addition of the standby NE 101B to the redundancy system. In these embodiments, at least a

portion of the routing protocol information has been dynamically established in the active NE 101A over a period of time prior to the standby NE joining the redundancy system 101.

[0039] Initialization process of the synchronization between the active NE and the standby NE

[0040] When NE 101B joins the redundancy system 101 at operation 1 (i.e., the NE 101B is added to the system for the first time or it is restarting from a failure), initial routing protocol information and transport protocol information are transmitted from NE 101A to NE 101B at operation 2a. Thus upon receipt of the current routing protocol information and current transport protocol information, NE 101B updates corresponding databases and data structures enabling the NE 101B to be synchronized with the current routing protocol information of NE 101A at this initial time of synchronization. While in some embodiments, all routing databases will be synchronized (e.g., RIB, Labels, etc.) from NE 101A to NE 101B, in other embodiments, only a subset of these databases is copied. For example, in one embodiment, the RIB-out database is copied from NE 101A to NE 101B, where the RIB-out database includes the best paths computed and advertised at NE 101A for each IP prefix.

[0041] The flow of operations then moves to operation 2b, at which information indicating a mapping between routing protocol sessions and transport protocol sessions is transmitted to NE 101B. According to one embodiment, for each BGP session (i.e., session established between NE 101A and a BGP peer network device), the session is mapped to a corresponding TCP socket on which the BGP session is established. In some embodiments, each BGP session is identified by a peer identifier (BGP-peer ID) uniquely identifying the peer network device, and a BGP instance identifier uniquely identifying a BGP instance to which the BGP session belongs. In one embodiment, NE 101A transmits the following information to NE 101B indicating the mapping between the BGP sessions and the TCP sockets established at the active NE 101A: a TCP identifier (TCP ID) identifying the TCP socket (e.g., a socket file descriptor (fd) identifying the active TCP socket (i.e., the TCP socket running on the active NE 101A) established between the NE 101A and the BGP peer; as well as TCP states associated with the TCP ID. The TCP states can be used to achieve a seamless TCP socket transition when a switchover occurs within the redundancy system. Thus instead of transmitting all TCP segments received at the active NE 101A to be processed by a TCP stack of the standby NE 101B for determining current TCP states and/or updated routing protocol information (transmitted within the TCP segments) as performed in prior art techniques, the embodiments of the present invention process the TCP segments at the active NE 101A and transmit only relevant information that may be needed to seamlessly transition the TCP sockets from the NE 101A to the NE 101B when a switchover occurs in the redundancy system consequently reducing the amount of data transmitted from NE

101A and NE 101B and the amount of processing performed at the standby NE 101B during runtime as will be described in further details below.

[0042] In some embodiments, the TCP state information for each TCP ID transmitted to NE 101B may include TCP socket's local and remote address tuples (e.g., family IP address of the peer network device (sin_family), the TCP port associated with the TCP socket for the peer network device (sin_port), the IP address of the peer network device (sin_addr)). The TCP state information may further include TCP control block flags associated with the TCP ID; and TCP control block fields (e.g., ts_recent, ts_val, t_family, the peer's maximum segment size, the local TCP's max segment size, the current segment size in use, the maximum segment life, the initial send sequence number, the send unacknowledged sequence number, the initial receive sequence number, the receive next, the highest sequence number sent, the send next, the send window, the interface maximum transmission unit (MTU), the path MTU, etc.).

[0043] In some embodiments, operation 2b is performed for each TCP socket established at the active NE 101A when the standby NE 101B joins the redundancy system and is ready to receive synchronization messages. In some embodiments the synchronization messages are transmitted through an inter process communication (IPC) channel established between NE 101A and NE 101B.

[0044] Following the receipt of the information indicating the mapping between the routing protocol session and the transport protocol session as well as the states associated with the transport protocol session, NE 101A is updated, at operation 3, to include the routing protocol information and transport protocol states received, and the information indicating the mapping between the higher layer routing protocol sessions (BGP sessions) and the transport protocol sessions (TCP sockets).

[0045] Figure 2 illustrates a flow diagram of operations performed to update the standby NE 101A upon receipt of information indicating a mapping between the routing protocol sessions and the transport protocol sessions in accordance with some embodiments. The operations in the flow diagram will be described with reference to the exemplary embodiments of Figure 1. However, it should be understood that the operations of the flow diagram can be performed by embodiments of the invention other than those discussed with reference to Figure 1, and the embodiments of the invention discussed with reference to Figure 1 can perform operations different than those discussed with reference to the flow diagrams.

[0046] At operation 202 upon receipt of a synchronization message including the information indicating the mapping between routing protocol sessions (e.g., BGP sessions) and transport protocol sessions (e.g., TCP sockets) of the active NE, NE 101B allocates a new local transport protocol session (e.g., a new local TCP socket (i.e., new TCP socket between the peer network

device 102 and NE 101B)). Flow then moves to operation 204 at which NE 101B obtains a new identifier (TCP ID) for this session. At operation 206, NE 101B creates and stores a mapping 141B between the transport protocol session on the active network element and the newly created transport protocol session on the standby NE. In some embodiments, the mapping includes a mapping between the TCP ID associated with a BGP session on the active NE and the TCP ID associated with that same BGP session on the standby NE (operation 216). The mapping 141B is stored at the NE 101B and will be used to enable a seamless switchover of the transport sessions for each routing protocol session.

[0047] The flow of operations then moves to operation 208 at which NE 101B updates the new transport protocol session allocated according to transport protocol information received from NE 101A. Once the update is completed the redundancy system 101 is determined to be in a nonstop routing (NSR) ready state such that if a switchover occurs at this moment in time a seamless transition of the routing protocol sessions (e.g., BGP sessions) and the transport protocol sessions (e.g., TCP sockets) occurs between NE 101A and NE 101B, where the transition is transparent to the peer network devices and does not require any processing at the peer network devices.

[0048] Figure 3 illustrates a block diagram of exemplary state transitions when performing an initialization process within the redundancy system in accordance with some embodiments. The state transitions in the block diagram will be described with reference to the exemplary embodiments of Figure 1. However, it should be understood that the state transitions of the block diagram of Figure 3 can be performed by embodiments of the invention other than those discussed with reference to Figure 1, and the embodiments of the invention discussed with reference to Figure 1 can perform operations different than those discussed with reference to the block diagram.

[0049] In some embodiments, the active NE 101A is in a “not ready” state prior to the initialization of the synchronization of the routing protocol information and the transport protocol information with the standby NE 101B. Prior to NE 101B joining the redundancy system (operation 1 of Figure 1), the NE 101B is set to be in a “not ready” NSR state indicating that the NE 101B is not ready to undergo a successful and seamless switchover of the redundancy system. Upon joining the redundancy system 101, the NSR state of the standby NE 101B is set to “init-sync-ready” indicating that the NE is ready to start the initial synchronization process. Thus at operation 304B, the synchronization process is in progress at the standby NE 101B and a message is transmitted to the NE 101A for requesting the initialization of the synchronization. At operation 304A, the active NE 101A starts the synchronization process upon receipt of the request from the standby NE and transmits a request to initiate the synchronization

process to the NE 101B. Upon receipt of the request, the NE 101B determines that the NE 101A may now start the synchronization process (operations 306A and 306B) at which the NE 101A transmits protocol routing information and mapping information (operations 2a and 2b of Figure 1) to the standby NE 101A. Once the synchronization process is completed (operation 308A), the NE 101A transmits an "end-init-sync" message indicating to the NE 101B that all information is synchronized and that the initial synchronization process is now terminated. Upon receipt of the "end-init-sync" message the standby NE 101B completes the synchronization process (operation 308B). At this moment, the NSR states of the two network element are set to "ready" such that if a switchover occurs at this moment a seamless transition occurs between the two network elements with respect to the routing protocols sessions and the transport protocol sessions.

[0050] The synchronization of states upon receipt of a routing protocol messages:

[0051] Following the initial synchronization process, the active NE 101A may receive higher layer routing protocol messages or updates (through static configuration) that may alter the routing protocol information stored at the active NE. The routing protocol information is then synchronized to the standby NE in order to enable nonstop routing at the redundancy system 101. Figure 4 illustrates a block diagram of operations performed for synchronization of NEs of a redundancy system to enable nonstop routing in accordance with some embodiments. In some embodiments, the active NE 101A receives, at operation 4a, one or more lower layer transport protocol packets including a routing protocol message over a lower layer transport protocol session (e.g., a TCP socket established between peer network device 102 and NE 101A). A lower layer transport protocol packet (e.g., a TCP segment) is not necessarily congruent with a higher layer routing protocol packet (e.g., a BGP message). For example, a TCP stack of the peer network device 102 may transmit a single BGP message within one or more TCP segments in order to conform with the protocol's and network requirements and capabilities. Upon receipt of these packets, NE 101A processes the transport protocol packets to retrieve, at operation 4b, the routing protocol message transmitted from the peer network device (e.g., a BGP message). In some embodiments, prior to transmitting a synchronization message to the standby NE 101B, the active NE 101A processes the higher layer routing protocol message (e.g., at the BGP stack of the NE 101A) and updates, at operation 4c, the routing protocol information 111B according to the routing protocol message (e.g., by updating or adding entries in one or more of the routing databases).

[0052] At operation 5, NE 101A transmits a synchronization message including the routing protocol message and an identifier of the transport protocol session through which the routing protocol was received from the peer network device 102. Flow then moves to operation 6a at

which NE 101B transmits an acknowledgment message to the active NE confirming the receipt of the synchronization message. The flow then moves to operation 6b, at which the standby NE 101B to update routing protocol information and transport protocol information according to the routing protocol and the transport session identifier and the states of the transport protocol session received in the synchronization message. The standby NE 101B updates routing protocol information and uses the mapping information stored at the NE to update the local transport protocol session associated with the transport protocol session of the active NE 101A identified in the synchronization message. Thus instead of transmitting all transport protocol packets (TCP segments) received at the active NE 101A to be processed by a transport stack (TCP stack) of the standby NE 101B for determining current transport states and/or updated routing protocol information (transmitted within the TCP segments) as performed in prior art techniques, the embodiments of the present invention process the transport packets at the active NE 101A and transmit only relevant information that may be needed to seamlessly transition the TCP sockets from the NE 101A to the NE 101B when a switchover occurs in the redundancy system consequently reducing the amount of data transmitted from NE 101A towards the NE 101B and the amount of processing performed at the standby NE 101B during this synchronization process.

[0053] Figure 5 illustrates a flow diagram of operations performed for synchronization of NEs of a redundancy system to enable nonstop routing in accordance with some embodiments. The operations in the flow diagram will be described with reference to the exemplary embodiments of the other figures. However, it should be understood that the operations of the flow diagram can be performed by embodiments of the invention other than those discussed with reference to the other figures, and the embodiments of the invention discussed with reference to these other figures can perform operations different than those discussed with reference to the flow diagram.

[0054] By way of example and not limitation, a single TCP socket and a single BGP session established between the NE 101A and the peer network device 102 will be described. One of ordinary skill in the art would understand that the invention is not so limited and that NE 101A may have multiple BGP sessions established over multiple TCP sockets with peer network devices. The embodiments of the present invention apply to each one of these BGP sessions established between a BGP peer network device and the NE 101A.

[0055] In some embodiments, the active NE 101A receives, at operation 502, one or more transport protocol packets including a routing protocol message over a transport protocol session (e.g., a TCP socket established between peer network device 102 and NE 101A). Upon receipt of these packets, NE 101A processes the packets to retrieve, at operation 504, the routing protocol message transmitted from the peer network device (e.g., a BGP message).

[0056] Flow then moves to operation 506, at which the active NE 101A transmits a synchronization message including the routing protocol message and an identifier of the transport protocol session (TCP ID) through which the routing protocol was received from the peer network device 102.The synchronization message further includes current transport protocol states associated with the transport protocol session established at the active NE. In some embodiments, prior to transmitting a synchronization message to the standby NE 101B, the active NE 101A processes the higher layer routing protocol message (e.g., at the BGP stack of the NE 101A) and updates the routing protocol information 111B according to the routing protocol message (e.g., by updating or adding entries in one or more of the routing databases). Flow then moves to optional operation 507 at which the active NE 101A registers to receive an acknowledgment message from the standby NE 101B confirming that the standby NE 101B has received the synchronization message. The receipt of the acknowledgment message will cause the active NE 101A to transmit a transport protocol acknowledgment message (e.g., TCP Ack) to the peer network device. In some embodiments, this operation can be skipped and the transport protocol acknowledgment message may be automatically transmitted to the peer network device when the active NE 101A transmits the synchronization message to the standby NE 101B.

[0057] Flow then moves to operation 508 at which the receipt of the synchronization message causes the standby NE 101B to update routing protocol information and transport protocol information according to the routing protocol and the transport session identifier and the states of the transport protocol session received in the synchronization message. The standby NE 101B updates routing protocol information and uses the mapping information stored at the NE to update the local transport protocol session associated with the transport protocol session of the active NE identified in the synchronization message. Thus instead of transmitting all transport protocol packets (TCP segments) received at the active NE 101A to be processed by a transport stack (TCP stack) of the standby NE 101B for determining current transport states and/or updated routing protocol information (transmitted within the TCP segments) as performed in prior art techniques, the embodiments of the present invention process the transport packets at the active NE 101A and transmit only relevant information that may be needed to seamlessly transition the TCP sockets from the NE 101A to the NE 101B when a switchover occurs in the redundancy system consequently reducing the amount of data transmitted from NE 101A towards the NE 101B and the amount of processing performed at the standby NE 101B during this synchronization process.

[0058] Figure 6A illustrates a flow diagram of exemplary detailed operations performed at the active network device upon receipt of transport protocol packets including a routing protocol

message in accordance with some embodiments. Figure 6A illustrates operations performed for a BGP session established between NE 101A and a peer network device and a TCP socket at the active NE 101A, where the TCP socket is associated with a transport protocol session identifier (e.g., TCP ID). At operation 602 the NE 101A receives one or more TCP segments that include a BGP message associated with the BGP session. Upon receipt of the BGP message, NE 101A initiates a synchronization procedure towards the standby NE 101B.

[0059] In some embodiments, upon receipt of the BGP message, the flow moves directly to operation 618 at which the message is transmitted to the standby NE 101B. In other embodiments, flow moves to operation 604 at which the BGP message is read from a socket buffer and added to a local buffer. Flow then moves to operation 606, at which NE 101A determines the value of a nonstop routing (NSR) state associated with the NE. In some embodiments, the NSR state of a network element indicates whether the initial synchronization of the NEs has been performed as described with reference to Figures 1-3. If the NSR state of the NE 101A is ready, flow moves to operation 610, at which the NE 101A determines whether the standby NE 101B has restarted or not. If the standby NE 101B has not restarted flow moves to operation 616 at which the NE determines if the NSR state is still in a ready state. If it is determined that the NSR state is not ready then flow moves to operation 622 at which NE signals the TCP stack to send an acknowledgment message to the peer network device 102 for the received TCP messages. Alternatively, if it is determined that the NSR state is ready, flow moves to operation 618 at which the NE 101A transmits a synchronization message to the standby NE including the BGP message, TCP states associated with the TCP socket on which the BGP message was received and the TCP ID. In some embodiments, the synchronization messages is transmitted through an IPC messaging channel and includes the TCP ID of the TCP socket on which the BGP message was received; a BGP instance identifier, a BGP peer identifier, and the BGP message.

[0060] Flow then moves to operation 620 at which the NE 101A registers to receive an acknowledgment message from the standby NE 101B (confirming reliable receipt of the synchronization message including the BGP message), which causes the NE 101A to transmit a TCP acknowledgment message to the peer network device 102 confirming that the TCP segments including the BGP message has been received. Flow then moves to operation 624 at which the BGP message is added to a read queue at the NE for processing. The NE 101A may then process the BGP message and update routing protocol information accordingly.

[0061] Referring back to operation 606, if the NSR state of the NE 101A is not determined to be ready, flow then moves to operation 608 at which the NE determines if the initial synchronization process (init-sync) has been completed. If the initial synchronization process is

complete the NSR state of the NE 101A is set to a ready state. If the initial synchronization process is not complete yet, the NSR state of the NE 101A is set to a "wait state" indicating that the NE needs to wait for the synchronization process between the NE 101A and NE 101B be complete prior to processing the new BGP message received. Once the initial synchronization process is complete flow moves to operation 616. Once the BGP message is added to a read queue (operation 624) flow moves to operation 626 at which the NE 101E starts processing following BGP messages received, if any.

[0062] Figure 6B illustrates a flow diagram of exemplary operations performed at the standby network element upon receipt of a synchronization message including a BGP message in accordance with some embodiments. At operation 632, the standby NE 101B receives the synchronization message including an identifier of an active transport protocol session, current states of the active transport protocol session, and a routing protocol message which was received through the active transport protocol session at the active network device. In some embodiments, the synchronization message further includes routing protocol instance identifier (BGP instance ID) and a peer session identifier (peer ID). Flow then moves to operation 634 at which the NE 101B determines a standby transport protocol session associated with the active transport protocol session on which the routing protocol message is to be transmitted from the active NE 101A. In some embodiments, the determination is performed by determining (operation 644) an identifier of a standby transport protocol session (e.g., TCP ID) based on the identifier of the active transport protocol session (which is established at the active network element 101A between the NE 101A and the peer network device 102) using the mapping 141B established between active transport protocol sessions (active TCP sockets) at the NE 101A and standby transport protocol sessions (standby TCP sockets) at the NE 101B. Flow then moves to operation 636 at which the NE 101B determines a standby routing protocol session (e.g., at least in part using BGP instance ID) at the standby NE and locates the peer network device to which the routing protocol session belongs according to the peer ID included in the synchronization message. Flow then moves to operation 638 at which the NE 101B adds the BGP message received within the synchronization message to a processing queue according to the peer pointer value determined at operation 636. Flow then moves to operation 640 at which the NE 101B processes the received BGP message and updates the routing protocol information 111B according to the message. In some embodiments, the NE 101B does not compute an updated best path based on the message received. Thus the BGP configuration is synchronized between the active NE 101A and the standby NE 101B such that the standby NE has up-to-date information of any configuration changes occurring on the active NE. For example, any changes

made to BGP instances and BGP peers are synchronized with the standby to keep the BGP-instance identifiers and peer identifiers mappings consistent.

[0063] New BGP message to be transmitted from the active NE

[0064] Figure 7 illustrates a block diagram of operations performed for synchronization of NEs of a redundancy system to enable nonstop routing in accordance with some embodiments. In some embodiments, the active NE 101A determines that a BGP message is to be transmitted to a peer network device (e.g., ND 102) over a lower layer transport protocol session (e.g., a TCP socket established between peer network device 102 and NE 101A). In order to enable nonstop routing by providing a seamless switchover within the redundancy system 101, the NE 101A synchronizes the BGP message with the standby NE 101B prior to transmitting the BGP message to the peer ND. At operation 7, NE 101A formats a BGP message including an update to routing protocol information to be transmitted over a given BGP session associated with a BGP instance identifier. The BGP message is to be forwarded to a peer network device (102) through a TCP socket associated with a TCP ID. Upon determination that a BGP message is to be transmitted, the NE 101A adds the BGP message to a TCP transmission buffer of the active NE without sending the message. At operation 8, the NE 101A constructs a synchronization message including an identifier of the TCP socket (TCP ID) associated with the BGP message to be transmitted; relevant lower layer protocol state information (TCP state) from current TCP session; an identifier of the peer ND and BGP instance identifier. Flow then moves to operation9a, at which upon receipt of the synchronization message, the standby NE 101B transmits an acknowledgment message to the active NE 101A confirming the receipt of the BGP message. At operation 9b, the NE 101B updates routing protocol information to include the last routing protocol message (BGP message) 131B advertised to a peer ND by the active NE 101A. At operation 9c, the standby NE 101B updates transport protocol session states according to the current transport protocol states received in the synchronization message. Upon receipt of the acknowledgement message from the standby NE 101B, the active NE 101A may now transmit the routing protocol message (BGP message), at operation 10, to the peer network device.

[0065] Figure 8A illustrates a flow diagram of exemplary detailed operations performed at the active network element for transmitting a BGP message in accordance with some embodiments. By way of example and not limitation, a single TCP socket and a single BGP session established between the NE 101A and the peer network device 102 will be described. One of ordinary skill in the art would understand that the invention is not so limited and that NE 101A may have multiple BGP sessions established over multiple TCP sockets with peer network devices. The embodiments of the present invention described with reference to Figure 8 apply to each one of these BGP sessions established between a BGP peer network device and the active NE 101A.

[0066] Figure 8A illustrates operations performed for a BGP session established between NE 101A and a peer network device 102 and an associated TCP socket at the active NE 101A, where the TCP socket is associated with a transport protocol session identifier (e.g., TCP ID). At operation 802 the NE 101A determines that a BGP IP prefix is to be updated. Flow then moves to operation 804, at which the NE 101A formats a BGP message, which is to be transmitted to the peer ND 102 to advertise the update. Flow then moves to operation 806, at which NE 101A determines whether the initial synchronization process of the active NE and the standby NE is complete. If the initial synchronization process has not been completed flow moves to operation 818 at which the BGP message is added to a socket buffer (socket associated with the TCP ID) and held there until a confirmation is received from the standby NE that the BGP message has been received. Referring back to operation 806, upon determination that the initial synchronization process is not complete, flow moves to operation 808 where the NE 101A determines whether the NSR state is ready. If the NSR state of NE 101A is not ready, flow then moves to operation 810 at which the NE determines whether the BGP session is established. If the BGP session is not established flow moves to operation 818. When the BGP session is established flow moves to operation 812 at which the NSR state is set to ready. Referring back to operation 808, if the NSR state is determined to be ready, flow then moves to operation 814 at which the NE 101A determines if the standby has been restarted. If the standby NE 101B has been restarted, the NSR state of the NE 101A is set to wait until the standby NE is synchronized with NE 101A and the NSR process is in a ready state. Flow then moves from operation 812, 814 or 812 to operation 818 at which the BGP message is added to a socket buffer (socket associated with the TCP ID) and held there until a confirmation is received from the standby NE 101B that the BGP message has been received. Flow then moves from operation 818 to operation 820 at which the NE 101A determines whether at least one of the conditions (a) is synchronization process completed, (b) is NSR state of the NE 101A ready, or (c) is the BGP session established is false. If at least one of (a), (b), or (c) is false flow then moves to operation 822 at which the NE 101A signals to the TCP socket to transmit the BGP message to the peer ND 102. If none of the conditions (a), (b), and (c) are false the flow moves to operation 818, at which the active NE 101A transmits a synchronization message to the standby NE 101B including the BGP message, TCP states associated with the TCP socket, and an identification of the TCP socket (TCP ID). Flow then moves to operation 820 at which the NE 101A registers to receive an acknowledgment message from the standby NE 101B (confirming reliable receipt of the synchronization message including the BGP message), which causes the NE 101A to transmit the BGP message to the peer network device. Flow then moves to operation 626 at which the NE 101E starts processing following BGP messages if any.

[0067] Figure 8B illustrates a flow diagram of exemplary operations performed at the standby network device upon receipt of a synchronization message including a BGP message in accordance with some embodiments. At operation 832, the standby NE 101B receives a synchronization message including an identifier of an active transport protocol session (TCP ID), a routing protocol message (BGP message) which is to be transmitted on the active transport protocol session to a peer network device, and current transport protocol states associated with the active transport protocol session (TCP states). In some embodiments, the synchronization message further includes routing protocol instance identifier (BGP instance ID) and a peer session identifier (peer ID). Flow then moves to operation 834 at which the NE 101B determines a standby transport protocol session associated with the active transport protocol session on which the routing protocol message is to be transmitted from the active NE 101A. In some embodiments, the determination is performed by determining (operation 844) an identifier of a standby transport protocol session (e.g., TCP ID) based on the identifier of the active transport protocol session (which is established at the active network element 101A between the NE 101A and the peer network device 102) using the mapping 141B established between active transport protocol sessions (active TCP sockets) at the NE 101A and standby transport protocol sessions (standby TCP sockets) at the NE 101B. Flow then moves to operation 836 at which the NE 101B determines a routing protocol session (using BGP instance ID) at the standby NE and locates the peer network device to which the routing protocol session belongs according to the peer id included in the synchronization message. Flow then moves to operation 838 at which the NE 101B adds the BGP message received within the synchronization message to a TCP transmission queue without transmitting the BGP message.

[0068] In some embodiments, the NE 101B further processes (operation 840) the BGP message which is to be transmitted by the active NE 101A to the peer network device 102. The NE 101B copies the message to a receiving buffer and set a flag indicating that the BGP message is to be transmitted at the active NE 101A towards the ND 102. The NE 101B then processes the BGP message by parsing the message and determining the IP prefix which is to be updated or alternatively withdrawn. The new routing information associated with that IP prefix is then added to the routing protocol information 111B of the standby NE 101B (e.g., an entry of the RIB-out database can be added/updated or alternatively withdrawn). Additional databases (labels, attributes, paths, etc.) may further be updated enabling the standby NE 101B to have routing protocol information synchronized with the NE 101A. In alternative embodiments, the BGP message is not parsed and a peer-bit is “set” in a prefix’s peer bitmap which indicates that the particular peer network device has been updated with the best path for this prefix. In some embodiments, a best route is not computed by the standby NE 101B until after a switchover

occurs as described in further details below with reference to Figure 9. In some embodiments, to conserve memory foot-print, peer-group can be used to store only a single last published update for all the peer network devices belonging to the group.

[0069] Switchover and reconciliation

[0070] Figure 9 illustrates a flow diagram of operations performed at the NE 101B when a switchover occurs and the NE assumes an active role within the redundancy system in which a nonstop routing is enabled in accordance with some embodiments. When a switchover occurs at the redundancy system 101, the NE 101A assumes a standby role and the NE 101B assumes an active role. The operations of Figure 9 will be described with reference to a single TCP socket to be established between a peer network device (e.g., 102) and the NE 101B by way of example and not limitation. At operation 932, the newly active NE 101B activates a transport protocol session (TCP socket) coupling the peer ND 102 and NE 101B. When the switchover occurs, the new active NE 101B includes a set of standby transport protocol sessions that were established during multiple synchronization processes (initial synchronization process and synchronization updates received from the previously active NE 101A during a runtime prior to the switchover). NE 101B upon detection of the switchover starts activating all these transport protocol sessions. Flow then moves to operation 934, at which NE 101B starts transmission of transport protocol keep-alive messages (TCP keepalive) in order to indicate that the transport session (TCP socket) is alive. In some embodiments this is performed by starting a "keep-alive" timer and initiating the transmission of the keepalive packets. At this step the NE 101B may further determine that the TCP states are adjusted (corrected) to the newly activated TCP socket.

[0071] Flow then moves to operation 936 at which the NE 101B starts the transmission of keep alive routing protocol message (BGP keepalive messages). This causes the routing protocol neighbors (e.g., peer ND 102) to not notice that the transport protocol sessions (TCP sockets) were transitioned from the NE 101A to the NE 101B enabling a nonstop routing at the redundancy system when the switchover occurs. Flow then moves to operation 938 at which the NE 101B transmits any routing protocol messages (BGP messages) stored in a transmission queue at the NE 101B which were not acknowledged by the ND 102 when a TCP socket was established between the ND 102 and the previously active NE 101A. Flow then moves to operation 940 at which NE 101B waits for routing protocols (running on the NE 101B) to converge causing the receipt of a route redistribution from the RIB. At operation 942, upon receipt of the route redistribution, the BGP stack of the NE 101B computes the BEST-PATH for each IP prefix. Flow then moves to operation 944 at which NE 101B determines whether the newly computed best path is different from the best path already stored in the routing protocol information 111B (e.g., stored in the RIB-out). If the best path is identical, this information is

discarded and not transmitted to the peer ND 102 (since the peer ND already includes this routing information). Alternatively, if the best path is different, the flow moves to operation 948 at which the NE 101B advertises the best path to the peer network device and updates routing protocol information database accordingly. The transmission of this new BGP message will cause the synchronization process of the redundancy system to be initiated such that the new standby NE 101A is synchronized according to the routing protocol states of NE 101B.

[0072] In alternative embodiments, when BGP messages advertised from the previously active NE 101A are not parsed at the previously standby NE 101B, upon switchover, the peer bit in the prefix's peer bitmap is checked to see if the best-path for this prefix has been advertised to this peer. If yes, then the update message being advertised to the peer is compared with a newly computed best path (which is determined at the NE 101B after the switchover occurs following the conversion of the routing protocol(s)) to see if it is different. In this embodiments, the best path for the prefix will be advertised to the peer only if at least one of the two conditions occurs the peer bit is set and the last published update for the peer is different from the newly computed best path, or the peer-bit is not set, indicating that no update was advertised to the peer for this prefix.

[0073] The embodiments described herein provide methods and apparatuses for enabling nonstop routing in a packet network. The embodiments provide a minimal runtime synchronization overhead by having synchronization messages including the BGP message and an identifier of the TCP sessions along with TCP state on which the BGP message is to be transmitted. A single routing protocol message is transmitted to peer network devices to achieve the nonstop routing and routing protocol synchronization after a switchover. Further in some embodiments, the message is only transmitted when the newly active NE determines that the peer network device does not include the updated routing protocol information. The embodiments provide a generic synchronization mechanism which is protocol independent and can be used with multiple version of routing protocols (for example multiple version of BGP, or MPLS LDP).

[0074] Architecture

[0075] An electronic device stores and transmits (internally and/or with other electronic devices over a network) code (which is composed of software instructions and which is sometimes referred to as computer program code or a computer program) and/or data using machine-readable media (also called computer-readable media), such as machine-readable storage media (e.g., magnetic disks, optical disks, read only memory (ROM), flash memory devices, phase change memory) and machine-readable transmission media (also called a carrier) (e.g., electrical, optical, radio, acoustical or other form of propagated signals – such as carrier

waves, infrared signals). Thus, an electronic device (e.g., a computer) includes hardware and software, such as a set of one or more processors coupled to one or more machine-readable storage media to store code for execution on the set of processors and/or to store data. For instance, an electronic device may include non-volatile memory containing the code since the non-volatile memory can persist code/data even when the electronic device is turned off (when power is removed), and while the electronic device is turned on that part of the code that is to be executed by the processor(s) of that electronic device is typically copied from the slower non-volatile memory into volatile memory (e.g., dynamic random access memory (DRAM), static random access memory (SRAM)) of that electronic device. Typical electronic devices also include a set or one or more physical network interface(s) to establish network connections (to transmit and/or receive code and/or data using propagating signals) with other electronic devices. One or more parts of an embodiment of the invention may be implemented using different combinations of software, firmware, and/or hardware.

[0076] A network device (ND) is an electronic device that communicatively interconnects other electronic devices on the network (e.g., other network devices, end-user devices). Some network devices are "multiple services network devices" that provide support for multiple networking functions (e.g., routing, bridging, switching, Layer 2 aggregation, session border control, Quality of Service, and/or subscriber management), and/or provide support for multiple application services (e.g., data, voice, and video).

[0077] Figure 10A illustrates connectivity between network devices (NDs) within an exemplary network, as well as three exemplary implementations of the NDs, according to some embodiments of the invention. Figure 10A shows NDs 1000A-H, and their connectivity by way of lines between 1000A-1000B, 1000B-1000C, 1000C-1000D, 1000D-1000E, 1000E-1000F, 1000F-1000G, and 1000A-1000G, as well as between 1000H and each of 1000A, 1000C, 1000D, and 1000G. These NDs are physical devices, and the connectivity between these NDs can be wireless or wired (often referred to as a link). An additional line extending from NDs 1000A, 1000E, and 1000F illustrates that these NDs act as ingress and egress points for the network (and thus, these NDs are sometimes referred to as edge NDs; while the other NDs may be called core NDs).

[0078] Two of the exemplary ND implementations in Figure 10A are: 1) a special-purpose network device 1002 that uses custom application–specific integrated–circuits (ASICs) and a special-purpose operating system (OS); and 2) a general purpose network device 1004 that uses common off-the-shelf (COTS) processors and a standard OS.

[0079] The special-purpose network device 1002 includes networking hardware 1010 comprising compute resource(s) 1012 (which typically include a set of one or more processors),

forwarding resource(s) 1014 (which typically include one or more ASICs and/or network processors), and physical network interfaces (NIs) 1016 (sometimes called physical ports), as well as non-transitory machine readable storage media 1018 having stored therein networking software 1020. A physical NI is hardware in a ND through which a network connection (e.g., wirelessly through a wireless network interface controller (WNIC) or through plugging in a cable to a physical port connected to a network interface controller (NIC)) is made, such as those shown by the connectivity between NDs 1000A-H. During operation, the networking software 1020 may be executed by the networking hardware 1010 to instantiate a set of one or more networking software instance(s) 1022. Each of the networking software instance(s) 1022, and that part of the networking hardware 1010 that executes that network software instance (be it hardware dedicated to that networking software instance and/or time slices of hardware temporally shared by that networking software instance with others of the networking software instance(s) 1022), form a separate virtual network element 1030A-R. Each of the virtual network element(s) (VNEs) 1030A-R includes a control communication and configuration module 1032A-R (sometimes referred to as a local control module or control communication module) and forwarding table(s) 1034A-R, such that a given virtual network element (e.g., 1030A) includes the control communication and configuration module (e.g., 1032A), a set of one or more forwarding table(s) (e.g., 1034A), and that portion of the networking hardware 1010 that executes the virtual network element (e.g., 1030A). The networking software 1020 further includes NonStop Routing Unit (NSRU) 1021, which when executed by the networking hardware 1010 instantiates a set of one or more NSRU 1033 part of the instances 1022 enabling the network device 1002 to perform operations described with reference to Figures 1-9.

[0080] The special-purpose network device 1002 is often physically and/or logically considered to include: 1) a ND control plane 1024 (sometimes referred to as a control plane) comprising the compute resource(s) 1012 that execute the control communication and configuration module(s) 1032A-R; and 2) a ND forwarding plane 1026 (sometimes referred to as a forwarding plane, a data plane, or a media plane) comprising the forwarding resource(s) 1014 that utilize the forwarding table(s) 1034A-R and the physical NIs 1016. By way of example, where the ND is a router (or is implementing routing functionality), the ND control plane 1024 (the compute resource(s) 1012 executing the control communication and configuration module(s) 1032A-R) is typically responsible for participating in controlling how data (e.g., packets) is to be routed (e.g., the next hop for the data and the outgoing physical NI for that data) and storing that routing information in the forwarding table(s) 1034A-R, and the ND forwarding plane 1026 is responsible for receiving that data on the physical NIs 1016 and

forwarding that data out the appropriate ones of the physical NIs 1016 based on the forwarding table(s) 1034A-R.

[0081] Figure 10B illustrates an exemplary way to implement the special-purpose network device 1002 according to some embodiments of the invention. Figure 10B shows a special-purpose network device including cards 1038 (typically hot pluggable). While in some embodiments the cards 1038 are of two types (one or more that operate as the ND forwarding plane 1026 (sometimes called line cards), and one or more that operate to implement the ND control plane 1024 (sometimes called control cards)), alternative embodiments may combine functionality onto a single card and/or include additional card types (e.g., one additional type of card is called a service card, resource card, or multi-application card). A service card can provide specialized processing (e.g., Layer 4 to Layer 7 services (e.g., firewall, Internet Protocol Security (IPsec), Secure Sockets Layer (SSL) / Transport Layer Security (TLS), Intrusion Detection System (IDS), peer-to-peer (P2P), Voice over IP (VoIP) Session Border Controller, Mobile Wireless Gateways (Gateway General Packet Radio Service (GPRS) Support Node (GGSN), Evolved Packet Core (EPC) Gateway)). By way of example, a service card may be used to terminate IPsec tunnels and execute the attendant authentication and encryption algorithms. These cards are coupled together through one or more interconnect mechanisms illustrated as backplane 1036 (e.g., a first full mesh coupling the line cards and a second full mesh coupling all of the cards).

[0082] Returning to Figure 10A, the general purpose network device 1004 includes hardware 1040 comprising a set of one or more processor(s) 1042 (which are often COTS processors) and network interface controller(s) 1044 (NICs; also known as network interface cards) (which include physical NIs 1046), as well as non-transitory machine readable storage media 1048 having stored therein software 1050. During operation, the processor(s) 1042 execute the software 1050 to instantiate one or more sets of one or more applications 1064A-R. The software 1050 further includes NonStop Routing Unit (NSRU) 1051, which when executed by the networking hardware 1040 enables the instance(s) 1052 of the network device 1004 to perform operations described with reference to Figures 1-9. While one embodiment does not implement virtualization, alternative embodiments may use different forms of virtualization. For example, in one such alternative embodiment the virtualization layer 1054 represents the kernel of an operating system (or a shim executing on a base operating system) that allows for the creation of multiple instances 1062A-R called software containers that may each be used to execute one (or more) of the sets of applications 1064A-R; where the multiple software containers (also called virtualization engines, virtual private servers, or jails) are user spaces (typically a virtual memory space) that are separate from each other and separate from the kernel

space in which the operating system is run; and where the set of applications running in a given user space, unless explicitly allowed, cannot access the memory of the other processes. In another such alternative embodiment the virtualization layer 1054 represents a hypervisor (sometimes referred to as a virtual machine monitor (VMM)) or a hypervisor executing on top of a host operating system, and each of the sets of applications 1064A-R is run on top of a guest operating system within an instance 1062A-R called a virtual machine (which may in some cases be considered a tightly isolated form of software container) that is run on top of the hypervisor - the guest operating system and application may not know they are running on a virtual machine as opposed to running on a "bare metal" host electronic device, or through para-virtualization the operating system and/or application may be aware of the presence of virtualization for optimization purposes. In yet other alternative embodiments, one, some or all of the applications are implemented as unikernel(s), which can be generated by compiling directly with an application only a limited set of libraries (e.g., from a library operating system (LibOS) including drivers/libraries of OS services) that provide the particular OS services needed by the application. As a unikernel can be implemented to run directly on hardware 1040, directly on a hypervisor (in which case the unikernel is sometimes described as running within a LibOS virtual machine), or in a software container, embodiments can be implemented fully with unikernels running directly on a hypervisor represented by virtualization layer 1054, unikernels running within software containers represented by instances 1062A-R, or as a combination of unikernels and the above-described techniques (e.g., unikernels and virtual machines both run directly on a hypervisor, unikernels and sets of applications that are run in different software containers).

[0083] The instantiation of the one or more sets of one or more applications 1064A-R, as well as virtualization if implemented, are collectively referred to as software instance(s) 1052. Each set of applications 1064A-R, corresponding virtualization construct (e.g., instance 1062A-R) if implemented, and that part of the hardware 1040 that executes them (be it hardware dedicated to that execution and/or time slices of hardware temporally shared), forms a separate virtual network element(s) 1060A-R.

[0084] The virtual network element(s) 1060A-R perform similar functionality to the virtual network element(s) 1030A-R - e.g., similar to the control communication and configuration module(s) 1032A and forwarding table(s) 1034A (this virtualization of the hardware 1040 is sometimes referred to as network function virtualization (NFV)). Thus, NFV may be used to consolidate many network equipment types onto industry standard high volume server hardware, physical switches, and physical storage, which could be located in Data centers, NDs, and customer premise equipment (CPE). While embodiments of the invention are illustrated with

each instance 1062A-R corresponding to one VNE 1060A-R, alternative embodiments may implement this correspondence at a finer level granularity (e.g., line card virtual machines virtualize line cards, control card virtual machine virtualize control cards, etc.); it should be understood that the techniques described herein with reference to a correspondence of instances 1062A-R to VNEs also apply to embodiments where such a finer level of granularity and/or unikernels are used.

[0085] In certain embodiments, the virtualization layer 1054 includes a virtual switch that provides similar forwarding services as a physical Ethernet switch. Specifically, this virtual switch forwards traffic between instances 1062A-R and the NIC(s) 1044, as well as optionally between the instances 1062A-R; in addition, this virtual switch may enforce network isolation between the VNEs 1060A-R that by policy are not permitted to communicate with each other (e.g., by honoring virtual local area networks (VLANs)).

[0086] The third exemplary ND implementation in Figure 10A is a hybrid network device 1006, which includes both custom ASICs/special-purpose OS and COTS processors/standard OS in a single ND or a single card within an ND. In certain embodiments of such a hybrid network device, a platform VM (i.e., a VM that that implements the functionality of the special-purpose network device 1002) could provide for para-virtualization to the networking hardware present in the hybrid network device 1006.

[0087] Regardless of the above exemplary implementations of an ND, when a single one of multiple VNEs implemented by an ND is being considered (e.g., only one of the VNEs is part of a given virtual network) or where only a single VNE is currently being implemented by an ND, the shortened term network element (NE) is sometimes used to refer to that VNE. Also in all of the above exemplary implementations, each of the VNEs (e.g., VNE(s) 1030A-R, VNEs 1060A-R, and those in the hybrid network device 1006) receives data on the physical NIs (e.g., 1016, 1046) and forwards that data out the appropriate ones of the physical NIs (e.g., 1016, 1046). For example, a VNE implementing IP router functionality forwards IP packets on the basis of some of the IP header information in the IP packet; where IP header information includes source IP address, destination IP address, source port, destination port (where "source port" and "destination port" refer herein to protocol ports, as opposed to physical ports of a ND), transport protocol (e.g., user datagram protocol (UDP), Transmission Control Protocol (TCP), and differentiated services code point (DSCP) values.

[0088] Figure 10C illustrates various exemplary ways in which VNEs may be coupled according to some embodiments of the invention. Figure 10C shows VNEs 1070A.1-1070A.P (and optionally VNEs 1070A.Q-1070A.R) implemented in ND 1000A and VNE 1070H.1 in ND 1000H. In Figure 10C, VNEs 1070A.1-P are separate from each other in the sense that they can

receive packets from outside ND 1000A and forward packets outside of ND 1000A; VNE 1070A.1 is coupled with VNE 1070H.1, and thus they communicate packets between their respective NDs; VNE 1070A.2-1070A.3 may optionally forward packets between themselves without forwarding them outside of the ND 1000A; and VNE 1070A.P may optionally be the first in a chain of VNEs that includes VNE 1070A.Q followed by VNE 1070A.R (this is sometimes referred to as dynamic service chaining, where each of the VNEs in the series of VNEs provides a different service – e.g., one or more layer 4-7 network services). While Figure 10C illustrates various exemplary relationships between the VNEs, alternative embodiments may support other relationships (e.g., more/fewer VNEs, more/fewer dynamic service chains, multiple different dynamic service chains with some common VNEs and some different VNEs).

[0089] The NDs of Figure 10A, for example, may form part of the Internet or a private network; and other electronic devices (not shown; such as end user devices including workstations, laptops, netbooks, tablets, palm tops, mobile phones, smartphones, phablets, multimedia phones, Voice Over Internet Protocol (VOIP) phones, terminals, portable media players, GPS units, wearable devices, gaming systems, set-top boxes, Internet enabled household appliances) may be coupled to the network (directly or through other networks such as access networks) to communicate over the network (e.g., the Internet or virtual private networks (VPNs) overlaid on (e.g., tunneled through) the Internet) with each other (directly or through servers) and/or access content and/or services. Such content and/or services are typically provided by one or more servers (not shown) belonging to a service/content provider or one or more end user devices (not shown) participating in a peer-to-peer (P2P) service, and may include, for example, public webpages (e.g., free content, store fronts, search services), private webpages (e.g., username/password accessed webpages providing email services), and/or corporate networks over VPNs. For instance, end user devices may be coupled (e.g., through customer premise equipment coupled to an access network (wired or wirelessly)) to edge NDs, which are coupled (e.g., through one or more core NDs) to other edge NDs, which are coupled to electronic devices acting as servers. However, through compute and storage virtualization, one or more of the electronic devices operating as the NDs in Figure 10A may also host one or more such servers (e.g., in the case of the general purpose network device 1004, one or more of the software instances 1062A-R may operate as servers; the same would be true for the hybrid network device 1006; in the case of the special-purpose network device 1002, one or more such servers could also be run on a virtualization layer executed by the compute resource(s) 1012); in which case the servers are said to be co-located with the VNEs of that ND.

[0090] A virtual network is a logical abstraction of a physical network (such as that in Figure 10A) that provides network services (e.g., L2 and/or L3 services). A virtual network can be

implemented as an overlay network (sometimes referred to as a network virtualization overlay) that provides network services (e.g., layer 2 (L2, data link layer) and/or layer 3 (L3, network layer) services) over an underlay network (e.g., an L3 network, such as an Internet Protocol (IP) network that uses tunnels (e.g., generic routing encapsulation (GRE), layer 2 tunneling protocol (L2TP), IPSec) to create the overlay network).

[0091] A network virtualization edge (NVE) sits at the edge of the underlay network and participates in implementing the network virtualization; the network-facing side of the NVE uses the underlay network to tunnel frames to and from other NVEs; the outward-facing side of the NVE sends and receives data to and from systems outside the network. A virtual network instance (VNI) is a specific instance of a virtual network on a NVE (e.g., a NE/VNE on an ND, a part of a NE/VNE on a ND where that NE/VNE is divided into multiple VNEs through emulation); one or more VNIs can be instantiated on an NVE (e.g., as different VNEs on an ND). A virtual access point (VAP) is a logical connection point on the NVE for connecting external systems to a virtual network; a VAP can be physical or virtual ports identified through logical interface identifiers (e.g., a VLAN ID).

[0092] Examples of network services include: 1) an Ethernet LAN emulation service (an Ethernet-based multipoint service similar to an Internet Engineering Task Force (IETF) Multiprotocol Label Switching (MPLS) or Ethernet VPN (EVPN) service) in which external systems are interconnected across the network by a LAN environment over the underlay network (e.g., an NVE provides separate L2 VNIs (virtual switching instances) for different such virtual networks, and L3 (e.g., IP/MPLS) tunneling encapsulation across the underlay network); and 2) a virtualized IP forwarding service (similar to IETF IP VPN (e.g., Border Gateway Protocol (BGP)/MPLS IPVPN) from a service definition perspective) in which external systems are interconnected across the network by an L3 environment over the underlay network (e.g., an NVE provides separate L3 VNIs (forwarding and routing instances) for different such virtual networks, and L3 (e.g., IP/MPLS) tunneling encapsulation across the underlay network)). Network services may also include quality of service capabilities (e.g., traffic classification marking, traffic conditioning and scheduling), security capabilities (e.g., filters to protect customer premises from network – originated attacks, to avoid malformed route announcements), and management capabilities (e.g., full detection and processing).

[0093] Fig. 10D illustrates a network with a single network element on each of the NDs of Figure 10A, and within this straight forward approach contrasts a traditional distributed approach (commonly used by traditional routers) with a centralized approach for maintaining reachability and forwarding information (also called network control), according to some

embodiments of the invention. Specifically, Figure 10D illustrates network elements (NEs) 1070A-H with the same connectivity as the NDs 1000A-H of Figure 10A.

[0094] Figure 10D illustrates that the distributed approach 1072 distributes responsibility for generating the reachability and forwarding information across the NEs 1070A-H; in other words, the process of neighbor discovery and topology discovery is distributed.

[0095] For example, where the special-purpose network device 1002 is used, the control communication and configuration module(s) 1032A-R of the ND control plane 1024 typically include a reachability and forwarding information module to implement one or more routing protocols (e.g., an exterior gateway protocol such as Border Gateway Protocol (BGP), Interior Gateway Protocol(s) (IGP) (e.g., Open Shortest Path First (OSPF), Intermediate System to Intermediate System (IS-IS), Routing Information Protocol (RIP), Label Distribution Protocol (LDP), Resource Reservation Protocol (RSVP) (including RSVP-Traffic Engineering (TE): Extensions to RSVP for LSP Tunnels and Generalized Multi-Protocol Label Switching (GMPLS) Signaling RSVP-TE)) that communicate with other NEs to exchange routes, and then selects those routes based on one or more routing metrics. Thus, the NEs 1070A-H (e.g., the compute resource(s) 1012 executing the control communication and configuration module(s) 1032A-R) perform their responsibility for participating in controlling how data (e.g., packets) is to be routed (e.g., the next hop for the data and the outgoing physical NI for that data) by distributively determining the reachability within the network and calculating their respective forwarding information. Routes and adjacencies are stored in one or more routing structures (e.g., Routing Information Base (RIB), Label Information Base (LIB), one or more adjacency structures) on the ND control plane 1024. The ND control plane 1024 programs the ND forwarding plane 1026 with information (e.g., adjacency and route information) based on the routing structure(s). For example, the ND control plane 1024 programs the adjacency and route information into one or more forwarding table(s) 1034A-R (e.g., Forwarding Information Base (FIB), Label Forwarding Information Base (LFIB), and one or more adjacency structures) on the ND forwarding plane 1026. For layer 2 forwarding, the ND can store one or more bridging tables that are used to forward data based on the layer 2 information in that data. While the above example uses the special-purpose network device 1002, the same distributed approach 1072 can be implemented on the general purpose network device 1004 and the hybrid network device 1006.

[0096] Figure 10D illustrates that a centralized approach 1074 (also known as software defined networking (SDN)) that decouples the system that makes decisions about where traffic is sent from the underlying systems that forwards traffic to the selected destination. The illustrated centralized approach 1074 has the responsibility for the generation of reachability and

forwarding information in a centralized control plane 1076 (sometimes referred to as a SDN control module, controller, network controller, OpenFlow controller, SDN controller, control plane node, network virtualization authority, or management control entity), and thus the process of neighbor discovery and topology discovery is centralized. The centralized control plane 1076 has a south bound interface 1082 with a data plane 1080 (sometime referred to the infrastructure layer, network forwarding plane, or forwarding plane (which should not be confused with a ND forwarding plane)) that includes the NEs 1070A-H (sometimes referred to as switches, forwarding elements, data plane elements, or nodes). The centralized control plane 1076 includes a network controller 1078, which includes a centralized reachability and forwarding information module 1079 that determines the reachability within the network and distributes the forwarding information to the NEs 1070A-H of the data plane 1080 over the south bound interface 1082 (which may use the OpenFlow protocol). Thus, the network intelligence is centralized in the centralized control plane 1076 executing on electronic devices that are typically separate from the NDs. The network controller 1078 further includes NonStop Routing Unit 1081, which enables the centralized control plane 1076 to perform operations described with reference to Figures 1-9.

[0097] For example, where the special-purpose network device 1002 is used in the data plane 1080, each of the control communication and configuration module(s) 1032A-R of the ND control plane 1024 typically include a control agent that provides the VNE side of the south bound interface 1082. In this case, the ND control plane 1024 (the compute resource(s) 1012 executing the control communication and configuration module(s) 1032A-R) performs its responsibility for participating in controlling how data (e.g., packets) is to be routed (e.g., the next hop for the data and the outgoing physical NI for that data) through the control agent communicating with the centralized control plane 1076 to receive the forwarding information (and in some cases, the reachability information) from the centralized reachability and forwarding information module 1079 (it should be understood that in some embodiments of the invention, the control communication and configuration module(s) 1032A-R, in addition to communicating with the centralized control plane 1076, may also play some role in determining reachability and/or calculating forwarding information – albeit less so than in the case of a distributed approach; such embodiments are generally considered to fall under the centralized approach 1074, but may also be considered a hybrid approach).

[0098] While the above example uses the special-purpose network device 1002, the same centralized approach 1074 can be implemented with the general purpose network device 1004 (e.g., each of the VNE 1060A-R performs its responsibility for controlling how data (e.g., packets) is to be routed (e.g., the next hop for the data and the outgoing physical NI for that

data) by communicating with the centralized control plane 1076 to receive the forwarding information (and in some cases, the reachability information) from the centralized reachability and forwarding information module 1079; it should be understood that in some embodiments of the invention, the VNEs 1060A-R, in addition to communicating with the centralized control plane 1076, may also play some role in determining reachability and/or calculating forwarding information – albeit less so than in the case of a distributed approach) and the hybrid network device 1006. In fact, the use of SDN techniques can enhance the NFV techniques typically used in the general purpose network device 1004 or hybrid network device 1006 implementations as NFV is able to support SDN by providing an infrastructure upon which the SDN software can be run, and NFV and SDN both aim to make use of commodity server hardware and physical switches.

[0099] Figure 10D also shows that the centralized control plane 1076 has a north bound interface 1084 to an application layer 1086, in which resides application(s) 1088. The centralized control plane 1076 has the ability to form virtual networks 1092 (sometimes referred to as a logical forwarding plane, network services, or overlay networks (with the NEs 1070A-H of the data plane 1080 being the underlay network)) for the application(s) 1088. Thus, the centralized control plane 1076 maintains a global view of all NDs and configured NEs/VNEs, and it maps the virtual networks to the underlying NDs efficiently (including maintaining these mappings as the physical network changes either through hardware (ND, link, or ND component) failure, addition, or removal).

[00100] While Figure 10D shows the distributed approach 1072 separate from the centralized approach 1074, the effort of network control may be distributed differently or the two combined in certain embodiments of the invention. For example: 1) embodiments may generally use the centralized approach (SDN) 1074, but have certain functions delegated to the NEs (e.g., the distributed approach may be used to implement one or more of fault monitoring, performance monitoring, protection switching, and primitives for neighbor and/or topology discovery); or 2) embodiments of the invention may perform neighbor discovery and topology discovery via both the centralized control plane and the distributed protocols, and the results compared to raise exceptions where they do not agree. Such embodiments are generally considered to fall under the centralized approach 1074, but may also be considered a hybrid approach.

[00101] While Figure 10D illustrates the simple case where each of the NDs 1000A-H implements a single NE 1070A-H, it should be understood that the network control approaches described with reference to Figure 10D also work for networks where one or more of the NDs 1000A-H implement multiple VNEs (e.g., VNEs 1030A-R, VNEs 1060A-R, those in the hybrid network device 1006). Alternatively or in addition, the network controller 1078 may also

emulate the implementation of multiple VNEs in a single ND. Specifically, instead of (or in addition to) implementing multiple VNEs in a single ND, the network controller 1078 may present the implementation of a VNE/NE in a single ND as multiple VNEs in the virtual networks 1092 (all in the same one of the virtual network(s) 1092, each in different ones of the virtual network(s) 1092, or some combination). For example, the network controller 1078 may cause an ND to implement a single VNE (a NE) in the underlay network, and then logically divide up the resources of that NE within the centralized control plane 1076 to present different VNEs in the virtual network(s) 1092 (where these different VNEs in the overlay networks are sharing the resources of the single VNE/NE implementation on the ND in the underlay network).

[00102] On the other hand, Figures 10E and 10F respectively illustrate exemplary abstractions of NEs and VNEs that the network controller 1078 may present as part of different ones of the virtual networks 1092. Figure 10E illustrates the simple case of where each of the NDs 1000A-H implements a single NE 1070A-H (see Figure 10D), but the centralized control plane 1076 has abstracted multiple of the NEs in different NDs (the NEs 1070A-C and G-H) into (to represent) a single NE 1070I in one of the virtual network(s) 1092 of Figure 10D, according to some embodiments of the invention. Figure 10E shows that in this virtual network, the NE 1070I is coupled to NE 1070D and 1070F, which are both still coupled to NE 1070E.

[00103] Figure 10F illustrates a case where multiple VNEs (VNE 1070A.1 and VNE 1070H.1) are implemented on different NDs (ND 1000A and ND 1000H) and are coupled to each other, and where the centralized control plane 1076 has abstracted these multiple VNEs such that they appear as a single VNE 1070T within one of the virtual networks 1092 of Figure 10D, according to some embodiments of the invention. Thus, the abstraction of a NE or VNE can span multiple NDs.

[00104] While some embodiments of the invention implement the centralized control plane 1076 as a single entity (e.g., a single instance of software running on a single electronic device), alternative embodiments may spread the functionality across multiple entities for redundancy and/or scalability purposes (e.g., multiple instances of software running on different electronic devices).

[00105] Similar to the network device implementations, the electronic device(s) running the centralized control plane 1076, and thus the network controller 1078 including the centralized reachability and forwarding information module 1079, may be implemented a variety of ways (e.g., a special purpose device, a general-purpose (e.g., COTS) device, or hybrid device). These electronic device(s) would similarly include compute resource(s), a set or one or more physical NICs, and a non-transitory machine-readable storage medium having stored thereon the

centralized control plane software. For instance, Figure 11 illustrates, a general purpose control plane device 1104 including hardware 1140 comprising a set of one or more processor(s) 1142 (which are often COTS processors) and network interface controller(s) 1144 (NICs; also known as network interface cards) (which include physical NIs 1146), as well as non-transitory machine readable storage media 1148 having stored therein centralized control plane (CCP) software 1150.

[00106] In embodiments that use compute virtualization, the processor(s) 1142 typically execute software to instantiate a virtualization layer 1154 (e.g., in one embodiment the virtualization layer 1154 represents the kernel of an operating system (or a shim executing on a base operating system) that allows for the creation of multiple instances 1162A-R called software containers (representing separate user spaces and also called virtualization engines, virtual private servers, or jails) that may each be used to execute a set of one or more applications; in another embodiment the virtualization layer 1154 represents a hypervisor (sometimes referred to as a virtual machine monitor (VMM)) or a hypervisor executing on top of a host operating system, and an application is run on top of a guest operating system within an instance 1162A-R called a virtual machine (which in some cases may be considered a tightly isolated form of software container) that is run by the hypervisor ; in another embodiment, an application is implemented as a unikernel, which can be generated by compiling directly with an application only a limited set of libraries (e.g., from a library operating system (LibOS) including drivers/libraries of OS services) that provide the particular OS services needed by the application, and the unikernel can run directly on hardware 1140, directly on a hypervisor represented by virtualization layer 1154 (in which case the unikernel is sometimes described as running within a LibOS virtual machine), or in a software container represented by one of instances 1162A-R). Again, in embodiments where compute virtualization is used, during operation an instance of the CCP software 1150 (illustrated as CCP instance 1176A) is executed (e.g., within the instance 1162A) on the virtualization layer 1154. In embodiments where compute virtualization is not used, the CCP instance 1176A is executed, as a unikernel or on top of a host operating system, on the "bare metal" general purpose control plane device 1104. The instantiation of the CCP instance 1176A, as well as the virtualization layer 1154 and instances 1162A-R if implemented, are collectively referred to as software instance(s) 1152.

[00107] In some embodiments, the CCP instance 1176A includes a network controller instance 1178. The network controller instance 1178 includes a centralized reachability and forwarding information module instance 1179 (which is a middleware layer providing the context of the network controller 1078 to the operating system and communicating with the various NEs), and an CCP application layer 1180 (sometimes referred to as an application layer)

over the middleware layer (providing the intelligence required for various network operations such as protocols, network situational awareness, and user – interfaces). At a more abstract level, this CCP application layer 1180 within the centralized control plane 1076 works with virtual network view(s) (logical view(s) of the network) and the middleware layer provides the conversion from the virtual networks to the physical view. The network controller instance further includes NonStop Routing Instance 1181, which enables the control plane device 1104 to perform operations described with reference to Figures 1-9.

[00108] The centralized control plane 1076 transmits relevant messages to the data plane 1080 based on CCP application layer 1180 calculations and middleware layer mapping for each flow. A flow may be defined as a set of packets whose headers match a given pattern of bits; in this sense, traditional IP forwarding is also flow–based forwarding where the flows are defined by the destination IP address for example; however, in other implementations, the given pattern of bits used for a flow definition may include more fields (e.g., 10 or more) in the packet headers. Different NDs/NEs/VNEs of the data plane 1080 may receive different messages, and thus different forwarding information. The data plane 1080 processes these messages and programs the appropriate flow information and corresponding actions in the forwarding tables (sometime referred to as flow tables) of the appropriate NE/VNEs, and then the NEs/VNEs map incoming packets to flows represented in the forwarding tables and forward packets based on the matches in the forwarding tables.

[00109] Standards such as OpenFlow define the protocols used for the messages, as well as a model for processing the packets. The model for processing packets includes header parsing, packet classification, and making forwarding decisions. Header parsing describes how to interpret a packet based upon a well-known set of protocols. Some protocol fields are used to build a match structure (or key) that will be used in packet classification (e.g., a first key field could be a source media access control (MAC) address, and a second key field could be a destination MAC address).

[00110] Packet classification involves executing a lookup in memory to classify the packet by determining which entry (also referred to as a forwarding table entry or flow entry) in the forwarding tables best matches the packet based upon the match structure, or key, of the forwarding table entries. It is possible that many flows represented in the forwarding table entries can correspond/match to a packet; in this case the system is typically configured to determine one forwarding table entry from the many according to a defined scheme (e.g., selecting a first forwarding table entry that is matched). Forwarding table entries include both a specific set of match criteria (a set of values or wildcards, or an indication of what portions of a packet should be compared to a particular value/values/wildcards, as defined by the matching

capabilities – for specific fields in the packet header, or for some other packet content), and a set of one or more actions for the data plane to take on receiving a matching packet. For example, an action may be to push a header onto the packet, for the packet using a particular port, flood the packet, or simply drop the packet. Thus, a forwarding table entry for IPv4/IPv6 packets with a particular transmission control protocol (TCP) destination port could contain an action specifying that these packets should be dropped.

[00111] Making forwarding decisions and performing actions occurs, based upon the forwarding table entry identified during packet classification, by executing the set of actions identified in the matched forwarding table entry on the packet.

[00112] However, when an unknown packet (for example, a "missed packet" or a "match-miss" as used in OpenFlow parlance) arrives at the data plane 1080, the packet (or a subset of the packet header and content) is typically forwarded to the centralized control plane 1076. The centralized control plane 1076 will then program forwarding table entries into the data plane 1080 to accommodate packets belonging to the flow of the unknown packet. Once a specific forwarding table entry has been programmed into the data plane 1080 by the centralized control plane 1076, the next packet with matching credentials will match that forwarding table entry and take the set of actions associated with that matched entry.

[00113] A network interface (NI) may be physical or virtual; and in the context of IP, an interface address is an IP address assigned to a NI, be it a physical NI or virtual NI. A virtual NI may be associated with a physical NI, with another virtual interface, or stand on its own (e.g., a loopback interface, a point-to-point protocol interface). A NI (physical or virtual) may be numbered (a NI with an IP address) or unnumbered (a NI without an IP address). A loopback interface (and its loopback address) is a specific type of virtual NI (and IP address) of a NE/VNE (physical or virtual) often used for management purposes; where such an IP address is referred to as the nodal loopback address. The IP address(es) assigned to the NI(s) of a ND are referred to as IP addresses of that ND; at a more granular level, the IP address(es) assigned to NI(s) assigned to a NE/VNE implemented on a ND can be referred to as IP addresses of that NE/VNE.

[00114] Next hop selection by the routing system for a given destination may resolve to one path (that is, a routing protocol may generate one next hop on a shortest path); but if the routing system determines there are multiple viable next hops (that is, the routing protocol generated forwarding solution offers more than one next hop on a shortest path – multiple equal cost next hops), some additional criteria is used - for instance, in a connectionless network, Equal Cost Multi Path (ECMP) (also known as Equal Cost Multi Pathing, multipath forwarding and IP multipath) may be used (e.g., typical implementations use as the criteria particular header fields

to ensure that the packets of a particular packet flow are always forwarded on the same next hop to preserve packet flow ordering). For purposes of multipath forwarding, a packet flow is defined as a set of packets that share an ordering constraint. As an example, the set of packets in a particular TCP transfer sequence need to arrive in order, else the TCP logic will interpret the out of order delivery as congestion and slow the TCP transfer rate down.

[00115] Some NDs include functionality for authentication, authorization, and accounting (AAA) protocols (e.g., RADIUS (Remote Authentication Dial-In User Service), Diameter, and/or TACACS+ (Terminal Access Controller Access Control System Plus). AAA can be provided through a client/server model, where the AAA client is implemented on a ND and the AAA server can be implemented either locally on the ND or on a remote electronic device coupled with the ND. Authentication is the process of identifying and verifying a subscriber. For instance, a subscriber might be identified by a combination of a username and a password or through a unique key. Authorization determines what a subscriber can do after being authenticated, such as gaining access to certain electronic device information resources (e.g., through the use of access control policies). Accounting is recording user activity. By way of a summary example, end user devices may be coupled (e.g., through an access network) through an edge ND (supporting AAA processing) coupled to core NDs coupled to electronic devices implementing servers of service/content providers. AAA processing is performed to identify for a subscriber the subscriber record stored in the AAA server for that subscriber. A subscriber record includes a set of attributes (e.g., subscriber name, password, authentication information, access control information, rate-limiting information, policing information) used during processing of that subscriber's traffic.

[00116] Certain NDs (e.g., certain edge NDs) internally represent end user devices (or sometimes customer premise equipment (CPE) such as a residential gateway (e.g., a router, modem)) using subscriber circuits. A subscriber circuit uniquely identifies within the ND a subscriber session and typically exists for the lifetime of the session. Thus, a ND typically allocates a subscriber circuit when the subscriber connects to that ND, and correspondingly de-allocates that subscriber circuit when that subscriber disconnects. Each subscriber session represents a distinguishable flow of packets communicated between the ND and an end user device (or sometimes CPE such as a residential gateway or modem) using a protocol, such as the point-to-point protocol over another protocol (PPPoX) (e.g., where X is Ethernet or Asynchronous Transfer Mode (ATM)), Ethernet, 802.1Q Virtual LAN (VLAN), Internet Protocol, or ATM). A subscriber session can be initiated using a variety of mechanisms (e.g., manual provisioning a dynamic host configuration protocol (DHCP), DHCP/client-less internet protocol service (CLIPS) or Media Access Control (MAC) address tracking). For example, the

point-to-point protocol (PPP) is commonly used for digital subscriber line (DSL) services and requires installation of a PPP client that enables the subscriber to enter a username and a password, which in turn may be used to select a subscriber record. When DHCP is used (e.g., for cable modem services), a username typically is not provided; but in such situations other information (e.g., information that includes the MAC address of the hardware in the end user device (or CPE)) is provided. The use of DHCP and CLIPS on the ND captures the MAC addresses and uses these addresses to distinguish subscribers and access their subscriber records.

[00117] A virtual circuit (VC), synonymous with virtual connection and virtual channel, is a connection oriented communication service that is delivered by means of packet mode communication. Virtual circuit communication resembles circuit switching, since both are connection oriented, meaning that in both cases data is delivered in correct order, and signaling overhead is required during a connection establishment phase. Virtual circuits may exist at different layers. For example, at layer 4, a connection oriented transport layer datalink protocol such as Transmission Control Protocol (TCP) may rely on a connectionless packet switching network layer protocol such as IP, where different packets may be routed over different paths, and thus be delivered out of order. Where a reliable virtual circuit is established with TCP on top of the underlying unreliable and connectionless IP protocol, the virtual circuit is identified by the source and destination network socket address pair, i.e. the sender and receiver IP address and port number. However, a virtual circuit is possible since TCP includes segment numbering and reordering on the receiver side to prevent out-of-order delivery. Virtual circuits are also possible at Layer 3 (network layer) and Layer 2 (datalink layer); such virtual circuit protocols are based on connection oriented packet switching, meaning that data is always delivered along the same network path, i.e. through the same NEs/VNEs. In such protocols, the packets are not routed individually and complete addressing information is not provided in the header of each data packet; only a small virtual channel identifier (VCI) is required in each packet; and routing information is transferred to the NEs/VNEs during the connection establishment phase; switching only involves looking up the virtual channel identifier in a table rather than analyzing a complete address. Examples of network layer and datalink layer virtual circuit protocols, where data always is delivered over the same path: X.25, where the VC is identified by a virtual channel identifier (VCI); Frame relay, where the VC is identified by a VCI; Asynchronous Transfer Mode (ATM), where the circuit is identified by a virtual path identifier (VPI) and virtual channel identifier (VCI) pair; General Packet Radio Service (GPRS); and Multiprotocol label switching (MPLS), which can be used for IP over virtual circuits (Each circuit is identified by a label).

[00118] Each VNE (e.g., a virtual router, a virtual bridge (which may act as a virtual switch instance in a Virtual Private LAN Service (VPLS) is typically independently administrable. For example, in the case of multiple virtual routers, each of the virtual routers may share system resources but is separate from the other virtual routers regarding its management domain, AAA (authentication, authorization, and accounting) name space, IP address, and routing database(s). Multiple VNEs may be employed in an edge ND to provide direct network access and/or different classes of services for subscribers of service and/or content providers.

[00119] Within certain NDs, "interfaces" that are independent of physical NIs may be configured as part of the VNEs to provide higher-layer protocol and service information (e.g., Layer 3 addressing). The subscriber records in the AAA server identify, in addition to the other subscriber configuration requirements, to which context (e.g., which of the VNEs/NEs) the corresponding subscribers should be bound within the ND. As used herein, a binding forms an association between a physical entity (e.g., physical NI, channel) or a logical entity (e.g., circuit such as a subscriber circuit or logical circuit (a set of one or more subscriber circuits)) and a context's interface over which network protocols (e.g., routing protocols, bridging protocols) are configured for that context. Subscriber data flows on the physical entity when some higher-layer protocol interface is configured and associated with that physical entity.

[00120] For example, while the flow diagrams in the figures show a particular order of operations performed by certain embodiments of the invention, it should be understood that such order is exemplary (e.g., alternative embodiments may perform the operations in a different order, combine certain operations, overlap certain operations, etc.).

[00121] While the invention has been described in terms of several embodiments, those skilled in the art will recognize that the invention is not limited to the embodiments described, can be practiced with modification and alteration within the spirit and scope of the appended claims. The description is thus to be regarded as illustrative instead of limiting.

CLAIMS

What is claimed is:

1. A method in a first network element coupled with a second network element, wherein the first and the second network elements are part of a redundancy system, and wherein the first network element is in an active state, of enabling nonstop routing, the method comprising:

receiving (502) one or more transport protocol packets from a peer network device that has a routing protocol session established with the first network element, wherein the one or more transport protocol packets include a routing protocol message associated with the routing protocol session;

processing (504) the transport protocol packets to retrieve the routing protocol message; and

transmitting (506) a synchronization message to the second network element, wherein the synchronization message includes the retrieved routing protocol message, an identifier of a transport protocol session associated with the routing protocol session, and current transport protocol states associated with the routing protocol session.

2. The method of claim 1, wherein the synchronization message causes (508) the second network element to update routing protocol information associated with a second transport protocol session to be used for transmitting routing protocol messages between the peer network device and the second network element when a switchover occurs within the redundancy system causing the second network element to assume an active role.

3. The method of claim 1, wherein the method further comprises prior to receiving the one or more transport protocol packets:

upon determination that the second network element has joined (1) the redundancy system, transmitting (2a) initial routing protocol information to the second network element;

transmitting (2b) information indicating a mapping between routing protocol sessions and transport protocol sessions established at the first network element; and

causing the second network element to allocate (202) a new transport protocol session associated with the peer network device and the second network element, wherein the new transport protocol session corresponds to a transport protocol session established at the first network element.

4. The method of claim 3, wherein the transmitting the information indicating the mapping between routing protocol sessions and transport protocol sessions further causes the second network element to,

store (206) a mapping between the new transport protocol session and the transport protocol session established at the first network element associated with the routing protocol session; and

update (208) the new transport protocol session according to the transport protocol session established at the first network element.

5. The method of claim 1, wherein the method further comprises registering (620) to receive an acknowledgment message from the second network element.

6. The method of claim 5, further comprising:

receiving an acknowledgment message from the second network element indicating that the synchronization message has been received, wherein the acknowledgment message causes the first network element to transmit a transport protocol acknowledgment message to the peer network device.

7. The method of claim 1, wherein the routing protocol is Border Gateway Protocol (BGP) and the transport protocol is Transmission Control Protocol (TCP).

8. A first network element to be coupled with a second network element, wherein the first and the second network elements are part of a redundancy system, and wherein the first network element is in an active state, for enabling nonstop routing, the first network element comprising:

a non-transitory computer readable medium to store instructions; and

a processor coupled with the non-transitory computer readable medium to process the stored instructions to:

receive (502) one or more transport protocol packets from a peer network device that has a routing protocol session established with the first network element, wherein the one or more transport protocol packets include a routing protocol message associated with the routing protocol session,

process (504) the transport protocol packets to retrieve the routing protocol message, and

transmit (506) a synchronization message to the second network element, wherein the synchronization message includes the retrieved routing protocol message, an identifier of a transport protocol session associated

with the routing protocol session, and current transport protocol states associated with the transport protocol session.

9. The first network element of claim 8, wherein the synchronization message causes (508) the second network element to update routing protocol information associated with a second transport protocol session to be used for transmitting routing protocol messages between the peer network device and the second network element when a switchover occurs within the redundancy system causing the second network element to assume an active role.

10. The first network element of claim 8, wherein the first network element is further, prior to receiving the one or more transport protocol packets, to:

upon determination that the second network element has joined (1) the redundancy system, transmit (2a) initial routing protocol information to the second network element;

transmit (2b) information indicating a mapping between routing protocol sessions and transport protocol sessions established at the first network element; and

cause the second network element to allocate (202) a new transport protocol session associated with the peer network device and the second network element, wherein the new transport protocol session corresponds to a transport protocol session established at the first network element.

11. The first network element of claim 10, wherein to transmit the information indicating the mapping between routing protocol sessions and transport protocol sessions further causes the second network element to,

store (206) a mapping between the new transport protocol session and the transport protocol session established at the first network element associated with the routing protocol session; and

update (208) the new transport protocol session according to the transport protocol session established at the first network element.

12. The first network element of claim 8, wherein the first network element is further to register (620) to receive an acknowledgment message from the second network element.

13. The first network element of claim 12, wherein the first network element is further to receive an acknowledgment message from the second network element indicating that the synchronization message has been received, wherein the acknowledgment message causes the first network element to transmit a transport protocol acknowledgment message to the peer network device.

14. The first network element of claim 8, wherein the routing protocol is Border Gateway Protocol (BGP) and the transport protocol is Transmission Control Protocol (TCP).

15. A non-transitory computer readable storage medium that provide instructions, which when executed by a processor of a first network element to be coupled with a second network element, wherein the first and the second network elements are part of a redundancy system, and wherein the first network element is in an active state, cause said processor to perform operations comprising:

receiving (502) one or more transport protocol packets from a peer network device that has a routing protocol session established with the first network element, wherein the one or more transport protocol packets include a routing protocol message associated with the routing protocol session;

processing (504) the transport protocol packets to retrieve the routing protocol message; and

transmitting (506) a synchronization message to the second network element, wherein the synchronization message includes the retrieved routing protocol message, an identifier of a transport protocol session associated with the routing protocol session, and current transport protocol states associated with the transport protocol session.

16. The non-transitory computer readable storage medium of claim 15, wherein the synchronization message causes (508) the second network element to update routing protocol information associated with a second transport protocol session to be used for transmitting routing protocol messages between the peer network device and the second network element when a switchover occurs within the redundancy system causing the second network element to assume an active role.

17. The non-transitory computer readable storage medium of claim 15, wherein the operations further comprise prior to receiving the one or more transport protocol packets:

upon determination that the second network element has joined (1) the redundancy system, transmitting (2a) initial routing protocol information to the second network element;

transmitting (2b) information indicating a mapping between routing protocol sessions and transport protocol sessions established at the first network element; and

causing the second network element to allocate (202) a new transport protocol session associated with the peer network device and the second network element,

wherein the new transport protocol session corresponds to a transport protocol session established at the first network element.

18. The non-transitory computer readable storage medium of claim 17, wherein the transmitting the information indicating the mapping between routing protocol sessions and transport protocol sessions further causes the second network element to,

store (206) a mapping between the new transport protocol session and the transport protocol session established at the first network element associated with the routing protocol session; and

update (208) the new transport protocol session according to the transport protocol session established at the first network element.

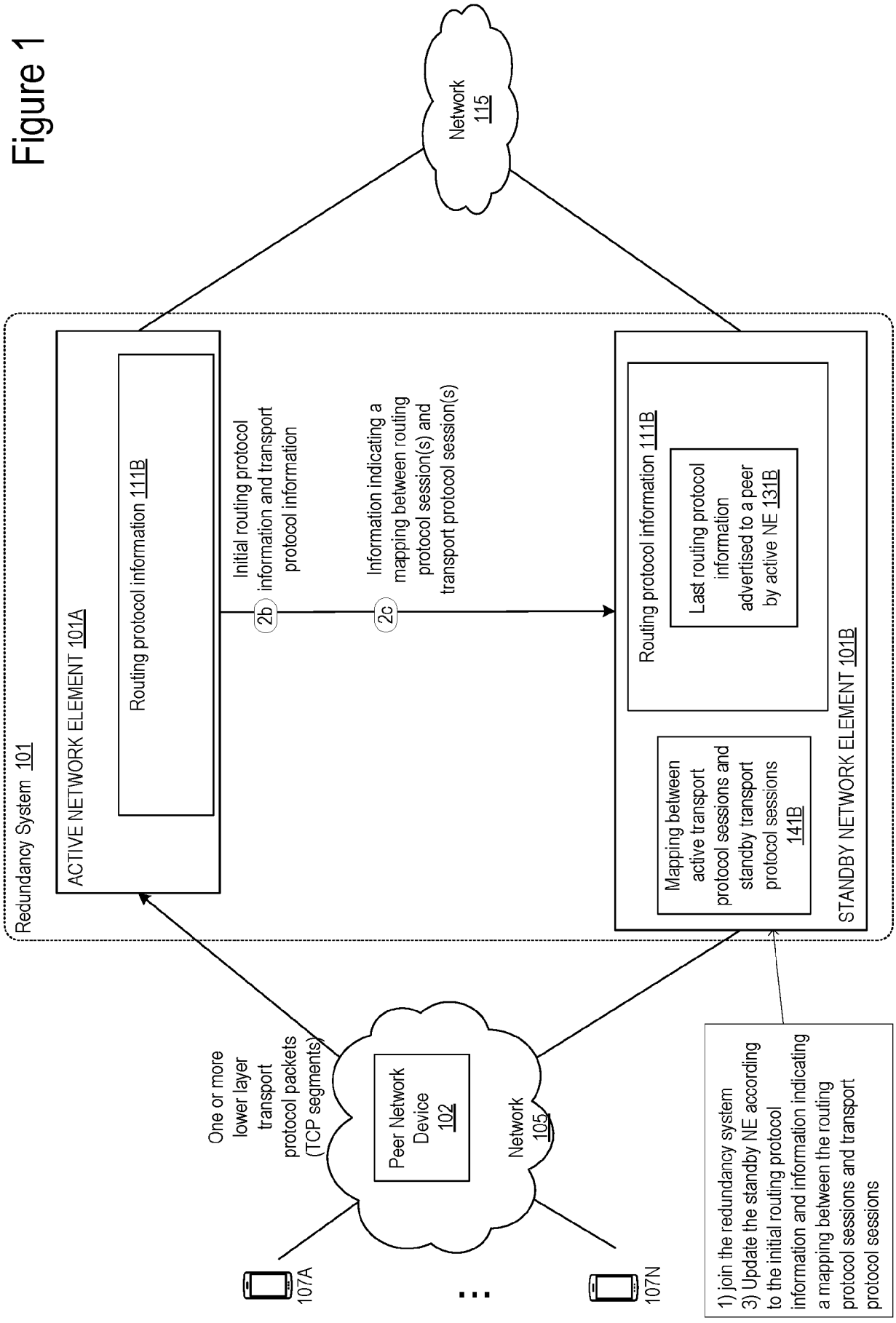
19. The non-transitory computer readable storage medium of claim 15, wherein the operations further comprise registering (620) to receive an acknowledgment message from the second network element.

20. The non-transitory computer readable storage medium of claim 19, wherein the operations further comprise:

receiving an acknowledgment message from the second network element indicating that the synchronization message has been received, wherein the acknowledgment message causes the first network element to transmit a transport protocol acknowledgment message to the peer network device.

21. The non-transitory computer readable storage medium of claim 15, wherein the routing protocol is Border Gateway Protocol (BGP) and the transport protocol is Transmission Control Protocol (TCP).

Figure 1


Network 115
Redundancy System 101
ACTIVE NETWORK ELEMENT 101A
Routing protocol information 111B
Initial routing protocol information and transport protocol information
2b
Information indicating a mapping between routing protocol session(s) and transport protocol session(s)
2c
Routing protocol information 111B
Last routing protocol information advertised to a peer by active NE 131B
Mapping between active transport protocol sessions and standby transport protocol sessions 141B
STANDBY NETWORK ELEMENT 101B
One or more lower layer transport protocol packets (TCP segments)
Peer Network Device 102
Network 105
107A
...
107N
1) join the redundancy system
3) Update the standby NE according to the initial routing protocol information and information indicating a mapping between the routing protocol sessions and transport protocol sessions

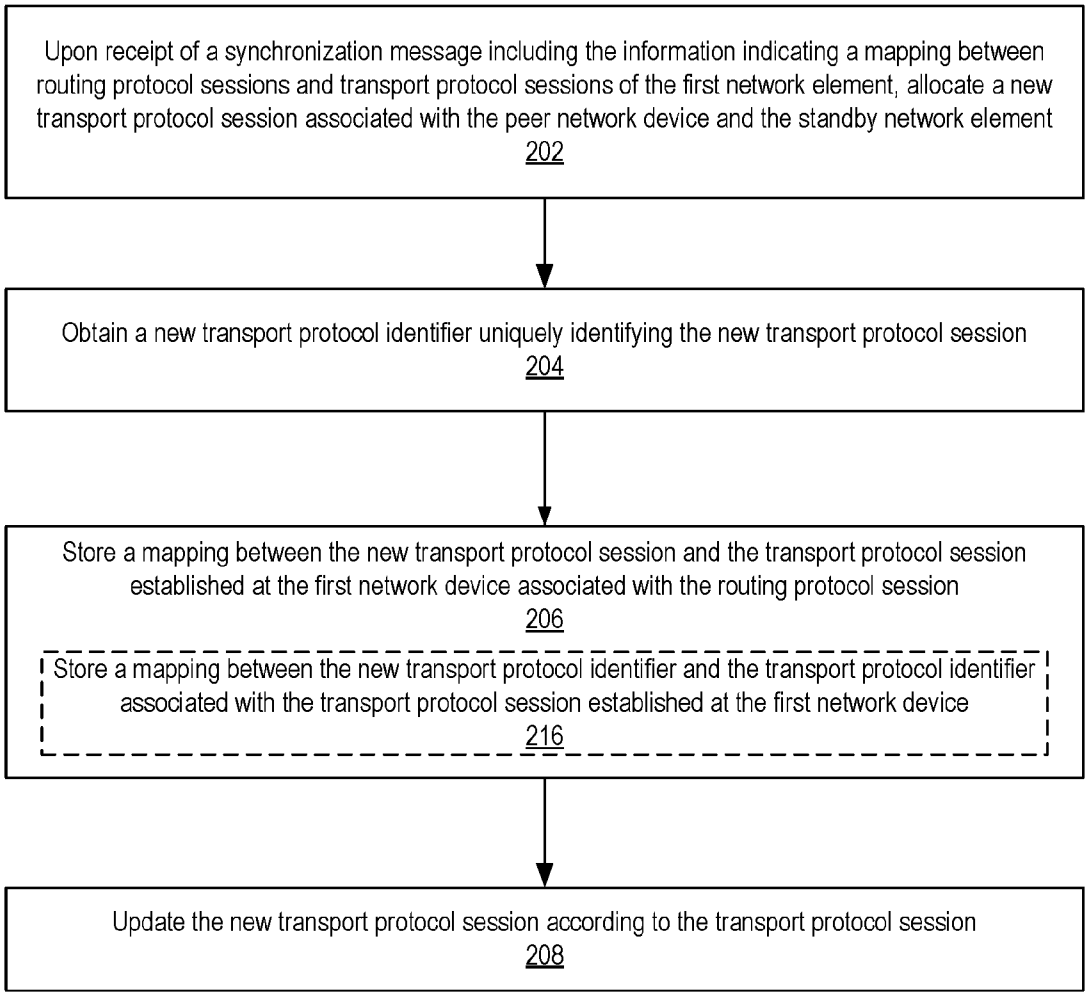

Upon receipt of a synchronization message including the information indicating a mapping between routing protocol sessions and transport protocol sessions of the first network element, allocate a new transport protocol session associated with the peer network device and the standby network element
202
Obtain a new transport protocol identifier uniquely identifying the new transport protocol session
204
Store a mapping between the new transport protocol session and the transport protocol session established at the first network device associated with the routing protocol session
206
Store a mapping between the new transport protocol identifier and the transport protocol identifier associated with the transport protocol session established at the first network device
216
Update the new transport protocol session according to the transport protocol session
208

Figure 2

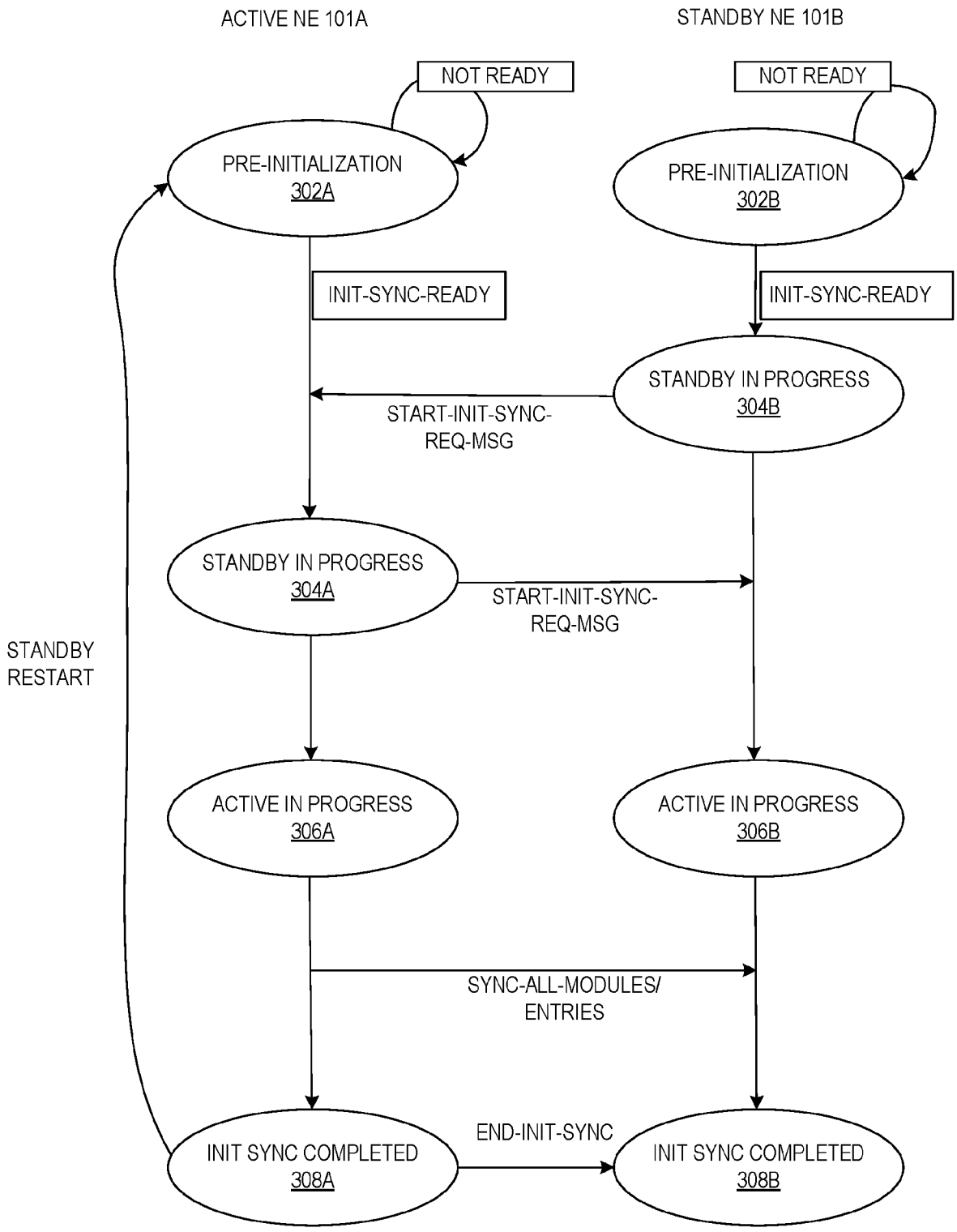
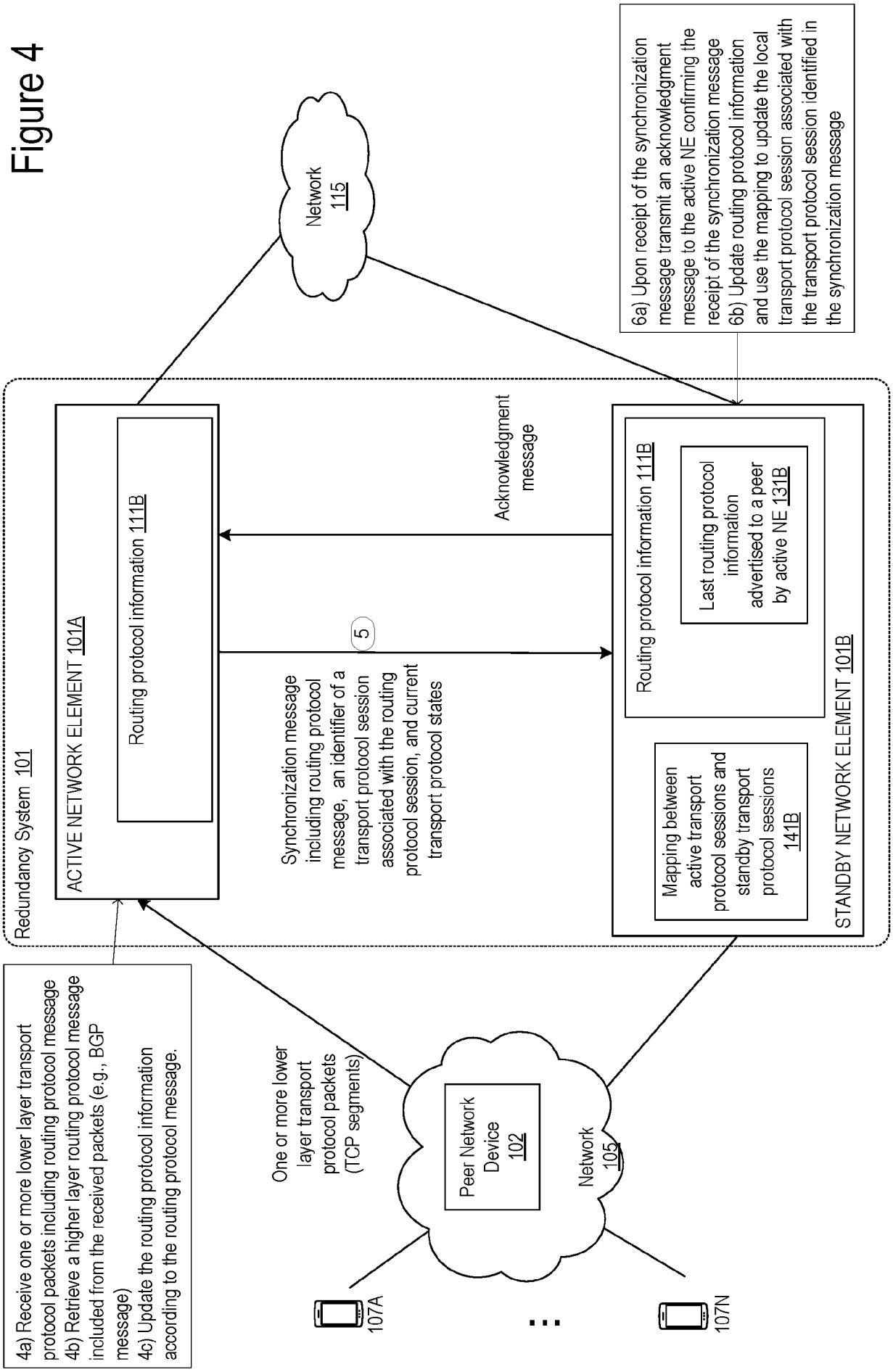

ACTIVE NE 101A
STANDBY NE 101B
NOT READY
NOT READY
PRE-INITIALIZATION
302A
PRE-INITIALIZATION
302B
INIT-SYNC-READY
INIT-SYNC-READY
STANDBY IN PROGRESS
304B
START-INIT-SYNC-
REQ-MSG
STANDBY IN PROGRESS
304A
START-INIT-SYNC-
REQ-MSG
STANDBY
RESTART
ACTIVE IN PROGRESS
306A
ACTIVE IN PROGRESS
306B
SYNC-ALL-MODULES/
ENTRIES
END-INIT-SYNC
INIT SYNC COMPLETED
308A
INIT SYNC COMPLETED
308B

Figure 3

Figure 4


Network 115
6a) Upon receipt of the synchronization message transmit an acknowledgment message to the active NE confirming the receipt of the synchronization message 6b) Update routing protocol information and use the mapping to update the local transport protocol session associated with the transport protocol session identified in the synchronization message
Redundancy System 101
ACTIVE NETWORK ELEMENT 101A
Routing protocol information 111B
Synchronization message including routing protocol message, an identifier of a transport protocol session associated with the routing protocol session, and current transport protocol states
5
Acknowledgment message
Routing protocol information 111B
Last routing protocol information advertised to a peer by active NE 131B
Mapping between active transport protocol sessions and standby transport protocol sessions 141B
STANDBY NETWORK ELEMENT 101B
4a) Receive one or more lower layer transport protocol packets including routing protocol message
4b) Retrieve a higher layer routing protocol message included from the received packets (e.g., BGP message)
4c) Update the routing protocol information according to the routing protocol message.
One or more lower layer transport protocol packets (TCP segments)
Peer Network Device 102
Network 105
107A
107N

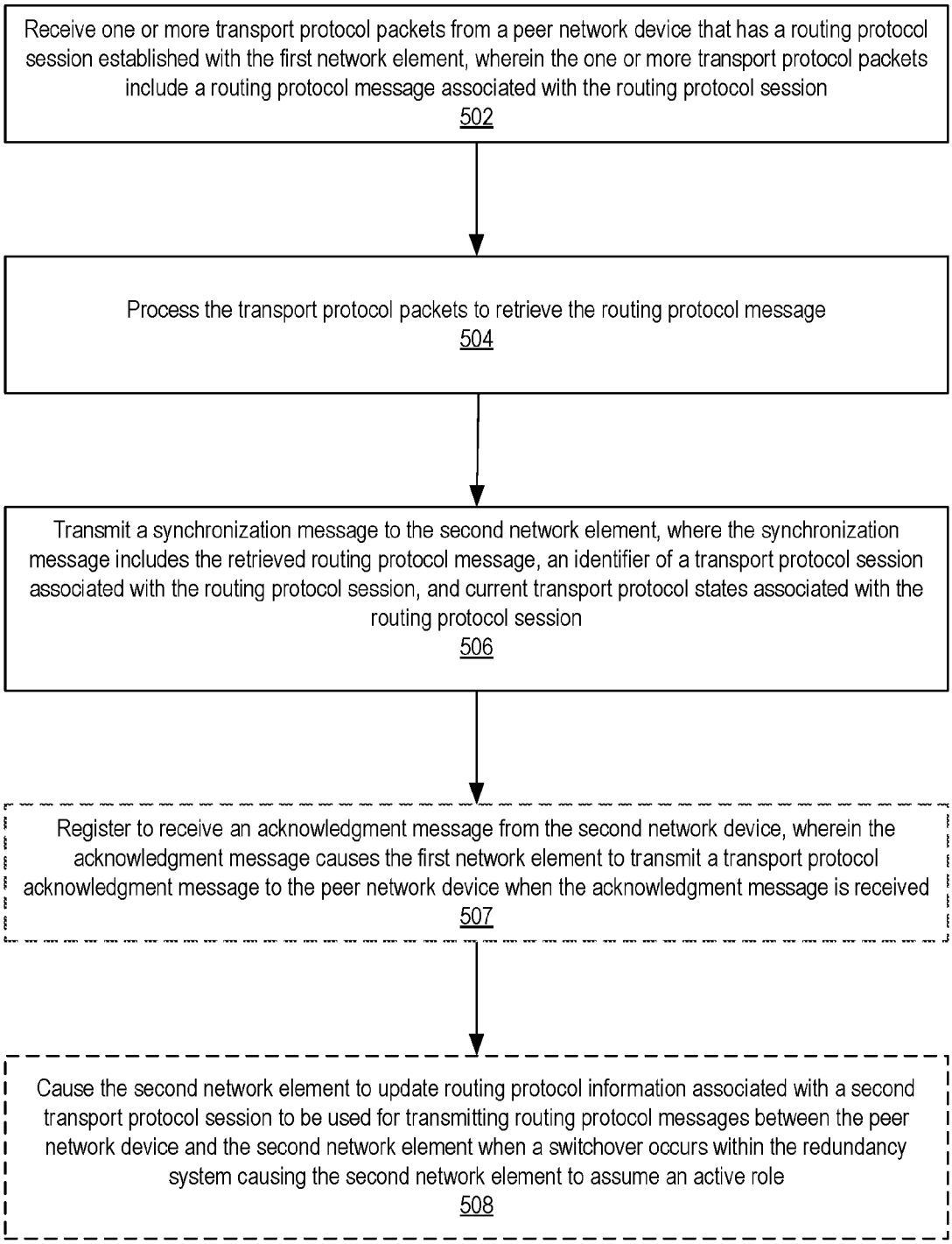

Receive one or more transport protocol packets from a peer network device that has a routing protocol session established with the first network element, wherein the one or more transport protocol packets include a routing protocol message associated with the routing protocol session
502
Process the transport protocol packets to retrieve the routing protocol message
504
Transmit a synchronization message to the second network element, where the synchronization message includes the retrieved routing protocol message, an identifier of a transport protocol session associated with the routing protocol session, and current transport protocol states associated with the routing protocol session
506
Register to receive an acknowledgment message from the second network device, wherein the acknowledgment message causes the first network element to transmit a transport protocol acknowledgment message to the peer network device when the acknowledgment message is received
507
Cause the second network element to update routing protocol information associated with a second transport protocol session to be used for transmitting routing protocol messages between the peer network device and the second network element when a switchover occurs within the redundancy system causing the second network element to assume an active role
508

Figure 5

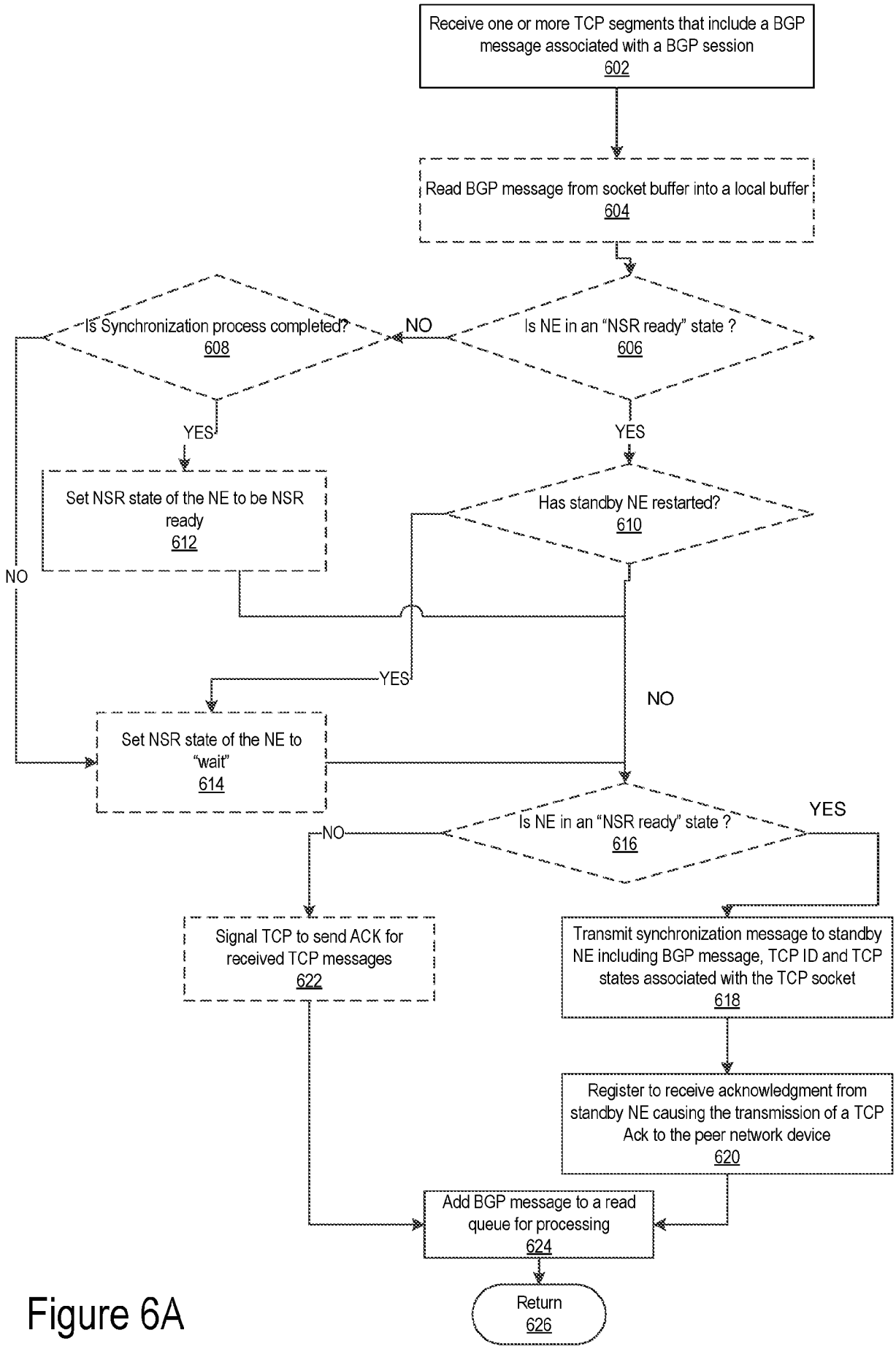

Receive one or more TCP segments that include a BGP message associated with a BGP session
602
Read BGP message from socket buffer into a local buffer
604
Is NE in an "NSR ready" state ?
606
NO
Is Synchronization process completed?
608
YES
YES
Set NSR state of the NE to be NSR ready
612
Has standby NE restarted?
610
NO
YES
NO
Set NSR state of the NE to "wait"
614
Is NE in an "NSR ready" state ?
616
NO
YES
Signal TCP to send ACK for received TCP messages
622
Transmit synchronization message to standby NE including BGP message, TCP ID and TCP states associated with the TCP socket
618
Register to receive acknowledgment from standby NE causing the transmission of a TCP Ack to the peer network device
620
Add BGP message to a read queue for processing
624
Return
626

Figure 6A

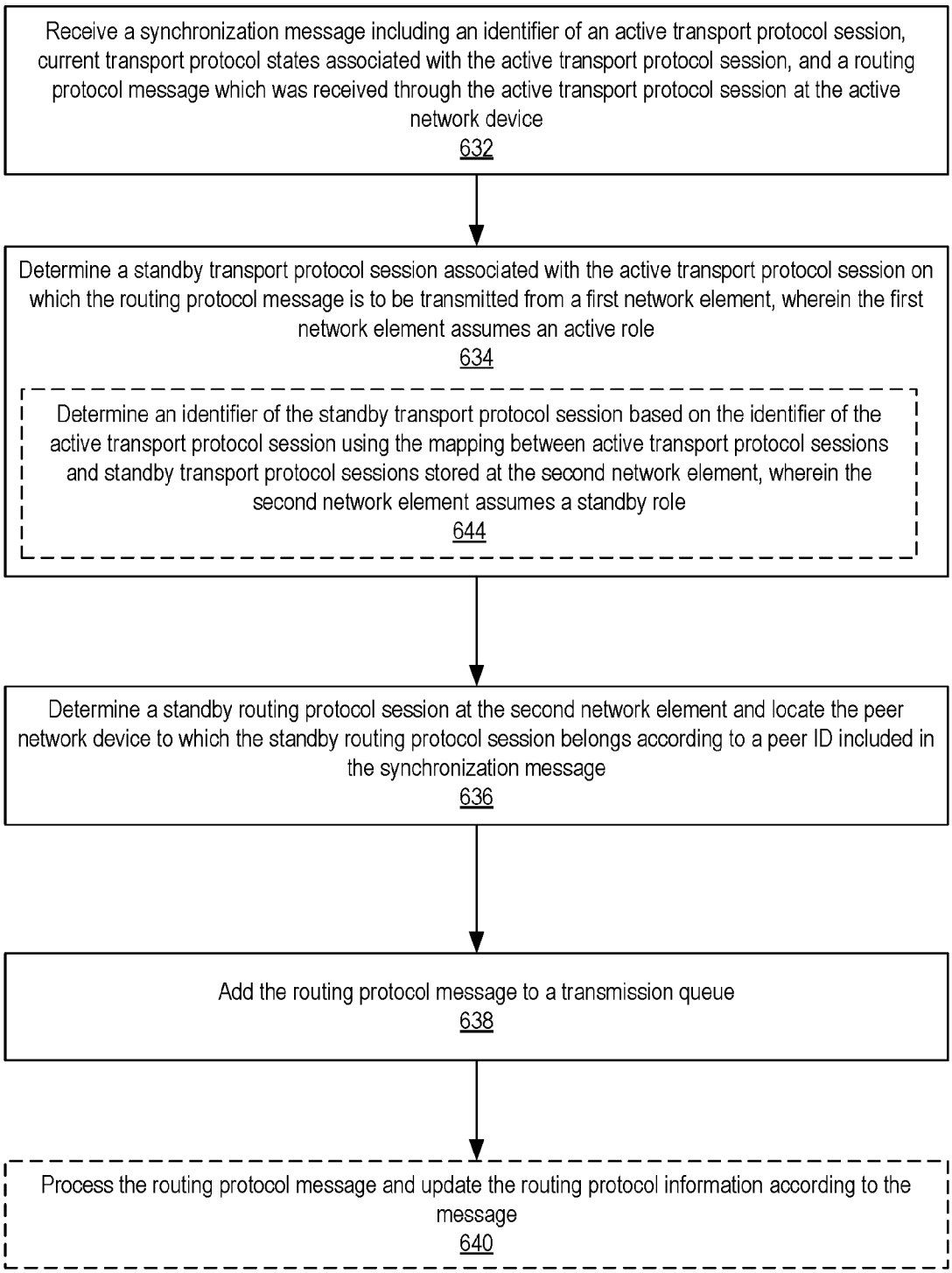
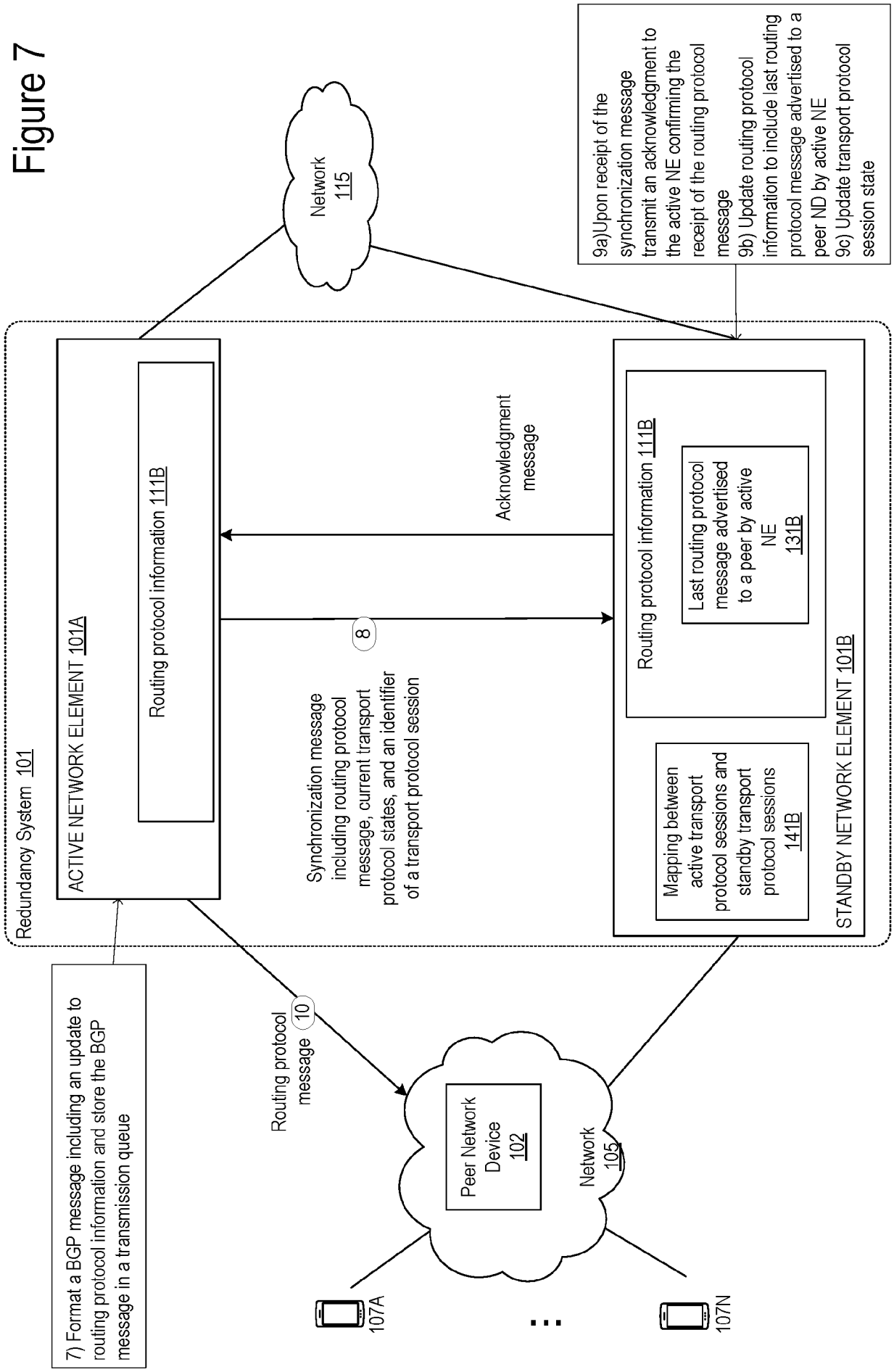

Receive a synchronization message including an identifier of an active transport protocol session, current transport protocol states associated with the active transport protocol session, and a routing protocol message which was received through the active transport protocol session at the active network device
632
Determine a standby transport protocol session associated with the active transport protocol session on which the routing protocol message is to be transmitted from a first network element, wherein the first network element assumes an active role
634
Determine an identifier of the standby transport protocol session based on the identifier of the active transport protocol session using the mapping between active transport protocol sessions and standby transport protocol sessions stored at the second network element, wherein the second network element assumes a standby role
644
Determine a standby routing protocol session at the second network element and locate the peer network device to which the standby routing protocol session belongs according to a peer ID included in the synchronization message
636
Add the routing protocol message to a transmission queue
638
Process the routing protocol message and update the routing protocol information according to the message
640

Figure 6B

Figure 7


Network 115
9a)Upon receipt of the synchronization message transmit an acknowledgment to the active NE confirming the receipt of the routing protocol message
9b) Update routing protocol information to include last routing protocol message advertised to a peer ND by active NE
9c) Update transport protocol session state
Redundancy System 101
ACTIVE NETWORK ELEMENT 101A
Routing protocol information 111B
Acknowledgment message
8
Synchronization message including routing protocol message, current transport protocol states, and an identifier of a transport protocol session
Routing protocol information 111B
Last routing protocol message advertised to a peer by active NE 131B
Mapping between active transport protocol sessions and standby transport protocol sessions 141B
STANDBY NETWORK ELEMENT 101B
7) Format a BGP message including an update to routing protocol information and store the BGP message in a transmission queue
Routing protocol message 10
Peer Network Device 102
Network 105
107A
...
107N

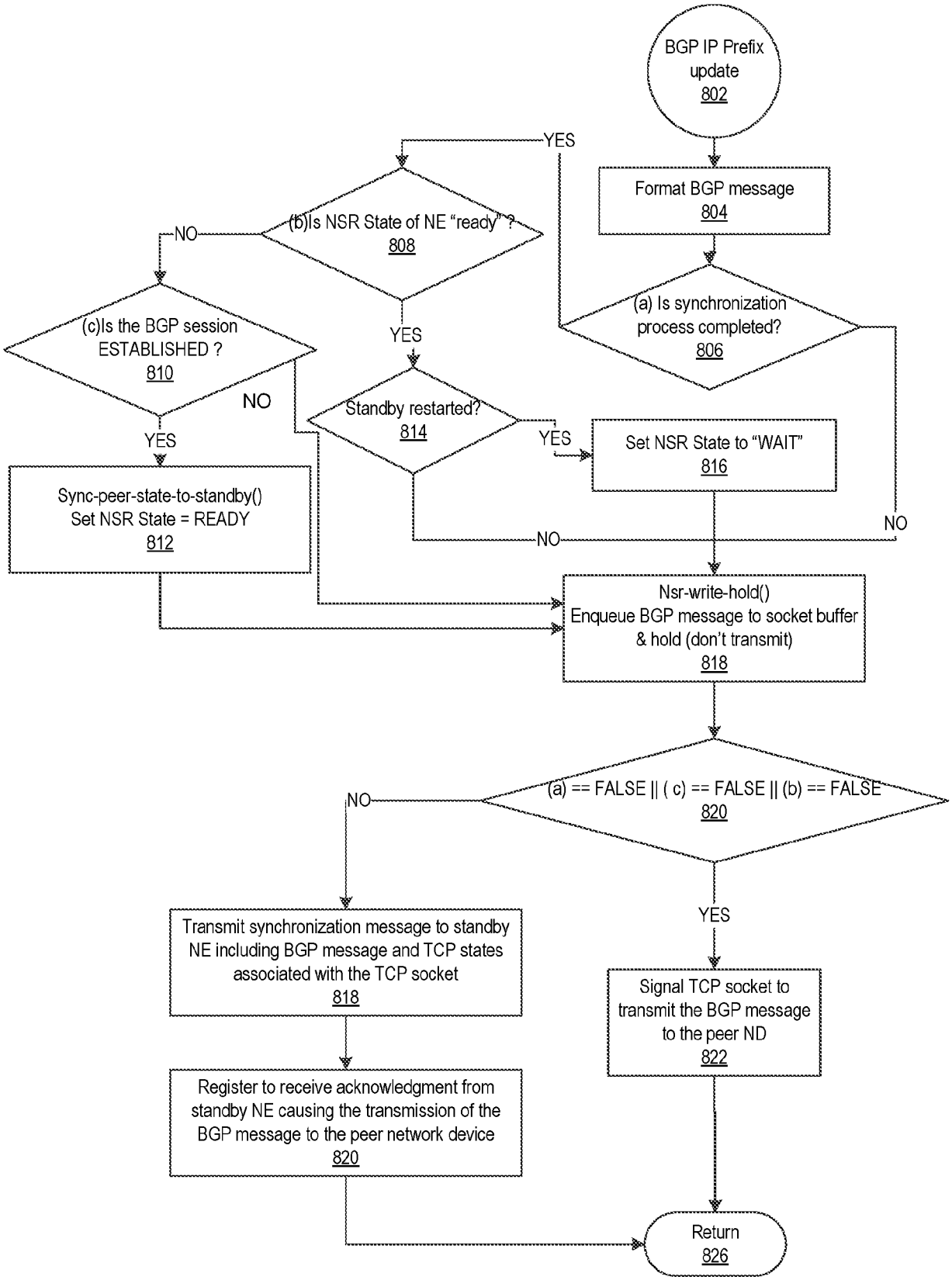

BGP IP Prefix update
802
Format BGP message
804
(a) Is synchronization process completed?
806
YES
NO
(b)Is NSR State of NE "ready" ?
808
YES
NO
(c)Is the BGP session ESTABLISHED ?
810
YES
NO
Standby restarted?
814
YES
NO
Set NSR State to "WAIT"
816
Sync-peer-state-to-standby()
Set NSR State = READY
812
Nsr-write-hold()
Enqueue BGP message to socket buffer & hold (don't transmit)
818
(a) == FALSE || (c) == FALSE || (b) == FALSE
820
NO
YES
Transmit synchronization message to standby NE including BGP message and TCP states associated with the TCP socket
818
Register to receive acknowledgment from standby NE causing the transmission of the BGP message to the peer network device
820
Signal TCP socket to transmit the BGP message to the peer ND
822
Return
826

Figure 8A

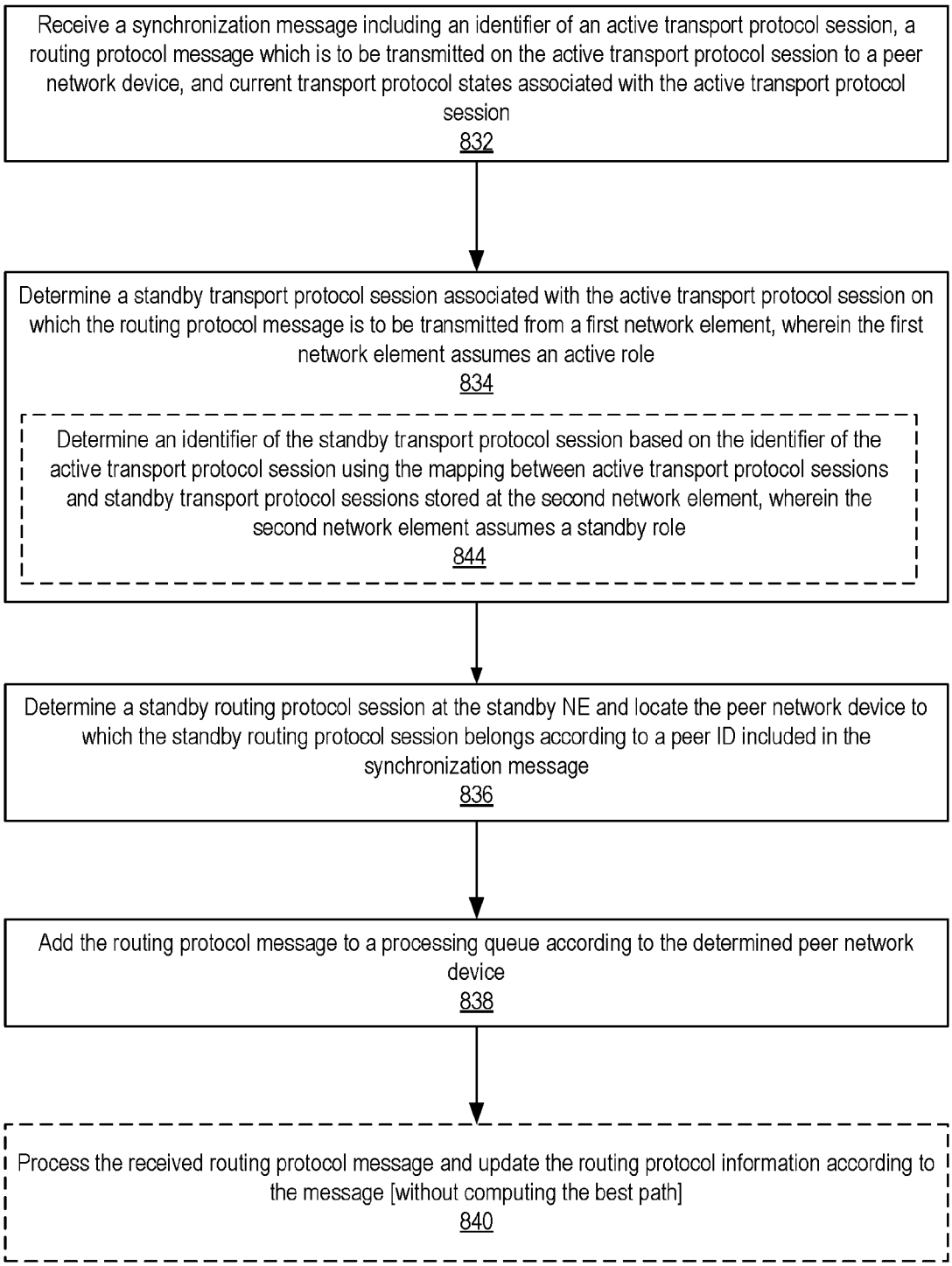

Receive a synchronization message including an identifier of an active transport protocol session, a routing protocol message which is to be transmitted on the active transport protocol session to a peer network device, and current transport protocol states associated with the active transport protocol session
832
Determine a standby transport protocol session associated with the active transport protocol session on which the routing protocol message is to be transmitted from a first network element, wherein the first network element assumes an active role
834
Determine an identifier of the standby transport protocol session based on the identifier of the active transport protocol session using the mapping between active transport protocol sessions and standby transport protocol sessions stored at the second network element, wherein the second network element assumes a standby role
844
Determine a standby routing protocol session at the standby NE and locate the peer network device to which the standby routing protocol session belongs according to a peer ID included in the synchronization message
836
Add the routing protocol message to a processing queue according to the determined peer network device
838
Process the received routing protocol message and update the routing protocol information according to the message [without computing the best path]
840

Figure 8B

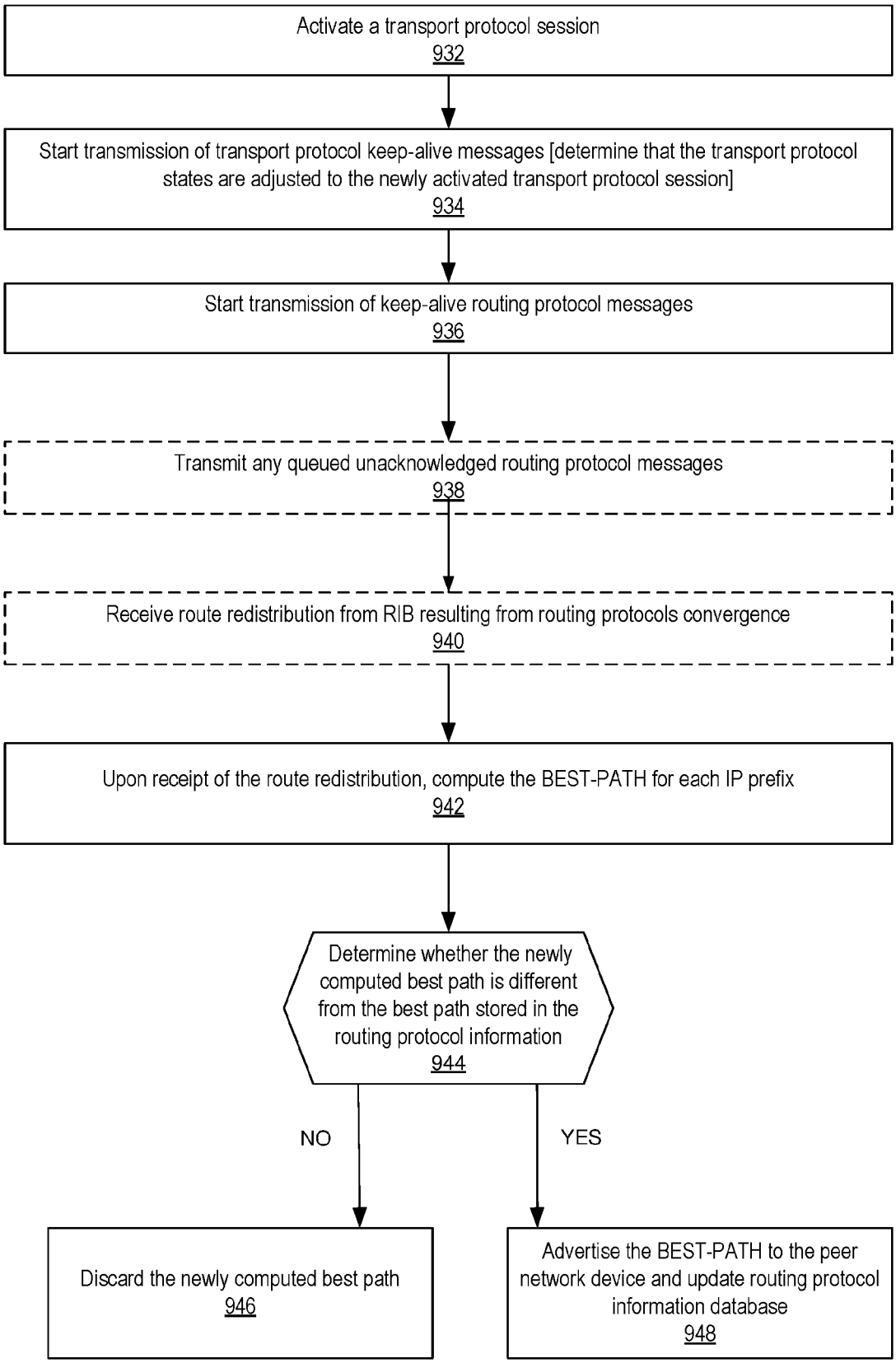

Activate a transport protocol session
932
Start transmission of transport protocol keep-alive messages [determine that the transport protocol states are adjusted to the newly activated transport protocol session]
934
Start transmission of keep-alive routing protocol messages
936
Transmit any queued unacknowledged routing protocol messages
938
Receive route redistribution from RIB resulting from routing protocols convergence
940
Upon receipt of the route redistribution, compute the BEST-PATH for each IP prefix
942
Determine whether the newly computed best path is different from the best path stored in the routing protocol information
944
NO
YES
Discard the newly computed best path
946
Advertise the BEST-PATH to the peer network device and update routing protocol information database
948

Figure 9

Fig. 10A

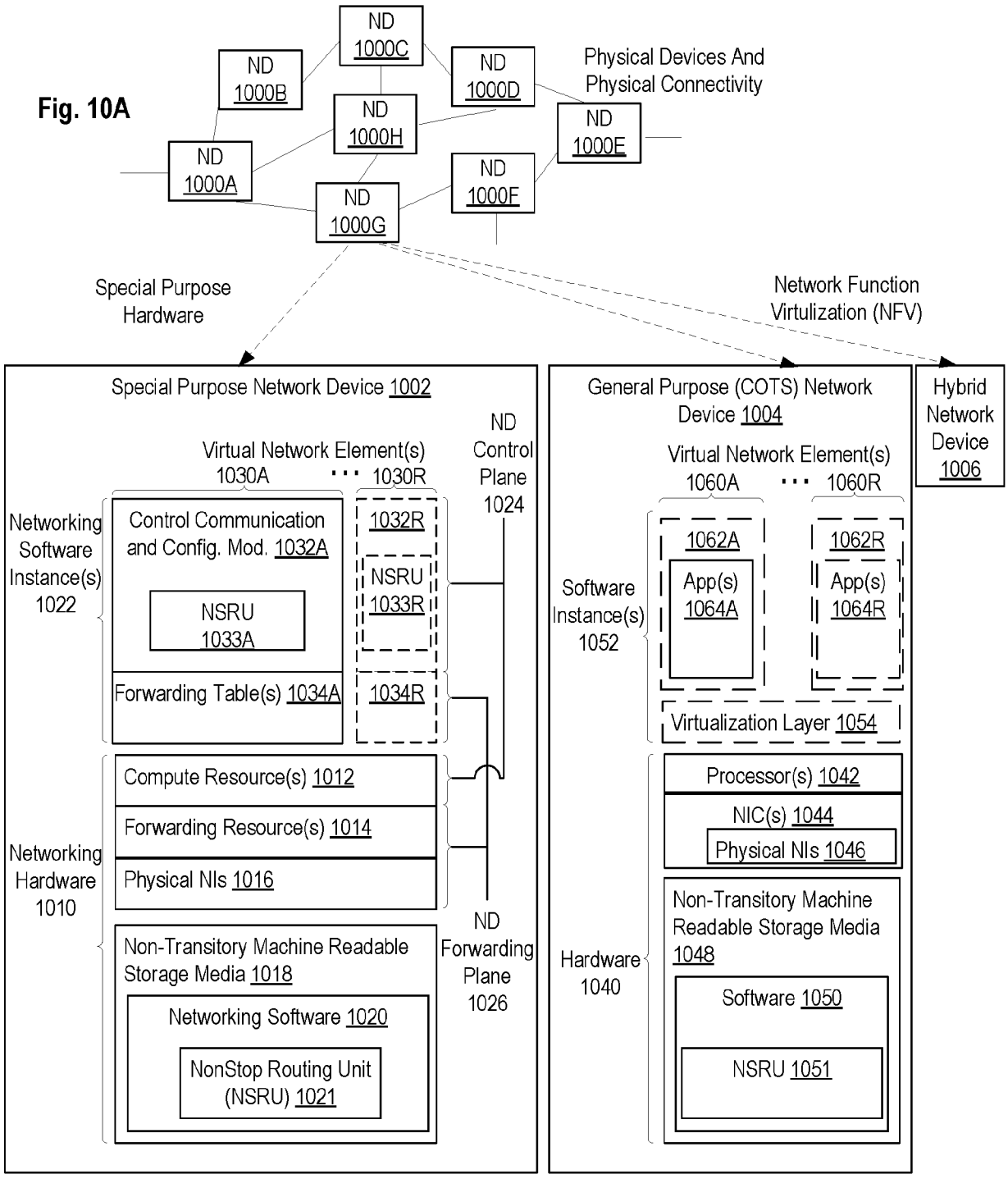


Fig. 10B

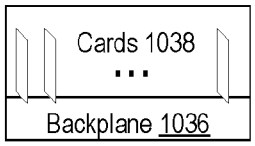


Fig. 10C

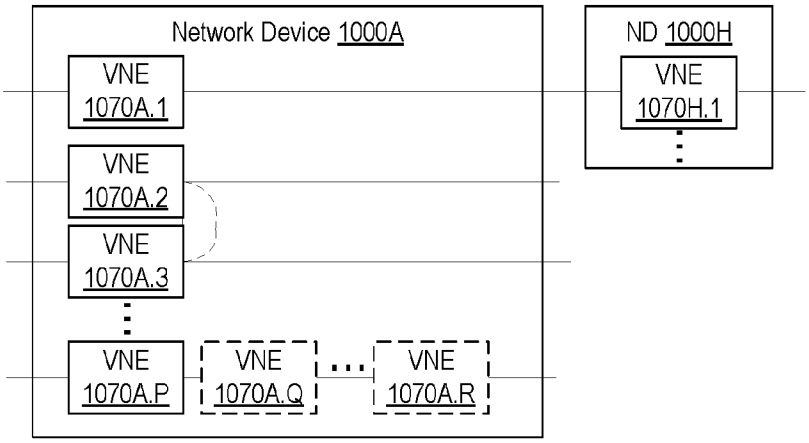


Fig. 10D

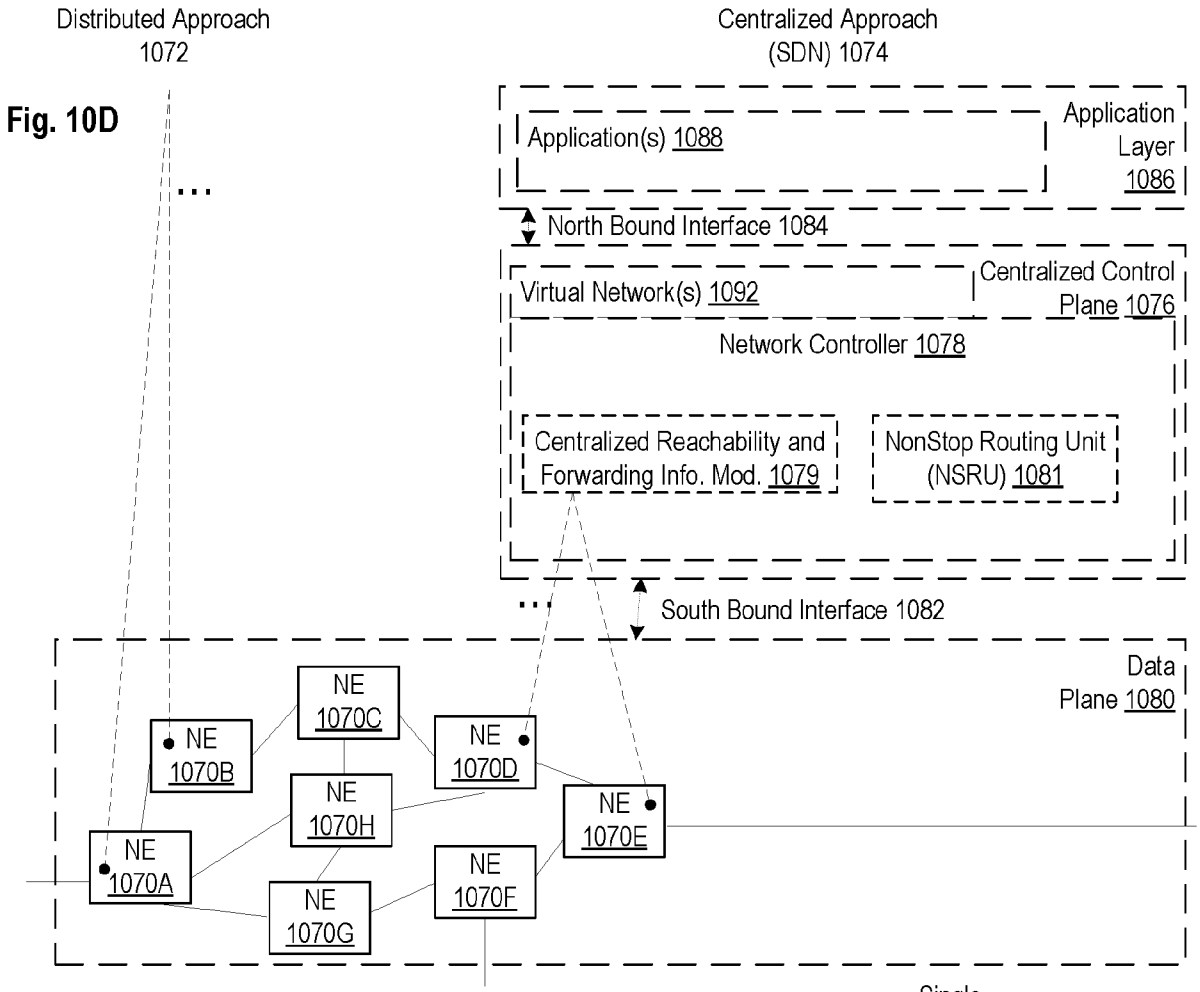


Fig. 10E

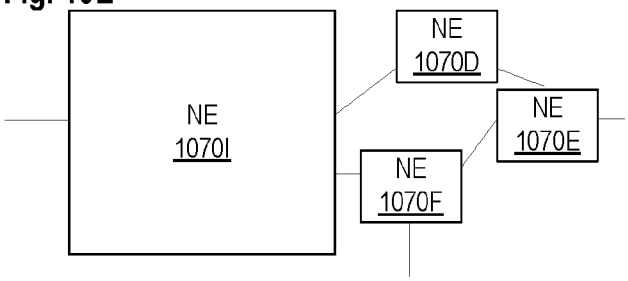


Fig. 10F

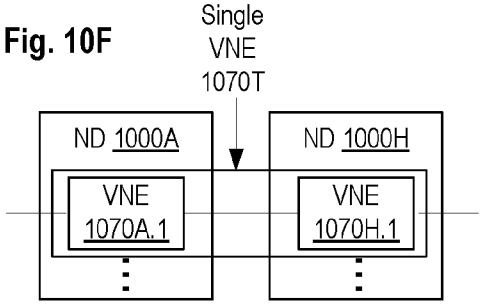
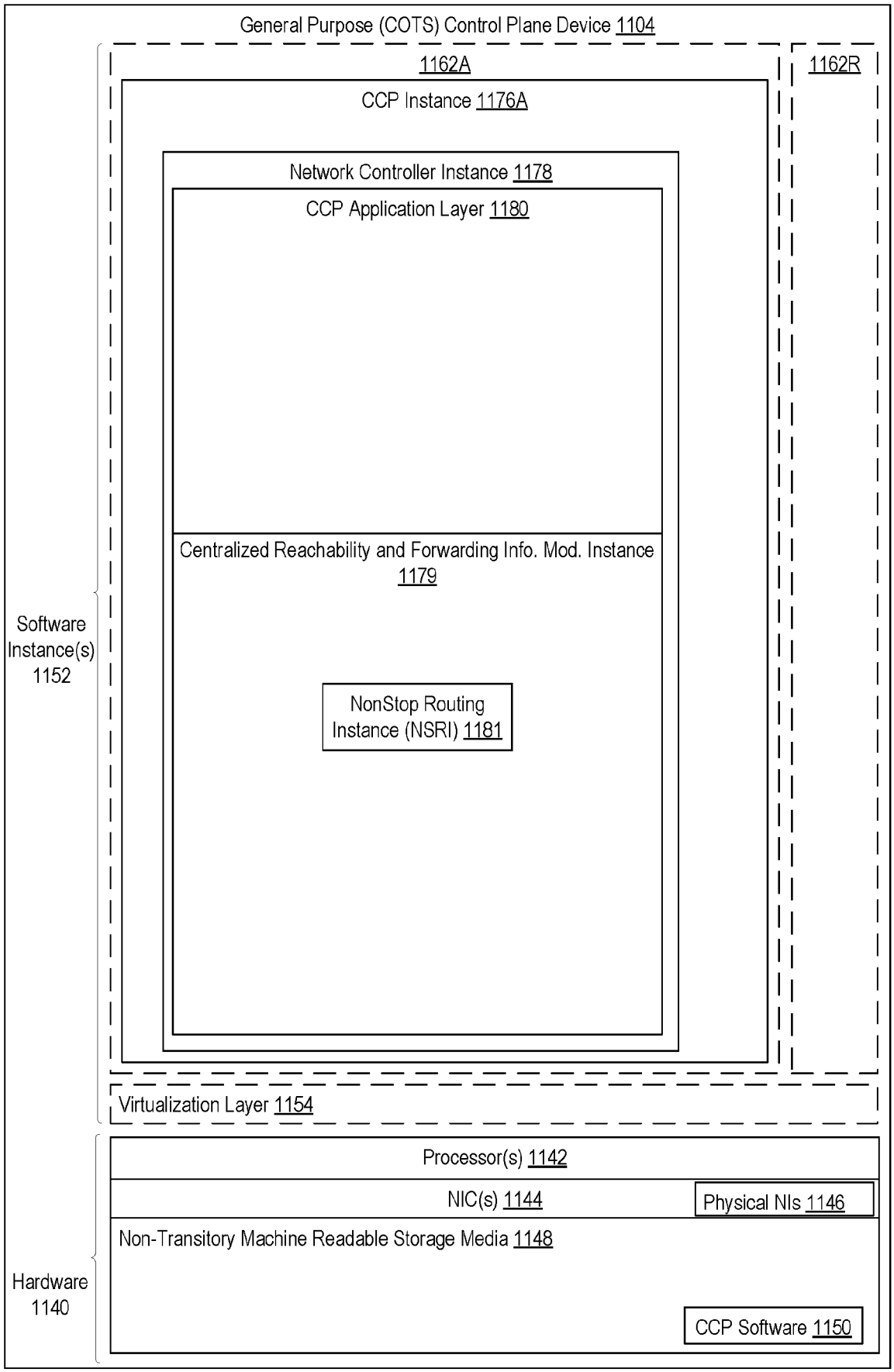


Fig. 11


General Purpose (COTS) Control Plane Device 1104
1162A
1162R
CCP Instance 1176A
Network Controller Instance 1178
CCP Application Layer 1180
Centralized Reachability and Forwarding Info. Mod. Instance 1179
NonStop Routing Instance (NSRI) 1181
Software Instance(s) 1152
Virtualization Layer 1154
Processor(s) 1142
NIC(s) 1144
Physical NIs 1146
Non-Transitory Machine Readable Storage Media 1148
CCP Software 1150
Hardware 1140

INTERNATIONAL SEARCH REPORT

International application No
PCT/IB2016/051951

A. CLASSIFICATION OF SUBJECT MATTER
INV. H04L12/755 H04L12/707
ADD.

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

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
EPO-Internal, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	EP 1 331 769 A1 (ALCATEL CANADA INC [CA]) 30 July 2003 (2003-07-30) paragraphs [0010] - [0048] -----	1-21
A	US 2013/258839 A1 (WANG DAWEI [US] ET AL) 3 October 2013 (2013-10-03) paragraphs [0007] - [0010], [0020] - [0074] -----	1-21
A	US 9 021 459 B1 (QU HUI [US]) 28 April 2015 (2015-04-28) column 4, line 7 - column 15, line 14 -----	1-21

☐ 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	Date of mailing of the international search report
11 November 2016	21/11/2016

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Köppl, Martin

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/IB2016/051951

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 1331769 A1	30-07-2003	DE 60301407 D1	06-10-2005
		DE 60301407 T2	22-06-2006
		EP 1331769 A1	30-07-2003
US 2013258839 A1	03-10-2013	CN 104205748 A	10-12-2014
		EP 2832060 A1	04-02-2015
		US 2013258839 A1	03-10-2013
		WO 2013144746 A1	03-10-2013
US 9021459 B1	28-04-2015	NONE	