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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
31 August 2017 (31.08.2017)

WIPO | PCT

(10) International Publication Number
WO 2017/144947 A1

(51) International Patent Classification:
H04L 12/759 (2013.01) *H04L 12/761* (2013.01)
H04L 12/18 (2006.01) *H04L 12/24* (2006.01)

(21) International Application Number:
PCT/IB2016/050985

(22) International Filing Date:
23 February 2016 (23.02.2016)

(25) Filing Language: English

(26) Publication Language: English

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

(72) Inventor: **ALLAN, David Ian**; 306 S. 15th Street, San Jose, California 95112 (US).

(74) Agent: **DE VOS, Daniel M.**; 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))*

WO 2017/144947 A1

(54) Title: METHOD AND APPARATUS FOR SPANNING TREES FOR COMPUTED SPRING MULTICAST

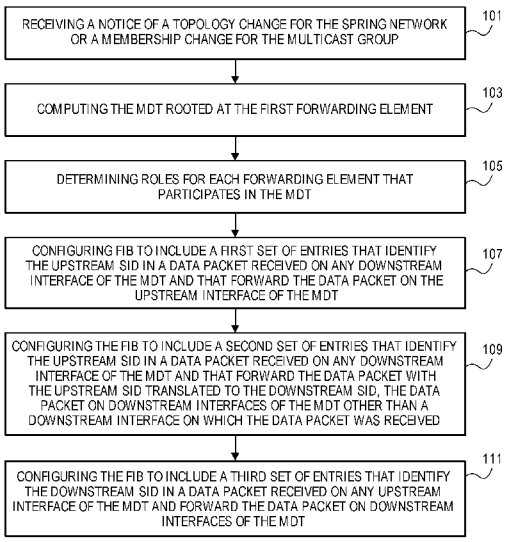


FIG. 1

(57) Abstract: A method is implemented by a network device to compute a multicast distribution tree (MDT) that is a spanning tree in a source packet in routing (SPRING) network for a multicast group, where a first forwarding element has advertised itself as the root of the multicast group and at least one forwarding element has advertised a send and/or receive interest in the multicast group. The result of the spanning tree computation is utilized to update the forwarding information base (FIB) of the network device.

METHOD AND APPARATUS FOR SPANNING TREES FOR COMPUTED SPRING MULTICAST

TECHNICAL FIELD

[0001] Embodiments of the invention relate to the field of multicast distribution tree generation in communication networks; and more specifically, to the implementation of multipoint to multipoint trees in source packet in routing (SPRING) networks.

BACKGROUND

[0002] Numerous techniques and protocols exist for configuring networks to handle multicast traffic. For Internet Protocol (IP) and/or multiprotocol label switching (MPLS) implementations the existing solutions for multicast are based on multicast label distribution protocol (mLDP) or protocol independent multicast (PIM). These are all techniques that depend on a unicast shortest path first (SPF) computation followed by handshaking between peers to sort out a loop free multicast distribution tree (MDT) for each multicast source. These techniques typically depend on a "flat" deployment of multicast state, in that all nodes transited by the traffic of a given MDT will install state for that MDT. At the same time numerous protocols exist that provide for unicast tunneling, and some (such as label based architectures like source packet in routing (SPRING) or MPLS-LDP) implement a full mesh of unicast tunnels as an artifact for normal operation.

[0003] Shortest path bridging (SPB) is a protocol related to computer networking for the configuration of computer networks that enables multipath routing. In one embodiment, the protocol is specified by the Institute of Electrical and Electronics Engineers (IEEE) 802.1aq standard. This protocol replaces prior standards such as spanning tree protocols. SPB enables all paths in the computing network to be active with multiple equal costs paths being utilized through load sharing and similar technologies. The standard enables the implementation of logical Ethernet networks in Ethernet infrastructures using a link state protocol to advertise the topology and logical network memberships of the nodes in the network. SPB implements large scale multicast as part of implementing virtualized broadcast domains. A key distinguishing feature of the SPB standard is that the MDTs are computed from the information in the routing system's link state database via an all-pairs-shortest-path algorithm, which minimizes the amount of control messaging to converge multicast.

[0004] SPRING is an exemplary profile of the use of MPLS technology whereby global identifiers are used in the form of a global label assigned per label switched route (LSR) used for forwarding to that LSR. A full mesh of unicast tunnels is constructed via every node in the

network computing the shortest path to every other node and installing the associated global labels accordingly. In the case of SPRING, this also allows explicit paths to be set up via the application of label stacks at the network ingress. Encompassed with this approach is the concept of a strict (every hop specified) or loose (some waypoints specified) route dependent on how exhaustively the ingress applied label stack specifies the path.

[0005] Proposals have been made to use global identifiers in the dataplane combined with the IEEE 802.1aq technique of advertising multicast registrations in the interior gateway protocol (IGP) and replicating the "all pairs shortest path" approach of IEEE 802.1aq to compute MDTs without the additional handshaking. Such an approach would inherit a lot of desirable properties embodied in the IEEE 802.1aq approach, primarily in the simplification of the amount of control plane exchange required to converge the network. Further proposals have been made to combine the IEEE 802.1aq approach with SPRING tunneling such that multicast distribution tree construction is a hybrid of sparsely deployed multicast state and unicast tunnels.

[0006] Given the above context, a node in the SPRING network could compute its role in implementing any given multicast (S, G) tree. An algorithm that starts with all pairs shortest path computation augmented with algorithms to identify the nodes with specific roles of root, leave or replication point may be employed by each node. Existing unicast tunnels may be used between sources, replication points and leaves of an MDT such that the overall amount of state in the network is minimized. However, the art does not support multipoint to multipoint trees in SPRING networks. In the domain of multipoint to multipoint trees a spanning tree can be considered to be fairly efficient in that the tree is completely bi-directional and employs split horizon forwarding such that traffic received by a node participating in the spanning tree will replicate the traffic on all interfaces except that of origin. The closest analogy in other technologies is core based trees which rely on tunneling traffic to a common root which then mirrors traffic back to all leaves. However, this tunneling is inefficient as all traffic is backhauled to the root, and the sender will receive its own traffic back which it then must discard.

SUMMARY

[0007] In one embodiment, a method is implemented by a network device to compute a multicast distribution tree (MDT) that is a spanning tree in a source packet in routing (SPRING) network for a multicast group, where a first forwarding element has advertised itself as the root of the multicast group and at least one forwarding element has advertised a send and/or receive interest in the multicast group, the result of the spanning tree computation utilized to update the forwarding information base (FIB) of the network device. The method includes receiving a

notice of a topology change for the SPRING network or a membership change for the multicast group, computing the MDT rooted at the first forwarding element, determining roles for each forwarding element that participates in the MDT, configuring the FIB to include a first set of entries that identify an upstream SID in a data packet received on any downstream interface of the MDT and that forward the data packet on the upstream interface of the MDT, configuring the FIB to include a second set of entries that identify the upstream SID in a data packet received on any downstream interface of the MDT and that forward the data packet with the upstream SID translated to the downstream SID, the data packet forwarded on the downstream interfaces of the MDT other than a downstream interface on which the data packet was received; and configuring the FIB to include a third set of entries that identify a downstream SID in a data packet received on any upstream interface of the MDT and that forward the data packet on the downstream interfaces of the MDT.

[0008] In another embodiment, a method is implemented by a network device to compute a multicast distribution tree (MDT) that is a spanning tree in a source packet in routing (SPRING) network for a multicast group, where a first forwarding element has advertised itself as the root of the multicast group and at least one forwarding element has advertised a send and/or receive interest in the multicast group, the result of the spanning tree computation utilized to update the forwarding information base (FIB) of the network device. The method includes receiving a notice of a topology change for the SPRING network or a membership change for the multicast group, computing the MDT rooted at the first forwarding element, determining roles for each forwarding element that participates in the MDT, configuring the FIB to include a first set of entries that identify an upstream SID for each source in the MDT in a data packet and that forward the data packet on the downstream interfaces and upstream interfaces of the MDT other than an interface of the received data packet where a receive interest has been received where the upstream SID is translated to a downstream SID for forwarding on downstream interfaces; and configuring the FIB to include a second set of entries that identify a single downstream SID in a data packet received on an upstream interface of the MDT and that forward the data packet with the single downstream SID on all downstream interfaces of the MDT.

[0009] In one embodiment, a network device computes a multicast distribution tree (MDT) that is a spanning tree in a source packet in routing (SPRING) network for a multicast group, where a first forwarding element has advertised itself as the root of the multicast group and at least one forwarding element has advertised a send and/or receive interest in the multicast group. The result of the spanning tree computation is utilized to update the forwarding information base (FIB) of the network device. The network device includes a non-transitory machine readable medium having stored therein a multicast spanning tree generator and the FIB, and a processor

coupled to the non-transitory machine readable medium. The processor is configured to execute the multicast spanning tree generator. The multicast spanning tree generator is configured to receive a notice of a topology change for the SPRING network or a membership change for the multicast group, to compute the MDT rooted at the first forwarding element, to determine roles for each forwarding element that participates in the MDT, to configure the FIB to include a first set of entries that identify an upstream SID in a data packet received on any downstream interface of the MDT and that forward the data packet on the upstream interface of the MDT, to configure the FIB to include a second set of entries that identify the upstream SID in a data packet received on any downstream interface of the MDT and that forward the data packet with the upstream SID translated to the downstream SID, the data packet forwarded on the downstream interfaces of the MDT other than a downstream interface on which the data packet was received, and configure the FIB to include a third set of entries that identify a downstream SID in a data packet received on any upstream interface of the MDT and that forward the data packet on the downstream interfaces of the MDT.

[0010] In another embodiment, a computing device is in communication with a network device. The computing device executes a plurality of virtual machines for implementing network function virtualization (NFV). The computing device computes a multicast distribution tree (MDT) that is a spanning tree in a source packet in routing (SPRING) network for a multicast group, where a first forwarding element has advertised itself as the root of the multicast group and at least one forwarding element has advertised a send and/or receive interest in the multicast group, the result of the spanning tree computation utilized to update the forwarding information base (FIB) of the network device. The computing device includes a non-transitory machine readable medium having stored therein a multicast spanning tree generator and the FIB, and a processor coupled to the non-transitory machine readable medium. The processor is configured to execute a virtual machine from the plurality of virtual machines. The virtual machine is configured to execute the multicast spanning tree generator. The multicast spanning tree generator is configured to receive a notice of a topology change for the SPRING network or a membership change for the multicast group, to compute the MDT rooted at the first forwarding element, to determine roles for each forwarding element that participates in the MDT, to configure the FIB to include a first set of entries that identify an upstream SID in a data packet received on any downstream interface of the MDT and that forward the data packet on the upstream interface of the MDT, to configure the FIB to include a second set of entries that identify the upstream SID in a data packet received on any downstream interface of the MDT and that forward the data packet with the upstream SID translated to the downstream SID, the data packet forwarded on the downstream interfaces of the MDT other than a downstream

interface on which the data packet was received, and configure the FIB to include a third set of entries that identify a downstream SID in a data packet received on any upstream interface of the MDT and that forward the data packet on the downstream interfaces of the MDT.

[0011] In a further embodiment, a control plane device is configured to implement a control plane of a software defined networking (SDN) network including a network device in a network with a plurality of network devices, wherein the control plane device is configured to configure the network device to compute a multicast distribution tree (MDT) that is a spanning tree in a source packet in routing (SPRING) network for a multicast group, where a first forwarding element has advertised itself as the root of the multicast group and at least one forwarding element has advertised a send and/or receive interest in the multicast group. The result of the spanning tree computation is utilized to update the forwarding information base (FIB) of the network device. The control plane device includes a non-transitory machine readable medium having stored therein a multicast spanning tree generator and the FIB, and a processor coupled to the non-transitory machine readable medium. The processor is configured to execute the multicast spanning tree generator. The multicast spanning tree generator is configured to receive a notice of a topology change for the SPRING network or a membership change for the multicast group, to compute the MDT rooted at the first forwarding element, to determine roles for each forwarding element that participates in the MDT, to configure the FIB to include a first set of entries that identify an upstream SID in a data packet received on any downstream interface of the MDT and that forward the data packet on the upstream interface of the MDT, to configure the FIB to include a second set of entries that identify the upstream SID in a data packet received on any downstream interface of the MDT and that forward the data packet with the upstream SID translated to the downstream SID, the data packet forwarded on the downstream interfaces of the MDT other than a downstream interface on which the data packet was received, and configure the FIB to include a third set of entries that identify a downstream SID in a data packet received on any upstream interface of the MDT and that forward the data packet on the downstream interfaces of the MDT.

BRIEF DESCRIPTION OF THE DRAWINGS

[0012] 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:

[0013] Figure 1 is a flowchart of one embodiment of process for generating spanning trees for a SPRING network.

[0014] Figure 2 is a flowchart of one embodiment of another process for generating spanning trees for a SPRING network.

[0015] Figure 3A is a diagram of examples of spanning trees in protocol independent multicast (PIM).

[0016] Figure 3B is a diagram of an example of a generic spanning tree.

[0017] Figure 4 is a diagram of one example implementation of spanning trees and forwarding information base (FIB) management in a SPRING network.

[0018] Figure 5 is a diagram of another example implementation of spanning trees and forwarding information base (FIB) management in a SPRING network.

[0019] Figure 6A 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.

[0020] Figure 6B illustrates an exemplary way to implement a special-purpose network device according to some embodiments of the invention.

[0021] Figure 6C illustrates various exemplary ways in which virtual network elements (VNEs) may be coupled according to some embodiments of the invention.

[0022] Figure 6D 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.

[0023] Figure 6E 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.

[0024] Figure 6F 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.

[0025] Figure 7 illustrates a general purpose control plane device with centralized control plane (CCP) software 750), according to some embodiments of the invention.

DETAILED DESCRIPTION

[0026] The following description describes methods and apparatus for implementing spanning trees and in particular multipoint to multipoint core rooted spanning trees. The embodiments

provide a process for computing minimum cost trees constructed from the root to the leaves that are bi-directional. Nodes are configured such that every replication point on the upstream path to the root and the root itself performs split-horizon replication. The nodes are configured to replicate multicast traffic on all logical and physical interfaces of the MDT with the exception of the interface of receipt. This requires a tree construction algorithm that avoids packet duplication on any physical or logical link, for which prior art exists. The nodes are configured such that the root employs the same forwarding topology as a per source specific tree but constructed so that the connectivity is bi-directional, and employs split horizon forwarding at every replication point. Split horizon in this context, refers to the technique of not sending traffic on the interface of receipt.

[0027] 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.

[0028] 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.

[0029] 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.

[0030] 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.

[0031] 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.

[0032] 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).

[0033] 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.

[0034] 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.

[0035] SPRING Networks

[0036] The embodiments provide a method of construction of spanning trees in a network that uses both interior gateway protocol (IGP) driven computation, and unicast tunnels in the construction of MDTs. The embodiments utilize the computations of multicast distribution trees (MDTs) and the exemplary information available in shortest path bridging (SPB) implementations such as IEEE 802.1aq adapted to other technologies. In IEEE 802.1aq multicast registrations are advertised in the interior gateway protocol (IGP), thus all nodes in the network have multicast group membership information about the other nodes in the network and are explicitly delegated with the task of determining their role in each MDT on the basis of information in the IGP.

[0037] IEEE 802.1aq performs an all-pairs shortest path computation that determines a path from all nodes in a network to all other nodes in the network that selects a shortest path to each node from a source node. When combined with the tie breaking algorithms specified in IEEE 802.1aq the result is an acyclic shortest path tree. Multicast distribution trees can also be computed in a similar manner as they can be derived from the shortest path trees using the notion of reverse path forwarding. With the use of multicast protocol label switching (MPLS) a global multicast label is assigned by the management system and used on a per (S, G) multicast distribution tree basis. The (S, G) notation indicates an S – source and G – multicast group

relation where the multicast tree has a single source and a group of listeners in a network. The multicast labels in the network are carried end to end (E2E). This is inherent to the operation of SPRING. A multicast implementation MPLS could also be envisioned that combined IGP registrations and an LDP signaled unicast tunnel mesh could also be adapted to carry the labels E2E.

[0038] The example embodiments utilize SPRING for unicast tunneling. As a consequence of SPRING operation, upon nodal convergence to align with the view of the network available via the IGP, the local forwarding information base (LFIB) of each network device in the network will have at least one unicast SPRING-label switched route to each other LSR. It is not necessary, but assumed that the network will also utilize penultimate-hop popping (PHP) on SPRING based LSPs where the outermost label is removed before being forwarded to the last hop to a destination.

[0039] After a shortest path first (SPF) tree (i.e., an (S, *) tree where S indicates the node is the source and * indicates the tree reaches all nodes) is computed for a given node that node can then determine whether it is a source, a leaf or a replication node for each possible (S, G) MDT. If the node has one of these three roles in a given MDT it will then install the appropriate state for each, whereas if the node does not participate in the MDT in any of the three roles then no state needs to be installed for that MDT. The installed state where the node is a root or a replicating node utilizes established *a priori* unicast tunnels to deliver multicast packets to downstream non-adjacent leaves or replicating nodes. Tunnels do not need to be established for downstream immediately adjacent nodes that have a role in the MDT as they will have installed state for the MDT. Knowledge of the role of each node in the network in relation to a given MDT is an artifact of the all-pairs shortest path computation.

[0040] Overview

[0041] The example embodiments utilize SPRING for constructing spanning trees rooted on a designated node in the network. Spanning trees trade bandwidth efficiency and latency against state and in the case of the computed approach, convergence time as the computation of one tree is sufficient for all sources in a multicast group to send traffic to all receivers, therefore for classes of multicast service where the tradeoff can be made, the use of spanning trees can be employed.

[0042] Bi-directional trees, such as protocol independent multicast (PIM) bi-directional trees, multicast label distribution protocol (mLDP) multipoint to multipoint trees and similar technology have inherent inefficiency in that all packets need to be tunneled to the rendezvous point which is also the MDT root before replication to the leaves begins. While the process is a standard paradigm whereby packets are simply tunneled to the rendezvous point, a further

inefficiency is that any sender of data will receive and have to discard a copy of the any data it sends to the rendezvous point. In a general a spanning tree is also a rooted tree, but replication occurs at every branch point in the spanning tree reducing both the bandwidth required and overall average latency of the tree.

[0043] The embodiments can employ efficient spanning tree construction techniques and minimal state techniques. These techniques can ensure that any tree constructed will only have one logical path to a leaf or set of leaves reachable by any individual physical or logical interface. Where there is more than one source downstream on a given interface, by the nature of spanning tree construction, all nodes will already have received a copy of any multicast packet for the group sent by any source whose connectivity in the spanning tree joins the current path downstream of the interface of arrival therefore there is no need to send any traffic on the physical or logical interface of arrival to ensure complete distribution of the traffic to all members of the multicast group served by the spanning tree.

[0044] The embodiments encompass at least two implementations for spanning trees. In one embodiment, an upstream multicast segment identifier (SID) and a downstream multicast SID are utilized. This separation reduces the vulnerability of transient loops. The technique also utilizes a per interface incoming label map (ILM) to facilitate split horizon forwarding in the context of the defined MPLS architecture. In the second embodiment, source specific SIDs and a single downstream SID are utilized. This embodiment also reduces the vulnerability of transient loops. The embodiment works when the MPLS implementation in the network is restricted to using a common per platform ILM. The per-interface ILM has advantages as it reduces the overall amount of state in the system. FIBs may otherwise get larger at nodes that are closer to the root of the spanning tree. However, the per-interface ILM may not be applicable to all scenarios.

[0045] In the embodiments, there are two types of relevant forwarding information base (FIB) entries, upstream SID and downstream SID entries. These entries define the next hop on a per upstream SID or downstream SID basis. These entries are mapped to one of four types of outgoing FIB entries. First, an upstream physical FIB entry defines the next upstream node in the MDT that is physically adjacent to the node of the FIB. Second, an upstream logical entry defines the next upstream node in the MDT that is not physically adjacent and will be reached via a tunnel which may egress the node via one of a set of one or more physical interfaces and may require additional processing (e.g. ECMP) to resolve to a physical interface. Third, a downstream physical entry that defines the next downstream node in the MDT that is physically adjacent to the node with the FIB. Fourth, the downstream logical entry which defines the next

downstream node in the MDT that is not physically adjacent and will be reached via a tunnel which may egress the node via one of a set of one or more physical interfaces.

[0046] In general, the elements of the embodiment using separate upstream and downstream SIDs involves configuration of nodes and interior gateway (IGP) advertisement. Note that the tree construction style (per interface or per platform ILM) may be communicated via the IGP, or may be a network wide configured attribute. A given node is configured to advertise itself via the IGP as the root for an MDT of a type 'spanning tree' for a multicast group G, with an upstream SID 'u' and a downstream SID 'd.' This is the root node for the spanning tree type MDT. Other nodes advertise themselves as senders and/or receivers of the multicast group G via the IGP. The advertisements may use a null SID value for consistency in encoding structure with existing mechanisms. Once these advertisements have been made, the nodes in the SPRING network may compute the spanning tree as if it were a (S,G) MDT with the node designated as the spanning tree root as the source, and determine their role in the tree for the multicast group G. A node participating in the topology of the spanning tree will have zero (if the root) or one upstream interface and zero (non-bud leaf, where a bud node is one that is both a leaf and a replication point) or more than one downstream interface, where both upstream and downstream interfaces can be physical or logical.

[0047] Figure 1 is a diagram of one embodiment of a process for configuring an FIB and the computation of the nodes for the upstream and downstream SID embodiment where per-interface ILMs are deployed in the network. The process involves a set of computing elements and forwarding elements. The computing elements are responsible for the implementation of the process and configuring of the forwarding elements. The computing elements may be implemented in the same network device as the forwarding element or separately implemented. The forwarding elements are the processes that implement the forwarding of data traffic in the SPRING network based on their FIBs. The 'nodes' of a network are synonymous with the forwarding elements.

[0048] In response to each forwarding element receiving a notice of topology or membership change for a SPRING network (Block 101), the process begins to compute a multicast distribution tree rooted at a first designated root forwarding element (Block 103). The computation generates a spanning tree rooted on the root forwarding element. The process determines a role for each forwarding element that participates in the MDT (i.e., a leaf, replicating node or root) (Block 105). The process then configures the FIB to implement a set of mappings. The FIB may be configured to include a first set of entries that identify the upstream SID in data packets received on any downstream interface (logical or physical) of the MDT and that forward the identified data packets on the upstream interface of the MDT (Block 107). The

FIB may also be configured to include a second set of entries that identify the upstream SID in data packets received on any downstream interface (logical or physical) of the MDT and that forward the identified data packets on each downstream interface (logical or physical) exclusive of the interface of arrival of the MDT (Block 109). In addition, the upstream SID of the received data packets is translated to the downstream SID before forwarding. Further, the FIB may be configured to include a third set of entries that identify the downstream SID in data packets received on any upstream interface (logical or physical) of the MDT and that forward the identified data packets on each downstream interface of the MDT (Block 111).

[0049] In general, the elements of the embodiment using source specific and downstream SIDs involves configuration of nodes and interior gateway (IGP) advertisement. A given node is configured to advertise itself via the IGP as the root for an MDT of a type 'spanning tree' for a multicast group G, with a downstream SID 'r.' This is the root node for the spanning tree type MDT. Other nodes advertise themselves as senders and/or receivers of the multicast group g via IGP. The advertisements may use separate upstream SID values for each node "node/g" and a common downstream SID value used for forwarding all traffic in the direction of away from the root. Once these advertisements have been made, the nodes in the SPRING network may compute the spanning tree and their role in the tree for the multicast group g.

[0050] Figure 2 is a diagram of one embodiment of a process for configuring an FIB and the computation of the nodes for the source specific and downstream SID embodiment. In response to each forwarding element receiving a notice of topology or membership change for a SPRING network (Block 201), the process begins to compute a multicast distribution tree rooted at a first designated root forwarding element (Block 203). The computation generates a spanning tree rooted on the root forwarding element that reaches all receivers in the served group. The process determines a role for each forwarding element that participates in the MDT (i.e., a leaf, replicating node or root) (Block 205). The process then configures the FIB to include a first set of entries that identify an upstream SID for each downstream source in the MDT in a data packet and that forward the data packet on the downstream interfaces in the MDT using the downstream SID and upstream interfaces (logical or physical) of the MDT using the source specific upstream SID other than an interface of the received data packet where a receive interest has been received (Block 207). Before forwarding, the upstream SID is translated to a downstream SID for forwarding on downstream interfaces, but is not translated on downstream interfaces. Then the process configures the FIB to include a second set of entries that identify the single downstream SID for the MDT in a data packet and that forward the data packet with the single downstream SID on each downstream interfaces (logical and physical) of the MDT (Block 209).

[0051] Figure 3A is a diagram of a core based tree implemented in protocol independent multicast (PIM) that is bi-directional (BI-DIR). In the basic core based tree shown here a tunnel exists from a source S to the root. Upstream and downstream refer to the direction to or direction from the root, respectively. Replication is performed on downstream interfaces in the MDT.

[0052] Figure 3B is a diagram of a further core based tree implemented in a PIM BI-DIR. The illustrated process defines a set of replication point such that upstream packet are replicated on all but interfaces of arrival at MDT replication points. Downstream packets are replicated on all downstream interfaces.

[0053] Figure 4 is a diagram of an example of the process for the upstream and downstream SID embodiment. In this example, a network topology is show, where node (i.e., forwarding element) A and D are leaves in the MDT. Node E is a replication point and for this example the computing (i.e., computing element) as well. Nodes B and C have no role in this MDT. For multicast group G1, the upstream identifier is "U" and the downstream SID is "D." The link between node E and node A is virtual and represented as a downstream logical interface. The FIB maintained by this node includes the illustrated entries. These are the FIB entries for example node E.

[0054] The first entry is for physical interface 1 which is a link to node B. The upstream SID U is associated with this entry and the replication instructions to replicate traffic from interface 1 to interface 3 and 4. The second entry is associated with the upstream SID U and the replication instructions to replicate traffic from interface 2 to interfaces 3 and 4. The third entry is association with the upstream SID U and the replication instructions indicate to replicate received traffic to interfaces 1 or 2 (based on ECMP) and interface 4. The fourth entry is associated with downstream SID d with the replication instructions to replicate received traffic on interface 4 to interfaces 1 or 2 (based on ECMP) and interface 3.

[0055] Figure 5 is a diagram of an example of the process for the source and downstream SID embodiment. In this example, a network topology is shown, where node (i.e., forwarding element) A and D are leaves in the MDT. Node E is a replication point and for this example the computing node (i.e., computing element) as well. Nodes B and C have no role in this MDT. For multicast group G1, the upstream identifiers are "a" and "d" and the downstream SID is "r." The link between node E and node A is virtual and represented as a downstream logical interface. The FIB maintained by this node E includes the illustrated entries. These are the FIB entries for example node E.

[0056] The first entry is for any interface. The upstream SID "a" is associated with this entry and the replication instructions to replicate traffic from any interface to interface 3 and 4. The

second entry is associated with the upstream SID "d" and the replication instructions to replicate traffic from any interface to interfaces 1 or 2 (based on ECMP) and 4. The third entry is association with the downstream SID "r" and the replication instructions indicate to replicate received traffic to interfaces 1 or 2 (based on ECMP) and interface 3.

[0057] The advantages of the embodiments include that each node can independently determine its role in the spanning tree and generate forwarding entries for its FIB to implement split horizon forwarding. One skilled in the art would understand that the refinement of the rules for logical interfaces is provided for explicitly to address the case of spanning trees constructed with unicast tunnels as a constituent component. It is possible to extend the spanning tree construction technique to include "flat trees" via collapsing the concept of logical interface to only refer to physical interfaces.

[0058] Architecture

[0059] Figure 6A 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 6A shows NDs 600A-H, and their connectivity by way of lines between 600A-600B, 600B-600C, 600C-600D, 600D-600E, 600E-600F, 600F-600G, and 600A-600G, as well as between 600H and each of 600A, 600C, 600D, and 600G. 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 600A, 600E, and 600F 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).

[0060] Two of the exemplary ND implementations in Figure 6A are: 1) a special-purpose network device 602 that uses custom application–specific integrated–circuits (ASICs) and a special-purpose operating system (OS); and 2) a general purpose network device 604 that uses common off-the-shelf (COTS) processors and a standard OS.

[0061] The special-purpose network device 602 includes networking hardware 610 comprising compute resource(s) 612 (which typically include a set of one or more processors), forwarding resource(s) 614 (which typically include one or more ASICs and/or network processors), and physical network interfaces (NIs) 616 (sometimes called physical ports), as well as non-transitory machine readable storage media 618 having stored therein networking software 620. 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 600A-H. During operation, the networking software 620 may be executed by the networking hardware 610 to instantiate a set of one or more networking

software instance(s) 622. Each of the networking software instance(s) 622, and that part of the networking hardware 610 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) 622), form a separate virtual network element 630A-R. Each of the virtual network element(s) (VNEs) 630A-R includes a control communication and configuration module 632A-R (sometimes referred to as a local control module or control communication module) and forwarding table(s) 634A-R, such that a given virtual network element (e.g., 630A) includes the control communication and configuration module (e.g., 632A), a set of one or more forwarding table(s) (e.g., 634A), and that portion of the networking hardware 610 that executes the virtual network element (e.g., 630A).

[0062] The special-purpose network device 601 can implement a multicast spanning tree generator 664. The multicast spanning tree generator 664 can be a program or similar set of instructions that implement the methods described herein above in reference to Figures 1-5 that provide spanning tree generation in a SPRING network. The multicast spanning tree generator 664 can be stored by the non-transitory machine readable storage media 618 and executed by the compute resources 612.

[0063] The special-purpose network device 602 is often physically and/or logically considered to include: 1) a ND control plane 624 (sometimes referred to as a control plane) comprising the compute resource(s) 612 that execute the control communication and configuration module(s) 632A-R; and 2) a ND forwarding plane 626 (sometimes referred to as a forwarding plane, a data plane, or a media plane) comprising the forwarding resource(s) 614 that utilize the forwarding table(s) 634A-R and the physical NIs 616. By way of example, where the ND is a router (or is implementing routing functionality), the ND control plane 624 (the compute resource(s) 612 executing the control communication and configuration module(s) 632A-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) 634A-R, and the ND forwarding plane 626 is responsible for receiving that data on the physical NIs 616 and forwarding that data out the appropriate ones of the physical NIs 616 based on the forwarding table(s) 634A-R.

[0064] Figure 6B illustrates an exemplary way to implement the special-purpose network device 602 according to some embodiments of the invention. Figure 6B shows a special-purpose network device including cards 638 (typically hot pluggable). While in some embodiments the cards 638 are of two types (one or more that operate as the ND forwarding plane 626 (sometimes called line cards), and one or more that operate to implement the ND

control plane 624 (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 636 (e.g., a first full mesh coupling the line cards and a second full mesh coupling all of the cards).

[0065] Returning to Figure 6A, the general purpose network device 604 includes hardware 640 comprising a set of one or more processor(s) 642 (which are often COTS processors) and network interface controller(s) 644 (NICs; also known as network interface cards) (which include physical NIs 646), as well as non-transitory machine readable storage media 648 having stored therein software 650. During operation, the processor(s) 642 execute the software 650 to instantiate one or more sets of one or more applications 664A-R. 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 654 represents the kernel of an operating system (or a shim executing on a base operating system) that allows for the creation of multiple instances 662A-R called software containers that may each be used to execute one (or more) of the sets of applications 664A-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 654 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 664A-R is run on top of a guest operating system within an instance 662A-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 640, 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 654, unikernels running within software containers represented by instances 662A-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).

[0066] The instantiation of the one or more sets of one or more applications 664A-R, as well as virtualization if implemented, are collectively referred to as software instance(s) 652. Each set of applications 664A-R, corresponding virtualization construct (e.g., instance 662A-R) if implemented, and that part of the hardware 640 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) 660A-R.

[0067] The virtual network element(s) 660A-R perform similar functionality to the virtual network element(s) 630A-R - e.g., similar to the control communication and configuration module(s) 632A and forwarding table(s) 634A (this virtualization of the hardware 640 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 662A-R corresponding to one VNE 660A-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 662A-R to VNEs also apply to embodiments where such a finer level of granularity and/or unikernels are used.

[0068] In certain embodiments, the virtualization layer 654 includes a virtual switch that provides similar forwarding services as a physical Ethernet switch. Specifically, this virtual switch forwards traffic between instances 662A-R and the NIC(s) 644, as well as optionally between the instances 662A-R; in addition, this virtual switch may enforce network isolation

between the VNEs 660A-R that by policy are not permitted to communicate with each other (e.g., by honoring virtual local area networks (VLANs)).

[0069] The general purpose network device 604 can implement a multicast spanning tree generator 664A-R. The multicast spanning tree generator 664A-R can be a program or similar set of instructions that implement the methods described herein above in reference to Figures 1-5 that provide for the generation of spanning trees for use in multicast distribution of frames in a SPRING network. The multicast spanning tree generator 664A-R can be stored by the non-transitory machine readable storage media 648 and executed by the software instances 652 and processors 642.

[0070] The third exemplary ND implementation in Figure 6A is a hybrid network device 606, 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 602) could provide for para-virtualization to the networking hardware present in the hybrid network device 606.

[0071] 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) 630A-R, VNEs 660A-R, and those in the hybrid network device 606) receives data on the physical NIs (e.g., 616, 646) and forwards that data out the appropriate ones of the physical NIs (e.g., 616, 646). 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 (DSCP) values.

[0072] Figure 6C illustrates various exemplary ways in which VNEs may be coupled according to some embodiments of the invention. Figure 6C shows VNEs 670A.1-670A.P (and optionally VNEs 670A.Q-670A.R) implemented in ND 600A and VNE 670H.1 in ND 600H. In Figure 6C, VNEs 670A.1-P are separate from each other in the sense that they can receive packets from outside ND 600A and forward packets outside of ND 600A; VNE 670A.1 is coupled with VNE 670H.1, and thus they communicate packets between their respective NDs; VNE 670A.2-670A.3 may optionally forward packets between themselves without forwarding

them outside of the ND 600A; and VNE 670A.P may optionally be the first in a chain of VNEs that includes VNE 670A.Q followed by VNE 670A.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 6C 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).

[0073] The NDs of Figure 6A, 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 6A may also host one or more such servers (e.g., in the case of the general purpose network device 604, one or more of the software instances 662A-R may operate as servers; the same would be true for the hybrid network device 606; in the case of the special-purpose network device 602, one or more such servers could also be run on a virtualization layer executed by the compute resource(s) 612); in which case the servers are said to be co-located with the VNEs of that ND.

[0074] A virtual network is a logical abstraction of a physical network (such as that in Figure 6A) 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).

[0075] 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).

[0076] 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).

[0077] Fig. 6D illustrates a network with a single network element on each of the NDs of Figure 6A, 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 6D illustrates network elements (NEs) 670A-H with the same connectivity as the NDs 600A-H of Figure 6A.

[0078] Figure 6D illustrates that the distributed approach 672 distributes responsibility for generating the reachability and forwarding information across the NEs 670A-H; in other words, the process of neighbor discovery and topology discovery is distributed.

[0079] For example, where the special-purpose network device 602 is used, the control communication and configuration module(s) 632A-R of the ND control plane 624 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 670A-H (e.g., the compute resource(s) 612 executing the control communication and configuration module(s) 632A-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 624. The ND control plane 624 programs the ND forwarding plane 626 with information (e.g., adjacency and route information) based on the routing structure(s). For example, the ND control plane 624 programs the adjacency and route information into one or more forwarding table(s) 634A-R (e.g., Forwarding Information Base (FIB), Label Forwarding Information Base (LFIB), and one or more adjacency structures) on the ND forwarding plane 626. 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 602, the same distributed approach 672 can be implemented on the general purpose network device 604 and the hybrid network device 606.

[0080] Figure 6D illustrates that a centralized approach 674 (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 674 has the responsibility for the generation of reachability and forwarding information in a centralized control plane 676 (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 676 has a south bound interface 682 with a data plane 680 (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 670A-H (sometimes referred to as switches, forwarding elements, data plane elements, or nodes). The centralized control plane 676 includes a network controller 678, which includes a centralized reachability and forwarding information module 679 that determines the reachability within the network and distributes the forwarding information to the NEs 670A-H of the data plane 680 over the south bound interface 682 (which may use the OpenFlow protocol). Thus, the network intelligence is centralized in the centralized control plane 676 executing on electronic devices that are typically separate from the NDs.

[0081] For example, where the special-purpose network device 602 is used in the data plane 680, each of the control communication and configuration module(s) 632A-R of the ND control plane 624 typically include a control agent that provides the VNE side of the south bound interface 682. In this case, the ND control plane 624 (the compute resource(s) 612 executing the control communication and configuration module(s) 632A-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 676 to receive the forwarding information (and in some cases, the reachability information) from the centralized reachability and forwarding information module 679 (it should be understood that in some embodiments of the invention, the control communication and configuration module(s) 632A-R, in addition to communicating with the centralized control plane 676, 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 674, but may also be considered a hybrid approach).

[0082] While the above example uses the special-purpose network device 602, the same centralized approach 674 can be implemented with the general purpose network device 604 (e.g., each of the VNE 660A-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 676 to receive the forwarding information (and in some cases, the reachability information) from the centralized reachability and forwarding information module 679; it should be understood that in some embodiments of the invention, the VNEs 660A-R, in addition to communicating with the centralized control plane 676, 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 606. In fact, the use of SDN techniques can enhance the NFV techniques typically used in the general purpose network device 604 or hybrid network device 606 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.

[0083] Figure 6D also shows that the centralized control plane 676 has a north bound interface 684 to an application layer 686, in which resides application(s) 688. The centralized control plane 676 has the ability to form virtual networks 692 (sometimes referred to as a logical forwarding plane, network services, or overlay networks (with the NEs 670A-H of the data plane 680 being the underlay network)) for the application(s) 688. Thus, the centralized control plane 676 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).

[0084] While Figure 6D shows the distributed approach 672 separate from the centralized approach 674, 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) 674, 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 674, but may also be considered a hybrid approach.

[0085] While Figure 6D illustrates the simple case where each of the NDs 600A-H implements a single NE 670A-H, it should be understood that the network control approaches described with reference to Figure 6D also work for networks where one or more of the NDs 600A-H implement multiple VNEs (e.g., VNEs 630A-R, VNEs 660A-R, those in the hybrid network device 606). Alternatively or in addition, the network controller 678 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 678 may present the implementation of a VNE/NE in a single ND as multiple VNEs in the virtual networks 692 (all in the same one of the virtual network(s) 692, each in different ones of the virtual network(s) 692, or some combination). For example, the network controller 678 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 676 to present different VNEs in the virtual network(s) 692 (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).

[0086] The centralized control plane 676 can implement a spanning tree generator 681. The spanning tree generator 681 can be a program or similar set of instructions that implement the methods described herein above in reference to Figures 1-5 that provide for the generation of spanning trees for distribution of multicast frames in a SPRING network.

[0087] On the other hand, Figures 6E and 6F respectively illustrate exemplary abstractions of NEs and VNEs that the network controller 678 may present as part of different ones of the virtual networks 692. Figure 6E illustrates the simple case of where each of the NDs 600A-H implements a single NE 670A-H (see Figure 6D), but the centralized control plane 676 has abstracted multiple of the NEs in different NDs (the NEs 670A-C and G-H) into (to represent) a single NE 670I in one of the virtual network(s) 692 of Figure 6D, according to some embodiments of the invention. Figure 6E shows that in this virtual network, the NE 670I is coupled to NE 670D and 670F, which are both still coupled to NE 670E.

[0088] Figure 6F illustrates a case where multiple VNEs (VNE 670A.1 and VNE 670H.1) are implemented on different NDs (ND 600A and ND 600H) and are coupled to each other, and where the centralized control plane 676 has abstracted these multiple VNEs such that they appear as a single VNE 670T within one of the virtual networks 692 of Figure 6D, according to some embodiments of the invention. Thus, the abstraction of a NE or VNE can span multiple NDs.

[0089] While some embodiments of the invention implement the centralized control plane 676 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).

[0090] Similar to the network device implementations, the electronic device(s) running the centralized control plane 676, and thus the network controller 678 including the centralized reachability and forwarding information module 679, 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 7 illustrates, a general purpose control plane device 704 including hardware 740 comprising a set of one or more processor(s) 742 (which are often COTS processors) and network interface controller(s) 744 (NICs; also known

as network interface cards) (which include physical NIs 746), as well as non-transitory machine readable storage media 748 having stored therein centralized control plane (CCP) software 750.

[0091] In embodiments that use compute virtualization, the processor(s) 742 typically execute software to instantiate a virtualization layer 754 (e.g., in one embodiment the virtualization layer 754 represents the kernel of an operating system (or a shim executing on a base operating system) that allows for the creation of multiple instances 762A-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 754 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 762A-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 740, directly on a hypervisor represented by virtualization layer 754 (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 762A-R). Again, in embodiments where compute virtualization is used, during operation an instance of the CCP software 750 (illustrated as CCP instance 776A) is executed (e.g., within the instance 762A) on the virtualization layer 754. In embodiments where compute virtualization is not used, the CCP instance 776A is executed, as a unikernel or on top of a host operating system, on the "bare metal" general purpose control plane device 704. The instantiation of the CCP instance 776A, as well as the virtualization layer 754 and instances 762A-R if implemented, are collectively referred to as software instance(s) 752.

[0092] In some embodiments, the CCP instance 776A includes a network controller instance 778. The network controller instance 778 includes a centralized reachability and forwarding information module instance 779 (which is a middleware layer providing the context of the network controller 678 to the operating system and communicating with the various NEs), and an CCP application layer 780 (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 780 within the centralized control plane 676 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.

[0093] The centralized control plane 676 transmits relevant messages to the data plane 680 based on CCP application layer 780 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 680 may receive different messages, and thus different forwarding information. The data plane 680 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.

[0094] The general purpose control plane device 404 can implement a spanning tree generator 481. The 481 can be a program or similar set of instructions that implement the methods described herein above in reference to Figures 1-5 that provide generation of spanning trees for distribution of multicast frames in a SPRING network. The spanning tree generator 481 can be stored by the non-transitory machine readable storage media 448 and executed by the software instances 452 and processors 442.

[0095] 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).

[0096] 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.

[0097] 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.

[0098] However, when an unknown packet (for example, a "missed packet" or a "match-miss" as used in OpenFlow parlance) arrives at the data plane 680, the packet (or a subset of the packet header and content) is typically forwarded to the centralized control plane 676. The centralized control plane 676 will then program forwarding table entries into the data plane 680 to accommodate packets belonging to the flow of the unknown packet. Once a specific forwarding table entry has been programmed into the data plane 680 by the centralized control plane 676, the next packet with matching credentials will match that forwarding table entry and take the set of actions associated with that matched entry.

[0099] 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 implemented by a network device to compute a multicast distribution tree (MDT) that is a spanning tree in a source packet in routing (SPRING) network for a multicast group, where a first forwarding element has advertised itself as a root of the multicast group and at least one forwarding element has advertised a send and/or receive interest in the multicast group, a result of the spanning tree computation utilized to update a forwarding information base (FIB) of the network device, the method comprising:

receiving (101) a notice of a topology change for the SPRING network or a membership change for the multicast group;

computing (103) the MDT rooted at the first forwarding element;

determining (105) roles for each forwarding element that participates in the MDT;

configuring (107) the FIB to include a first set of entries that identify an upstream SID in a data packet received on any downstream interface of the MDT and that forward the data packet on an upstream interface of the MDT;

configuring (109) the FIB to include a second set of entries that identify the upstream SID in a data packet received on any downstream interface of the MDT and that forward the data packet with the upstream SID translated to the downstream SID, the data packet forwarded on the downstream interfaces of the MDT other than a downstream interface on which the data packet was received; and

configuring (111) the FIB to include a third set of entries that identify a downstream SID in a data packet received on any upstream interface of the MDT and that forward the data packet on the downstream interfaces of the MDT.

2. The method of claim 1, wherein the roles for each forwarding element are leaves, replication points and roots of the MDT.

3. The method of claim 1, wherein configuring the FIB to include the second set of entries includes FIB entries that forward the data packet on all downstream interfaces other than the downstream interface on which the data packet was received.

4. The method of claim 1, wherein configuring the FIB to include the third set of entries includes FIB entries that forward the data packet on all downstream interfaces of the MDT.

5. A method implemented by a network device to compute a multicast distribution tree (MDT) that is a spanning tree in a source packet in routing (SPRING) network for a multicast group, where a first forwarding element has advertised itself as a root of the multicast group and at least one forwarding element has advertised a send and/or receive interest in the multicast group, a result of the spanning tree computation utilized to update a forwarding information base (FIB) of the network device, the method comprising:

receiving (201) a notice of a topology change for the SPRING network or a membership change for the multicast group;

computing (203) the MDT rooted at the first forwarding element;

determining (205) roles for each forwarding element that participates in the MDT;

configuring (207) the FIB to include a first set of entries that identify an upstream SID for each downstream source in the MDT in a data packet and that forward the data packet on downstream interfaces and upstream interfaces of the MDT other than an interface of a received data packet where a receive interest has been received where the upstream SID is translated to a downstream SID for forwarding on downstream interfaces; and

configuring (209) the FIB to include a second set of entries that identify a single downstream SID in a data packet received on an upstream interface of the MDT and that forward the data packet with the single downstream SID on all downstream interfaces of the MDT.

6. The method of claim 5, wherein the roles for each forwarding element are leaves, replication points and roots of the MDT.

7. A network device to compute a multicast distribution tree (MDT) that is a spanning tree in a source packet in routing (SPRING) network for a multicast group, where a first forwarding element has advertised itself as a root of the multicast group and at least one forwarding element has advertised a send and/or receive interest in the multicast group, a result of the spanning tree computation utilized to update the forwarding information base (FIB) of the network device, the network device comprising:

a non-transitory machine readable medium having stored therein a multicast spanning tree generator and the FIB; and

a processor coupled to the non-transitory machine readable medium, the processor configured to execute the multicast spanning tree generator, the multicast spanning tree generator configured to receive a notice of a topology change for

the SPRING network or a membership change for the multicast group, to compute the MDT rooted at the first forwarding element, to determine roles for each forwarding element that participates in the MDT, to configure the FIB to include a first set of entries that identify an upstream SID in a data packet received on any downstream interface of the MDT and that forward the data packet on an upstream interface of the MDT, to configure the FIB to include a second set of entries that identify the upstream SID in a data packet received on any downstream interface of the MDT and that forward the data packet with the upstream SID translated to a downstream SID, the data packet forwarded on the downstream interfaces of the MDT other than a downstream interface on which the data packet was received, and configure the FIB to include a third set of entries that identify a downstream SID in a data packet received on any upstream interface of the MDT and that forward the data packet on the downstream interfaces of the MDT.

8. The network device of claim 7, wherein the roles for each forwarding element are leaves, replication points and roots of the MDT.

9. The network device of claim 7, wherein configuring the FIB to include the second set of entries includes FIB entries that forward the data packet on all downstream interfaces other than the downstream interface on which the data packet was received.

10. The network device of claim 7, wherein configuring the FIB to include the third set of entries includes FIB entries that forward the data packet on all downstream interfaces of the MDT.

11. A computing device in communication with a network device, the computing device to execute a plurality of virtual machines for implementing network function virtualization (NFV), the computing device to compute a multicast distribution tree (MDT) that is a spanning tree in a source packet in routing (SPRING) network for a multicast group, where a first forwarding element has advertised itself as a root of the multicast group and at least one forwarding element has advertised a send and/or receive interest in the multicast group, a result of the spanning tree computation utilized to update a forwarding information base (FIB) of the network device, the computing device comprising:
a non-transitory machine readable medium having stored therein a multicast spanning tree generator and the FIB; and

a processor coupled to the non-transitory machine readable medium, the processor configured to execute a virtual machine from the plurality of virtual machines, the virtual machine configured to execute the multicast spanning tree generator, the multicast spanning tree generator configured to receive a notice of a topology change for the SPRING network or a membership change for the multicast group, to compute the MDT rooted at the first forwarding element, to determine roles for each forwarding element that participates in the MDT, to configure the FIB to include a first set of entries that identify an upstream SID in a data packet received on any downstream interface of the MDT and that forward the data packet on an upstream interface of the MDT, to configure the FIB to include a second set of entries that identify the upstream SID in a data packet received on any downstream interface of the MDT and that forward the data packet with the upstream SID translated to a downstream SID, the data packet forwarded on downstream interfaces of the MDT other than a downstream interface on which the data packet was received, and configure the FIB to include a third set of entries that identify a downstream SID in a data packet received on any upstream interface of the MDT and that forward the data packet on the downstream interfaces of the MDT.

12. The computing device of claim 11, wherein the roles for each forwarding element are leaves, replication points and roots of the MDT.

13. The computing device of claim 11, wherein configuring the FIB to include the second set of entries includes FIB entries that forward the data packet on all downstream interfaces other than the downstream interface on which the data packet was received.

14. The computing device of claim 11, wherein configuring the FIB to include the third set of entries includes FIB entries that forward the data packet on all downstream interfaces of the MDT.

15. A control plane device is configured to implement a control plane of a software defined networking (SDN) network including a network device in a network with a plurality of network devices, wherein the control plane device is configured to configure the network device to compute a multicast distribution tree (MDT) that is a spanning tree in a source packet in routing (SPRING) network for a multicast group, where a first forwarding element has advertised itself as a root of the multicast group and at least one forwarding element has advertised a send and/or receive interest in the multicast group, a result of

the spanning tree computation utilized to update a forwarding information base (FIB) of the network device, the control plane device comprising:

a non-transitory machine readable medium having stored therein a multicast spanning tree generator and the FIB; and

a processor coupled to the non-transitory machine readable medium, the processor configured to execute the multicast spanning tree generator, the multicast spanning tree generator configured to receive a notice of a topology change for the SPRING network or a membership change for the multicast group, to compute the MDT rooted at the first forwarding element, to determine roles for each forwarding element that participates in the MDT, to configure the FIB to include a first set of entries that identify an upstream SID in a data packet received on any downstream interface of the MDT and that forward the data packet on an upstream interface of the MDT, to configure the FIB to include a second set of entries that identify the upstream SID in a data packet received on any downstream interface of the MDT and that forward the data packet with the upstream SID translated to a downstream SID, the data packet forwarded on the downstream interfaces of the MDT other than a downstream interface on which the data packet was received, and configure the FIB to include a third set of entries that identify a downstream SID in a data packet received on any upstream interface of the MDT and that forward the data packet on the downstream interfaces of the MDT.

16. The control plane device of claim 15, wherein the roles for each forwarding element are leaves, replication points and roots of the MDT.

17. The control plane device of claim 15, wherein configuring the FIB to include the second set of entries includes FIB entries that forward the data packet on all downstream interfaces other than the downstream interface on which the data packet was received.

18. The control plane device of claim 15, wherein configuring the FIB to include the third set of entries includes FIB entries that forward the data packet on all downstream interfaces of the MDT.

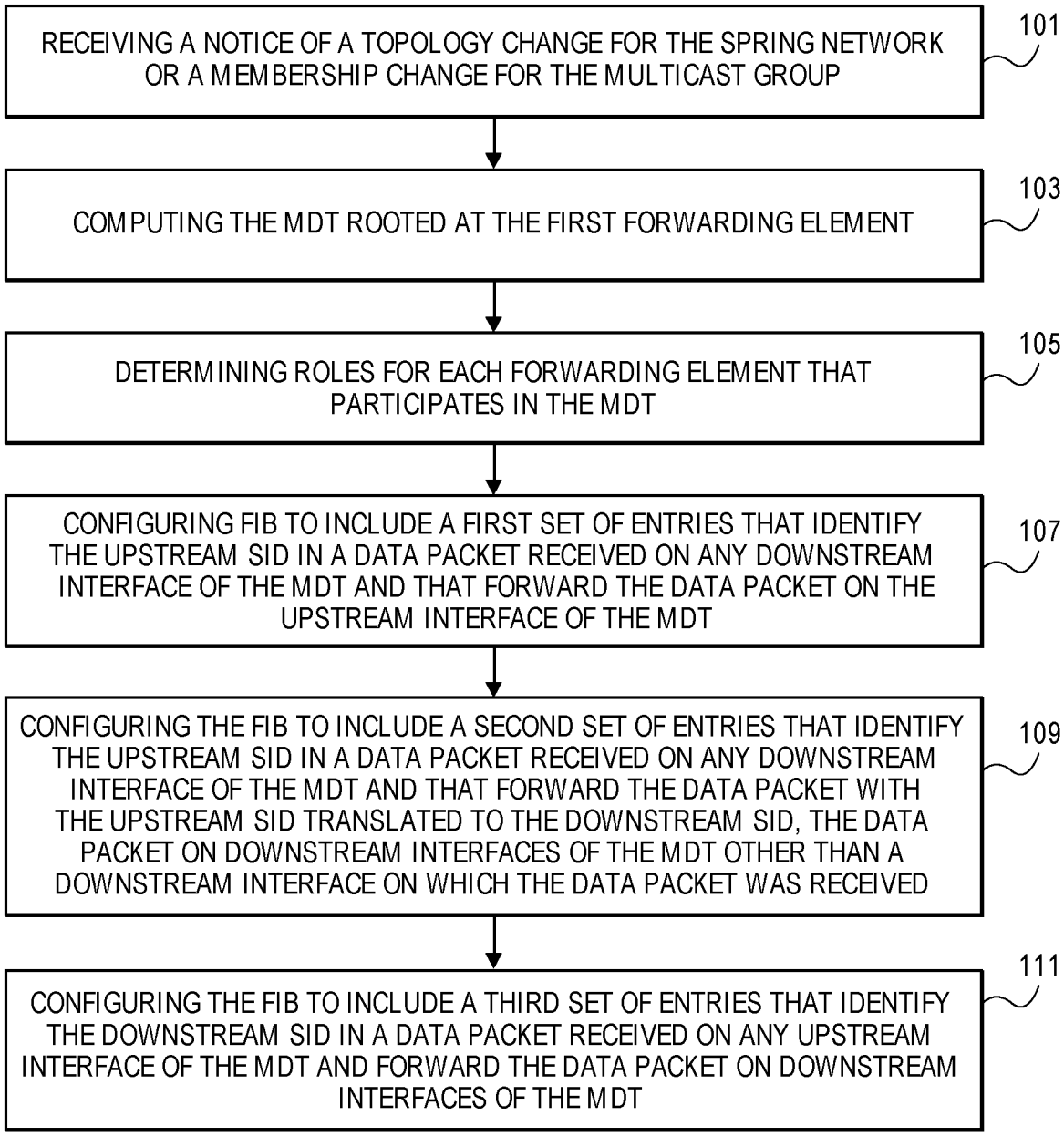

RECEIVING A NOTICE OF A TOPOLOGY CHANGE FOR THE SPRING NETWORK OR A MEMBERSHIP CHANGE FOR THE MULTICAST GROUP
101
COMPUTING THE MDT ROOTED AT THE FIRST FORWARDING ELEMENT
103
DETERMINING ROLES FOR EACH FORWARDING ELEMENT THAT PARTICIPATES IN THE MDT
105
CONFIGURING FIB TO INCLUDE A FIRST SET OF ENTRIES THAT IDENTIFY THE UPSTREAM SID IN A DATA PACKET RECEIVED ON ANY DOWNSTREAM INTERFACE OF THE MDT AND THAT FORWARD THE DATA PACKET ON THE UPSTREAM INTERFACE OF THE MDT
107
CONFIGURING THE FIB TO INCLUDE A SECOND SET OF ENTRIES THAT IDENTIFY THE UPSTREAM SID IN A DATA PACKET RECEIVED ON ANY DOWNSTREAM INTERFACE OF THE MDT AND THAT FORWARD THE DATA PACKET WITH THE UPSTREAM SID TRANSLATED TO THE DOWNSTREAM SID, THE DATA PACKET ON DOWNSTREAM INTERFACES OF THE MDT OTHER THAN A DOWNSTREAM INTERFACE ON WHICH THE DATA PACKET WAS RECEIVED
109
CONFIGURING THE FIB TO INCLUDE A THIRD SET OF ENTRIES THAT IDENTIFY THE DOWNSTREAM SID IN A DATA PACKET RECEIVED ON ANY UPSTREAM INTERFACE OF THE MDT AND FORWARD THE DATA PACKET ON DOWNSTREAM INTERFACES OF THE MDT
111

FIG. 1

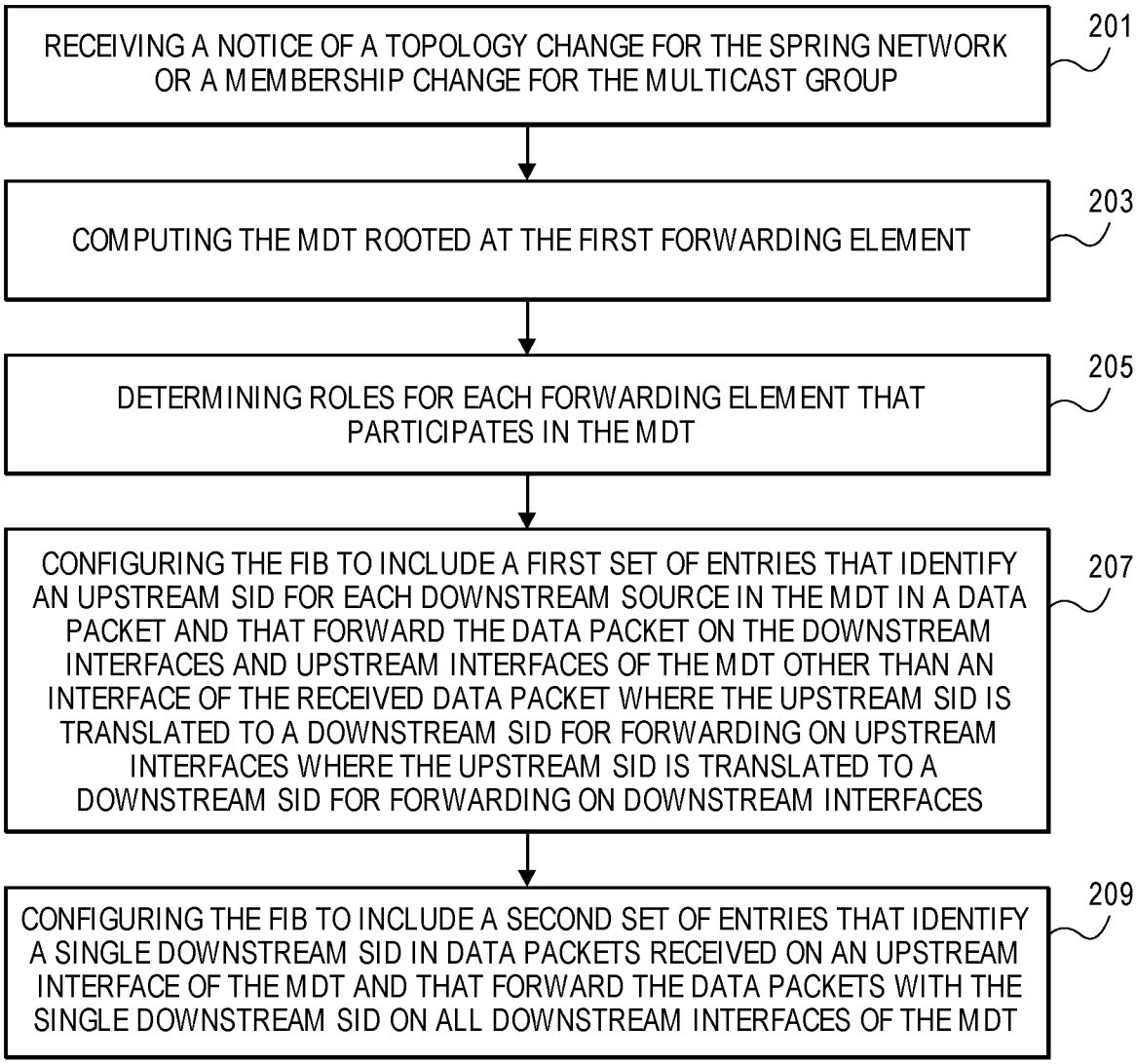
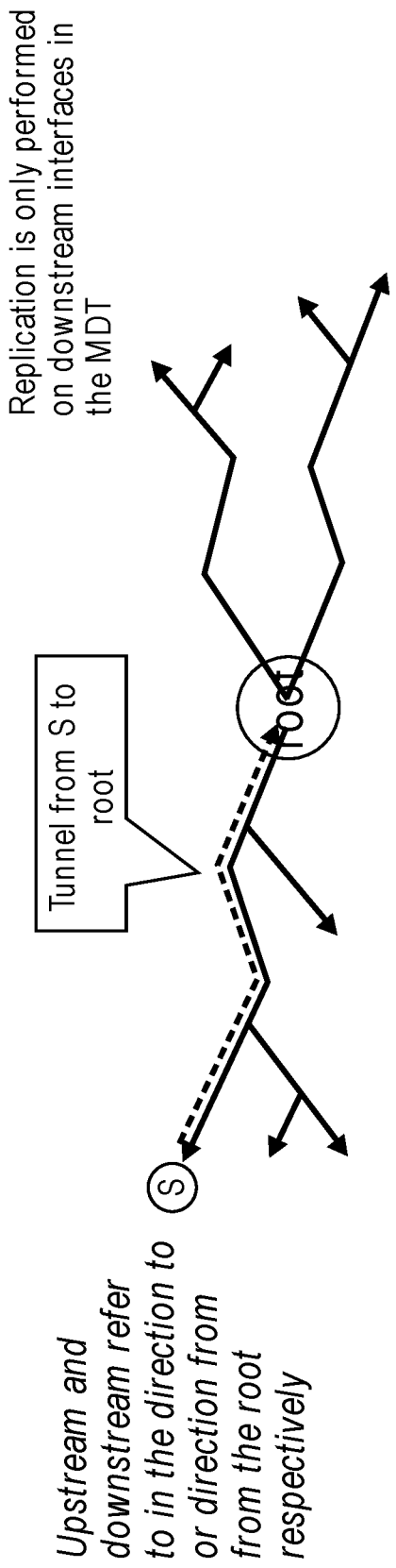
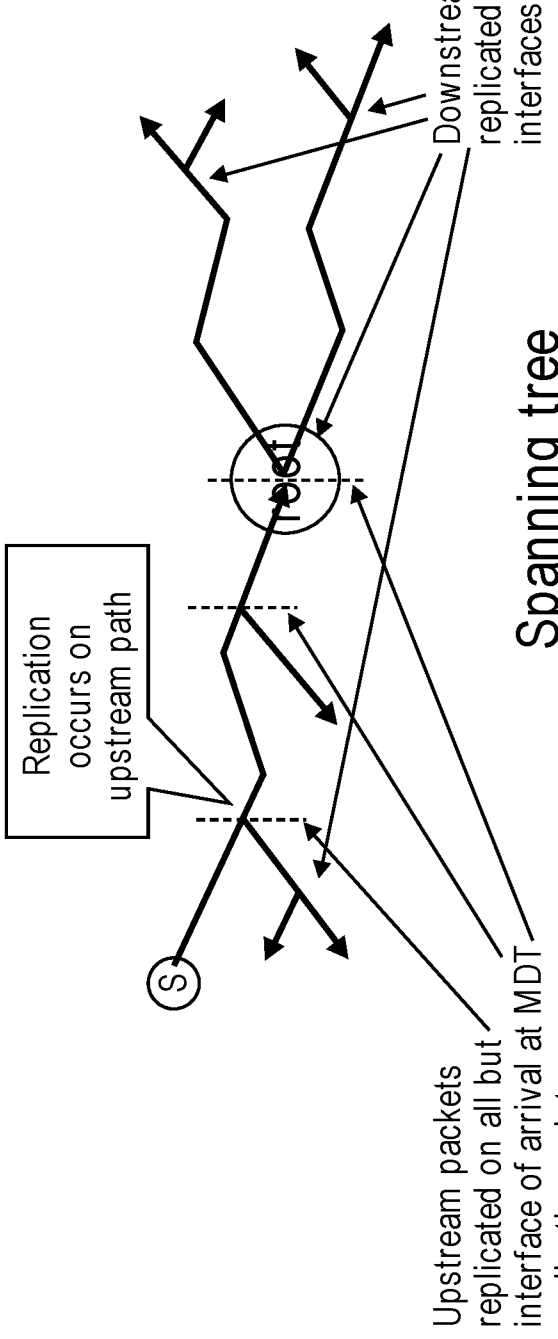

RECEIVING A NOTICE OF A TOPOLOGY CHANGE FOR THE SPRING NETWORK OR A MEMBERSHIP CHANGE FOR THE MULTICAST GROUP
201
COMPUTING THE MDT ROOTED AT THE FIRST FORWARDING ELEMENT
203
DETERMINING ROLES FOR EACH FORWARDING ELEMENT THAT PARTICIPATES IN THE MDT
205
CONFIGURING THE FIB TO INCLUDE A FIRST SET OF ENTRIES THAT IDENTIFY AN UPSTREAM SID FOR EACH DOWNSTREAM SOURCE IN THE MDT IN A DATA PACKET AND THAT FORWARD THE DATA PACKET ON THE DOWNSTREAM INTERFACES AND UPSTREAM INTERFACES OF THE MDT OTHER THAN AN INTERFACE OF THE RECEIVED DATA PACKET WHERE THE UPSTREAM SID IS TRANSLATED TO A DOWNSTREAM SID FOR FORWARDING ON UPSTREAM INTERFACES WHERE THE UPSTREAM SID IS TRANSLATED TO A DOWNSTREAM SID FOR FORWARDING ON DOWNSTREAM INTERFACES
207
CONFIGURING THE FIB TO INCLUDE A SECOND SET OF ENTRIES THAT IDENTIFY A SINGLE DOWNSTREAM SID IN DATA PACKETS RECEIVED ON AN UPSTREAM INTERFACE OF THE MDT AND THAT FORWARD THE DATA PACKETS WITH THE SINGLE DOWNSTREAM SID ON ALL DOWNSTREAM INTERFACES OF THE MDT
209

FIG. 2



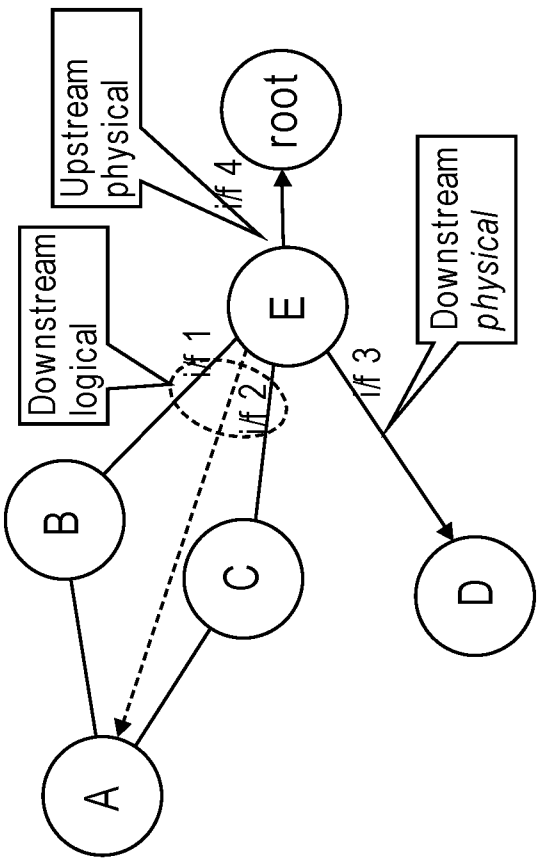
PIM BI-DIR

FIG. 3A



Spanning tree

FIG. 3B



Node E's FIB

I/F	SID	Action
1	u	Replicate • u→d on i/f 3 • u→u on i/f 4
2	u	Replicate • u→d on i/f 3 • u→u on i/f 4
3	u	Replicate • u→A/d on i/f 1 or 2 (ECMP) • u→d on i/f 4
4	d	Replicate • d→A/d on i/f 1 or 2 ECMP • d→d on i/f 3

FIG. 4

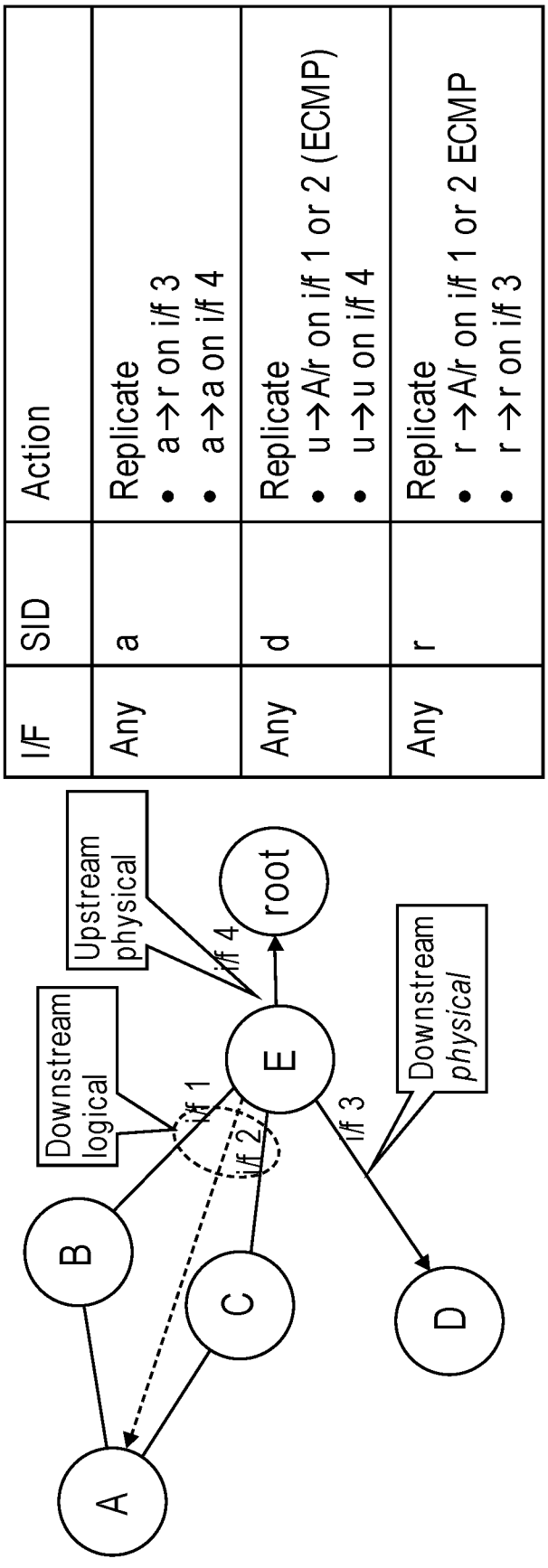

I/F
SID
Action
Any
a
Replicate
• a→r on i/f 3
• a→a on i/f 4
Any
d
Replicate
• u→A/r on i/f 1 or 2 (ECMP)
• u→u on i/f 4
Any
r
Replicate
• r →A/r on i/f 1 or 2 ECMP
• r →r on i/f 3
Upstream physical
Downstream logical
Downstream physical
i/f 1
i/f 2
i/f 3
i/f 4
root
A
B
C
D
E

FIG. 5

FIG. 6A

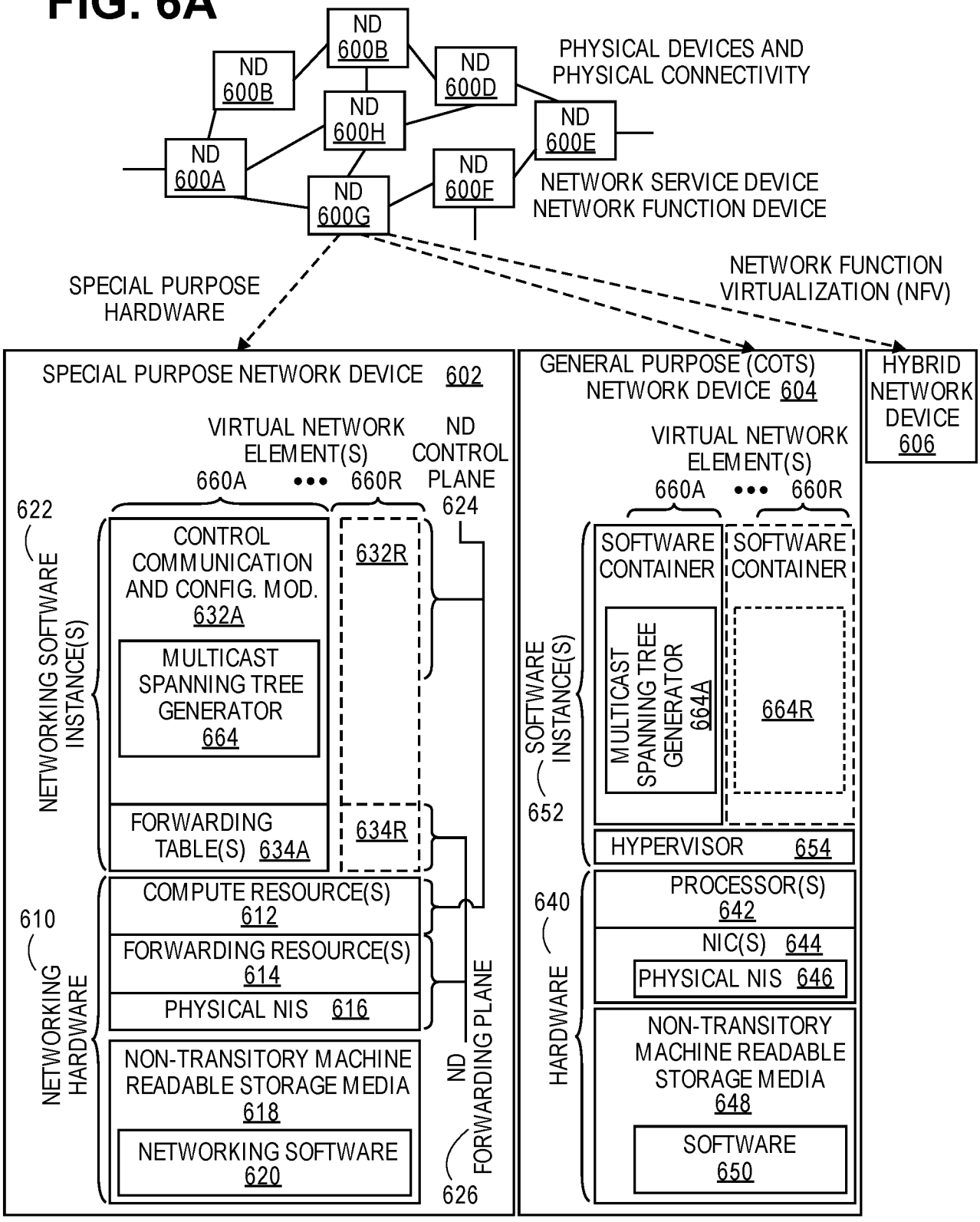

ND 600A
ND 600B
ND 600B
ND 600D
ND 600E
ND 600F
ND 600G
ND 600H
PHYSICAL DEVICES AND PHYSICAL CONNECTIVITY
NETWORK SERVICE DEVICE NETWORK FUNCTION DEVICE
SPECIAL PURPOSE HARDWARE
NETWORK FUNCTION VIRTUALIZATION (NFV)
SPECIAL PURPOSE NETWORK DEVICE 602
VIRTUAL NETWORK ELEMENT(S)
660A ••• 660R
ND CONTROL PLANE 624
622
NETWORKING SOFTWARE INSTANCE(S)
CONTROL COMMUNICATION AND CONFIG. MOD. 632A
632R
MULTICAST SPANNING TREE GENERATOR 664
FORWARDING TABLE(S) 634A
634R
610
NETWORKING HARDWARE
COMPUTE RESOURCE(S) 612
FORWARDING RESOURCE(S) 614
PHYSICAL NIS 616
NON-TRANSITORY MACHINE READABLE STORAGE MEDIA 618
NETWORKING SOFTWARE 620
ND FORWARDING PLANE
626
GENERAL PURPOSE (COTS) NETWORK DEVICE 604
VIRTUAL NETWORK ELEMENT(S)
660A ••• 660R
SOFTWARE INSTANCE(S)
652
SOFTWARE CONTAINER
MULTICAST SPANNING TREE GENERATOR 664A
SOFTWARE CONTAINER
664R
HYPERVISOR 654
640
HARDWARE
PROCESSOR(S) 642
NIC(S) 644
PHYSICAL NIS 646
NON-TRANSITORY MACHINE READABLE STORAGE MEDIA 648
SOFTWARE 650
HYBRID NETWORK DEVICE 606

FIG. 6B

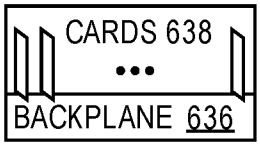

CARDS 638
•••
BACKPLANE 636

FIG. 6C

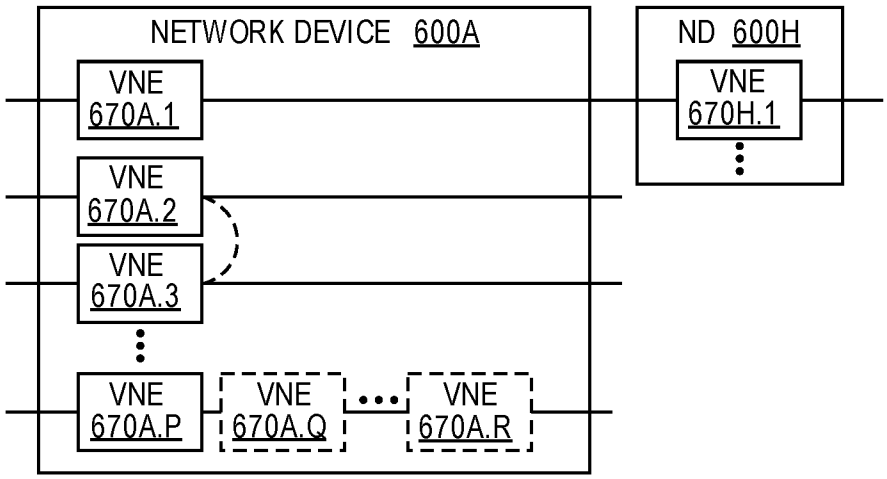


FIG. 6D

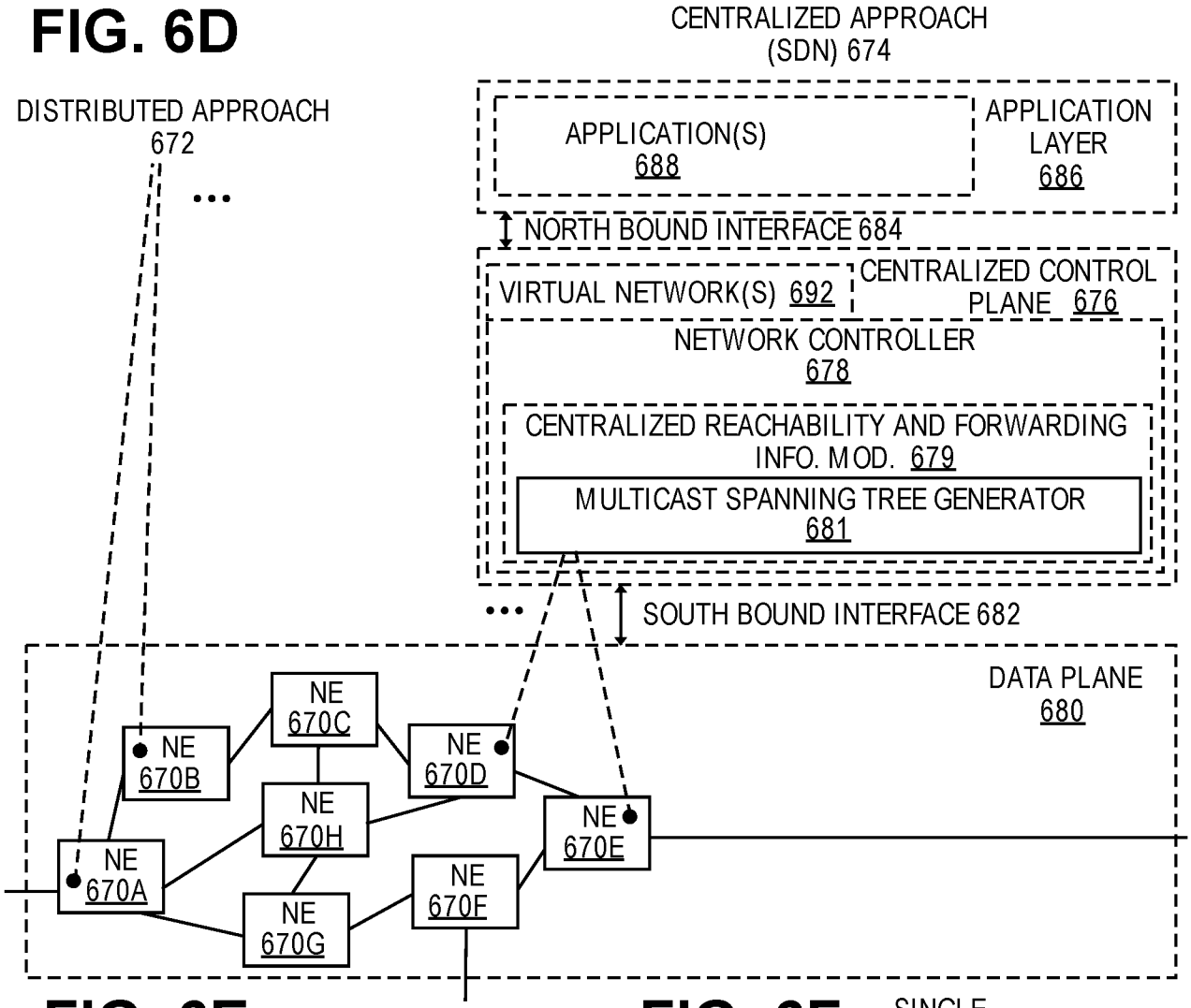


FIG. 6E

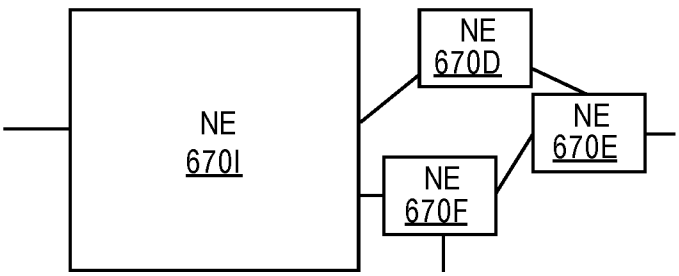


FIG. 6F

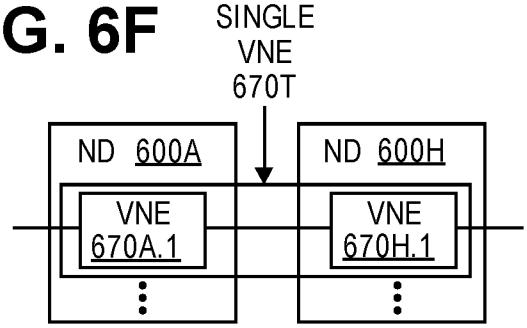
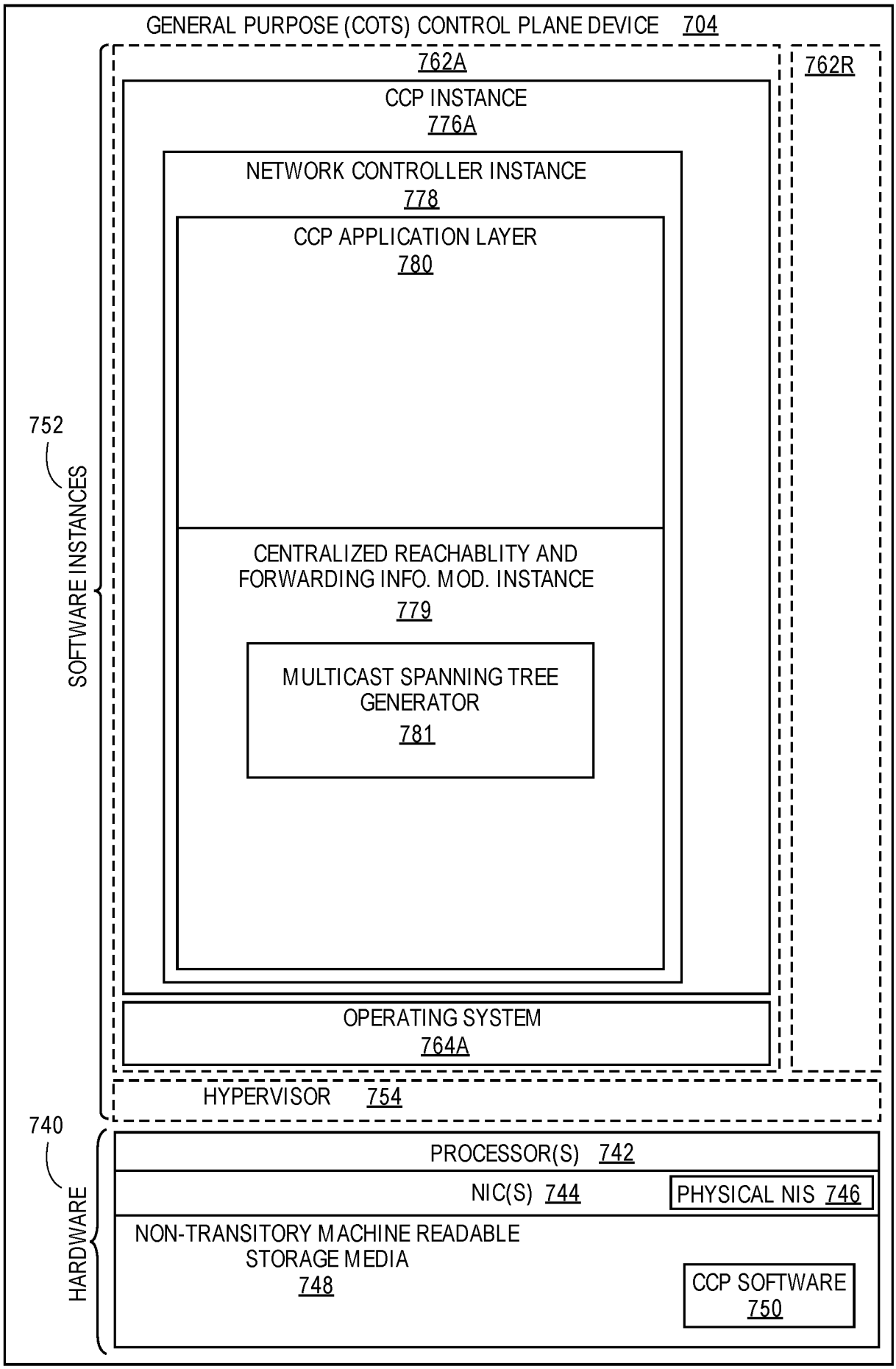


FIG. 7


GENERAL PURPOSE (COTS) CONTROL PLANE DEVICE 704
762A
762R
CCP INSTANCE
776A
NETWORK CONTROLLER INSTANCE
778
CCP APPLICATION LAYER
780
752
SOFTWARE INSTANCES
CENTRALIZED REACHABLITY AND FORWARDING INFO. MOD. INSTANCE
779
MULTICAST SPANNING TREE GENERATOR
781
OPERATING SYSTEM
764A
HYPERVISOR 754
740
HARDWARE
PROCESSOR(S) 742
NIC(S) 744
PHYSICAL NIS 746
NON-TRANSITORY MACHINE READABLE STORAGE MEDIA
748
CCP SOFTWARE
750

INTERNATIONAL SEARCH REPORT

International application No
PCT/IB2016/050985

A. CLASSIFICATION OF SUBJECT MATTER
INV. H04L12/759
ADD. H04L12/18 H04L12/761 H04L12/24

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	US 2015/188771 A1 (ALLAN DAVID IAN [US] ET AL) 2 July 2015 (2015-07-02) the whole document ----- -/--	1-18

[X] Further documents are listed in the continuation of Box C.

[X] 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
2 November 2016	09/11/2016

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	Fischer, Erik

INTERNATIONAL SEARCH REPORT

International application No
PCT/IB2016/050985

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	IJ WIJNANDS ET AL: "Label Distribution Protocol Extensions for Point-to-Multipoint and Multipoint-to-Multipoint Label Switched Paths; rfc6388.txt", LABEL DISTRIBUTION PROTOCOL EXTENSIONS FOR POINT-TO-MULTIPOINT AND MULTIPOINT-TO-MULTIPOINT LABEL SWITCHED PATHS; RFC6388.TXT, INTERNET ENGINEERING TASK FORCE, IETF; STANDARD, INTERNET SOCIETY (ISOC) 4, RUE DES FALAISES CH-1205 GENEVA, SWITZERLAND, 5 November 2011 (2011-11-05), pages 1-39, XP015081335, [retrieved on 2011-11-05] page 1, paragraph Abstract page 3, paragraph 1. Introduction - page 4, paragraph 1. Introduction page 14, paragraph 3. Setting up MP2MP LSPs with LDP - page 26, paragraph 7. Root Node Redundancy -----	1-18
A	US 2012/075988 A1 (LU WENHU [US] ET AL) 29 March 2012 (2012-03-29) paragraph [0112] - paragraph [0145]; figures 13-20 -----	1-18
A	US 2008/298360 A1 (WIJNANDS JACOB IJSBRAND [BE] ET AL) 4 December 2008 (2008-12-04) paragraph [0018] - paragraph [0074]; figures 1-10 -----	1-18

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/IB2016/050985

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2015188771 A1	02-07-2015	US 2015188771 A1	02-07-2015
		WO 2015097576 A1	02-07-2015
US 2012075988 A1	29-03-2012	CN 103155485 A	12-06-2013
		EP 2622803 A2	07-08-2013
		JP 5876493 B2	02-03-2016
		JP 2013542662 A	21-11-2013
		KR 20140001882 A	07-01-2014
		US 2012075988 A1	29-03-2012
		US 2014313880 A1	23-10-2014
		WO 2012042440 A2	05-04-2012
US 2008298360 A1	04-12-2008	US 2008298360 A1	04-12-2008
		US 2013177018 A1	11-07-2013