US 20130318219A1

(19) United States

(12) Patent Application Publication

(10) Pub. No.: US 2013/0318219 A1

(43) Pub. Date: Nov. 28, 2013

Kancherla

(54) **LAYER-3 OVERLAY GATEWAYS**

(71) Applicant: **BROCADE COMMUNICATIONS SYSTEMS, INC**, San Jose, CA (US)

(72) Inventor: **Mani Prasad Kancherla**, Cupertino, CA (US)

(73) Assignee: **BROCADE COMMUNICATIONS SYSTEMS, INC**, San Jose, CA (US)

(21) Appl. No.: **13/899,849**

(22) Filed: **May 22, 2013**

Related U.S. Application Data

(60) Provisional application No. 61/650,943, filed on May 23, 2012.

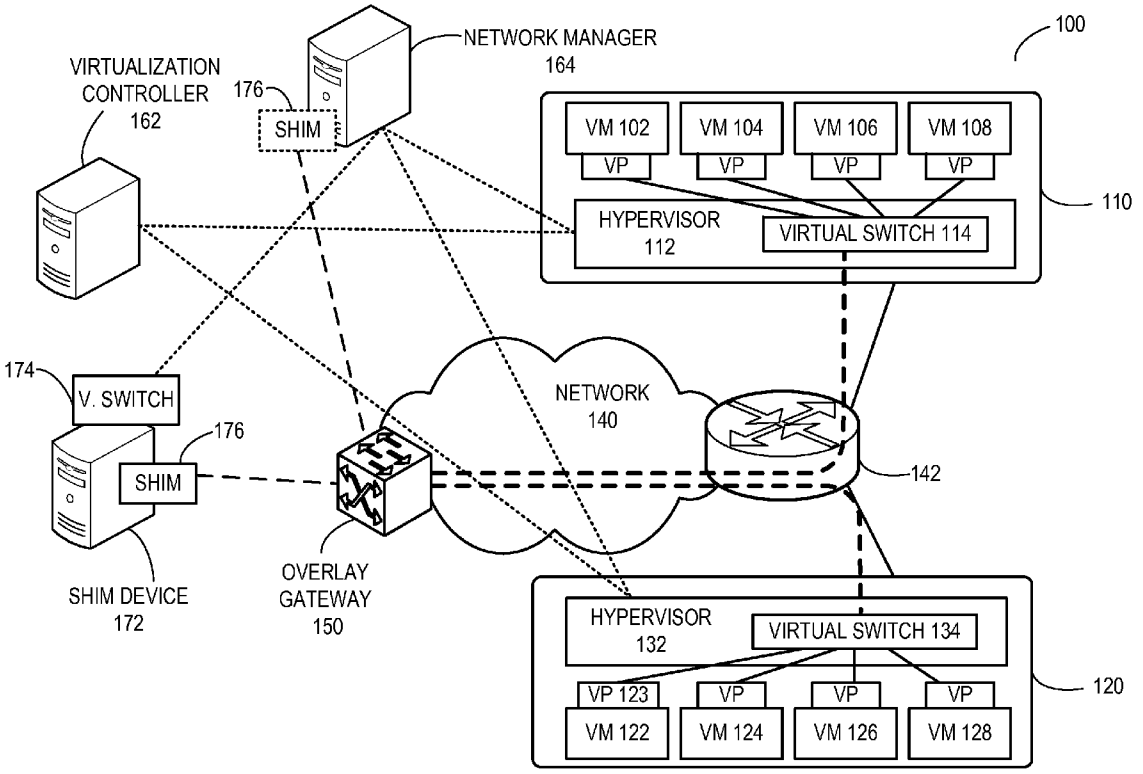
Publication Classification

(51) **Int. Cl.**
H04L 12/24 (2006.01)

(52) **U.S. Cl.**
CPC *H04L 41/0806* (2013.01)
USPC **709/222**

(57) **ABSTRACT**

One embodiment of the present invention provides a computing system. The computing system includes a processor and a computer-readable storage medium for storing instructions. Based on the instructions, the processor operates the computing system as an overlay gateway. The computing system initiates and terminates an overlay tunnel associated with a virtual machine. During operation, the computing system maps a virtual Internet Protocol (IP) address of the virtual machine to a second IP address used to terminate the overlay tunnel based on information received from a configuration system. The computing system then determines an output port for a data packet based on the second IP address. The data packet comprises an inner packet and the destination address of this inner packet corresponds to the virtual IP address.



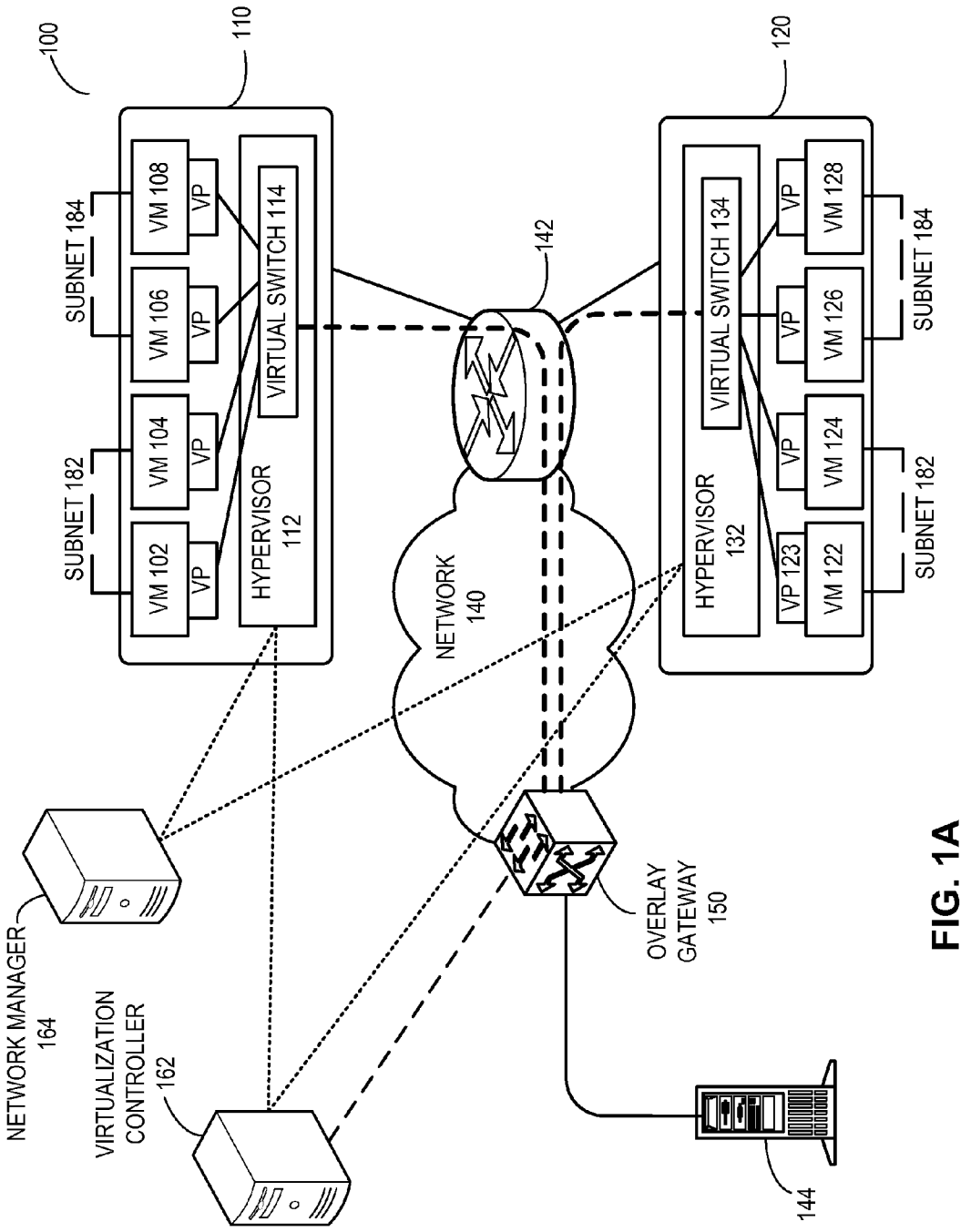

100
110
120
SUBNET 184
SUBNET 182
VM 108
VM 106
VM 104
VM 102
VP
VIRTUAL SWITCH 114
HYPERVISOR 112
142
NETWORK 140
VIRTUAL SWITCH 134
HYPERVISOR 132
VM 128
VM 126
VM 124
VM 122
VP 123
NETWORK MANAGER 164
VIRTUALIZATION CONTROLLER 162
OVERLAY GATEWAY 150
144

FIG. 1A

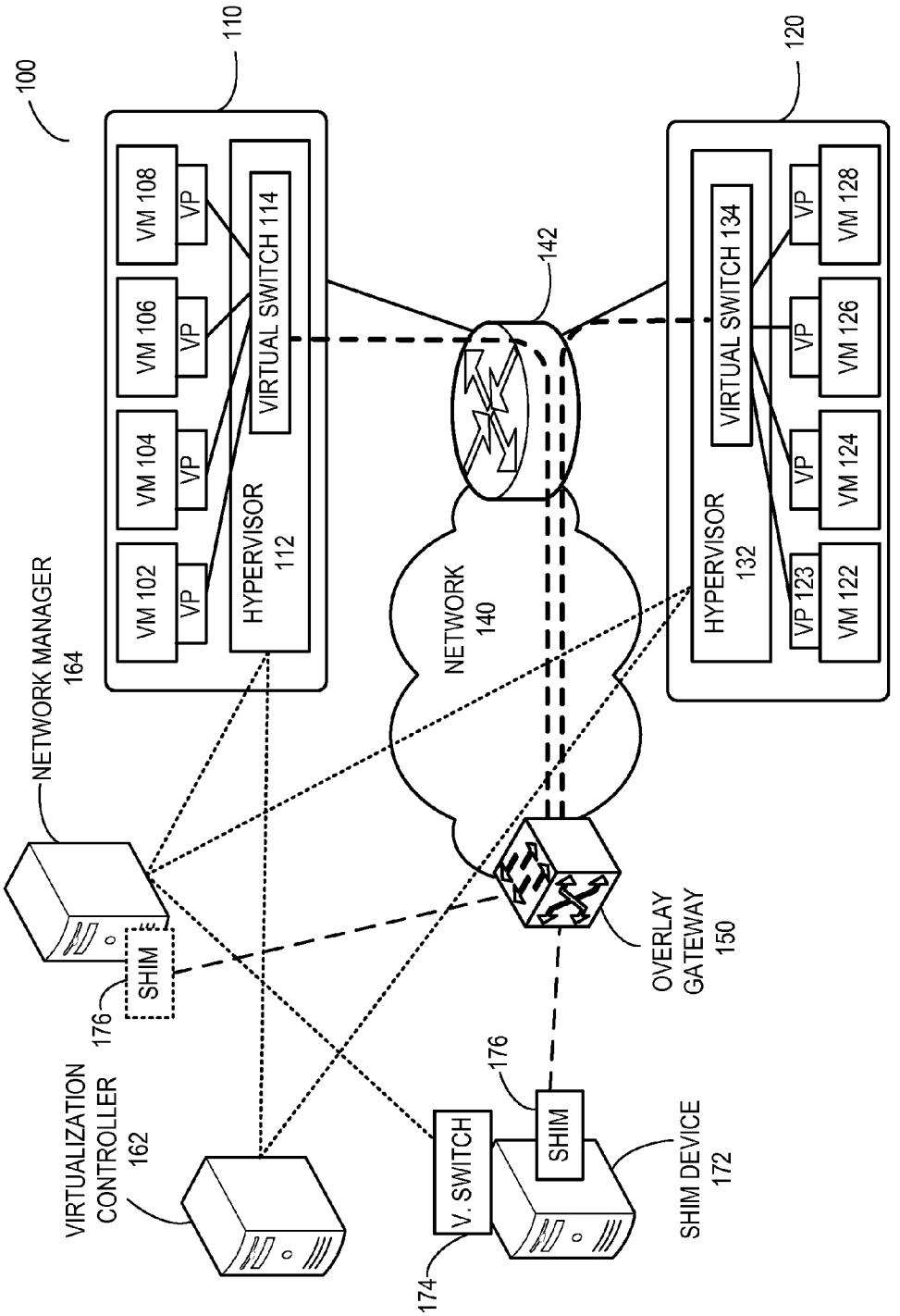

100
110
120
VM 108
VP
VM 106
VP
VM 104
VP
VM 102
VP
VIRTUAL SWITCH 114
HYPERVISOR 112
142
NETWORK 140
VIRTUAL SWITCH 134
VP
VM 128
VP
VM 126
VP
VM 124
VP 123
VM 122
HYPERVISOR 132
NETWORK MANAGER 164
SHIM
176
OVERLAY GATEWAY 150
VIRTUALIZATION CONTROLLER 162
V. SWITCH
174
SHIM DEVICE 172

FIG. 1B

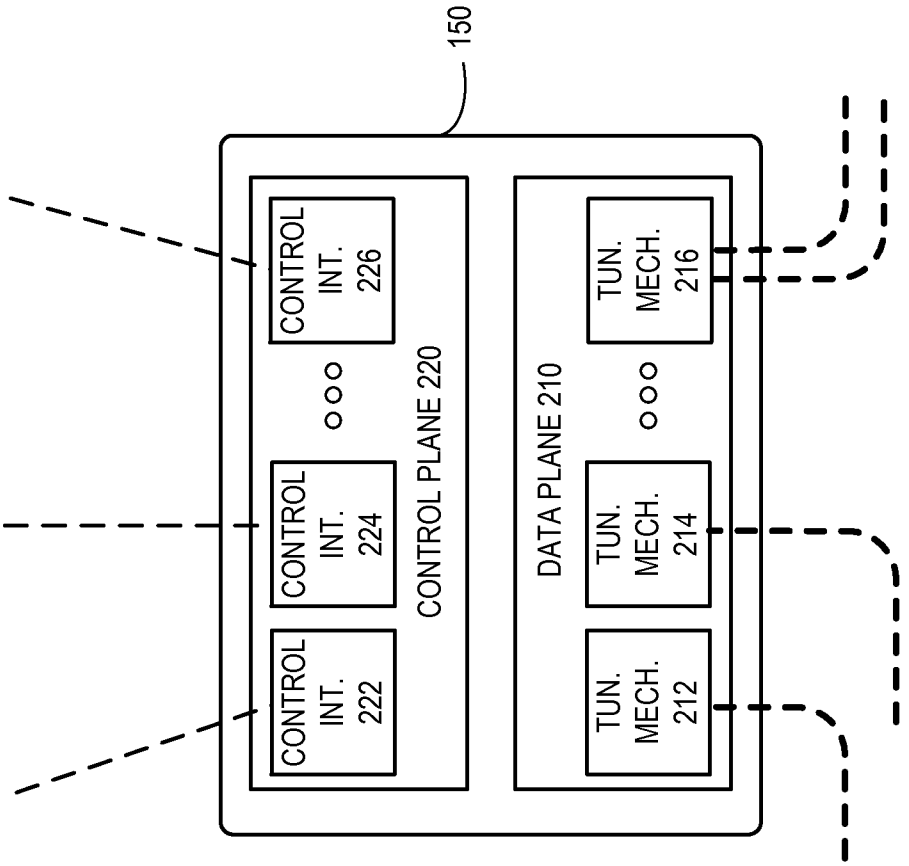
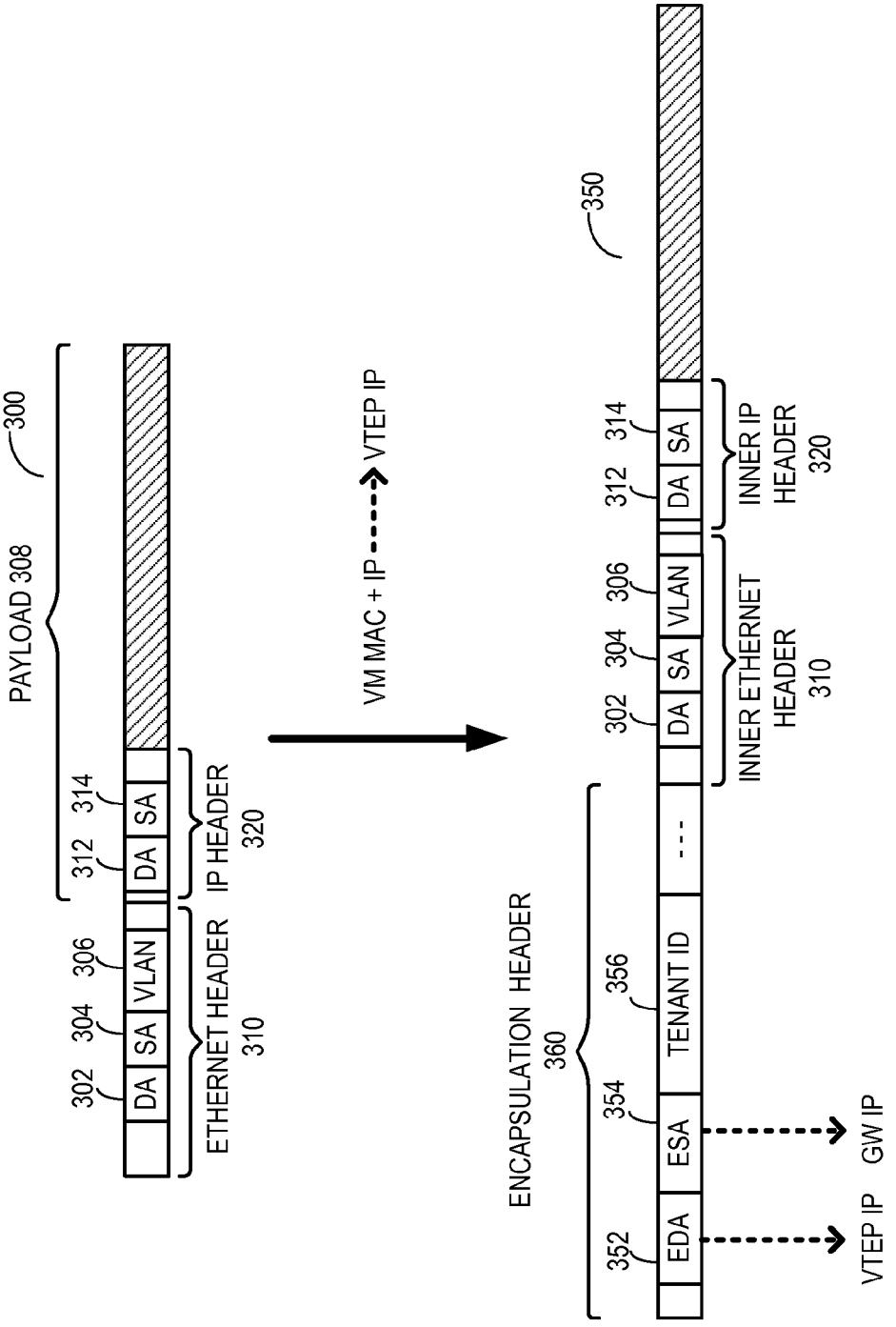

150
CONTROL INT. 226
CONTROL INT. 224
CONTROL INT. 222
CONTROL PLANE 220
DATA PLANE 210
TUN. MECH. 216
TUN. MECH. 214
TUN. MECH. 212

FIG. 2


PAYLOAD 308
300
302
304
306
312
314
DA
SA
VLAN
DA
SA
ETHERNET HEADER
310
IP HEADER
320
VM MAC + IP
VTEP IP
350
ENCAPSULATION HEADER
360
352
354
356
EDA
ESA
TENANT ID

DA
SA
VLAN
DA
SA
INNER ETHERNET HEADER
310
INNER IP HEADER
320
VTEP IP
GW IP

FIG. 3

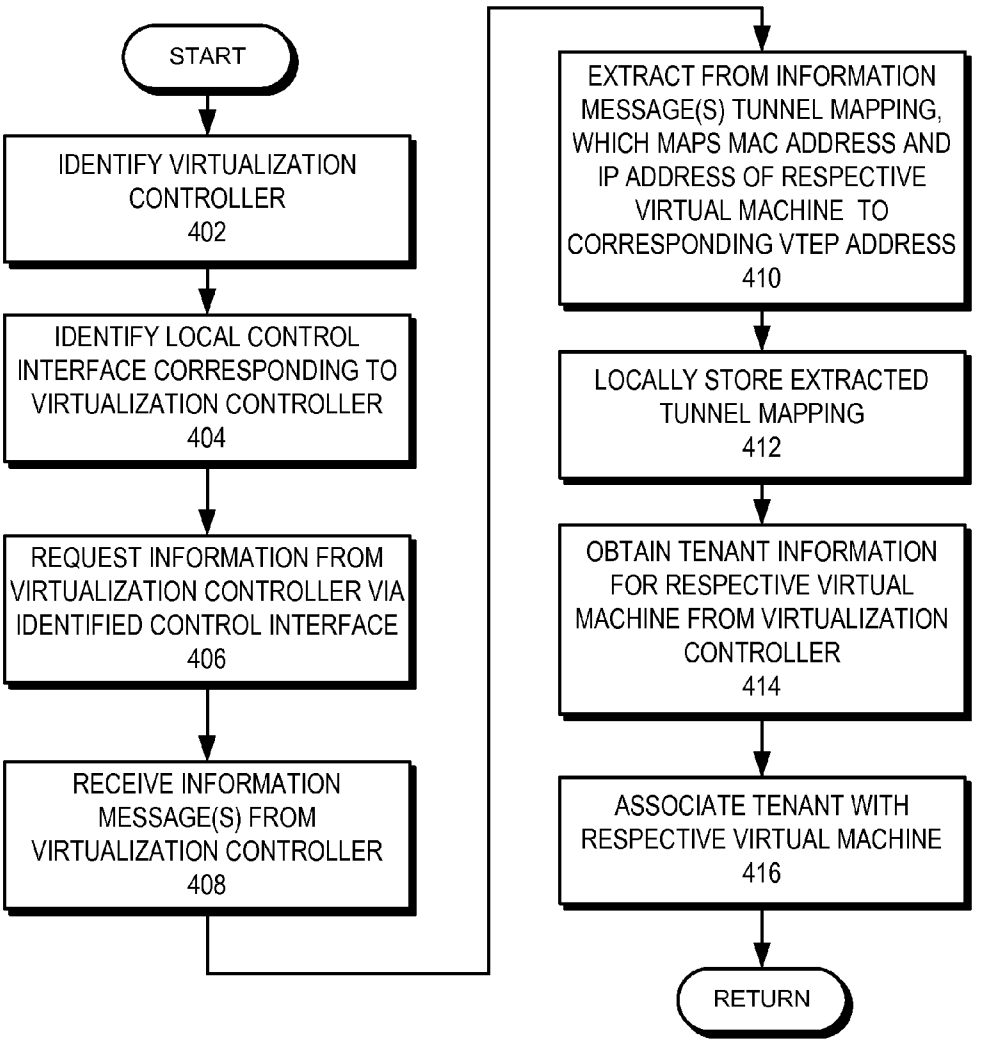

START
IDENTIFY VIRTUALIZATION CONTROLLER 402
IDENTIFY LOCAL CONTROL INTERFACE CORRESPONDING TO VIRTUALIZATION CONTROLLER 404
REQUEST INFORMATION FROM VIRTUALIZATION CONTROLLER VIA IDENTIFIED CONTROL INTERFACE 406
RECEIVE INFORMATION MESSAGE(S) FROM VIRTUALIZATION CONTROLLER 408
EXTRACT FROM INFORMATION MESSAGE(S) TUNNEL MAPPING, WHICH MAPS MAC ADDRESS AND IP ADDRESS OF RESPECTIVE VIRTUAL MACHINE TO CORRESPONDING VTEP ADDRESS 410
LOCALLY STORE EXTRACTED TUNNEL MAPPING 412
OBTAIN TENANT INFORMATION FOR RESPECTIVE VIRTUAL MACHINE FROM VIRTUALIZATION CONTROLLER 414
ASSOCIATE TENANT WITH RESPECTIVE VIRTUAL MACHINE 416
RETURN

FIG. 4A

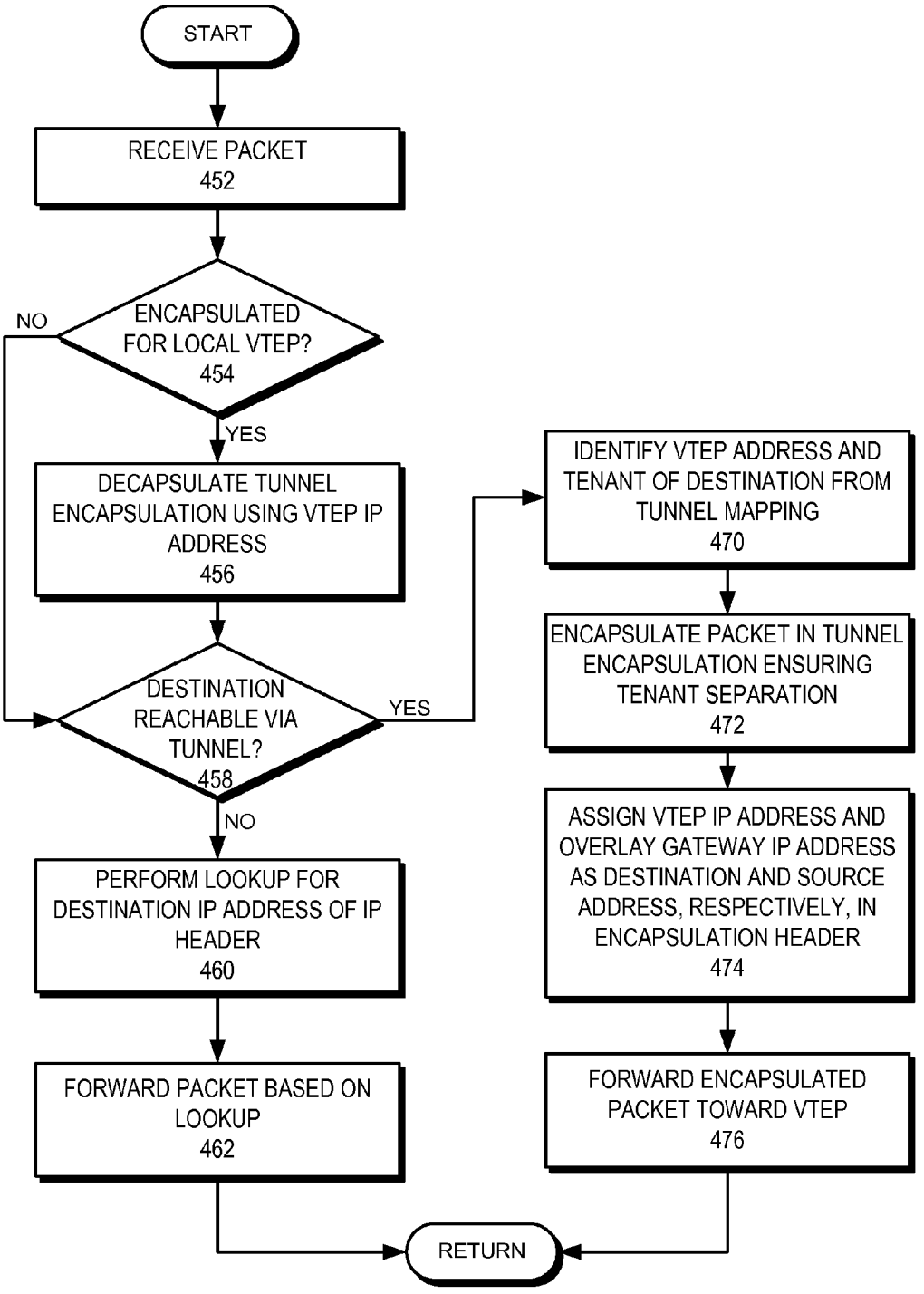

START
RECEIVE PACKET
452
ENCAPSULATED FOR LOCAL VTEP?
454
NO
YES
DECAPSULATE TUNNEL ENCAPSULATION USING VTEP IP ADDRESS
456
DESTINATION REACHABLE VIA TUNNEL?
458
YES
NO
IDENTIFY VTEP ADDRESS AND TENANT OF DESTINATION FROM TUNNEL MAPPING
470
ENCAPSULATE PACKET IN TUNNEL ENCAPSULATION ENSURING TENANT SEPARATION
472
ASSIGN VTEP IP ADDRESS AND OVERLAY GATEWAY IP ADDRESS AS DESTINATION AND SOURCE ADDRESS, RESPECTIVELY, IN ENCAPSULATION HEADER
474
FORWARD ENCAPSULATED PACKET TOWARD VTEP
476
PERFORM LOOKUP FOR DESTINATION IP ADDRESS OF IP HEADER
460
FORWARD PACKET BASED ON LOOKUP
462
RETURN

FIG. 4B

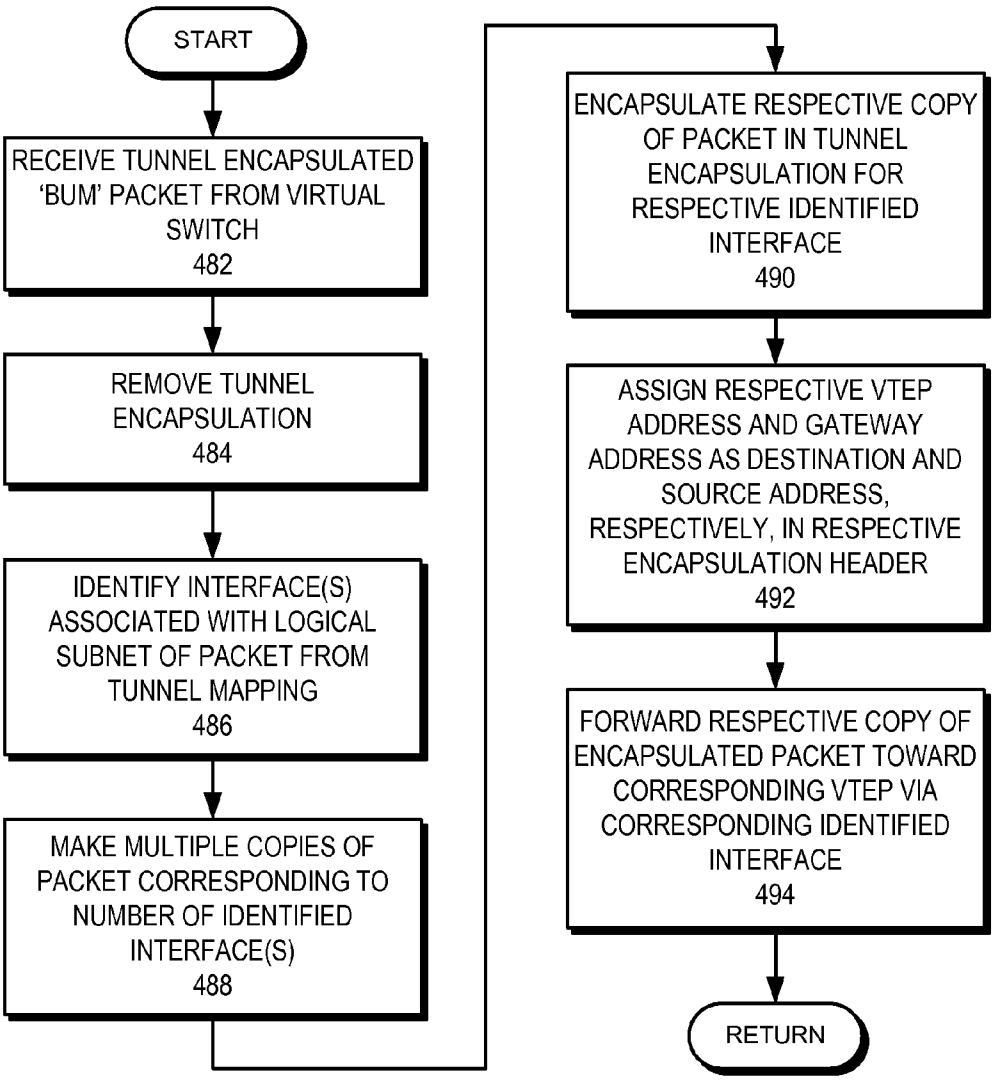

START
RECEIVE TUNNEL ENCAPSULATED 'BUM' PACKET FROM VIRTUAL SWITCH 482
REMOVE TUNNEL ENCAPSULATION 484
IDENTIFY INTERFACE(S) ASSOCIATED WITH LOGICAL SUBNET OF PACKET FROM TUNNEL MAPPING 486
MAKE MULTIPLE COPIES OF PACKET CORRESPONDING TO NUMBER OF IDENTIFIED INTERFACE(S) 488
ENCAPSULATE RESPECTIVE COPY OF PACKET IN TUNNEL ENCAPSULATION FOR RESPECTIVE IDENTIFIED INTERFACE 490
ASSIGN RESPECTIVE VTEP ADDRESS AND GATEWAY ADDRESS AS DESTINATION AND SOURCE ADDRESS, RESPECTIVELY, IN RESPECTIVE ENCAPSULATION HEADER 492
FORWARD RESPECTIVE COPY OF ENCAPSULATED PACKET TOWARD CORRESPONDING VTEP VIA CORRESPONDING IDENTIFIED INTERFACE 494
RETURN

FIG. 4C

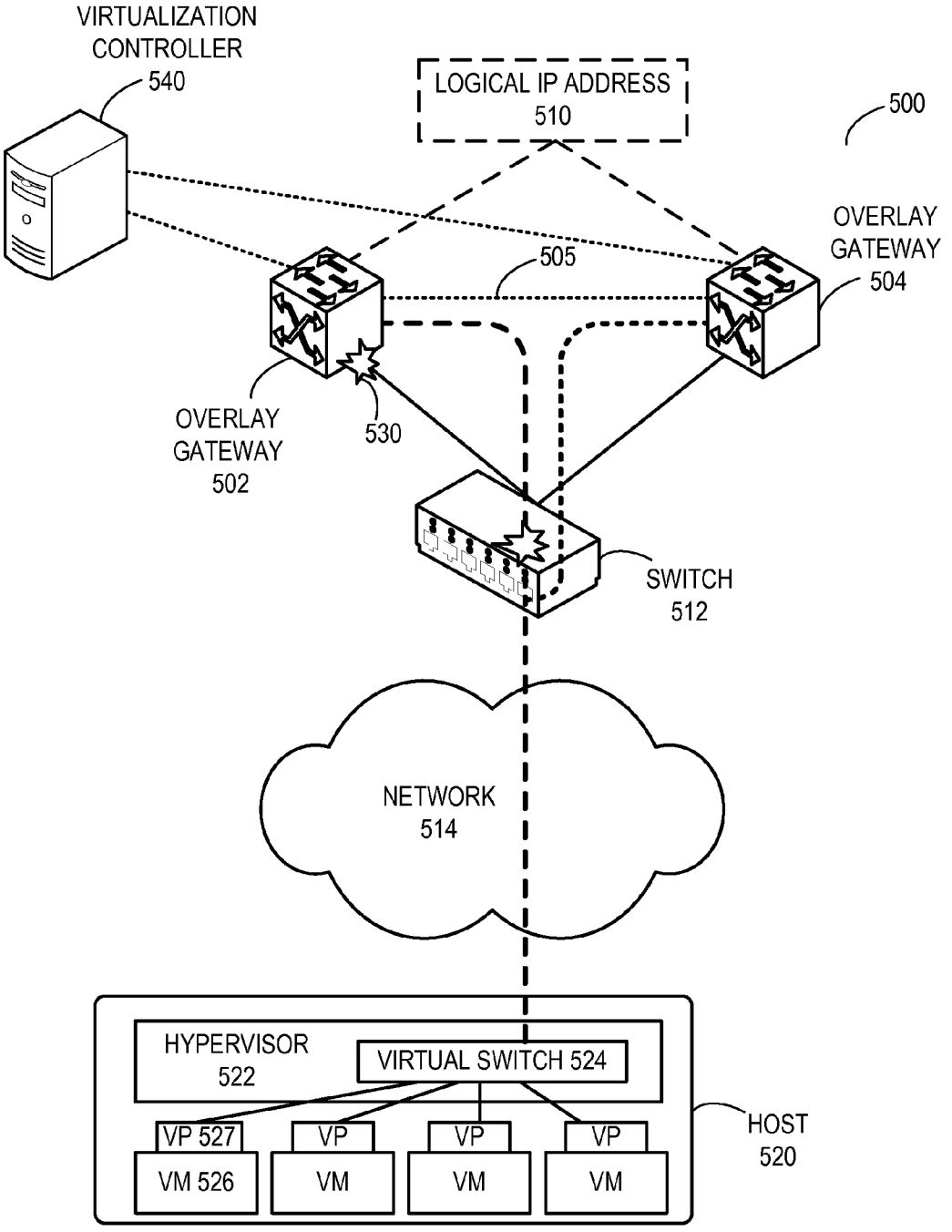

VIRTUALIZATION CONTROLLER
540
LOGICAL IP ADDRESS
510
500
OVERLAY GATEWAY
504
505
OVERLAY GATEWAY
502
530
SWITCH
512
NETWORK
514
HYPERVISOR
522
VIRTUAL SWITCH 524
VP 527
VP
VP
VP
VM 526
VM
VM
VM
HOST
520

FIG. 5A

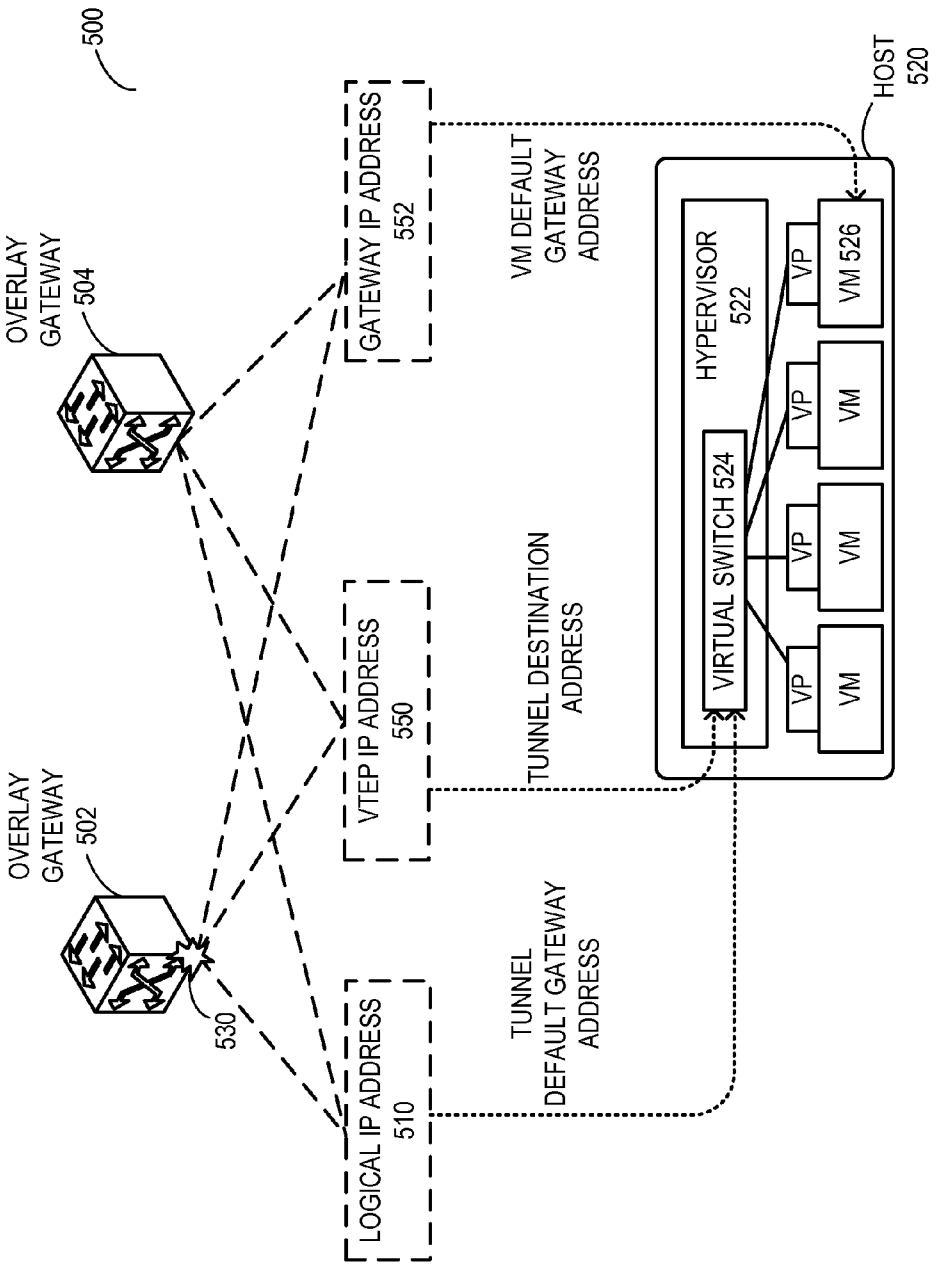

500
OVERLAY GATEWAY 504
OVERLAY GATEWAY 502
530
GATEWAY IP ADDRESS 552
VTEP IP ADDRESS 550
LOGICAL IP ADDRESS 510
VM DEFAULT GATEWAY ADDRESS
TUNNEL DESTINATION ADDRESS
TUNNEL DEFAULT GATEWAY ADDRESS
HOST 520
HYPERVISOR 522
VIRTUAL SWITCH 524
VP
VM
VM 526

FIG. 5B

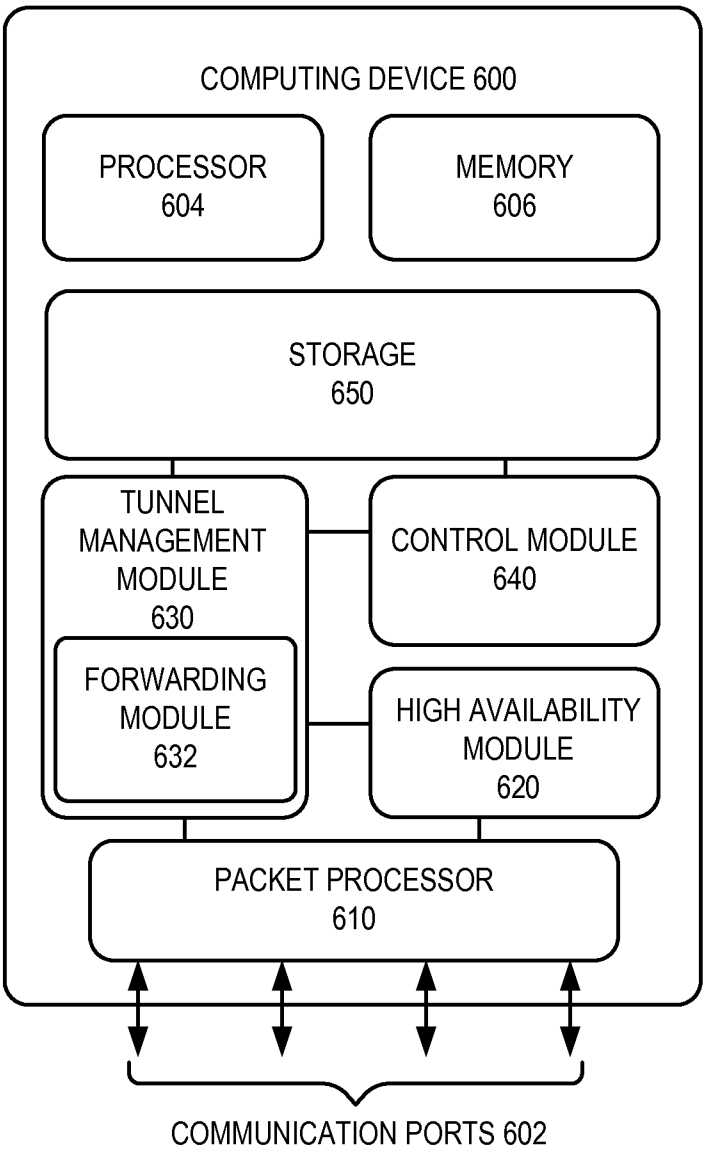

COMPUTING DEVICE 600
PROCESSOR 604
MEMORY 606
STORAGE 650
TUNNEL MANAGEMENT MODULE 630
FORWARDING MODULE 632
CONTROL MODULE 640
HIGH AVAILABILITY MODULE 620
PACKET PROCESSOR 610
COMMUNICATION PORTS 602

FIG. 6

LAYER-3 OVERLAY GATEWAYS

RELATED APPLICATIONS

[0001] This application claims the benefit of U.S. Provisional Application No. 61/650,943, Attorney Docket Number BRCD-3141.0.1.US.PSP, titled "L3 Overlay Gateways," by inventor Mani Prasad Kancherla, filed 23 May 2012, the disclosure of which is incorporated by reference herein.

[0002] The present disclosure is related to:

[0003] U.S. Patent Publication No. 2010/0246388, Attorney Docket Number BRCD-112-0439US, titled "Redundant Host Connection in a Routed Network," by inventors Somesh Gupta, Anoop Ghanwani, Phanidhar Koganti, and Shunjia Yu, filed 16 Mar. 2010; and

[0004] U.S. patent application Ser. No. 13/801,858, Attorney Docket Number BRCD-3135.1.US.NP, titled "Overlay Tunnel in a Fabric Switch," by inventor Phanidhar Koganti, filed 13 Mar. 2013;

[0005] the disclosures of which are incorporated by reference herein.

BACKGROUND

[0006] 1. Field

[0007] The present disclosure relates to network management. More specifically, the present disclosure relates to layer-3 overlays in a network.

[0008] 2. Related Art

[0009] The exponential growth of the Internet has made it a popular delivery medium for a variety of applications running on physical and virtual devices. Such applications have brought with them an increasing demand for bandwidth. As a result, equipment vendors race to build larger and faster switches with versatile capabilities, such as awareness of virtual machine migration, to move more traffic efficiently. However, the size of a switch cannot grow infinitely. It is limited by physical space, power consumption, and design complexity, to name a few factors. Furthermore, switches with higher capability are usually more complex and expensive. More importantly, because an overly large and complex system often does not provide economy of scale, simply increasing the size and capability of a switch may prove economically unviable due to the increased per-port cost.

[0010] As Internet traffic is becoming more diverse, virtual computing in a network is becoming progressively more important as a value proposition for network architects. The evolution of virtual computing has placed additional requirements on the network. However, conventional layer-2 network architecture often cannot readily accommodate the dynamic nature of virtual machines. For example, in conventional datacenter architecture, host machines can be interconnected by layer-2 (e.g., Ethernet) interconnects forming a layer-2 broadcast domain. Because of the physical reach limitation of a layer-2 broadcast domain, a datacenter is typically segmented into different layer-2 broadcast domains. Consequently, any communication to outside of a layer-2 broadcast domain is carried over layer-3 networks. As the locations of virtual machines become more mobile and dynamic, and data communication from the virtual machine becomes more diverse, it is often desirable that the network infrastructure can provide layer-3 network overlay tunnels to assist the data communication acrosslayer-2 broadcast domains.

[0011] While overlays bring many desirable features to a network, some issues remain unsolved in providing a logical subnet across layer-2 broadcast domains.

SUMMARY

[0012] One embodiment of the present invention provides a computing system. The computing system includes a processor and a computer-readable storage medium for storing instructions. Based on the instructions, the processor operates the computing system as an overlay gateway. The computing system initiates and terminates an overlay tunnel associated with a virtual machine. During operation, the computing system maps a virtual Internet Protocol (IP) address of the virtual machine to a second IP address used to terminate the overlay tunnel based on information received from a configuration system. The computing system then determines an output port for a data packet based on the second IP address. The data packet comprises an inner packet and the destination address of this inner packet corresponds to the virtual IP address.

[0013] In a variation on this embodiment, the mapping is also based on a virtual media access control (MAC) address corresponding to the virtual IP address.

[0014] In a variation on this embodiment, the computing system updates the mapping by mapping the virtual IP address of the virtual machine to a third IP address used to determine the output port for the data packet.

[0015] In a variation on this embodiment, the configuration system is one or more of: a virtualization controller, a network manager, and a shim device. The virtualization controller allocates the virtual machine to a hypervisor in a host machine and assigns the virtual IP addresses to the virtual machine. The network manager notifies the hypervisor regarding networking information. The shim device obtains networking information from the network manager.

[0016] In a further variation, the computing system also includes a shim control plane layer, which recognizes a plurality of virtualization controllers. A respective virtualization controller can correspond to a different virtualization mechanism.

[0017] In a variation on this embodiment, the computing system further comprises a shim data plane layer, which recognizes a plurality of overlay tunneling mechanisms.

[0018] In a further variation, a tunneling mechanism is associated with one or more of: a Virtual Extensible Local Area Network (VXLAN), a Generic Routing Encapsulation (GRE) protocol, a Network Virtualization using GRE (NVGRE) protocol, and an openvSwitch GRE protocol.

[0019] In a variation on this embodiment, the computing system identifies in a data packet a logical IP address associated with the computing system and a remote computing system, wherein the data packet is associated with the overlay tunnel.

[0020] In a further variation, the computing system determines an active status of the computing system in conjunction with the remote computing system. If the computing system is not active, the processor precludes the computing system from processing a packet associated with the logical IP address.

[0021] In a further variation, the computing system detects a failure of the remote computing system. Upon detecting the failure, the computing system starts processing a packet associated with the logical IP address.

[0022] In a further variation, the computing system identifies a tunnel termination IP address associated with the com-

puting system and a remote computing system, wherein the data packet is associated with the overlay tunnel. This tunnel termination IP address belongs to a subnet different from a subnet to which the logical IP address belongs.

BRIEF DESCRIPTION OF THE FIGURES

[0023] FIG. **1**A illustrates an exemplary virtualized network environment with an overlay gateway, in accordance with an embodiment of the present invention.

[0024] FIG. **1**B illustrates an exemplary virtualized network environment with a shim device assisting an overlay gateway, in accordance with an embodiment of the present invention.

[0025] FIG. **2** illustrates an exemplary overlay gateway supporting multiple control interfaces and tunneling mechanisms, in accordance with an embodiment of the present invention.

[0026] FIG. **3** illustrates an exemplary header format for a conventional packet and its tunnel encapsulation provided by an overlay gateway, in accordance with an embodiment of the present invention.

[0027] FIG. **4**A presents a flowchart illustrating the process of an overlay gateway obtaining a tunnel mapping from a virtualization controller, in accordance with an embodiment of the present invention.

[0028] FIG. **4**B presents a flowchart illustrating the process of an overlay gateway forwarding a received packet, in accordance with an embodiment of the present invention.

[0029] FIG. **4**C presents a flowchart illustrating the process of an overlay gateway forwarding a broadcast, unknown unicast, or multicast packet in a logical subnet, in accordance with an embodiment of the present invention.

[0030] FIG. **5**A illustrates an exemplary an overlay gateway with high availability, in accordance with an embodiment of the present invention.

[0031] FIG. **5**B illustrates an exemplary usage of multiple addresses of an overlay gateway with high availability, in accordance with an embodiment of the present invention.

[0032] FIG. **6** illustrates an exemplary computing system operating as an overlay gateway, in accordance with an embodiment of the present invention.

[0033] In the figures, like reference numerals refer to the same figure elements.

DETAILED DESCRIPTION

[0034] The following description is presented to enable any person skilled in the art to make and use the invention, and is provided in the context of a particular application and its requirements. Various modifications to the disclosed embodiments will be readily apparent to those skilled in the art, and the general principles defined herein may be applied to other embodiments and applications without departing from the spirit and scope of the present invention. Thus, the present invention is not limited to the embodiments shown, but is to be accorded the widest scope consistent with the claims.

Overview

[0035] In embodiments of the present invention, the problem of facilitating a logical sub network (subnet) beyond a physical subnet boundary is solved by incorporating an overlay gateway which provides virtual tunneling between physical subnets to form the logical subnet. This logical subnet logically couples the virtual machines belonging to the logical subnet but residing in host machines belonging to different physical subnets. In this way, the physical infrastructure of a network is often virtualized to accommodate multi-tenancy. One of the challenges in network virtualization is to bridge the physical network topology with the virtualized network subnet.

[0036] For example, a datacenter can include virtual machines associated with a customer (or tenant), running on hypervisors residing on different physical hosts. These virtual machines can be part of the same logical subnet. A virtualization controller of the datacenter typically allocates a respective virtual machine to a hypervisor in a host machine, and assigns the Media Access Control (MAC) and Internet Protocol (IP) addresses to the virtual machine. Typically, hypervisors or virtual switches use layer-3 virtual tunneling to allow virtual machines belonging to the same logical subnet to communicate. These hypervisors or virtual switches can be referred to as virtual tunnel end points (VTEPs). However, if the host machine of one of the virtual machines does not have the equivalent tunneling configuration, the other virtual machine may not be able to communicate via a virtual tunnel.

[0037] To solve this problem, an overlay gateway facilitates virtual tunneling to a respective VTEP (e.g., a hypervisor or a virtual switch) of a respective host machine. The overlay gateway, in turn, communicates with a destination, such as a physical server, which does not support the same tunneling mechanism. However, to associate the virtual tunnel with a virtual machine, the overlay gateway needs to identify a VTEP for the virtual machine. To facilitate this identification, the overlay gateway maintains a tunnel mapping between the MAC address of a virtual machine, and the corresponding VTEP address. Note that the tunnel mapping can also include the mapping between the MAC address and the IP address of the virtual machine.

[0038] In some embodiments, the overlay gateway communicates with the virtualization controller and obtains the tunnel mapping for a respective virtual machine. Whenever the mapping is updated, the overlay gateway obtains the updated mapping from the virtualization controller. In some embodiments, the overlay gateway can include two "shim layers." A shim layer operates as a communication interface between two devices. One shim layer operates as the control plane and interfaces with the virtualization controller for obtaining the mapping. The other shim layer operates as the data plane and facilitates tunnel encapsulation of packets to and from the virtual machines. As a result, the same overlay gateway can support multiple overlay networks comprising multiple virtualization and tunneling mechanisms.

[0039] In some embodiments, the interconnection in the datacenter includes an Ethernet fabric switch. In an Ethernet fabric switch, any number of switches coupled in an arbitrary topology may logically operate as a single switch. Any new switch may join or leave the fabric switch in "plug-and-play" mode without any manual configuration. A fabric switch appears as a single logical switch to an external device. In some further embodiments, the fabric switch is a Transparent Interconnection of Lots of Links (TRILL) network and a respective member switch of the fabric switch is a TRILL routing bridge (RBridge).

[0040] The term "external device" can refer to any device to which a VTEP cannot directly establish a tunnel. An external device can be a host, a server, a conventional layer-2 switch, a layer-3 router, or any other type of physical or virtual device.

Additionally, an external device can be coupled to other switches or hosts further away from a network. An external device can also be an aggregation point for a number of network devices to enter the network. The terms "device" and "machine" are used interchangeably.

[0041] The term "hypervisor" is used in a generic sense, and can refer to any virtual machine manager. Any software, firmware, or hardware that creates and runs virtual machines can be a "hypervisor." The term "virtual machine" also used in a generic sense and can refer to software implementation of a machine or device. Any virtual device which can execute a software program similar to a physical device can be a "virtual machine." A host external device on which a hypervisor runs one or more virtual machines can be referred to as a "host machine."

[0042] The term "tunnel" refers to a data communication where one or more networking protocols are encapsulated using another networking protocol. Although the present disclosure is presented using examples based on a layer-3 encapsulation of a layer-2 protocol, "tunnel" should not be interpreted as limiting embodiments of the present invention to layer-2 and layer-3 protocols. A "tunnel" can be established for any networking layer, sub-layer, or a combination of networking layers.

[0043] The term "packet" refers to a group of bits that can be transported together across a network. "Packet" should not be interpreted as limiting embodiments of the present invention to layer-3 networks. "Packet" can be replaced by other terminologies referring to a group of bits, such as "frame," "cell," or "datagram."

[0044] The term "switch" is used in a generic sense, and it can refer to any standalone or fabric switch operating in any network layer. "Switch" should not be interpreted as limiting embodiments of the present invention to layer-2 networks. Any device that can forward traffic to an external device or another switch can be referred to as a "switch." Examples of a "switch" include, but are not limited to, a layer-2 switch, a layer-3 router, a TRILL RBridge, or a fabric switch comprising a plurality of similar or heterogeneous smaller physical switches.

[0045] The term "RBridge" refers to routing bridges, which are bridges implementing the TRILL protocol as described in Internet Engineering Task Force (IETF) Request for Comments (RFC) "Routing Bridges (RBridges): Base Protocol Specification," available at http://tools.ietf.org/html/rfc6325, which is incorporated by reference herein. Embodiments of the present invention are not limited to application among RBridges. Other types of switches, routers, and forwarders can also be used.

[0046] The term "switch identifier" refers to a group of bits that can be used to identify a switch. If the switch is an RBridge, the switch identifier can be an "RBridge identifier." The TRILL standard uses "RBridge ID" to denote a 48-bit Intermediate-System-to-Intermediate-System (IS-IS) ID assigned to an RBridge, and "RBridge nickname" to denote a 16-bit value that serves as an abbreviation for the "RBridge ID." In this disclosure, "switch identifier" is used as a generic term, is not limited to any bit format, and can refer to any format that can identify a switch. The term "RBridge identifier" is used in a generic sense, is not limited to any bit format, and can refer to "RBridge ID," "RBridge nickname," or any other format that can identify an RBridge.

Network Architecture

[0047] FIG. **1**A illustrates an exemplary virtualized network environment with an overlay gateway, in accordance with an embodiment of the present invention. As illustrated in FIG. **1**A, a virtualized network environment **100**, which can be in a datacenter, includes a number of host machines **110** and **120** coupled to a layer-3 router **142** in network **140** via one or more hops. A number of virtual machines **102**, **104**, **106**, and **108** run on hypervisor **112** in host machine **110**. A respective virtual machine has a virtual port (VP, or virtual network interface card, VNIC). The virtual port of a respective virtual machine running on hypervisor **112** is logically coupled to a virtual switch **114**, which is provided by hypervisor **112**. Virtual switch **114** is responsible for dispatching outgoing and incoming traffic of virtual machines **102**, **104**, **106**, and **108**. Similarly, a number of virtual machines **122**, **124**, **126**, and **128** run on hypervisor **132** in host machine **120**. The virtual port of a respective virtual machine running on hypervisor **132** is logically coupled to a virtual switch **134** which is provided by a hypervisor **132**. Logically, virtual switches **114** and **134** function as aggregation points and couple router **142** via one or more links.

[0048] Also included are a virtualization controller **162** and a network manager **164**. Virtualization controller **162**, often based on an instruction from a network administrator, allocates a respective virtual machine to a hypervisor in a host machine, and assigns virtual MAC and IP addresses to the virtual machine. For example, virtualization controller **162** allocates virtual machine **122** to hypervisor **132** in host machine **120**, and assigns virtual MAC and IP addresses to virtual port **123** of virtual machine **122**. An Ethernet frame generated by virtual machine **122** has the virtual MAC of virtual port **123** as its source address. In this example, host machines **110** and **120** are parts of two different physical subnets in network **140**. However, virtual machines **102** and **104** in host machine **110** and virtual machines **122** and **124** in host machine **120** are part of logical subnet **182**. Similarly, virtual machines **106** and **108** in host machine **110** and virtual machines **126** and **128** in host machine **120** are part of the same logical subnet **184**. Usually a logical subnet corresponds to a tenant.

[0049] In some embodiments, virtual switches **114** and **134** are logically coupled to network manager **164**, which provides virtual switches **114** and **134** with networking information required to communicate with each other. For example, because virtual machines **102** and **122** are part of the same logical subnet, virtual machine **102** can communicate with virtual machine **122** via layer-2. However, these virtual machines reside on host machines in different physical subnets. Hence, virtual switch **114** needs to know that virtual machine **122** is logically coupled to virtual switch **134** (e.g., virtual switch **134** is the VTEP for virtual machine **122**). By providing this networking information, network manager **164** enable virtual switches **114** and **134** to operate as VTEPs for virtual machines **102** and **122**, respectively, and use layer-3 virtual tunneling to facilitate communication between these virtual machines. However, because an external device, such as physical server **144**, may not have the equivalent tunneling configuration, a virtual machine, such virtual machine **122**, may not be able to communicate with server **144** via a virtual tunnel.

[0050] In order to communicate with server **144**, an overlay gateway **150** allows a respective VTEP to establish virtual tunneling via network **140**. Overlay gateway **150**, in turn,

communicates with physical server **144**. During operation, virtual machine **122** sends a packet to virtual server **144** via logically coupled virtual switch **134**. Virtual switch **134** encapsulates the packet in a tunnel header and forwards the encapsulated packet to gateway **150**. Upon receiving the encapsulated packet, overlay gateway **150** removes the tunnel encapsulation and forwards the packet to server **144** based on the destination address of the packet. When server **144** sends a packet back to virtual machine **122**, overlay gateway **150** receives the packet. However, to efficiently forward this packet to virtual machine **122**, overlay gateway **150** needs to identify the virtual switch (i.e., the VTEP) to which virtual machine **122** is logically coupled. To facilitate the identification, overlay gateway **150** maintains a tunnel mapping between the MAC address of virtual machine **122**, and the corresponding VTEP address of virtual switch **134**. Note that the tunnel mapping can also include the mapping between the MAC address and the IP address of virtual machine **122**.

[0051] For example, overlay gateway **150** can obtain such mapping for virtual machine **122** by sending a broadcast (e.g., an Address Resolution Protocol (ARP)) query with virtual machine **122**'s IP address to obtain the corresponding VTEP address. However, in a large datacenter with a large number of virtual machines, sending a large number of broadcast queries can be inefficient. In some embodiments, overlay gateway **150** communicates with virtualization controller **162** and obtains the tunnel mapping for a respective virtual machine. For virtual machine **122**, such mapping can include an identifier to host machine **120** (e.g., a MAC address of a physical network interface of host machine **120**), the MAC address of virtual port **123**, and the corresponding VTEP address of virtual switch **134**. If the mapping is updated (e.g., due to a virtual machine migration) in virtualization controller **162**, overlay gateway **150** obtains the updated tunnel mapping from virtualization controller **162**.

[0052] Based on the obtained tunnel mapping, overlay gateway **150** identifies virtual switch **134** as the VTEP for virtual machine **122**, encapsulates the packet from server **144** in a tunnel header, and forwards the encapsulated packet to virtual switch **134**. Upon receiving the encapsulated packet, virtual switch **134** removes the encapsulation and provides the packet to virtual machine **122**. Suppose that virtualization controller **162** migrates virtual machine **122** to host machine **110**. Consequently, the tunnel mapping for virtual machine **122** is updated in virtualization controller **162**. The updated mapping for virtual machine **122** includes an identifier to host machine **110** and the corresponding VTEP address of virtual switch **114**. Overlay gateway **150** can receive an update message comprising the updated tunnel mapping for from virtualization controller **162**.

[0053] In some embodiments, overlay gateway **150** can obtain tunnel mapping from network manager **164**. FIG. **1**B illustrates an exemplary virtualized network environment with a shim device assisting an overlay gateway in conjunction with the example in FIG. **1**A, in accordance with an embodiment of the present invention. Network manager **164** provides a respective virtual switch with networking information required to communicate with each other. To obtain information from network manager **164**, virtualized network environment **100** includes a shim device **172**, which runs a virtual switch **174**. This virtual switch **174** is logically coupled to network manager **164**, which considers virtual switch **174** as another virtual switch in a hypervisor. Consequently, network manager **164** provides virtual switch **174** with networking information required to communicate with virtual machines logically coupled to other virtual switches. For virtual machine **122**, such information can include an identifier to host machine **120** (e.g., a MAC address of a physical network interface of host machine **120**), the MAC address of virtual port **123**, and the corresponding VTEP address of virtual switch **134**.

[0054] Shim device **172** can include a shim layer **176** which communicates with overlay gateway **150**. Overlay gateway **150** obtains the networking information via shim layer **176** and constructs the tunnel mapping. Note that networking information may not include the virtual MAC addresses of the virtual machines. Under such a scenario, overlay gateway **150** uses broadcast queries using the virtual IP addresses of the virtual machines to obtain the corresponding virtual MAC addresses. In some embodiments, shim layer **176** can reside on network manager **164** (denoted with dotted lines) and provide networking information to overlay **150**, thereby bypassing the shim device **172**. However, integrating shim layer **176** with network manager **164** creates additional memory and processing requirements in the physical hardware and may degrade the performance of network manager **164**.

[0055] FIG. **2** illustrates an exemplary overlay gateway supporting multiple control interfaces and tunneling mechanisms, in accordance with an embodiment of the present invention. Overlay gateway **150** can include two shim layers. One shim layer operates as the control plane **220** and interfaces with virtualization controller **162** for obtaining the mapping. The other shim layer operates as the data plane **210** and facilitates tunnel encapsulation to packets to and from the virtual machines. As a result, overlay gateway **150** can support multiple overlay networks comprising multiple virtualization and tunneling mechanisms.

[0056] Control plane **220** includes a number of control interfaces **222**, **224**, and **226**. A respective control interface is capable of communicate with a different virtualization manager. Examples of a control interface include, but are not limited to, interface for VMWareNSX, interface for Microsoft System Center, and interface for OpenStack. For example, control interface **222** can communicate with OpenStack while control interface **224** can communicate with Microsoft System Center. Data plane **210** supports a number of tunneling mechanism **212**, **214**, and **216**. A respective tunneling mechanism is capable of establishing a different overlay tunnel by facilitating a corresponding tunnel encapsulation (i.e., operating as a VTEP for different tunneling mechanisms). Examples of a tunneling mechanism include, but are not limited to, Virtual Extensible Local Area Network (VXLAN), Generic Routing Encapsulation (GRE), and its variations, such as Network Virtualization using GRE (NVGRE) and openvSwitch GRE. For example, tunneling mechanism **212** can represent VXLAN while tunneling mechanism **214** can represent GRE.

[0057] With the support of different interfaces and tunneling mechanism, if a datacenter includes a plurality of virtualized network environments from different vendors, the same overlay gateway **150** can serve these environments. In the example in FIG. **1**A, if virtual switch **114** supports VXLAN while virtual switch **134** supports GRE, gateway **150** can use tunneling mechanism **212** and **214**, respectively, to provide tunnel-encapsulated overlay with virtual switches **114** and **134**, respectively. If virtualization controller **162** runs OpenStack, overlay gateway **150** can use interface **222** to

obtain the tunnel mapping. Similarly, if virtualization controller **162** is a Microsoft System Center, overlay gateway **150** can use interface **224** to obtain the tunnel mapping.

Packet Format

[0058] FIG. **3** illustrates an exemplary header format for a conventional packet and its tunnel encapsulation provided by an overlay gateway, in accordance with an embodiment of the present invention. In this example, a conventional Ethernet packet **300** typically includes a payload **308** and an Ethernet header **310**. Typically, payload **308** can include an IP packet which includes an IP header **320**. IP header **320** includes an IP destination address (DA) **312** and an IP source address (SA) **314**. Ethernet header **310** includes a MAC DA **302**, a MAC SA **304**, and optionally a virtual local area network (VLAN) tag **306**.

[0059] Suppose that packet **300** is a packet from server **144** to virtual machine **122** in FIG. **1**A. In one embodiment, overlay gateway **150** encapsulates conventional packet **300** into an encapsulated packet **350** based on the tunnel mapping. Encapsulated packet **350** typically includes an encapsulation header **360**, which corresponds to an encapsulation mechanism, as described in conjunction with FIG. **2**. Encapsulation header **360** contains an encapsulation DA **352**, which corresponds to the VTEP IP address of virtual switch **134**, and an encapsulation SA **354**, which corresponds to the IP address of overlay gateway **150**. In the example in FIG. **1**A, encapsulated packet **350** is forwarded via network **140** based on encapsulation DA **352**. In some embodiments, encapsulation header **360** also includes a tenant identifier **356**, which uniquely identifies a tenant in virtualized network environment **100**. For example, if encapsulation header **360** corresponds to a tunnel for virtual machine **122**, tenant identifier **356** identifies the tenant to which virtual machine **122** belong. In this way, gateway **150** can maintain tenant isolation by using separate tunnel encapsulation for packets for different tenants.

[0060] Typically, an upper layer application in server **144** generates an IP packet destined for virtual machine **122**, using the virtual IP address of virtual machine **122** as IP DA address **312**, and the physical IP address of server **144** as IP SA address **314**. This IP packet becomes payload **308**. The layer-2 in server **144** then generates Ethernet header **310** to encapsulate payload **308**. If server **144** and virtual machine **122** reside within the same logical subnet, MAC DA **302** of Ethernet header **310** is assigned the MAC address of virtual machine **122**. MAC SA **304** of Ethernet header **310** is server **144**'s MAC address. Server **144** then sends Ethernet packet **300** to virtual machine **122** via overlay gateway **150**.

[0061] When overlay gateway **150** receives Ethernet packet **300** from server **144**, overlay gateway **150** inspects Ethernet header **310**, and optionally IP header **308** and its payload (e.g., the layer-**4** header). Based on this information, overlay gateway **150** determines that Ethernet packet **300** is destined to virtual machine **122** within the same logical subnet. Subsequently, overlay gateway **150** assembles the encapsulation header **360** (corresponding to an encapsulation mechanism). Encapsulation DA **352** of encapsulation header **360** is assigned the IP address of VTEP IP address of virtual switch **134**. Encapsulation SA **354** of encapsulation header **360** is overlay gateway **150**'s IP address. Note that overlay gateway **150**'s IP address can be a logical IP address as well. Overlay gateway **150** then attaches tenant identifier **356** and forwards encapsulated packet **350** to VTEP virtual switch **134**. Upon receiving packet **350**, virtual switch **134** removes encapsulation header **360**, examines Ethernet header **310** in decapsulated packet **300**, and provides decapsulated packet **300** to virtual machine **122**.

Operations

[0062] In the example in FIG. **1**A, overlay gateway **150** communicates with virtualization controller **162** to obtain a tunnel mapping and forwards received packets via tunnel encapsulation based on the obtained tunnel mapping. FIG. **4**A presents a flowchart illustrating the process of an overlay gateway obtaining a tunnel mapping from a virtualization controller, in accordance with an embodiment of the present invention. During operation, the overlay gateway identifies the virtualization controller (operation **402**) and identifies the local control interface corresponding to the identified virtualization controller (operation **404**), as described in conjunction with FIG. **2**. The overlay gateway then requests information from the virtualization controller via the identified control interface (operation **406**). In response, the virtual controller sends an information message comprising the relevant tunnel mapping.

[0063] The overlay gateway receives this information message (operation **408**) and extracts the tunnel mapping from the information message (operation **410**). This tunnel mapping maps the MAC address of a respective virtual machine to a corresponding VTEP address. Note that the tunnel mapping can also include the mapping between the MAC address and the IP address of the virtual machine. The overlay gateway then locally stores the extracted tunnel mapping (operation **412**). The overlay gateway can also obtain tenant information for a respective virtual machine from the virtualization controller (operation **414**) and associates the tenant with the corresponding virtual machine (operation **416**). In some embodiments, the overlay gateway can obtain the tenant information as a part of the tunnel mapping.

[0064] FIG. **4**B presents a flowchart illustrating the process of an overlay gateway forwarding a received packet, in accordance with an embodiment of the present invention. Upon receiving a packet (operation **452**), the overlay gateway checks whether the packet is encapsulated for a local VTEP (i.e., destined to a VTEP associated with the gateway) (operation **454**), as described in conjunction with FIG. **3**. If the packet is encapsulated and destined to a VTEP associated with the gateway, the overlay gateway decapsulates the tunnel encapsulation of the packet using VTEP IP address (operation **456**). If the packet received by the overlay gateway is not encapsulated (operation **454**) or the encapsulation has been decapsulated with overlay gateway VTEP IP address (operation **456**), the gateway checks whether the destination of the packet is reachable via a tunnel (e.g., is destined to a virtual machine via the tunnel) (operation **458**). When the tunnel encapsulation has been removed from the packet (operation **456**) and the destination is not reachable via tunnel (operation **458**), the overlay gateway performs a lookup based on the IP address of the IP header (operation **460**) and forwards the packet based on the lookup (operation **462**). Note that if the packet has been encapsulated, the IP header refers to the inner IP header.

[0065] If the packet received by the overlay gateway is not encapsulated for a local VTEP (operation **454**) or has been decapsulated with overlay gateway VTEP IP address (operation **456**), and the destination is reachable via a tunnel (operation **458**), the overlay gateway identifies the VTEP address,

and the tenant of the destination from tunnel mapping (operation **470**). The overlay gateway can identify the destination by examining the destination IP and/or MAC address of the packet. The overlay gateway then encapsulates the packet in tunnel encapsulation ensuring tenant separation (operation **472**). In some embodiments, the overlay gateway uses separate tunnels for separate tenants and can include an identifier to a tenant in the encapsulation header. The overlay gateway assigns the identified VTEP IP address as the destination IP address and the IP address of the overlay gateway as the source IP address of in the encapsulation header (operation **474**). Note that if the encapsulation mechanism is based on a layer other than layer-3, the overlay gateway can use VTEP and gateway addresses of the corresponding layer. The overlay gateway then forwards the encapsulated packet toward the VTEP (operation **476**).

[0066] Typically broadcast, unknown unicast, or multicast traffic (which can be referred to as "BUM" traffic) is distributed to multiple recipients. For ease of deployment, logical switches typically make multiple copies of a packet belonging to such traffic and individually unicast the packets based on tunnel encapsulation towards the virtual switches associated with the same logical subnet. This often leads to inefficient usage of processing capability of the hypervisors, especially in a large scale deployment. To solve this problem, an overlay gateway can facilitate efficient distribution of such traffic. A virtual switch can simply encapsulate the "BUM" packet in tunnel encapsulation and forward the packet to the overlay gateway. The overlay gateway, in turn, forwards the packet in the logical subnet.

[0067] FIG. **4**C presents a flowchart illustrating the process of an overlay gateway forwarding a broadcast, unknown unicast, or multicast packet in a logical subnet, in accordance with an embodiment of the present invention. During operation, the overlay gateway receives a tunnel encapsulated packet belonging to broadcast, unknown unicast, or multicast traffic (operation **482**). The overlay gateway removes the tunnel encapsulation (operation **484**) and identifies the interface(s) associated with the logical subnet of the packet from tunnel mapping (operation **486**). Because the virtual switch from which the overlay gateway has received the packet is responsible for distributing the packet to the member virtual machines of the logical subnet, the overlay gateway does not forward the packet toward that virtual switch.

[0068] The overlay gateway then makes multiple copies of the packet corresponding to the number of identified interface (s) (operation **488**) and encapsulate a respective copy of the packet in tunnel encapsulation for respective identified interface (operation **490**). Because the overlay gateway supports multiple tunneling mechanisms, as described in conjunction with FIG. **2**, the overlay gateway can still distribute the packet if different virtual switches associated with the logical subnet support different tunneling mechanisms. The overlay gateway assigns a respective identified VTEP IP address as the destination IP address, and the IP address of the overlay gateway as the source IP address of in a respective encapsulation header (operation **492**). Note that if the encapsulation mechanism is based on a layer other than layer-3, the overlay gateway can use VTEP and gateway addresses of the corresponding layer. The overlay gateway then forwards respective copy of the encapsulated packet toward the corresponding VTEP via a corresponding identified interface (operation **494**).

High Availability

[0069] In the example in FIG. **1**A, if overlay gateway **150** fails or encounters a link failure, overlay gateway **150** can no longer operate as the gateway. Hence, providing high-availability to overlay gateway **150** is essential. FIG. **5**A illustrates an exemplary an overlay gateway with high availability, in accordance with an embodiment of the present invention. As illustrated in FIG. **5**A, a virtualized network environment **500**, which can be in a datacenter, includes a host machine **520** coupled to a switch **512** in network **514** via one or more hops. A number of virtual machines run on hypervisor **522** in host machine **520**. A respective virtual machine has a virtual port. The virtual port of a respective virtual machine running on hypervisor **522** is logically coupled to a virtual switch **524** which is provided by hypervisor **522**. Also included is a virtualization controller **540**, which allocates a respective virtual machine to a hypervisor in a host machine, and assigns virtual MAC and IP addresses to the virtual machine.

[0070] Virtualized network environment **500** also includes overlay gateways **502** and **504**, coupled to each other via logical link **505**. Logical link **505** can include one or more physical links, interconnected via layer-2 and/or layer-3. In this example, overlay gateway **502** remains actively operational while overlay gateway **504** operates as a standby gateway for overlay gateway **502**. In some embodiments, overlay gateway **502** communicates with virtualization controller **540** and obtains the corresponding tunnel mapping for a respective virtual machine. In some embodiments, upon obtaining the tunnel mapping, overlay gateway sends an information message comprising the tunnel mapping to overlay gateway **504**. In this way, both overlay gateways **502** and **504** can have the same tunnel mapping. If the mapping is updated (e.g., due to a virtual machine migration) in virtualization controller **540**, as described in conjunction with FIG. **1**A, overlay gateway **502** obtains the updated tunnel mapping from virtualization controller **540** and sends an information message comprising the updated tunnel mapping to overlay gateway **504**. In some embodiments, overlay gateways **502** and **504** individually obtain the tunnel mapping from virtualization controller **540**.

[0071] Overlay gateways **502** and **504** can share a logical IP address **510**. While operational, active overlay gateway **502** uses logical IP address **510** as the VTEP address while standby overlay gateway **504** suppresses the operations (e.g., ARP response) associated with logical IP address **510**. As a result, only overlay gateway **502** responds to any ARP query for logical IP address **510**. Consequently, switch **512** only learns the MAC address of overlay gateway **502** and forwards all subsequent packets to overlay gateway **502**.

[0072] During regular operation, overlay gateway **502** facilitate virtual tunneling to logical switch **524**, which is a VTEP for virtual machine **526** in host machine **520**, via network **514**. Upon obtaining a packet from virtual machine **526**, virtual switch **524** encapsulates the packet in a tunnel header and forwards the encapsulated packet toward overlay gateway **502**. Because switch **512** has only learned the MAC address of overlay gateway **502**, switch **512** forwards the packet to overlay gateway **502**. Upon receiving the encapsulated packet, overlay gateway **502** removes the tunnel encapsulation and forwards the packet toward the destination address of the packet.

[0073] Overlay gateways **502** and **504** can exchange "keep alive" messages via link **505** to notify each other regarding their active status. Suppose that failure **530** causes a link or

device failure which makes overlay gateway **502** unavailable. Overlay gateway **504** detects failure **530** by not receiving the keep alive message from overlay gateway **502** for a predetermined period of time and assumes the operations associated with logical IP address **510**. Due to the failure of overlay gateway **502**, switch **512** typically clears the learned MAC address of gateway **502**. In some embodiments, overlay gateway **504** sends a gratuitous ARP response message, which allows switch **512** to learn the MAC address of overlay gateway **504** and update its forwarding table accordingly. Based on the updated forwarding table, switch **512** forwards the subsequent packets from virtual machine **526** to overlay gateway **504**. Upon receiving the encapsulated packet, overlay gateway **504** removes the tunnel encapsulation and forwards the packet toward the destination address of the packet.

[0074] FIG. **5**B illustrates an exemplary usage of multiple addresses of an overlay gateway with high availability, in accordance with an embodiment of the present invention. In this example, overlay gateways **502** and **504** can have different IP addresses for different purposes. For example, besides logical IP address **510**, overlay gateways **502** and **504** can have VTEP IP address **550** and gateway IP address **552**. Virtual machine **524** uses gateway IP address **552** as the default gateway address. Hence, if virtual machine **526** needs to send a packet outside of its logical subnet, virtual machine **526** sends ARP request to gateway IP address **552**. In some embodiments, overlay gateways **502** and **504** can have a respective gateway IP address for a respective logical subnet to operate as the default gateway for the logical subnet.

[0075] Virtual switch **524** uses logical IP address **510** as the default gateway address and VTEP IP address **550** as the default tunnel destination address. VTEP IP address **550** can be outside of the logical subnet(s) associated with the virtual machines in host machine **520**. For sending a tunnel-encapsulated packet to VTEP IP address **550**, virtual switch **524** sends ARP request to logical IP address **510**. Because all encapsulated packets destined to VTEP IP address **550** is directed toward logical IP address **510**, overlay gateway **502** receives the packet and takes appropriate action. In this way, a single VTEP IP address **550**, which can be outside of the logical subnets associated with a respective virtual machine, operate as the tunnel destination address for all logical subnets.

[0076] Upon detecting failure **530**, overlay gateway **504** assumes the operations associated with logical IP address **510**, as described in conjunction with FIG. **5**A. Because all encapsulated packets destined to VTEP IP address **550** are directed toward logical IP address **510**, the encapsulated packets are directed toward overlay gateway **504**. Hence, providing high availability to only logical IP address **510** is sufficient to ensure high availability to tunnel encapsulated packets destined to VTEP IP address **550**. However, if VTEP IP address **550** is in the logical subnet of virtual machine **526**, virtual machine **526** directly sends packets to VTEP IP address **550**. Under such a scenario, providing high availability to VTEP IP address **550** is also necessary.

Exemplary Overlay Gateway

[0077] FIG. **6** illustrates an exemplary computing system operating as an overlay gateway, in accordance with an embodiment of the present invention. In this example, a computing system **600** includes a general purpose processor **604**, a memory **606**, a number of communication ports **602**, a packet processor **610**, a tunnel management module **630**, a forwarding module **632**, a control module **640**, a high availability module **620**, and a storage **650**. Processor **604** executes instructions stored in memory **606** to operate computing system **600** as an overlay gateway, which initiates or terminates an overlay tunnel associated with a virtual machine.

[0078] During operation, one of the communication ports **602** receives a packet from a configuration system. This configuration system can be one or more of: a virtualization controller, a network manager, and a shim device. Packet processor **610**, in conjunction with control module **640**, extracts a tunnel mapping from the received packet. This tunnel mapping maps a virtual IP address and/or a MAC address of the virtual machine to a VTEP IP address. Control module **640** stores the tunnel mapping in storage **650** and loads in memory **606** during operation. When the mapping is updated, control module **640** also updates the mapping, as described in conjunction with FIG. **1**A. Tunnel management module **630** recognizes a plurality of overlay tunneling mechanisms. When a data packet destined to the virtual machine is received, forwarding module **632** obtain the VTEP IP address from the mapping for the virtual machine, encapsulates the packet based on a recognized tunneling mechanism, and determines an output port among the communication ports **602** for the data packet based on the VTEP IP address.

[0079] High availability module **620** associates computing system **600** with a logical IP address, which is also associated with a remote computing system, as described in conjunction with FIG. **5**A. High availability module **620** determines whether computing system **600** is an active or a standby overlay gateway. If computing system **600** is a standby overlay gateway, processor **604** precludes packet processor **610** from processing a packet associated with the logical IP address. When high availability module **620** detects a failure of the remote computing system, packet processor **610** starts processing packets associated with the logical IP address. In some embodiments, high availability module **620** also associates computing system **600** with a VTEP address, which belongs to a subnet different from a subnet to which the logical IP address belongs, as described in conjunction with FIG. **5**B.

[0080] Note that the above-mentioned modules can be implemented in hardware as well as in software. In one embodiment, these modules can be embodied in computer-executable instructions stored in a memory which is coupled to one or more processors in computing system **600**. When executed, these instructions cause the processor(s) to perform the aforementioned functions.

[0081] In summary, embodiments of the present invention provide a computing system and a method for facilitating layer-3 overlay tunneling. In one embodiment, the computing system includes a processor and a computer-readable storage medium for storing instructions. Based on the instructions, the processor operates the computing system as an overlay gateway. The computing system initiates and terminates an overlay tunnel associated with a virtual machine. During operation, the computing system maps a virtual Internet Protocol (IP) address of the virtual machine to a second IP address used to terminate the overlay tunnel based on information received from a configuration system. The computing system then determines an output port for a data packet based on the second IP address. The data packet comprises an inner packet and the destination address of this inner packet corresponds to the virtual IP address.

[0082] The methods and processes described herein can be embodied as code and/or data, which can be stored in a computer-readable non-transitory storage medium. When a computer system reads and executes the code and/or data stored on the computer-readable non-transitory storage medium, the computer system performs the methods and processes embodied as data structures and code and stored within the medium.

[0083] The methods and processes described herein can be executed by and/or included in hardware modules or apparatus. These modules or apparatus may include, but are not limited to, an application-specific integrated circuit (ASIC) chip, a field-programmable gate array (FPGA), a dedicated or shared processor that executes a particular software module or a piece of code at a particular time, and/or other programmable-logic devices now known or later developed. When the hardware modules or apparatus are activated, they perform the methods and processes included within them.

[0084] The foregoing descriptions of embodiments of the present invention have been presented only for purposes of illustration and description. They are not intended to be exhaustive or to limit this disclosure. Accordingly, many modifications and variations will be apparent to practitioners skilled in the art. The scope of the present invention is defined by the appended claims.

What is claimed is:

1. A computing system, comprising:

a processor;

a computer-readable storage medium storing instructions which when executed by the processor causes the processor to perform a method, the method comprising:

initiating or terminating an overlay tunnel associated with a virtual machine;

mapping a virtual Internet Protocol (IP) address of the virtual machine to a second IP address used to terminate the overlay tunnel based on information received from a configuration system; and

determining an output port for a data packet comprising an inner packet based on the second IP address, wherein the destination address of the inner packet corresponds to the virtual IP address.

2. The computing system of claim **1**, wherein the mapping is further based on a virtual media access control (MAC) address corresponding to the virtual IP address.

3. The computing system of claim **1**, wherein the method further comprises updating the mapping which maps the virtual IP address of the virtual machine to a third IP address used to determine the output port for the data packet.

4. The computing system of claim **1**, wherein the configuration system is one or more of:

a virtualization controller which allocates the virtual machine to a hypervisor in a host machine and assigns the virtual IP addresses to the virtual machine;

a network manager which notifies the hypervisor regarding networking information; and

a shim device which obtains networking information from the network manager.

5. The computing system of claim **4**, further comprising a shim control plane layer operable to recognize a plurality of virtualization controllers, wherein a respective virtualization controller corresponds to a different virtualization mechanism.

6. The computing system of claim **1**, further comprising a shim data plane layer operable to recognize a plurality of overlay tunneling mechanisms.

7. The computing system of claim **6**, wherein a tunneling mechanism is associated with one or more of:

a Virtual Extensible Local Area Network (VXLAN);

a Generic Routing Encapsulation (GRE) protocol;

a Network Virtualization using GRE (NVGRE) protocol; and

an openvSwitch GRE protocol.

8. The computing system of claim **1**, wherein the method further comprises identifying in a data packet a logical IP address associated with the computing system and a remote computing system, wherein the data packet is associated with the overlay tunnel.

9. The computing system of claim **8**, wherein the method further comprises:

determining an active status of the computing system in conjunction with the remote computing system; and

precluding the computing system from processing a packet associated with the logical IP address in response to detecting the computing system not being active.

10. The computing system of claim **9**, wherein the method further comprises:

detecting a failure of the remote computing system; and

processing a packet associated with the logical IP address in response to detecting the failure.

11. The computing system of claim **8**, wherein the method further comprises:

identifying a tunnel termination IP address associated with the computing system and a remote computing system, wherein the data packet is associated with the overlay tunnel; and

wherein the tunnel termination IP address belongs to a subnet different from a subnet to which the logical IP address belongs.

12. A method, comprising:

initiating or terminating, by a computing system, an overlay tunnel associated with a virtual machine;

mapping a virtual Internet Protocol (IP) address of the virtual machine to a second IP address used to terminate the overlay tunnel based on information received from a configuration system; and

determining an output port for a data packet comprising an inner packet based on the second IP address, wherein the destination address of the inner packet corresponds to the virtual IP address.

13. The method of claim **12**, wherein the mapping is further based on a virtual media access control (MAC) address corresponding to the virtual IP address.

14. The method of claim **12**, further comprising updating the mapping which maps the virtual IP address of the virtual machine to a third IP address used to determine the output port for the data packet.

15. The method of claim **12**, wherein the configuration system is one or more of:

a virtualization controller which allocates the virtual machine to a hypervisor in a host machine and assigns the virtual IP addresses to the virtual machine;

a network manager which notifies the hypervisor regarding networking information; and

a shim device which obtains networking information from the network manager.

16. The method of claim **15**, further comprising recognizing a plurality of virtualization controllers, wherein a respective virtualization controller corresponds to a different virtualization mechanism.

17. The method of claim **12**, further comprising recognizing a plurality of overlay tunneling mechanisms.

18. The method of claim **17**, wherein a tunneling mechanism is associated with one or more of:

a Virtual Extensible Local Area Network (VXLAN);

a Generic Routing Encapsulation (GRE) protocol;

a Network Virtualization using GRE (NVGRE) protocol; and

an openvSwitch GRE protocol.

19. The method of claim **12**, further comprising identifying in a data packet a logical IP address associated with the computing system and a remote computing system, wherein the data packet is associated with the overlay tunnel.

20. The method claim **19**, further comprising:

determining an active status of the computing system in conjunction with the remote computing system; and

precluding the computing system from processing a packet associated with the logical IP address in response to detecting the computing system not being active.

21. The method of claim **20**, further comprising:

detecting a failure of the remote computing system; and

processing a packet associated with the logical IP address in response to detecting the failure.

22. The method of claim **19**, further comprising:

identifying a tunnel termination IP address associated with the computing system and a remote computing system, wherein the data packet is associated with the overlay tunnel; and

wherein the tunnel termination IP address belongs to a subnet different from a subnet to which the logical IP address belongs.

23. A computing means, comprising:

a tunneling means for initiating or terminating an overlay tunnel associated with a virtual machine;

a mapping means for mapping a virtual Internet Protocol (IP) address of the virtual machine to a second IP address used to terminate the overlay tunnel based on information received from a configuration system; and

a forwarding means for determining an output port for a data packet comprising an inner packet based on the second IP address, wherein the destination address of the inner packet corresponds to the virtual IP address.

* * * * *