Internet Engineering Task Force
Internet Draft
Intended Status: Experimental
Expires: July 22, 2014

M. Mahalingam
Storvisor
D. Dutt
Cumulus Networks
K. Duda
Arista
P. Agarwal
Broadcom
L. Kreeger
Cisco
T. Sridhar
VMware
M. Bursell
Citrix
C. Wright
Red Hat

January 23, 2014

VXLAN: A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks

draft-mahalingam-dutt-dcops-vxlan-07.txt

Status of this Memo

Copyright Notice

Abstract

This document describes Virtual eXtensible Local Area Network (VXLAN), which is used to address the need for overlay networks within virtualized data centers accommodating multiple tenants. The scheme and the related protocols can be used in cloud service provider and enterprise data center networks. This memo documents the deployed VXLAN protocol for the benefit of the IETF community. The IETF consensus on this RFC represents consensus to publish this memo, and not consensus on the text itself.

Table of Contents

1. Introduction

Server virtualization has placed increased demands on the physical network infrastructure. A physical server now has multiple virtual machines (VMs) each with its own MAC address. This requires larger MAC address tables in the switched Ethernet network due to potential attachment of and communication among hundreds of thousands of VMs.

In the case when the VMs in a data center are grouped according to their Virtual LAN (VLAN, one might need thousands of VLANs to partition the traffic according to the specific group that the VM may belong to. The current VLAN limit of 4094 is inadequate in such situations.

Another type of demand that is being placed on data centers is the need to host multiple tenants, each with their own isolated network domain. This is not economical to realize with dedicated infrastructure, so network administrators opt to implement this over a shared network. A common problem is that each tenant may independently assign MAC addresses and VLAN IDs leading to potential duplication of these on the physical network.

Another requirement for virtualized environments using a Layer 2 physical infrastructure is having the Layer 2 network scale across the entire data center or even between data centers for efficient allocation of compute, network and storage resources. In such networks, using traditional approaches like the Spanning Tree Protocol (STP) for a loop free topology can result in a large number of disabled links.

The last scenario is the case where the network operator prefers to use IP for interconnection of the physical infrastructure (e.g. to achieve multipath scalability through Equal Cost Multipath (ECMP), thus avoiding disabled links). Even in such environments, there is a need to preserve the Layer 2 model for inter-VM communication.

The scenarios described above lead to a requirement for an overlay network. This overlay is used to carry the MAC traffic from the individual VMs in an encapsulated format over a logical "tunnel".

This document details a framework termed Virtual eXtensible Local Area Network (VXLAN) which provides such an encapsulation scheme to address the various requirements specified above. This memo documents the deployed VXLAN protocol for the benefit of the IETF community. The IETF consensus on this RFC represents consensus to publish this memo, and not consensus on the text itself.

1.1. Acronyms & Definitions

ACL - Access Control List

ECMP - Equal Cost Multipath

IGMP - Internet Group Management Protocol

PIM - Protocol Independent Multicast

SPB - Shortest Path Bridging

STP - Spanning Tree Protocol

ToR - Top of Rack

TRILL - Transparent Interconnection of Lots of Links

VXLAN - Virtual eXtensible Local Area Network

VXLAN Segment - VXLAN Layer 2 overlay network over which VMs communicate

VXLAN Overlay Network - another term for VXLAN Segment

VXLAN Gateway - an entity which forwards traffic between VXLAN and non-VXLAN environments

VTEP - VXLAN Tunnel End Point - an entity which originates and/or terminates VXLAN tunnels

VLAN - Virtual Local Area Network

VM - Virtual Machine

VNI - VXLAN Network Identifier (or VXLAN Segment ID)

2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC-2119 [RFC2119].

3. VXLAN Problem Statement

This section provides further details on the areas that VXLAN is intended to address. The focus is on the networking infrastructure within the data center and the issues related to them.

3.1. Limitations imposed by Spanning Tree & VLAN Ranges

Current Layer 2 networks use the IEEE 802.1D Spanning Tree Protocol (STP) [802.1D] to avoid loops in the network due to duplicate paths. STP blocks the use of links to avoid the replication and looping of frames. Some data center operators see this as a problem with Layer 2 networks in general since with STP they are effectively paying for more ports and links than they can really use. In addition, resiliency due to multipathing is not available with the STP model. Newer initiatives such as TRILL [RFC6325] and SPB[802.1aq]) have been proposed to help with multipathing and thus surmount some of the problems with STP. STP limitations may also be avoided by configuring servers within a rack to be on the same Layer 3 network with switching happening at Layer 3 both within the rack and between racks. However, this is incompatible with a Layer 2 model for inter-VM communication.

Another characteristic of Layer 2 data center networks is their use of Virtual LANs (VLANs) to provide broadcast isolation. A 12 bit VLAN ID is used in the Ethernet data frames to divide the larger Layer 2 network into multiple broadcast domains. This has served well for several data centers which require fewer than 4094 VLANs. With the growing adoption of virtualization, this upper limit is seeing pressure. Moreover, due to STP, several data centers limit the number of VLANs that could be used. In addition, requirements for multitenant environments accelerate the need for larger VLAN limits, as discussed in Section 3.3.

3.2. Multitenant Environments

Cloud computing involves on demand elastic provisioning of resources for multi-tenant environments. The most common example of cloud computing is the public cloud, where a cloud service provider offers these elastic services to multiple customers/tenants over the same physical infrastructure.

Isolation of network traffic by tenant could be done via Layer 2 or Layer 3 networks. For Layer 2 networks, VLANs are often used to segregate traffic - so a tenant could be identified by its own VLAN, for example. Due to the large number of tenants that a cloud provider might service, the 4094 VLAN limit is often inadequate. In addition, there is often a need for multiple VLANs per tenant, which exacerbates the issue.

Another use case is cross pod expansion. A pod typically consists of one or more racks of servers with associated network and storage connectivity. Tenants may start off on a pod and, due to expansion, require servers/VMs on other pods, especially in the case when tenants on the other pods are not fully utilizing all their resources. This use case requires a "stretched" Layer 2 environment connecting the individual servers/VMs.

Layer 3 networks are not a comprehensive solution for multi tenancy either. Two tenants might use the same set of Layer 3 addresses within their networks which requires the cloud provider to provide isolation in some other form. Further, requiring all tenants to use IP excludes customers relying on direct Layer 2 or non-IP Layer 3 protocols for inter VM communication.

3.3. Inadequate Table Sizes at ToR Switch

Today's virtualized environments place additional demands on the MAC address tables of Top of Rack (ToR) switches which connect to the servers. Instead of just one MAC address per server link, the ToR now has to learn the MAC addresses of the individual VMs (which could range in the 100s per server). This is needed because traffic from/to the VMs to the rest of the physical network will traverse the link between the server and the switch. A typical ToR switch could connect to 24 or 48 servers depending upon the number of its server facing ports. A data center might consist of several racks, so each ToR switch would need to maintain an address table for the communicating VMs across the various physical servers. This places a much larger demand on the table capacity compared to non-virtualized environments.

If the table overflows, the switch may stop learning new addresses until idle entries age out, leading to significant flooding of subsequent unknown destination frames.

4. Virtual eXtensible Local Area Network (VXLAN)

VXLAN (Virtual eXtensible Local Area Network) addresses the above requirements of the Layer 2 and Layer 3 data center network infrastructure in the presence of VMs in a multi-tenant environment. It runs over the existing networking infrastructure and provides a means to "stretch" a Layer 2 network. In short, VXLAN is a Layer 2 overlay scheme over a Layer 3 network. Each overlay is termed a VXLAN segment. Only VMs within the same VXLAN segment can communicate with each other. Each VXLAN segment is identified through a 24 bit segment ID, hereafter termed the VXLAN Network Identifier (VNI). This allows up to 16M VXLAN segments to coexist within the same administrative domain.

The VNI identifies the scope of the inner MAC frame originated by the individual VM. Thus, you could have overlapping MAC addresses across segments but never have traffic "cross over" since the traffic is isolated using the VNI. The VNI is in an outer header which encapsulates the inner MAC frame originated by the VM. In the following sections, the term "VXLAN segment" is used interchangeably with the term "VXLAN overlay network".

Due to this encapsulation, VXLAN could also be termed a tunneling scheme to overlay Layer 2 networks on top of Layer 3 networks. The tunnels are stateless, so each frame is encapsulated according to a set of rules. The end point of the tunnel (VXLAN Tunnel End Point or VTEP) discussed in the following sections is located within the hypervisor on the server which hosts the VM. Thus, the VNI and VXLAN related tunnel/outer header encapsulation are known only to the VTEP

- the VM never sees it (see Figure 1). Note that it is possible that VTEPs could also be on a physical switch or physical server and could be implemented in software or hardware. One use case where the VTEP is a physical switch is discussed in Section 6 on VXLAN deployment scenarios.

The following sections discuss typical traffic flow scenarios in a VXLAN environment using one type of control scheme - data plane learning. Here, the association of VM's MAC to VTEP's IP address is discovered via source address learning. Multicast is used for carrying unknown destination, broadcast and multicast frames.

In addition to a learning based control plane, there are other schemes possible for the distribution of the VTEP IP to VM MAC mapping information. Options could include a central authority/directory based lookup by the individual VTEPs, distribution of this mapping information to the VTEPs by the central authority, and so on. These are sometimes characterized as push and pull models respectively. This draft will focus on the data plane learning scheme as the control plane for VXLAN.

4.1. Unicast VM to VM communication

Consider a VM within a VXLAN overlay network. This VM is unaware of VXLAN. To communicate with a VM on a different host, it sends a MAC frame destined to the target as normal. The VTEP on the physical host looks up the VNI to which this VM is associated. It then determines if the destination MAC is on the same segment and if there is a mapping of the destination MAC address to the remote VTEP. If so, an outer header comprising an outer MAC, outer IP header and VXLAN header (see Figure 1 in Section 5 for frame format) are prepended to the original MAC frame. The encapsulated packet is forwarded towards the remote VTEP. Upon reception, the remote VTEP verifies the validity of the VNI and if there is a VM on that VNI using a MAC address that matches the inner destination MAC address. If so, the packet is stripped of its encapsulating headers and passed on to the destination VM. The destination VM never knows about the VNI or that the frame was transported with a VXLAN encapsulation.

In addition to forwarding the packet to the destination VM, the remote VTEP learns the Inner Source MAC to outer Source IP address mapping. It stores this mapping in a table so that when the destination VM sends a response packet, there is no need for an "unknown destination" flooding of the response packet.

Determining the MAC address of the destination VM prior to the transmission by the source VM is performed as with non-VXLAN environments except as described in Section 4.2. Broadcast frames are used but are encapsulated within a multicast packet, as detailed in the Section 4.2.

4.2. Broadcast Communication and Mapping to Multicast

Consider the VM on the source host attempting to communicate with the destination VM using IP. Assuming that they are both on the same subnet, the VM sends out an ARP broadcast frame. In the non-VXLAN environment, this frame would be sent out using MAC broadcast across all switches carrying that VLAN.

With VXLAN, a header including the VXLAN VNI is inserted at the beginning of the packet along with the IP header and UDP header. However, this broadcast packet is sent out to the IP multicast group on which that VXLAN overlay network is realized.

To effect this, we need to have a mapping between the VXLAN VNI and the IP multicast group that it will use. This mapping is done at the management layer and provided to the individual VTEPs through a management channel. Using this mapping, the VTEP can provide IGMP membership reports to the upstream switch/router to join/leave the VXLAN related IP multicast groups as needed. This will enable pruning of the leaf nodes for specific multicast traffic addresses based on whether a member is available on this host using the specific multicast address (see [RFC4541]). In addition, use of multicast routing protocols like Protocol Independent Multicast - Sparse Mode (PIM-SM see [RFC4601]) will provide efficient multicast trees within the Layer 3 network.

The VTEP will use (*,G) joins. This is needed as the set of VXLAN tunnel sources is unknown and may change often, as the VMs come up/go down across different hosts. A side note here is that since each VTEP can act as both the source and destination for multicast packets, a protocol like PIM-bidir (see [RFC5015]) would be more efficient.

The destination VM sends a standard ARP response using IP unicast. This frame will be encapsulated back to the VTEP connecting the originating VM using IP unicast VXLAN encapsulation. This is possible since the mapping of the ARP response's destination MAC to the VXLAN tunnel end point IP was learned earlier through the ARP request.

Another point to note is that multicast frames and "unknown MAC destination" frames are also sent using the multicast tree, similar to the broadcast frames.

4.3. Physical Infrastructure Requirements

When IP multicast is used within the network infrastructure, a multicast routing protocol like PIM-SM can be used by the individual Layer 3 IP routers/switches within the network. This is used to build efficient multicast forwarding trees so that multicast frames are only sent to those hosts which have requested to receive them.

Similarly, there is no requirement that the actual network connecting the source VM and destination VM should be a Layer 3 network - VXLAN can also work over Layer 2 networks. In either case, efficient multicast replication within the Layer 2 network can be achieved using IGMP snooping.

5. VXLAN Frame Format

The VXLAN frame format is shown below. Parsing this from the bottom of the frame - above the outer frame check sequence (FCS), there is an inner MAC frame with its own Ethernet header with source, destination MAC addresses along with the Ethernet type plus an optional VLAN. See Section 6 for further details of inner VLAN tag handling.

The inner MAC frame is encapsulated with the following four headers (starting from the innermost header):

O VXLAN Header: This is an 8 byte field which has:

o Flags (8 bits)- where the I flag MUST be set to 1 for a valid VXLAN Network ID (VNI). The other 7 bits (designated "R") are reserved fields and MUST be set to zero on transmit and ignored on receive.

o VXLAN Segment ID/VXLAN Network Identifier (VNI) - this is a 24 bit value used to designate the individual VXLAN overlay network on which the communicating VMs are situated. VMs in different VXLAN overlay networks cannot communicate with each other.

o Reserved fields (24 bits and 8 bits) - MUST be set to zero on transmit and ignored on receive.

O Outer UDP Header: This is the outer UDP header with a source port provided by the VTEP and the destination port being a well-known UDP port. IANA has assigned the value 4789 for the VXLAN UDP port and this value SHOULD be used by default as the destination UDP port. Some early implementations of VXLAN have used other values for the destination port. To enable interoperability with these implementations, the destination port SHOULD be configurable. It is recommended that the source port number be calculated using a hash of fields from the inner packet - one example being a hash of the inner Ethernet frame`s headers. This is to enable a level of entropy for ECMP/load balancing of the VM to VM traffic across the VXLAN overlay.

The UDP checksum field SHOULD be transmitted as zero. When a packet is received with a UDP checksum of zero, it MUST be accepted for decapsulation. Optionally, if the encapsulating endpoint includes a non-zero UDP checksum, it MUST be correctly calculated across the entire packet including the IP header, UDP header, VXLAN header and encapsulated MAC frame. When a decapsulating endpoint receives a packet with a non-zero checksum it MAY choose to verify the checksum value. If it chooses to perform such verification, and the verification fails, the packet MUST be dropped. If the decapsulating destination chooses not to perform the verification, or performs it successfully, the packet MUST be accepted for decapsulation.

O Outer IP Header: This is the outer IP header with the source IP address indicating the IP address of the VTEP over which the communicating VM (as represented by the inner source MAC address) is running. The destination IP address can be a unicast or multicast IP address (see Sections 4.1 and 4.2). When it is a unicast IP address, it represents the IP address of the VTEP connecting the communicating VM as represented by the inner destination MAC address. For multicast destination IP addresses, please refer to the scenarios detailed in Section 4.2.

O Outer Ethernet Header (example): Figure 1 is an example of an inner Ethernet frame encapsulated within an outer Ethernet + IP + UDP + VXLAN header. The outer destination MAC address in this frame may be the address of the target VTEP or of an intermediate Layer 3 router. The outer VLAN tag is optional. If present, it may be used for delineating VXLAN traffic on the LAN.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1

  Outer Ethernet Header:
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |             Outer Destination MAC Address                     |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    | Outer Destination MAC Address | Outer Source MAC Address      |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                Outer Source MAC Address                       |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |OptnlEthtype = C-Tag 802.1Q    | Outer.VLAN Tag Information    |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    | Ethertype = 0x0800            |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

  Outer IPv4 Header:
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |Version|  IHL  |Type of Service|          Total Length         |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |         Identification        |Flags|      Fragment Offset    |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |  Time to Live |Protocl=17(UDP)|   Header Checksum             |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                       Outer Source IPv4 Address               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
       |                   Outer Destination IPv4 Address              |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

    Outer UDP Header:
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |       Source Port = xxxx      |       Dest Port = VXLAN Port  |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |           UDP Length          |        UDP Checksum           |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

    VXLAN Header:
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |R|R|R|R|I|R|R|R|            Reserved                           |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                VXLAN Network Identifier (VNI) |   Reserved    |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

    Inner Ethernet Header:
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |             Inner Destination MAC Address                     |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       | Inner Destination MAC Address | Inner Source MAC Address      |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                Inner Source MAC Address                       |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |OptnlEthtype = C-Tag 802.1Q    | Inner.VLAN Tag Information    |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

    Payload:
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       | Ethertype of Original Payload |                               |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               |
       |                                  Original Ethernet Payload    |
       |                                                               |
       |(Note that the original Ethernet Frame's FCS is not included)  |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

    Frame Check Sequence:
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |   New FCS (Frame Check Sequence) for Outer Ethernet Frame     |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1 VXLAN Frame Format with IPv4 Outer Header

The frame format above shows tunneling of Ethernet frames using IPv4 for transport. Use of VXLAN with IPv6 transport is detailed below.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1

  Outer Ethernet Header:
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |             Outer Destination MAC Address                     |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    | Outer Destination MAC Address | Outer Source MAC Address      |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                Outer Source MAC Address                       |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |OptnlEthtype = C-Tag 802.1Q    | Outer.VLAN Tag Information    |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    | Ethertype = 0x86DD            |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

  Outer IPv6 Header:
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |Version| Traffic Class |           Flow Label                  |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |         Payload Length        | NxtHdr=17(UDP)|   Hop Limit   |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    +                                                               +
    |                                                               |
    +                     Outer Source IPv6 Address                 +
    |                                                               |
    +                                                               +
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    +                                                               +
    |                                                               |
    +                  Outer Destination IPv6 Address               +
    |                                                               |
    +                                                               +
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

  Outer UDP Header:
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |       Source Port = xxxx      |       Dest Port = VXLAN Port  |
```

```
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |           UDP Length          |        UDP Checksum           |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+



   VXLAN Header:
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |R|R|R|R|I|R|R|R|            Reserved                           |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                VXLAN Network Identifier (VNI) |   Reserved    |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

   Inner Ethernet Header:
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |             Inner Destination MAC Address                     |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      | Inner Destination MAC Address | Inner Source MAC Address      |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                Inner Source MAC Address                       |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |OptnlEthtype = C-Tag 802.1Q    | Inner.VLAN Tag Information    |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

   Payload:
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      | Ethertype of Original Payload |                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               |
      |                                 Original Ethernet Payload     |
      |                                                               |
      |(Note that the original Ethernet Frame's FCS is not included)  |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

   Frame Check Sequence:
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |   New FCS (Frame Check Sequence) for Outer Ethernet Frame     |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2 VXLAN Frame Format with IPv6 Outer Header

6. VXLAN Deployment Scenarios

VXLAN is typically deployed in data centers on virtualized hosts, which may be spread across multiple racks. The individual racks may be parts of a different Layer 3 network or they could be in a single Layer 2 network. The VXLAN segments/overlay networks are overlaid on top of these Layer 2 or Layer 3 networks.

Consider Figure 3 below depicting two virtualized servers attached to a Layer 3 infrastructure. The servers could be on the same rack, or on different racks or potentially across data centers within the same administrative domain. There are 4 VXLAN overlay networks identified by the VNIs 22, 34, 74 and 98. Consider the case of VM1-1 in Server 1 and VM2-4 on Server 2 which are on the same VXLAN overlay network identified by VNI 22. The VMs do not know about the overlay networks and transport method since the encapsulation and decapsulation happen transparently at the VTEPs on Servers 1 and 2. The other overlay networks and the corresponding VMs are: VM1-2 on Server 1 and VM2-1 on Server 2 both on VNI 34, VM1-3 on Server 1 and VM2-2 on Server 2 on VNI 74, and finally VM1-4 on Server 1 and VM2-3 on Server 2 on VNI 98.

```
     +------------+-------------+
     |        Server 1          |
     | +----+----+  +----+----+ |
     | |VM1-1    |  |VM1-2    | |
     | |VNI 22   |  |VNI 34   | |
     | |         |  |         | |
     | +---------+  +---------+ |
     |                          |
     | +----+----+  +----+----+ |
     | |VM1-3    |  |VM1-4    | |
     | |VNI 74   |  |VNI 98   | |
     | |         |  |         | |
     | +---------+  +---------+ |
     | Hypervisor VTEP (IP1)    |
     +--------------------------+
                           |
                           |
                           |
                           |   +-------------+
                           |   |   Layer 3   |
                           |---|   Network   |
                               |             |
                               +-------------+
                                   |
                                   |
                                   +-----------+
                                               |
                                               |
                                  +------------+-------------+
                                  |        Server 2          |
                                  | +----+----+  +----+----+ |
                                  | |VM2-1    |  |VM2-2    | |
                                  | |VNI 34   |  |VNI 74   | |
                                  | |         |  |         | |
                                  | +---------+  +---------+ |
                                  |                          |
                                  | +----+----+  +----+----+ |
                                  | |VM2-3    |  |VM2-4    | |
                                  | |VNI 98   |  |VNI 22   | |
                                  | |         |  |         | |
                                  | +---------+  +---------+ |
                                  | Hypervisor VTEP (IP2)    |
                                  +--------------------------+
```

Figure 3   VXLAN Deployment - VTEPs across a Layer 3 Network

One deployment scenario is where the tunnel termination point is a physical server which understands VXLAN. Another scenario is where nodes on a VXLAN overlay network need to communicate with nodes on legacy networks which could be VLAN based. These nodes may be physical nodes or virtual machines. To enable this communication, a network can include VXLAN gateways (see Figure 4 below with a switch acting as a VXLAN gateway) which forward traffic between VXLAN and non-VXLAN environments.

Consider Figure 4 for the following discussion. For incoming frames on the VXLAN connected interface, the gateway strips out the VXLAN header and forwards to a physical port based on the destination MAC address of the inner Ethernet frame. Decapsulated frames with the inner VLAN ID SHOULD be discarded unless configured explicitly to be passed on to the non-VXLAN interface. In the reverse direction, incoming frames for the non-VXLAN interfaces are mapped to a specific VXLAN overlay network based on the VLAN ID in the frame. Unless configured explicitly to be passed on in the encapsulated VXLAN frame, this VLAN ID is removed before the frame is encapsulated for VXLAN.

These gateways which provide VXLAN tunnel termination functions could be ToR/access switches or switches higher up in the data center network topology - e.g. core or even WAN edge devices. The last case (WAN edge) could involve a Provider Edge (PE) router which terminates VXLAN tunnels in a hybrid cloud environment. Note that in all these instances, the gateway functionality could be implemented in software or hardware.

```
      +---+-----+---+                                   +---+-----+---+
      |    Server 1 |                                   |  Non VXLAN  |
      (VXLAN enabled)<-----+                       +---->|  server     |
      +-------------+      |                       |     +-------------+
                           |                       |
      +---+-----+---+      |                       |     +---+-----+---+
      |Server 2     |      |                       |     |  Non VXLAN  |
      (VXLAN enabled)<-----+   +---+-----+---+     +---->|    server   |
      +-------------+      |   |Switch acting|     |     +-------------+
                           |---|  as VXLAN   |-----|
      +---+-----+---+      |   |   Gateway   |
      | Server 3    |      |   +-------------+
      (VXLAN enabled)<-----+
      +-------------+      |
                           |
      +---+-----+---+      |
      | Server 4    |      |
      (VXLAN enabled)<-----+
      +-------------+
```

Figure 4   VXLAN Deployment - VXLAN Gateway

6.1. Inner VLAN Tag Handling

Inner VLAN Tag Handling in VTEP and VXLAN Gateway should conform to the following:

Decapsulated VXLAN frames with the inner VLAN tag SHOULD be discarded unless configured otherwise. On the encapsulation side, a VTEP SHOULD NOT include an inner VLAN tag on tunnel packets unless configured otherwise. When a VLAN-tagged packet is a candidate for VXLAN tunneling, the encapsulating VTEP SHOULD strip the VLAN tag unless configured otherwise.

7. Security Considerations

Traditionally, layer 2 networks can only be attacked from 'within' by rogue endpoints - either by having inappropriate access to a LAN and snooping on traffic or by injecting spoofed packets to 'take over' another MAC address or by flooding and causing denial of service. A MAC-over-IP mechanism for delivering Layer 2 traffic significantly extends this attack surface. This can happen by rogues injecting themselves into the network by subscribing to one or more multicast groups that carry broadcast traffic for VXLAN segments and also by sourcing MAC-over-UDP frames into the transport network to inject spurious traffic, possibly to hijack MAC addresses.

This document does not, at this time, incorporate specific measures against such attacks, relying instead on other traditional mechanisms layered on top of IP. This section, instead, sketches out some possible approaches to security in the VXLAN environment.

Traditional Layer 2 attacks by rogue end points can be mitigated by limiting the management and administrative scope of who deploys and manages VMs/gateways in a VXLAN environment. In addition, such administrative measures may be augmented by schemes like 802.1X for admission control of individual end points. Also, the use of the UDP based encapsulation of VXLAN enables configuration and use of the 5 tuple based ACLs (Access Control Lists) functionality in physical switches.

Tunneled traffic over the IP network can be secured with traditional security mechanisms like IPsec that authenticate and optionally encrypt VXLAN traffic. This will, of course, need to be coupled with an authentication infrastructure for authorized endpoints to obtain and distribute credentials.

VXLAN overlay networks are designated and operated over the existing LAN infrastructure. To ensure that VXLAN end points and their VTEPs are authorized on the LAN, it is recommended that a VLAN be designated for VXLAN traffic and the servers/VTEPs send VXLAN traffic over this VLAN to provide a measure of security.

In addition, VXLAN requires proper mapping of VNIs and VM membership in these overlay networks. It is expected that this mapping be done and communicated to the management entity on the VTEP and the gateways using existing secure methods.

8. IANA Considerations

A well-known UDP port (4789) has been assigned by the IANA Service Name and Transport Protocol Port Number Registry for VXLAN. See Section 5 for discussion of the port number.

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

9.2. Informative References

[802.1D] "Standard for Local and Metropolitan Area Networks/ Media Access Control (MAC) Bridges, IEEE P802.1D-2004".

[RFC4601] Fenner, B., Handley, M., Holbrook, H., and Kouvelas, I., "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification", RFC 4601, August 2006.

[RFC5015] Handley, M., Kouvelas, I., Speakman, T., and Vicisano, L., "Bidirectional Protocol Independent Multicast (BIDIR-PIM)", RFC 5015, October 2007.

[RFC4541] Christensen, M., Kimball, K., and Solensky, F., "Considerations for Internet Group Management Protocol (IGMP) and Multicast Listener Discovery (MLD) Snooping Switches", RFC 4541, May 2006.

[RFC6325] Perlman, R., Eastlake, D., Dutt, D., Gai, S., and A. Ghanwani, "RBridges: Base Protocol Specification", RFC 6325, July 2011.

[802.1aq] "Standard for Local and Metropolitan Area Networks / Virtual Bridged Local Area Networks / Amendment20: Shortest Path Bridging, IEEE P802.1aq-2012".

10. Acknowledgments

The authors wish to thank Ajit Sanzgiri for contributions to the Security Considerations section and editorial inputs, Joseph Cheng, Margaret Petrus, Milin Desai, Nial de Barra, Jeff Mandin and Siva Kollipara for their editorial reviews, inputs and comments.

Authors' Addresses

Mallik Mahalingam
Storvisor
333 W.El Camino Real
Sunnyvale, CA 94087

Email: mallik_mahalingam@yahoo.com

Dinesh G. Dutt
Cumulus Networks
140C S.Whisman Road
Mountain View, CA 94041

Email: ddutt.ietf@hobbesdutt.com

Kenneth Duda
Arista Networks
5470 Great America Parkway
Santa Clara, CA 95054

Email: kduda@aristanetworks.com

Puneet Agarwal
Broadcom Corporation
3151 Zanker Road
San Jose, CA 95134

Email: pagarwal@broadcom.com

Lawrence Kreeger
Cisco Systems, Inc.
170 W. Tasman Avenue
Palo Alto, CA 94304

Email: kreeger@cisco.com

T. Sridhar
VMware Inc.
3401 Hillview
Palo Alto, CA 94304

Email: tsridhar@vmware.com

Mike Bursell
Citrix Systems Research & Development Ltd.
Building 101
Cambridge Science Park
Milton Road
Cambridge CB4 0FY
United Kingdom

Email: mike.bursell@citrix.com

Chris Wright
Red Hat Inc.
1801 Varsity Drive
Raleigh, NC 27606

Email: chrisw@redhat.com