Network Working Group
Internet-Draft
Intended Status: Experimental
Expires: November 3, 2014

H. Zhou
C. Li
eBay Inc.

May 2, 2014

Segmentation Offloading Extension for VXLAN
draft-zhou-li-vxlan-soe-01

Abstract

Segmentation offloading is nowadays common in network stack implementation and well supported by para-virtualized network device drivers for virtual machine (VM)s. This draft describes an extension to Virtual eXtensible Local Area Network (VXLAN) so that segmentation can be decoupled from physical/underlay networks and offloaded further to the remote end-point thus improving data-plane performance for VMs running on top of overlay networks.

Table of Contents

1 Introduction

Network virtualization over L3 transport is evolved along with server virtualization in data-centers, and data plane performance is one of the keys to the success of this combination. One of the most critical improvements in OS kernel TCP/IP stack in recent years is segmentation offloading, and now hypervisor providers support same mechanism in para-virtualized Ethernet drivers so that virtual servers can benefit from the same mechanism in virtualized world by offloading segmentation tasks to the lowest layer on hypervisors or NICs (if TSO/UFO is supported by the NICs equipped in the hypervisor).

While the general idea of segmentation offloading is to postpone segmentation to the latest point of packet transmission, this draft introduces a mechanism to avoid overlay segmentation completely in some situation.

Essentially, overlay networks has its own advantage comparing with physical underlay networks in that it does not have a hard MTU limitation. Therefore, segmentation offloading can be pushed to the remote end-point of the transport tunnel, where segmentation can be completely omitted (e.g. the remote end-point is a hypervisor), unless it is going to be forwarded to physical networks (e.g. the remote end-point is a gateway).

However, this advantage is not utilized when the transport of the overlay is based on the Virtual eXtensible Local Area Network [I-D.mahalingam-dutt-dcops-vxlan], which provides a transport mechanism for logically isolated L2 overlay networks between hypervisors. Lacking segmentation information in the VXLAN header, hypervisor implementations have to make pessimistic decisions to always segment the packet in the size specified by VMs before delivering to hypervisors' IP stack, because it does not know whether the remote end-point is bridged to a physical network with hard MTU limitations. It is worth noting that the segmentation here is not the IP fragmentation in terms of the physical network MTU, which may still follow if the segment size resulting from the process above plus the tunnel outer header is greater than the physical network MTU.

To fulfill the potential of segmentation offloading on overlay, this draft introduces segmentation metadata in VXLAN header. With the capability of carrying segmentation metadata in packets, hypervisors can offload the segmentation decision further to the remote tunnel end-point, where decision can be made whether segmentation is omitted, performed, or offloaded further to NIC hardware or next hop tunnel end-point.

This mechanism decouples segmentation for overlay from physical limitations of underlay, providing higher flexibility to hyerpervisor implementations to achieve significant performance gains in a major part of VXLAN deployment scenarios.

Although the performance gains can be achieved is affected by the physical network MTU, there is inherently no mandatory requirement to physical layer:

1) When physical network MTU is far bigger than overlay MTU, the offloading reduces the number of packets being transmitted by TX hypervisors and received in RX hypervisors and RX VMs.

2) When physical network MTU is close to overlay MTU, the number of packets being transmitted in physical network (resulted in IP fragmentation) may not be reduced significantly, but on RX side after IP reassembling, the number of packets being delivered from the hypervisor to the receiving VM is largely reduced, thus saving the cost of hypervisor <-> VM interaction and protocol stack traversing of the receiving VM. Furthermore, a minor cost saving is that the bytes being transmitted over physical network is slightly reduced because only one copy of headers (inner L2-L4 header, VXLAN header and outer UDP header) is transmitted for a large overlay packet.

In addition, offloading features support from NIC hardware is NOT required to the performance gains discussed above.

1.1 Requirements Notation

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].

1.2 Definition of Terms

GSO: Generic Segmentation Offload.

TSO: TCP Segmentation Offload.

UFO: UDP Segmentation Offload.

LRO: Large Receive Offload.

GRO: Generic Receive Offload.

NIC: Network Interface Card.

VM: Virtual Machine.

TX: Sending side.

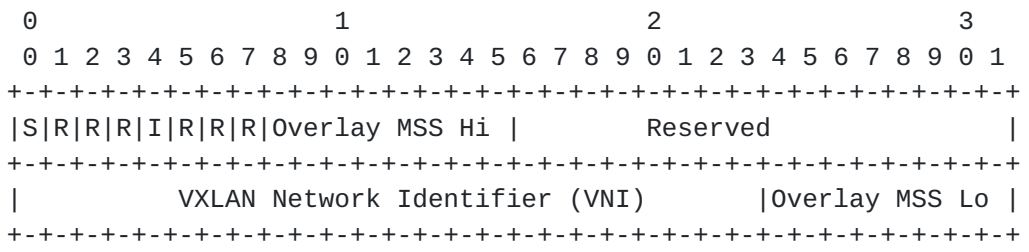
RX: Receiving side.

VTEP: Virtual Tunnel End Point.

2. Approach

2.1 VXLAN Header Extension

The new VXLAN Segmentation Offloading Extension (VXLAN-soe) header is defined as:



The changes to VXLAN are:

S Bit: Flag bit 0 is defined as the S (Segmentation Offloading Extension) bit.

S = 1 indicates that VXLAN-soe is applied to the encapsulated overlay packet, and the Overlay MSS fields (see below) are valid.

S = 0 indicates that VXLAN-soe is NOT applied, and the Overlay MSS fields MUST be set to 0 in accordance with VXLAN.

Overlay MSS: bit 8 - 15 and bit 56 - 63 together is defined as the Overlay Max Segment Size (16 bit unsigned integer) specified by TX VM for the segmentation being offloaded.

Overlay MSS Hi: bit 8 - 15 carries the higher 8 bits of the 16 bit value.

Overlay MSS Lo: bit 56 - 63 carries the lower 8 bits of the 16 bit value.

Definition of the 16 bit value depends on the inner packet type. For TCP packets, it is defined as the max size of TCP payload; for UDP packets, it is defined as the max size of IP payload. This definition follows the convention of Linux kernel implementation, thus GSO size passed from VM to hypervisor can be directly utilized. Definition for other inner packet types can be added in the future.

This field is valid only if the S bit is set.

2.2 TX VTEP

VTEP at TX side MUST set the S bit to 1 if the packet to be encapsulated is NOT segmented and it decides to offload the segmentation to the remote end-point. In such case the Overlay MSS field MUST be set accordingly. This is the typical use case when the TX VTEP is a hypervisor transmitting TCP stream of VMs with large sliding windows.

VTEP at TX side MUST clear the S bit if the packet to be encapsulated is segmented already or does NOT need to be segmented in terms of the overlay MTU. In such case, the encapsulation is in the same format as specified in VXLAN. This is the typical use case when the TX VTEP is a hypervisor transmitting small size overlay packets, or a gateway forwarding overlay packets to physical networks.

2.3 RX VTEP - Hypervisors

When a VTEP at RX side is on a hypervisor, where the packet is delivered to a receiving VM, the hypervisor checks the S bit. If the S bit is NOT set, the packet is handled as a normal VXLAN packet. In this case a packet with size larger than the MTU setting of the receiving VM's virtual interface is usually dropped by the hypervisor. If the S bit is set, the hypervisor SHALL NOT perform MTU check against the virtual interface of the receiving VM.

2.4 RX VTEP - Gateways

When a VTEP at RX side is on a gateway node that connects overlay networks and physical networks, the S bit MUST be checked and the VTEP MUST ensure the segmentation specified by the Overlay MSS field is performed by the VTEP itself or offloaded further - it MAY offload the segmentation again to the subsequent transmission mechanisms: such as TSO/UFO/GSO, or, if the link to the next hop is also an overlay based on VXLAN-soe (or other tunneling protocols that supports segmentation offloading), pass the segmentation metadata to the next hop.

3 IP Fragmentation

Skipping overlay segmentation results in big size packets being encapsulated in VXLAN and outer UDP/IP header. When the encapsulated packet size is bigger than physical network MTU, IP fragmentation has to be enforced. This can leads to two problems.

The first problem is that a single IP fragment loss will result in a

drop of the whole IP packet, which will result in waste of band-width and pose negative impact to the throughput. Because of this, it is recommended to implement VXLAN-soe as a configurable feature, which should be enabled only if physical network is highly reliable. Data center is the typical environment to enable this feature.

Another problem is that inner packet size plus the outer headers can exceed 65535 bytes, which is the upper limit of IP packet size. In this situation special handling can be implemented to avoid oversized IP packets, such as falling back to overlay segmentation. Other optimal solutions are possible but out of the scope of this document.

4 Interoperability

In addition to offload segmentation requests from VMs, VXLAN-soe enabled VTEP is able to offload segmentation requests from STT [I-D.davie-stt] overlay. The metadata required in VXLAN-soe header is a subset of STT metadata, and the additional segmentation offloading information carried in STT metadata such as L4 offset can be obtained by examine inner headers of the packets.

VXLAN-soe is compatible with VXLAN-gpe [I-D.quinn-vxlan-gpe], another extension of VXLAN. For example, if the packet being encapsulated is a TCP/IP packet without L2 header, TCP segmentation can also be skipped at TX side and offloaded to the RX side. See the example in section 5.2.

5 Deployment Examples

5.1 Example 1

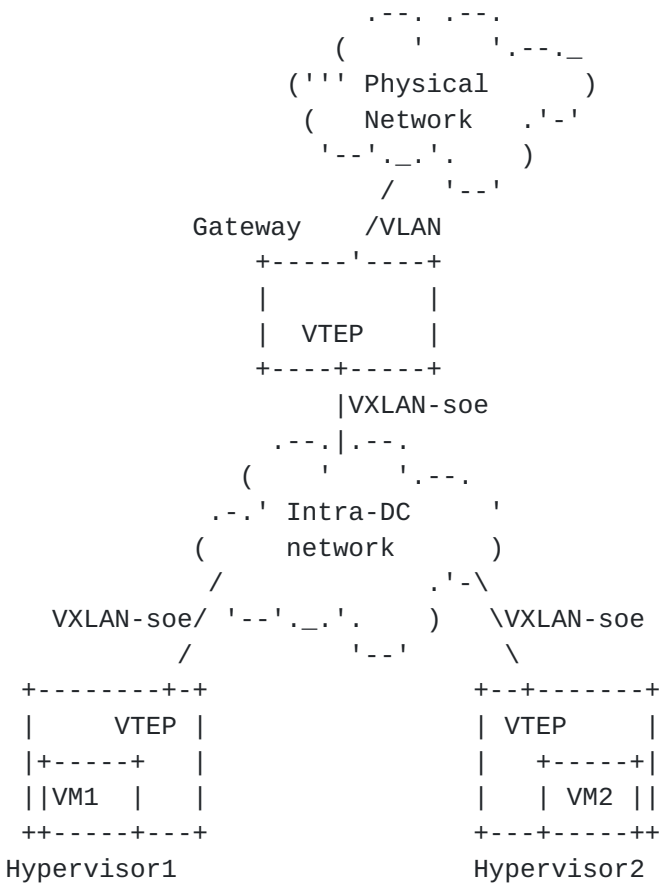


Figure 1

Figure 1 shows basic scenarios of VXLAN-soe usage. Take TCP stream as an example, when VM1 on Hypervisor1 send a big data buffer to VM2 on Hypervisor2, TCP segmentation is offloaded from VM1 to Hypervisor1, and because of VXLAN-soe, it is offloaded from Hypervisor1 to Hypervisor2: the VXLAN-soe encapsulated packet is fragmented in IP fragments according to physical network MTU and transmitted to Hypervisor2. On Hypervisor2, after IP reassembling, the big TCP data buffer is delivered directly to VM2.

When VM1 send a big data buffer to some host behind the Gateway, same process happens on Hypervisor1, but after the IP fragments are reassembled on the Gateway, TCP segmentation must be performed according to the overlay MSS in VXLAN-soe header. The Gateway can be deployed as a ToR switch or a generic server. If the Gateway is a generic server with TSO supported NIC, it can offload the

segmentation task to NIC hardware. In both cases, packets transmitted to the physical VLAN are already segmented according to the overlay MSS.

When TCP segments destined to VM1 are received from the physical VLAN on the Gateway, and if the Gateway is a generic server, NIC hardware with LRO/GRO support can accumulate small TCP segments to bigger TCP packets, which can be delivered to VM1 efficiently with the help of VXLAN-soe.

5.2 Example 2

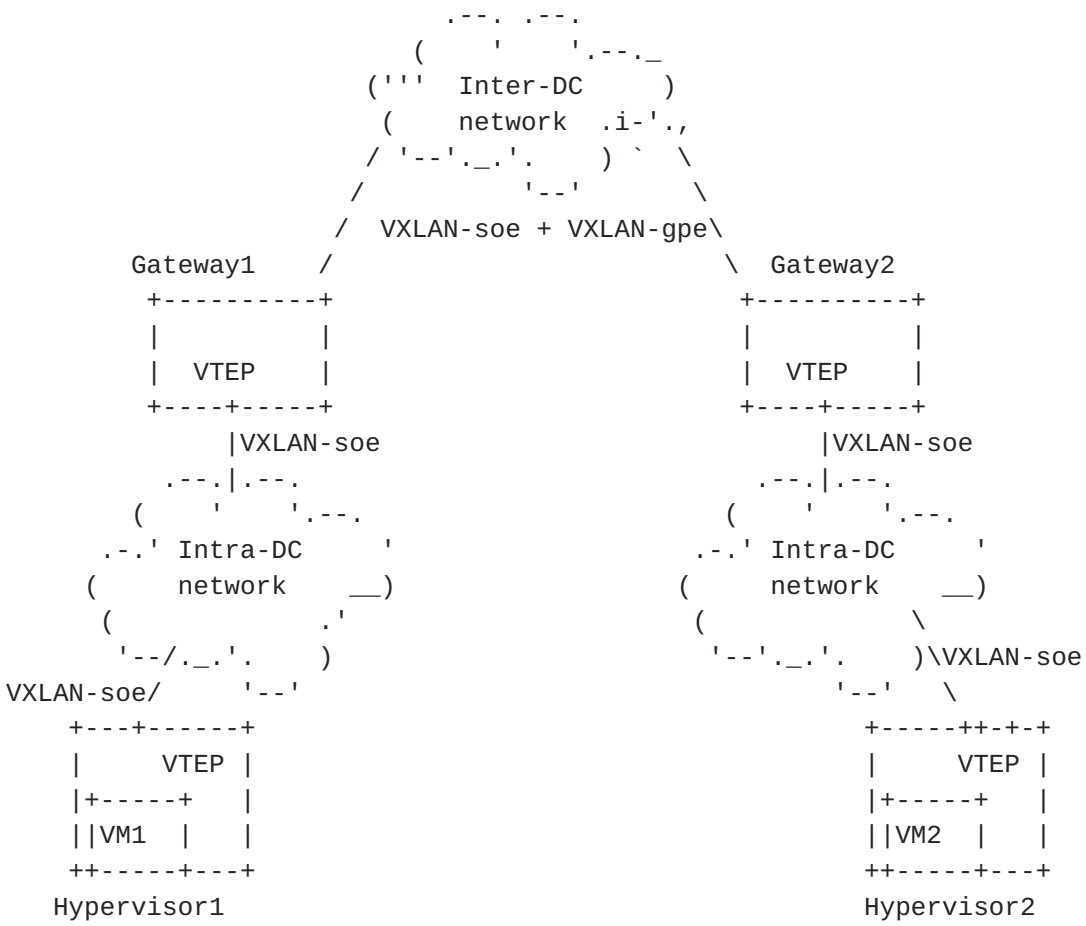


Figure 2

Figure 2 shows how VXLAN-soe and VXLAN-gpe works together. In this example, traffic from VM1 to VM2 needs to traverse inter-DC network connected by Gateway1 and Gateway2. In this case VXLAN-gpe is used between Gateway1 and Gateway2 to encapsulate L3 packets directly. When a big TCP buffer is sent from VM1, TCP segmentation is firstly offloaded to Hyervisor1 and then to Gateway1. With the help of VXLAN-soe between Gateway1 and Gateway2, TCP segmentation is offloaded further to Gateway2 and then to Hypervisor2, where the big TCP buffer is delivered directly to VM2.

6 Security Considerations

There is no special security issues introduced by this extension to VXLAN.

7 IANA Considerations

This document creates no new requirements on IANA namespaces [RFC5226].

8 References

8.1 Normative References

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

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

8.2 Informative References

[I-D.mahalingam-dutt-dcops-vxlan] Mahalingam, M., Dutt, D., Duda, K., Agarwal, P., Kreeger, L., Sridhar, T., Bursell, M., and C. Wright, "VXLAN: A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks", draft-mahalingam-dutt-dcops-vxlan-08 (work in progress), February 2014.

[I-D.davie-stt] Davie, B. and J. Gross, "A Stateless Transport Tunneling Protocol for Network Virtualization (STT)", draft-davie-stt-05(work in progress), March 2014.

[I-D.quinn-vxlan-gpe] Agarwal, P., Fernando, R., Kreeger, L., Lewis, D., Maino, F., Quinn, P., Yong, L., Xu, X., Smith, M., Yadav, N., and U. Elzur, "Generic Protocol Extension for VXLAN", draft-quinn-vxlan-gpe-02 (work in progress), December 2013.

Authors' Addresses

Han Zhou
eBay, Inc.

EMail: hzhou8@ebay.com

Chengyuan Li
eBay, Inc.

Email: chengyli@ebay.com