NVO3 WG
Internet-Draft
Intended status: Standards Track
Expires: April 21, 2019

T. Ao
ZTE Corporation
G. Mirsky
ZTE Corp.
Y. Fan
China Telecom

October 18, 2018

Architecture for Overlay Virtual Sub-Network Interconnection
draft-ao-nvo3-overlay-subnet-architecture-00

Abstract

An virtualized overlay network may be divided into several subnets for the reasons of geographical location, management, or using different technologies being used. For example, different customer have their own preference. But all these subnets need to work together to provide an end-to-end connectionif in a virtual network, An extended architecture of the NVO3 and propose a new component to provide the connection fucntion are introduced in this document.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 21, 2019.

Copyright Notice

Table of Contents

1. Background

Network virtualization using Overlays over Layer 3 (NVO3) is a technology that is used to address issues that arise in building large, multi-tenant data centers that make extensive use of server virtualization.

As described [I-D.defoy-coms-subnet-interconnection] and [I-D.homma-coms-slice-gateway], for network slicing in 5G, there are number of reasons to compose an end-to-end network slice instance by using subnets and stitching operations, thus enabling hierarchical/ recursive management of slices. These subnets are the part of the network slice instance, but not isolate from network slice. An overlay virtual network may also be constructed of several subnets. In such scenario, subnets interconnection to a virtual network is required. We also can consider such interconnection as stitiching. With such stitiching, several subnets can provide a end to end connection in an overlay virtual network.

Moreover, with the progress in NVO3 WG, some of the data plane encapsulations have been put forward, some are outstanding dataplane for an overlay network, such as VxLAN-GPE [I-D.ietf-nvo3-vxlan-gpe], GENEVE [I-D.ietf-nvo3-geneve] and GUE [I-D.ietf-nvo3-encap], etc. The consideration about these overlay encapsulations has been analyzed in [I-D.ietf-nvo3-encap]. The fact is that each of them has its custmers, and furthermore, some of them have been already deployed in the network. So that a problem arises: for a virtual network, all the hosts that connect to the same VN and want to

communicate with each other are required to have the same data plane encapsulation. This problem limits the network scalability and capacity. Especially, when the NVE is located on the vSwitch, the encapsulation method on the NVE is not predictable. Allowing as many types of accession as possible is more attractive for a virtualized overlay network. So in such a scenario, the NVEs using same techonology can be connected to the same subnet.

To improve the scalability and capacity of the virtualized overlay network and to satisfy the subnet interconnection requirement in network slicing, we propose a subnet interconnection architecture in NVO3, and provide a stitching gateway between the subnets in a virtual network in this document. With such architecture, the subnets in an overlay virtual network with different technologies and with seperate management can be interconnected. The gateway that provides the connection between subnets is referred to as Stitching Gateway. In particular, the Stitching Gateway generally would be located in a certain kind of NVE, such NVE is called as S-NVE in the following descrption.

2. Conventions used in this document

2.1. Terminology

NVO3: Network Virtualization using Overlay over Layer 3

NVA: Network Virtualization Authority

TS: Tenant System

VxLAN-GPE: Virtual extension LAN with Generic Protocol Extension

GENEVE: Generic Network Virtualization Encapsulation

GUE: Generic UDP Encapsulation

S-GW: Stitching Gateway. A gateway that does the stitching for seperate subnet to enable them to communicate with each other.

S-NVE: A NVE that complete the stitching functionn as a Stitching Gateway for subnets in an overlay virtual network.

Subnet: An Overlay Virtual Network is combined with several Subnets which may use different technology or different management.

Network slice in this document is used as shortened version of network slice instance.

2.2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Architecture for subnet

As Generic Reference Model desribed in [RFC7364], it's a DC reference model for network virtualization overlays where NVEs provide a logical interconnect between Tenant Systems that belong to a specific VN. But if we need to support a overlay virtual network, an extended reference model is shown as follows in the NVO3 architecture.

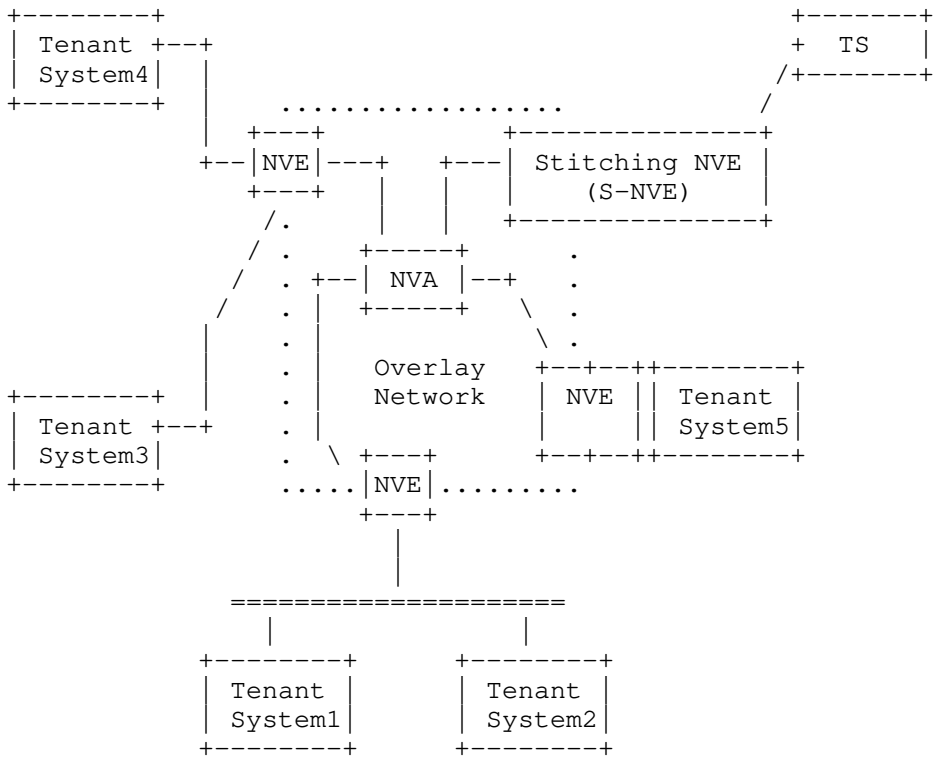


Figure 1 Reference Model supporting subnet

Figure 1: Reference Model supporting subnet

In the above figure, a specific Stitching Gateway(S-Gateway) component is introduced. Generally, the gateway is located on a NVE, so we call it as S-NVE. For the TSs in the same virtual network, if the NVEs which they are connected to are using defferent overlay technology, want to communicate with each other, Stitching NVE(S-NVE) should take over responsibilityas a gateway to provide a "bridge" for the communication. Or for the TSs connecting to different subnets, but belonging to a virtual network, need to communicate with each other through S-NVE.

The difference between NVE and S-NVE is that NVE is the connection between different VN, while S-NVE is the connection between subnet. From the view of NVE, S-NVE is a intermidiate relay node. For the two Tenant Systems belong to different subnet have to communicate through a S-NVE, even though they belong to a same virtual network.

That is, when different NVEs want to set up tunnel, if they can’t connect each other directly because they are on the different subnets, they can set up a tunnel with S-NVE seperately, so that the S-NVE connects the two tunnels as a stitcher. There could be more than one S-NVE in a virtual network.

3.1. Stitching NVE

Stitching NVE(S-NVE) is a certain kind of NVE that maybe appointed by NVA or by the manager. As an essential component in the NVO supporting subnet, the requirements for a Stitching NVE is :

1. Provide the connection for the two subnet of a virtual network.

2. Provides identification/ classification of customer and service traffic, performs the mapping of the two tunnels.

3. Support at least two kinds of encapsulations, translation between technologies/encapsulations when it is stitching two subnets with two different technology .

4. Fault and performance monitoring for underlay and overlay networks.

Regarding the [RFC8014], the Stitching NVE has a reference model as showed in Figure 2.

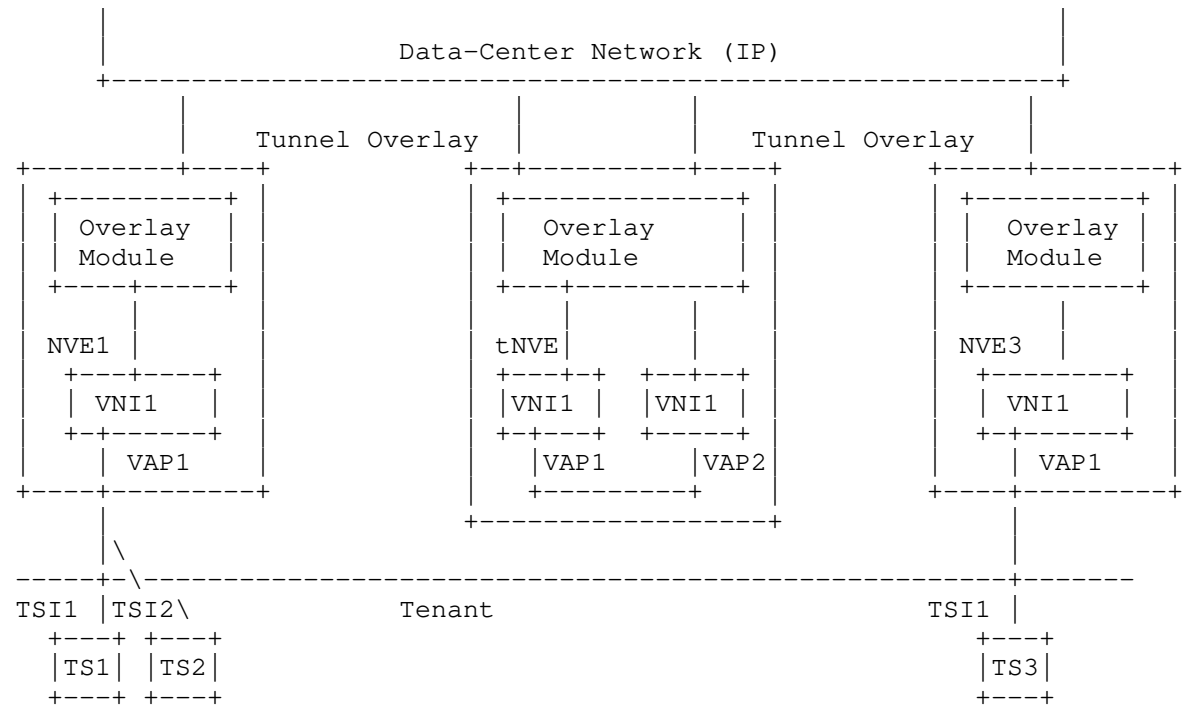


Figure 2 S-NVE Reference Model

Figure 2: S-NVE reference model

S-NVE is a key component of the connection between NVE1 and NVE2. It can be a dedicated device and be a NVE that also provide the overlay network for the TSs. When the NVE takes the role of stitching different subnets for different TSs, it will not forward the traffic to TS, but to another VAP that supports the encapsulation the destination NVE owned.

Take the Figure 2 as an example to illustrate how does S-NVE work. NVE1 belongs to subnet1 and only support VxLAN-GPE, and NVE2 belongs to subnet2 and only support GENEVE. For the two communicating TSs: TS1 needs to send packets to TS3, and TS3 also needs to reply to TS1. They are in the same VNID1, but the NVE they are connected to is using a different encapsulation, and the they belongs to two different subnet. So if the two TSes want to communicate with each other, packets have to transfer at S-NVE first. For NVE1, it has no sense that TS3 is connecting to NVE3, instead of assuming that TS3 is connecting to S-NVE. In the same way, for NVE3, it has no sense that TS1 is connecting to NVE1, instead of assuming that TS1 is connecting to S-NVE. So because of the existence of the tNVE, no matter TS1/TS3 or NVE1/NVE3, they never perceive that they are in the different data

plane. NVE1 getting the packets from TS1 encapsulates them in Vxlan-GPE and then send the packets to S-NVE. The S-NVE receives the packets from the Vxlan-GPE tunnel and then de-encapsulate the vxLAN-GPE to VAP1. Next, the S-NVE forwards packets to the Overlay Module from VAP2 to have another encapsulation GENEVE on the packets. At last S-NVE forwards the packet in the GENEVE tunnel to NVE3.

From the above, S-NVE is like a stitcher between TS1 and TS2. And owning to S-NVE, even though NVE1 and NVE2 which TS1 and TS2 connecting belong to separate subnets and seperately have different encapsulation, as long as they are in the same virtual network, they would communicate each other as a Larger L2 network and no need to know that they are in different subnets or using different technology.

4. IANA Considerations

TBD.

5. References

5.1. Normative References

[I-D.ietf-intarea-gue]
Herbert, T., Yong, L., and O. Zia, "Generic UDP Encapsulation", draft-ietf-intarea-gue-06 (work in progress), August 2018.

[I-D.ietf-nvo3-geneve]
Gross, J., Ganga, I., and T. Sridhar, "Geneve: Generic Network Virtualization Encapsulation", draft-ietf-nvo3-geneve-08 (work in progress), October 2018.

[I-D.ietf-nvo3-vxlan-gpe]
Maino, F., Kreeger, L., and U. Elzur, "Generic Protocol Extension for VXLAN", draft-ietf-nvo3-vxlan-gpe-06 (work in progress), April 2018.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC7365] Lasserre, M., Balus, F., Morin, T., Bitar, N., and Y. Rekhter, "Framework for Data Center (DC) Network Virtualization", RFC 7365, DOI 10.17487/RFC7365, October 2014, <https://www.rfc-editor.org/info/rfc7365>.

[RFC8014] Black, D., Hudson, J., Kreeger, L., Lasserre, M., and T. Narten, "An Architecture for Data-Center Network Virtualization over Layer 3 (NVO3)", RFC 8014, DOI 10.17487/RFC8014, December 2016, <https://www.rfc-editor.org/info/rfc8014>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

5.2. Informational References

[I-D.ao-nvo3-multi-encap-interconnect] Ao, T., Mirsky, G., and F. Yongbing, "Multi-encapsulation interconnection for Overlay Virtual Network", draft-ao-nvo3-multi-encap-interconnect-00 (work in progress), March 2018.

[I-D.defoy-coms-subnet-interconnection] Foy, X., Rahman, A., Galis, A., kiran.makhijani@huawei.com, k., Qiang, L., Homma, S., and P. Martinez-Julia, "Interconnecting (or Stitching) Network Slice Subnets", draft-defoy-coms-subnet-interconnection-03 (work in progress), March 2018.

[I-D.homma-coms-slice-gateway] Homma, S., Foy, X., and A. Galis, "Gateway Function for Network Slicing", draft-homma-coms-slice-gateway-01 (work in progress), March 2018.

[I-D.ietf-nvo3-encap] Boutros, S., "NVO3 Encapsulation Considerations", draft-ietf-nvo3-encap-02 (work in progress), September 2018.

[RFC7364] Narten, T., Ed., Gray, E., Ed., Black, D., Fang, L., Kreeger, L., and M. Napierala, "Problem Statement: Overlays for Network Virtualization", RFC 7364, DOI 10.17487/RFC7364, October 2014, <https://www.rfc-editor.org/info/rfc7364>.

Authors' Addresses

Ting Ao
ZTE Corporation
No.889, BiBo Road
Shanghai 201203
China

Phone: +86 21 68897642
Email: ao.ting@zte.com.cn

Greg Mirsky
ZTE Corp.
1900 McCarthy Blvd. #205
Milpitas, CA 95035
USA

Email: gregimirsky@gmail.com

Yongbin
China Telecom
No.109, Zhongshan Avenue
Guangzhou 510630
China

Email: fanyb@gsta.com

NVO3 WG
Internet-Draft
Intended status: Standards Track
Expires: December 7, 2019

T. Ao
ZTE Corporation
G. Mirsky
ZTE Corp.
Y. Fan
China Telecom
June 5, 2019

Architecture for Overlay Virtual Sub-Network Interconnection draft-ao-nvo3-overlay-subnet-architecture-02

Abstract

An virtualized overlay network may be divided into several subnets for the reasons of geographical location, management, or using different technologies being used. For example, different customer have their own preference. But all these subnets need to work together to provide an end-to-end connectionif in a virtual network, An extended architecture of the NVO3 and propose a new component to provide the connection fucntion are introduced in this document.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on December 7, 2019.

Copyright Notice

Table of Contents

1. Background

Network virtualization using Overlays over Layer 3 (NVO3) is a technology that is used to address issues that arise in building large, multi-tenant data centers that make extensive use of server virtualization.

As described [I-D.defoy-coms-subnet-interconnection] and [I-D.homma-coms-slice-gateway], for network slicing in 5G, there are number of reasons to compose an end-to-end network slice instance by using subnets and stitching operations, thus enabling hierarchical/ recursive management of slices. These subnets are the part of the network slice instance, but not isolate from network slice. An overlay virtual network may also be constructed of several subnets. In such scenario, subnets interconnection to a virtual network is required. We also can consider such interconnection as stitiching. With such stitiching, several subnets can provide a end to end connection in an overlay virtual network.

Moreover, with the progress in NVO3 WG, some of the data plane encapsulations have been put forward, some are outstanding dataplane for an overlay network, such as VxLAN-GPE [I-D.ietf-nvo3-vxlan-gpe], GENEVE [I-D.ietf-nvo3-geneve] and GUE [I-D.ietf-nvo3-encap], etc. The consideration about these overlay encapsulations has been analyzed in [I-D.ietf-nvo3-encap]. The fact is that each of them has its custmers, and furthermore, some of them have been already deployed in the network. So that a problem arises: for a virtual network, all the hosts that connect to the same VN and want to

communicate with each other are required to have the same data plane encapsulation. This problem limits the network scalability and capacity. Especially, when the NVE is located on the vSwitch, the encapsulation method on the NVE is not predictable. Allowing as many types of accession as possible is more attractive for a virtualized overlay network. So in such a scenario, the NVEs using same techonology can be connected to the same subnet.

To improve the scalability and capacity of the virtualized overlay network and to satisfy the subnet interconnection requirement in network slicing, we propose a subnet interconnection architecture in NVO3, and provide a stitching gateway between the subnets in a virtual network in this document. With such architecture, the subnets in an overlay virtual network with different technologies and with seperate management can be interconnected. The gateway that provides the connection between subnets is referred to as Stitching Gateway. In particular, the Stitching Gateway generally would be located in a certain kind of NVE, such NVE is called as S-NVE in the following descrption.

2. Conventions used in this document

2.1. Terminology

NVO3: Network Virtualization using Overlay over Layer 3

NVA: Network Virtualization Authority

TS: Tenant System

VxLAN-GPE: Virtual extension LAN with Generic Protocol Extension

GENEVE: Generic Network Virtualization Encapsulation

GUE: Generic UDP Encapsulation

S-GW: Stitching Gateway. A gateway that does the stitching for seperate subnet to enable them to communicate with each other.

S-NVE: A NVE that complete the stitching functionn as a Stitching Gateway for subnets in an overlay virtual network.

Subnet: An Overlay Virtual Network is combined with several Subnets which may use different technology or different management.

Network slice in this document is used as shortened version of network slice instance.

2.2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Architecture for subnet

As Generic Reference Model desribed in [RFC7364], it's a DC reference model for network virtualization overlays where NVEs provide a logical interconnect between Tenant Systems that belong to a specific VN. But if we need to support a overlay virtual network, an extended reference model is shown as follows in the NVO3 architecture.

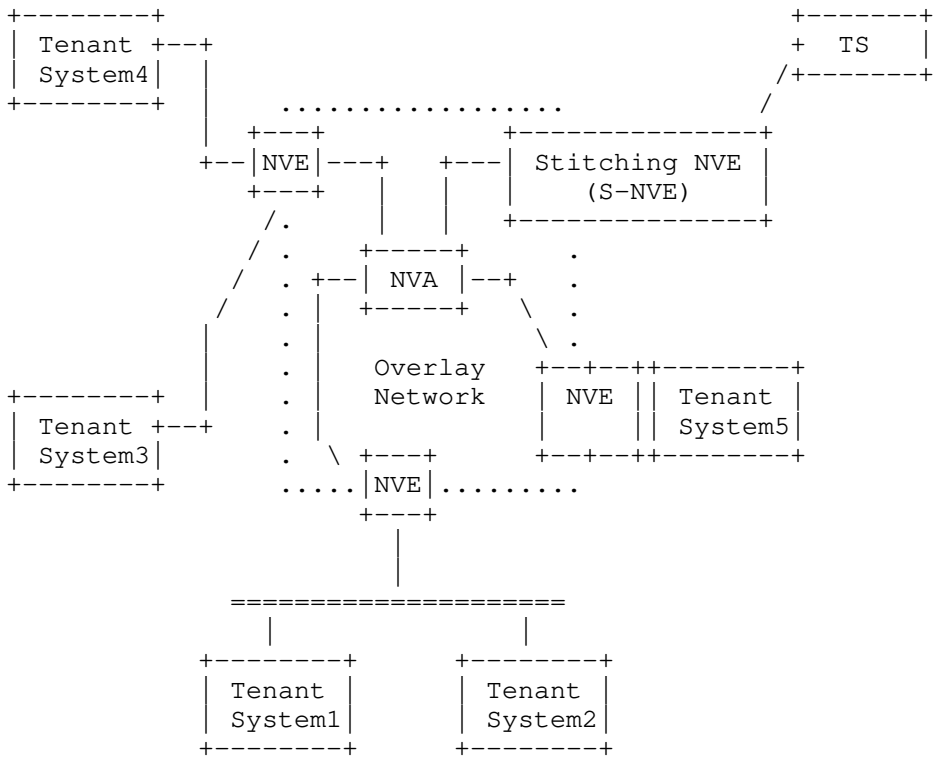


Figure 1 Reference Model supporting subnet

Figure 1: Reference Model supporting subnet

In the above figure, a specific Stitching Gateway(S-Gateway) component is introduced. Generally, the gateway is located on a NVE, so we call it as S-NVE. For the TSs in the same virtual network, if the NVEs which they are connected to are using defferent overlay technology, want to communicate with each other, Stitching NVE(S-NVE) should take over responsibilityas a gateway to provide a "bridge" for the communication. Or for the TSs connecting to different subnets, but belonging to a virtual network, need to communicate with each other through S-NVE.

The difference between NVE and S-NVE is that NVE is the connection between different VN, while S-NVE is the connection between subnet. From the view of NVE, S-NVE is a intermidiate relay node. For the two Tenant Systems belong to different subnet have to communicate through a S-NVE, even though they belong to a same virtual network.

That is, when different NVEs want to set up tunnel, if they can’t connect each other directly because they are on the different subnets, they can set up a tunnel with S-NVE seperately, so that the S-NVE connects the two tunnels as a stitcher. There could be more than one S-NVE in a virtual network.

3.1. Stitching NVE

Stitching NVE(S-NVE) is a certain kind of NVE that maybe appointed by NVA or by the manager. As an essential component in the NVO supporting subnet, the requirements for a Stitching NVE is :

1. Provide the connection for the two subnet of a virtual network.

2. Provides identification/ classification of customer and service traffic, performs the mapping of the two tunnels.

3. Support at least two kinds of encapsulations, translation between technologies/encapsulations when it is stitching two subnets with two different technology .

4. Fault and performance monitoring for underlay and overlay networks.

Regarding the [RFC8014], the Stitching NVE has a reference model as showed in Figure 2.

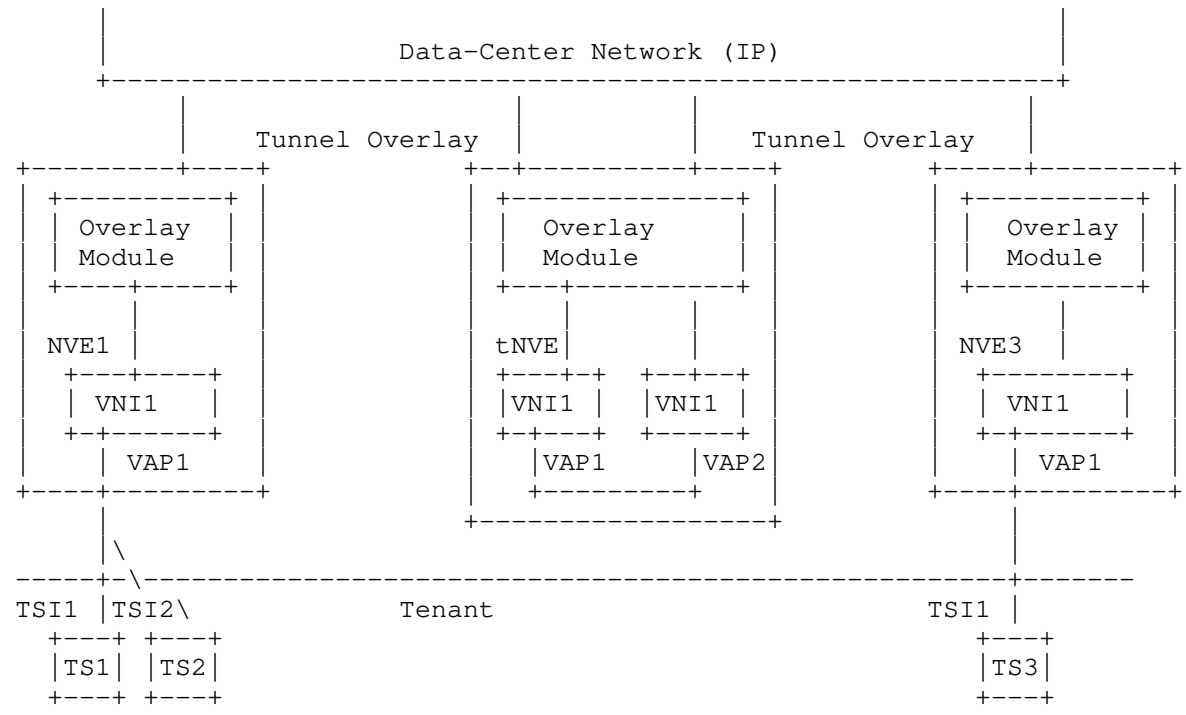


Figure 2 S-NVE Reference Model

Figure 2: S-NVE reference model

S-NVE is a key component of the connection between NVE1 and NVE2. It can be a dedicated device and be a NVE that also provide the overlay network for the TSs. When the NVE takes the role of stitching different subnets for different TSs, it will not forward the traffic to TS, but to another VAP that supports the encapsulation the destination NVE owned.

Take the Figure 2 as an example to illustrate how does S-NVE work. NVE1 belongs to subnet1 and only support VxLAN-GPE, and NVE2 belongs to subnet2 and only support GENEVE. For the two communicating TSs: TS1 needs to send packets to TS3, and TS3 also needs to reply to TS1. They are in the same VNID1, but the NVE they are connected to is using a different encapsulation, and the they belongs to two different subnet. So if the two TSes want to communicate with each other, packets have to transfer at S-NVE first. For NVE1, it has no sense that TS3 is connecting to NVE3, instead of assuming that TS3 is connecting to S-NVE. In the same way, for NVE3, it has no sense that TS1 is connecting to NVE1, instead of assuming that TS1 is connecting to S-NVE. So because of the existence of the tNVE, no matter TS1/TS3 or NVE1/NVE3, they never perceive that they are in the different data

plane. NVE1 getting the packets from TS1 encapsulates them in Vxlan-GPE and then send the packets to S-NVE. The S-NVE receives the packets from the Vxlan-GPE tunnel and then de-encapsulate the vxLAN-GPE to VAP1. Next, the S-NVE forwards packets to the Overlay Module from VAP2 to have another encapsulation GENEVE on the packets. At last S-NVE forwards the packet in the GENEVE tunnel to NVE3.

From the above, S-NVE is like a stitcher between TS1 and TS2. And owning to S-NVE, even though NVE1 and NVE2 which TS1 and TS2 connecting belong to separate subnets and seperately have different encapsulation, as long as they are in the same virtual network, they would communicate each other as a Larger L2 network and no need to know that they are in different subnets or using different technology.

4. IANA Considerations

TBD.

5. References

5.1. Normative References

[I-D.ietf-intarea-gue]
Herbert, T., Yong, L., and O. Zia, "Generic UDP Encapsulation", draft-ietf-intarea-gue-07 (work in progress), March 2019.

[I-D.ietf-nvo3-geneve]
Gross, J., Ganga, I., and T. Sridhar, "Geneve: Generic Network Virtualization Encapsulation", draft-ietf-nvo3-geneve-13 (work in progress), March 2019.

[I-D.ietf-nvo3-vxlan-gpe]
Maino, F., Kreeger, L., and U. Elzur, "Generic Protocol Extension for VXLAN", draft-ietf-nvo3-vxlan-gpe-07 (work in progress), April 2019.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC7365] Lasserre, M., Balus, F., Morin, T., Bitar, N., and Y. Rekhter, "Framework for Data Center (DC) Network Virtualization", RFC 7365, DOI 10.17487/RFC7365, October 2014, <https://www.rfc-editor.org/info/rfc7365>.

[RFC8014] Black, D., Hudson, J., Kreeger, L., Lasserre, M., and T. Narten, "An Architecture for Data-Center Network Virtualization over Layer 3 (NVO3)", RFC 8014, DOI 10.17487/RFC8014, December 2016, <https://www.rfc-editor.org/info/rfc8014>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

5.2. Informational References

[I-D.ao-nvo3-multi-encap-interconnect] Ao, T., Mirsky, G., and F. Yongbing, "Multi-encapsulation interconnection for Overlay Virtual Network", draft-ao-nvo3-multi-encap-interconnect-00 (work in progress), March 2018.

[I-D.defoy-coms-subnet-interconnection] Foy, X., Rahman, A., Galis, A., kiran.makhijani@huawei.com, k., Qiang, L., Homma, S., and P. Martinez-Julia, "Interconnecting (or Stitching) Network Slice Subnets", draft-defoy-coms-subnet-interconnection-03 (work in progress), March 2018.

[I-D.homma-coms-slice-gateway] Homma, S., Foy, X., and A. Galis, "Gateway Function for Network Slicing", draft-homma-coms-slice-gateway-01 (work in progress), March 2018.

[I-D.ietf-nvo3-encap] Boutros, S., "NVO3 Encapsulation Considerations", draft-ietf-nvo3-encap-02 (work in progress), September 2018.

[RFC7364] Narten, T., Ed., Gray, E., Ed., Black, D., Fang, L., Kreeger, L., and M. Napierala, "Problem Statement: Overlays for Network Virtualization", RFC 7364, DOI 10.17487/RFC7364, October 2014, <https://www.rfc-editor.org/info/rfc7364>.

Authors' Addresses

Ting Ao
ZTE Corporation
No.889, BiBo Road
Shanghai 201203
China

Phone: +86 17721209283
Email: 18555817@qq.com

Greg Mirsky
ZTE Corp.
1900 McCarthy Blvd. #205
Milpitas, CA 95035
USA

Email: gregimirsky@gmail.com

Yongbin
China Telecom
No.109, Zhongshan Avenue
Guangzhou 510630
China

Email: fanyb@gsta.com

INTERNET-DRAFT
Intended Status: Standard Track

Sami Boutros, Ed.
VMware
Dharma Rajan
Philip Kippen
Pierluigi Rolando
VMware

Expires: March 18, 2019

September 14, 2018

Geneve applicability for service function chaining
draft-boutros-nvo3-geneve-applicability-for-sfc-02

Abstract

This document describes the applicability of using Generic Network Virtualization Encapsulation (Geneve), to carry the service function path (SFP) information, and the network service header (NSH) encapsulation. The SFP information will be carried in Geneve option TLV(s).

Table of Contents

1. Introduction

The Service Function Chaining (SFC) Architecture [rfc7665] defines a service function chain (SFC) as (1) the instantiation of an ordered set of service functions and (2) the subsequent "steering" of traffic through them.

SFC defines a Service Function Path (SFP) as the exact set of service function forwarders (SFF)/service functions (SF)s the packet will visit when it actually traverses the network.

An optimized SFP helps to build an efficient Service function chain (SFC) that can be used to steer traffic based on classification rules, and metadata information to provide services for Network Function Virtualization (NFV). Metadata are typically passed between service functions and Service function forwarders SFF(s) along a service function path.

In a Network Virtualization Overlays (NVO3) domain, Network Virtualization Edges (NVE)s can be implemented on hypervisors hosting virtual network functions (VNF)s implementing service functions, or on physical routers connected to service function appliances. NVO3 domain uses tunneling and encapsulation protocols such as Geneve to provide connectivity for tenants workloads and service function running in its domain. NVEs in an NVO3 domain are typically controlled by a centralized network virtualization authority NVA.

[RFC8300] defines a new encapsulation protocol, network service header (NSH) to encode the SFP and the metadata.

1.1 Requirement for SFC in NVO3 domain

The requirement is to provide service function chaining in an NVO3 domain without the need to implement yet another control plane for service topology.

1.2 Proposed solution for SFC in NVO3 domain

This document specifies the applicability of using Generic Network Virtualization Encapsulation (Geneve), to carry the service function path (SFP) information, and the network service header (NSH) encapsulation.

The SFP will be implemented using a new Geneve Service Function List (SFL) option for use strictly between Network Virtualization Edges (NVEs) performing the service forwarding function (SFF) in the same Network Virtualization Overlay over Layer 3 NVO3 domain. The next protocol in the Geneve Header will be the NSH EtherType, 0x894F. The

NSH encapsulation will include the Service Path Identifier (SPI) and the Service Index (SI). The NSH SI will serve as an index to the VNF hop to visit in the SFL.

In the absence of the SFL we would need a service topology control plane. The Geneve overlay will encap the NSH encapsulation and the next protocol on Geneve will be the NSH Ethertype.

2. Terminology

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. Abbreviations

NVO3 Network Virtualization Overlays over Layer 3

OAM Operations, Administration, and Maintenance

TLV Type, Length, and Value

VNI Virtual Network Identifier

NVE Network Virtualization Edge

NVA Network Virtualization Authority

NIC Network interface card

VTEP Virtual Tunnel End Point

Transit device Underlay network devices between NVE(s).

Service Function (SF): Defined in [RFC7665].

Service Function Chain (SFC): Defined in [RFC7665].

Service Function Forwarder (SFF): Defined in [RFC7665].

Service Function Path (SFP): Defined in [RFC7665].

Metadata: Defined in [[draft-ietf-sfc-nsh]

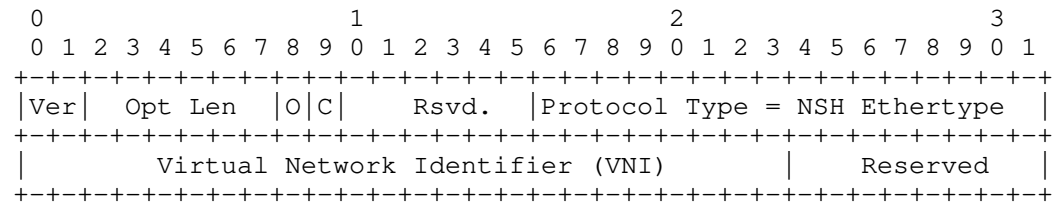
NFV: Network function virtualization.

VNF: Virtual network function

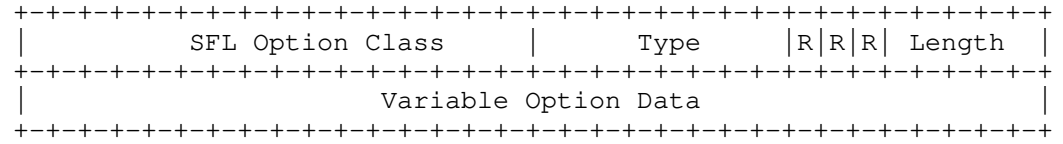
4. Geneve Option TLV(s)

4.1 Geneve Service Function List (SFL) Option TLV

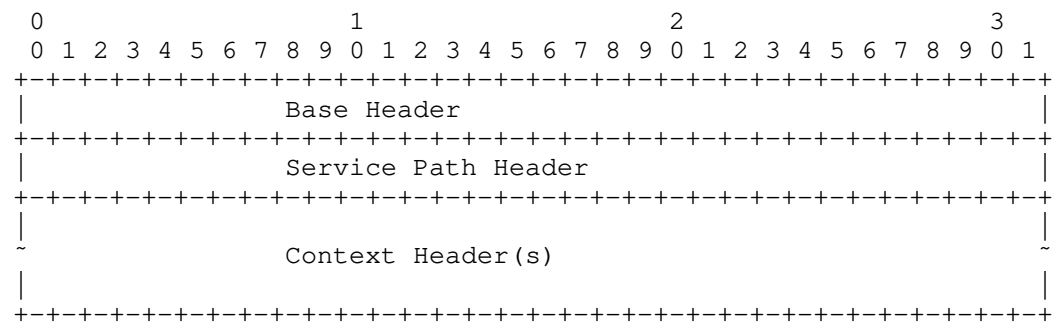
Geneve Header:



Geneve Option Header:



Followed by the NSH encapsulation which is composed of a 4-byte Base Header, a 4-byte Service Path Header, and optional Context Headers.



SFL Option Class = To be assigned by IANA

Type = To be assigned by IANA

'C' bit set, indicating endpoints must drop if they do not recognize this option)

Length = variable.

Variable option data:

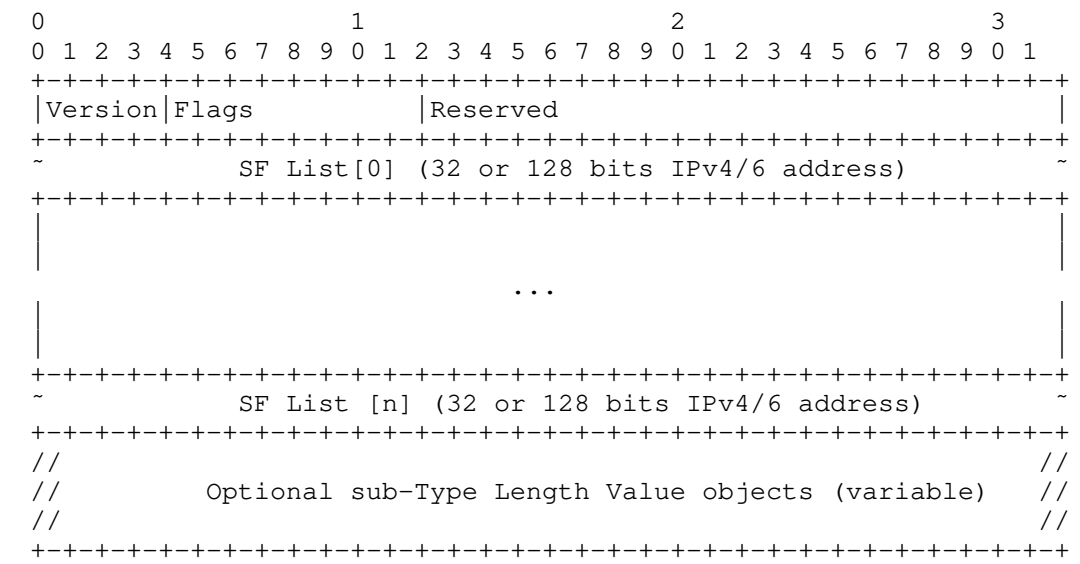


Figure 1: Service Function List (SFL) Option TLV.

Reserved: 12 bits. SHOULD be unset on transmission and MUST be ignored on receipt.

Flags:

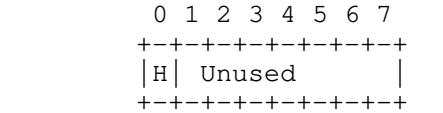


Figure 2: SFL flags

H-flag: HMAC flag. If set, the HMAC sub-TLV is present and is encoded as the last sub-TLV.

SF List[n]: 32 or 128 bits IPv4/6 addresses representing the nth service function ip address in the List.

The SF List is encoded starting from the last hop of the path. I.e., the first element of the list (SF List [0]) contains the last service function of the path while the last element of the SF List (SF List[n]) contains the first service function in the path.

HMAC sub-TLV is optional and contains the HMAC information. The

HMAC sub-TLV has the following format:

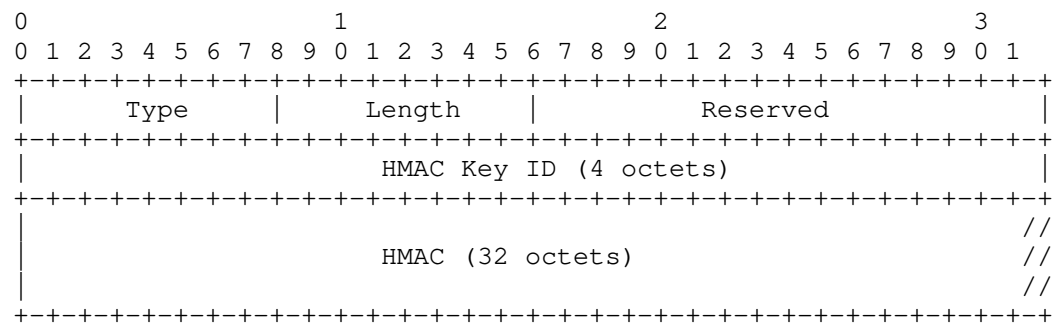


Figure 3: SFL HMAC sub-TLV.

Type: to be assigned by IANA (suggested value 1).
Length: 38.
Reserved: 2 octets. SHOULD be unset on transmission and MUST be ignored on receipt.
HMAC Key ID: 4 octets.
HMAC: 32 octets.
HMAC and HMAC Key ID usage is described in Operation section.

The Following applies to the HMAC TLV:

When present, the HMAC sub-TLV MUST be encoded as the last sub-TLV

If the HMAC sub-TLV is present, the H-Flag (Figure 2) MUST be set.

When the H-flag is set, the NVE inspecting the Geneve Service Function List Option TLV MUST find the HMAC sub-TLV in the last 38 octets of the option TLV.

5.. Operation

The mechanisms described in this section should work with both ipv4 and ipv6 for both customer inner payload and Geneve tunnel packets.

5.1 Operation at Ingress

A Source NVE acting as a service function classifier and a service function forwarder can be any node in an NVO3 domain, originating based on a classification policy for some customer inner payload an IP Geneve tunnel packet with the service function list (SFL) option TLV. The service functions in the SFL represent the IP addresses of the service functions that the inner customer packets needs to be inspected by. A controller can program the ingress NVE node to classify traffic and identify a service function paths i.e the set of

service functions in the path. The mechanism through which an SFL is derived by a controller or any other mechanisms is outside of the scope of this document.

The ingress NVE node fills in the list of service functions in the path, to the Geneve Service Function List option TLV, putting the first service function ip address as the last element in the list and the last service function ip address as the first element, setting of the NSH service index to the first element. The ingress NVE node, then, resolves the service first function ip address, to the NVE virtual tunnel endpoint node hosting or directly connected to the service function.

The Geneve tunnel destination is then set to the NVE tunnel endpoint hosting the first service function, and the service index is decremented to n-1 (where n is the number of elements in the SFL), and set on the SFL option TLV. An NSH metadata can also be set on the packet by the NVE ingress node.

The Geneve packet is sent out towards the first NVE.

HMAC optional sub-TLV may be set too.

5.2 Operation at each NVE along the service function path

The NVE node along the service function path corresponding to the Geneve tunnel destination of the packet, receives the packet, perform the service function forwarder function and identifies the SFL option, and locates the service function in the list based on the service index.

The Geneve tunnel header and option TLV(s) will be stripped and the packet will be delivered to the service function or virtual network function (VNF). The NVE maintains state related to the association of the SFL option TLV and the NSH service path identifier. The packet passed to the service function encaped with the NSH header and NSH context, if the SF is NSH aware, other encapsulations like vlan or q-in-q encap may be used to pass the metadata and NSH SPI to the SF too.

When the packet comes back from the service function along with the service path identifier (SPI) context, based on SPI on the packet the NVE acting as the SFF will be able to locate the SFL option TLV.

If the metadata context indicate (1) that some service functions need to be bypassed the NVE should bypass in the SFL the service functions to be skipped and update the NSH service index accordingly. (2) A new

classification need to be performed on the packet, in that case the NVE can re-classify the packet or sent it to an NVE node capable of classification.

The NVE node, then, resolves the next service function ip address, to the NVE virtual tunnel endpoint node hosting or directly connected to the service function.

The NVE then sets the Geneve tunnel destination to the next NVE tunnel endpoint, and the NSH service index is decremented by 1 and set on the NSH Header, along with other NSH metadata option TLV.

The Geneve ip packet is sent out towards the next NVE.

5.3 Operation at Egress

At the last NVE node along the service function path, the NVE locates the service function in the SFL option TLV based on the NSH service index. The service index received at the last NVE node will be set to 1.

The Geneve tunnel header and option TLV(s) will be stripped and the packet will be delivered to the service function. The NVE maintains state related to the association of the SFL option TLV and the NSH service path identifier. The packet passed to the service function encaped with the NSH header and NSH context, if the SF is NSH aware, other encapsulations like vlan or q-in-q encap may be used to pass the metadata and NSH SPI to the SF too.

When the packet comes back from the service function, based on NSH SPI on the packet or based the NVE will be able to locate the SFL option TLV.

Given that the service index will be set to 1, the last NVE will now deliver the packet to the NVE hosting or directly connected to the inner packet destination.

A packet received with a service function index of 0 MUST be dropped.

6. Security Considerations

Only NVE(s) that are the destinations of the Geneve tunnel packet will be inspecting the List of Service Function next hops Option. A Source routing option has some well-known security issues as described in [RFC4942] and [RFC5095].

The main use case for the use of the Geneve List of Service Function next hops Option will be within a single NVO3 administrative domain

where only trusted NVE nodes are enabled and configured participate, this is the same model as in [RFC6554].

NVE nodes MUST ignore the Geneve List of Service Function next hops Option created by outsiders based on NVA or trusted control plane information.

There is a need to prevent non-participating NVE node from using the Geneve Service Function List option TLV, as described in [draft-ietf-6man-segment-routing-header], we will use a security sub-TLV in the Service Function List option TLV, the security sub-TLV will be based on a key-hashed message authentication code (HMAC).

HMAC sub-TLV will contain:

HMAC Key-id, 32 bits wide;

HMAC, 256 bits wide (optional, exists only if HMAC Key-id is not 0).

The HMAC field is the output of the HMAC computation (per RFC 2104 [RFC2104]) using a pre-shared key identified by HMAC Key-id and of the text which consists of the concatenation of:

The source IPv4/IPv6 Geneve tunnel address

Version and Flags

HMAC Key-id.

All addresses in the List.

The purpose of the HMAC optional sub-TLV is to verify the validity, the integrity and the authorization of the Geneve Service Function List option TLV itself.

The HMAC optional sub-TLV is located at the end of the Geneve Service Function List option TLV.

The HMAC Key-id field serves as an index to the right combination of pre-shared key and hash algorithm and except that a value of 0 means that there is no HMAC field.

The HMAC Selection of a hash algorithm and Pre-shared key management will follow the procedures described in [draft-ietf-6man-segment-routing-header] section 6.2.

7. Management Considerations

The Source NVE can receive its information through any form of north bound Orchestrator. These could be from any open networking automation platform (ONAP) or others. The ingress to egress tunnel is built and managed by the service function classifier and service function forwarder by each node in an NVO3 domain. Error handling, is handled by the classifier reporting to north bound management systems.

8. Acknowledgements

The authors would like to acknowledge Jim Guichard for his feedback and valuable comments to this document.

9. IANA Considerations

This document makes the following registrations in the "Geneve Option Class" registry maintained by IANA:

Suggested Value	Description	Reference
XX	Geneve List of Service Function next hops	This document

In addition, this document request IANA to create and maintain a new Registry: "Geneve List of Service Function next hops Type-Value Objects".

The following code-point are requested from the registry:

Registry: Geneve List of Service Function next hops Type-Value Objects

Suggested Value	Description	Reference
1	HMAC TLV	This document

10. References

10.1 Normative References

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

10.2 Informative References

[Geneve] "Generic Network Virtualization Encapsulation", [I-D.ietf-

nvo3-geneve]

[RFC8300] Quinn, P., Elzur, U., and C. Pignataro, "Network Service Header (NSH)", RFC 8300, January 2018, <http://www.rfc-editor.org/info/rfc8300>.

[RFC4942] Davies, E., Krishnan, S., and P. Savola, "IPv6 Transition/Co-existence Security Considerations", RFC 4942, DOI 10.17487/RFC4942, September 2007, <http://www.rfc-editor.org/info/rfc4942>.

[RFC6554] Hui, J., Vasseur, JP., Culler, D., and V. Manral, "An IPv6 Routing Header for Source Routes with the Routing Protocol for Low-Power and Lossy Networks (RPL)", RFC 6554, DOI 10.17487/RFC6554, March 2012, <http://www.rfc-editor.org/info/rfc6554>.

[draft-ietf-6man-segment-routing-header] Previdi, S., et all, "IPv6 Segment Routing Header (SRH)", July 20, 2017, draft-ietf-6man-segment-routing-header-07

[RFC5095] Abley, J., Savola, P., and G. Neville-Neil, "Deprecation of Type 0 Routing Headers in IPv6", RFC 5095, DOI 10.17487/RFC5095, December 2007, <http://www.rfc-editor.org/info/rfc5095>.

[RFC7665] Halpern, J., Ed. and C. Pignataro, Ed., "Service Function Chaining (SFC) Architecture", RFC 7665, DOI 10.17487/RFC7665, October 2015, <https://www.rfc-editor.org/info/rfc7665>.

Authors' Addresses

Sami Boutros
VMware
Email: sboutros@vmware.com

Dharma Rajan
VMware
Email: drajan@vmware.com

Philip Kippen
VMware
Email: pkippen@vmware.com

Pierluigi Rolando
VMware
Email: prolando@vmware.com

INTERNET-DRAFT
Intended Status: Standard Track

Sami Boutros, Ed.
VMware
Dharma Rajan
Philip Kippen
Pierluigi Rolando
VMware
Jim Guichard
Huawei
Sam Aldrin
Google

Expires: March 19, 2020 September 16, 2019

Geneve applicability for service function chaining
draft-boutros-nvo3-geneve-applicability-for-sfc-04

Abstract

This document describes the applicability of using Generic Network Virtualization Encapsulation (Geneve), to carry the service function path (SFP) information, and the network service header (NSH) encapsulation. The SFP information will be carried in Geneve option TLV(s).

Table of Contents

1. Introduction

The Service Function Chaining (SFC) Architecture [rfc7665] defines a service function chain (SFC) as (1) the instantiation of an ordered set of service functions and (2) the subsequent "steering" of traffic through them.

SFC defines a Service Function Path (SFP) as the exact set of service function forwarders (SFF)/service functions (SF)s the packet will visit when it actually traverses the network.

An optimized SFP helps to build an efficient Service function chain (SFC) that can be used to steer traffic based on classification rules, and metadata information to provide services for Network Function Virtualization (NFV). Metadata are typically passed between service functions and Service function forwarders SFF(s) along a service function path.

In a Network Virtualization Overlays (NVO3) domain, Network Virtualization Edges (NVE)s can be implemented on hypervisors hosting virtual network functions VNF(s) or cloud native functions CNF(s) implementing service functions, or on CNFs on bare metal servers or on physical routers connected to service function appliances. NVO3 domain uses tunneling and encapsulation protocols such as Geneve to provide connectivity for tenants workloads and service function running in its domain. NVEs in an NVO3 domain are typically controlled by a centralized network virtualization authority NVA.

[RFC8300] defines a new encapsulation protocol, network service header (NSH) to encode the SFP and the metadata.

1.1 Requirement for SFC in NVO3 domain

The requirement is to provide service function chaining in an NVO3 domain without the need to implement yet another control plane for service topology.

1.2 Proposed solution for SFC in NVO3 domain

This document specifies the applicability of using Generic Network Virtualization Encapsulation (Geneve), to carry the service function path (SFP) information, and the network service header (NSH) encapsulation.

The SFP will be implemented using a new Geneve Service Function List (SFL) option for use strictly between Network Virtualization Edges (NVEs) performing the service forwarding function (SFF) in the same Network Virtualization Overlay over Layer 3 NVO3 domain. The next

protocol in the Geneve Header will be the NSH EtherType, 0x894F. The NSH encapsulation will include the Service Path Identifier (SPI) and the Service Index (SI). The NSH SI will serve as an index to the VNF/CNF hop to visit in the SFL.

In the absence of the SFL we would need a service topology control plane. The Geneve overlay will encap the NSH encapsulation and the next protocol on Geneve will be the NSH Ethertype.

2. Terminology

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. Abbreviations

NVO3 Network Virtualization Overlays over Layer 3

OAM Operations, Administration, and Maintenance

TLV Type, Length, and Value

VNI Virtual Network Identifier

NVE Network Virtualization Edge

NVA Network Virtualization Authority

NIC Network interface card

VTEP Virtual Tunnel End Point

Transit device Underlay network devices between NVE(s).

Service Function (SF): Defined in [RFC7665].

Service Function Chain (SFC): Defined in [RFC7665].

Service Function Forwarder (SFF): Defined in [RFC7665].

Service Function Path (SFP): Defined in [RFC7665].

Metadata: Defined in [[draft-ietf-sfc-nsh]

NFV: Network function virtualization.

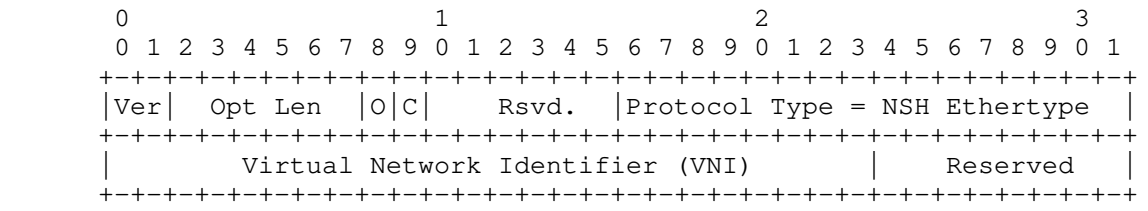
VNF: Virtual network function

CNF: Cloud native function

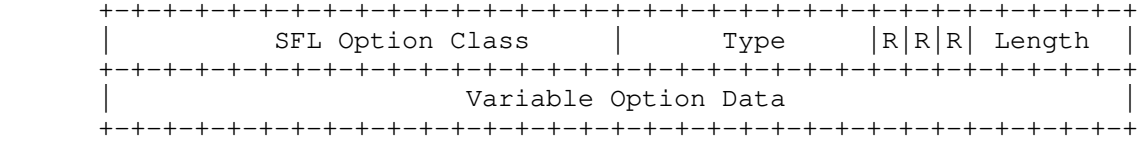
4. Geneve Option TLV(s)

4.1 Geneve Service Function List (SFL) Option TLV

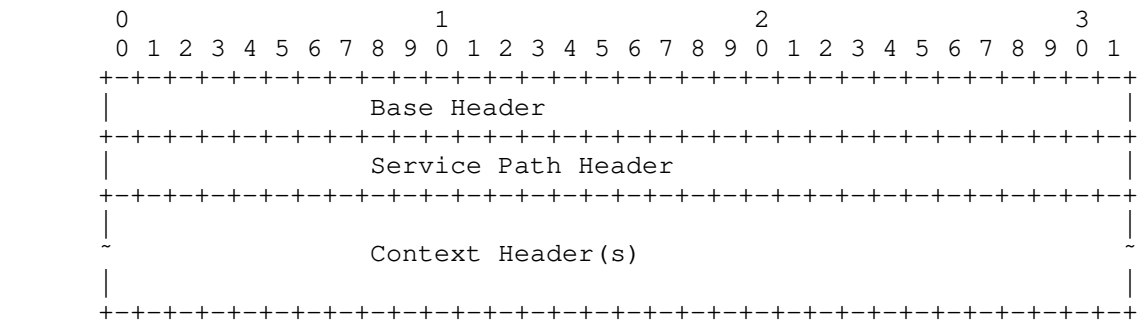
Geneve Header:



Geneve Option Header:



Followed by the NSH encapsulation which is composed of a 4-byte Base Header, a 4-byte Service Path Header, and optional Context Headers.



SFL Option Class = To be assigned by IANA

Type = To be assigned by IANA

'C' bit set, indicating endpoints must drop if they do not recognize this option)

Length = variable.

Variable option data:

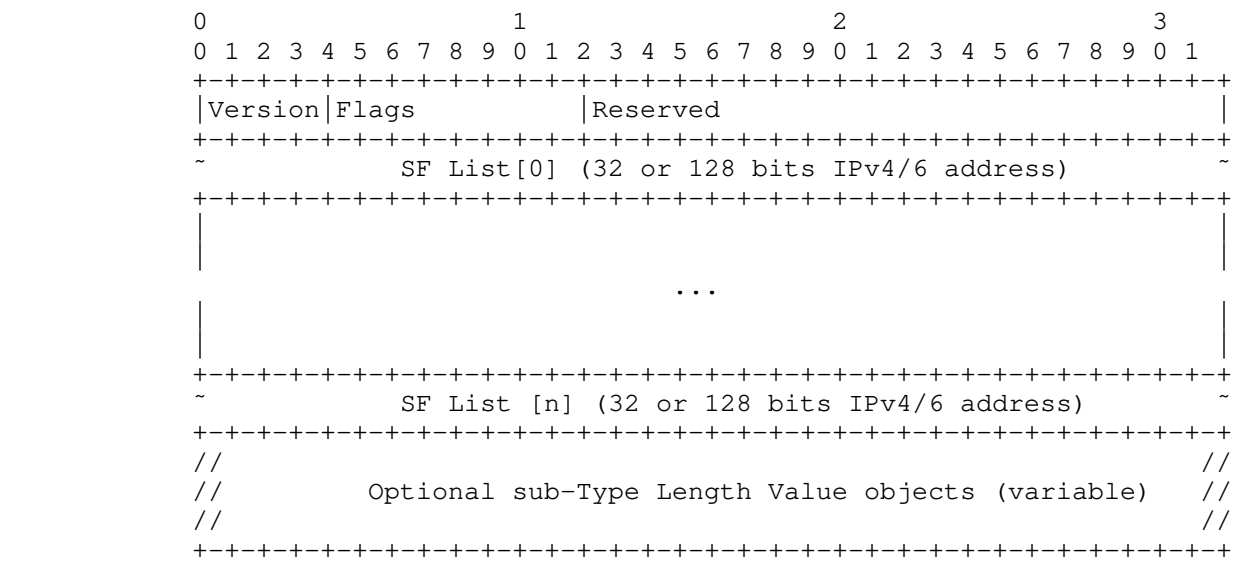


Figure 1: Service Function List (SFL) Option TLV.

Reserved: 12 bits. SHOULD be unset on transmission and MUST be ignored on receipt.

Flags:

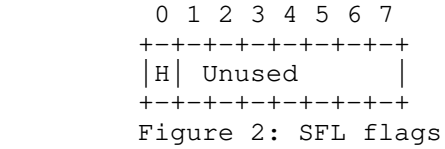


Figure 2: SFL flags

H-flag: HMAC flag. If set, the HMAC sub-TLV is present and is encoded as the last sub-TLV.

SF List[n]: 32 or 128 bits IPv4/6 addresses representing the nth service function ip address in the List.

The SF List is encoded starting from the last hop of the path. I.e., the first element of the list (SF List [0]) contains the last service function of the path while the last element of the SF List (SF List[n]) contains the first service function in the path.

HMAC sub-TLV is optional and contains the HMAC information. The

HMAC sub-TLV has the following format:

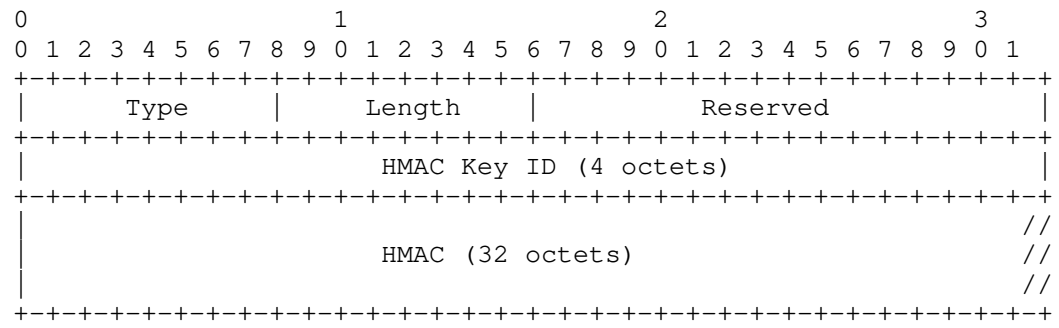


Figure 3: SFL HMAC sub-TLV.

Type: to be assigned by IANA (suggested value 1).
Length: 38.
Reserved: 2 octets. SHOULD be unset on transmission and MUST be ignored on receipt.
HMAC Key ID: 4 octets.
HMAC: 32 octets.
HMAC and HMAC Key ID usage is described in Operation section.

The Following applies to the HMAC TLV:

When present, the HMAC sub-TLV MUST be encoded as the last sub-TLV

If the HMAC sub-TLV is present, the H-Flag (Figure 2) MUST be set.

When the H-flag is set, the NVE inspecting the Geneve Service Function List Option TLV MUST find the HMAC sub-TLV in the last 38 octets of the option TLV.

5.. Operation

The mechanisms described in this section should work with both ipv4 and ipv6 for both customer inner payload and Geneve tunnel packets.

5.1 Operation at Ingress

A Source NVE acting as a service function classifier and a service function forwarder can be any node in an NVO3 domain, originating based on a classification policy for some customer inner payload an IP Geneve tunnel packet with the service function list (SFL) option TLV. The service functions in the SFL represent the IP addresses of the service functions that the inner customer packets needs to be inspected by. A controller can program the ingress NVE node to classify traffic and identify a service function paths i.e the set of

service functions in the path. The mechanism through which an SFL is derived by a controller or any other mechanisms is outside of the scope of this document.

The ingress NVE node fills in the list of service functions in the path, to the Geneve Service Function List option TLV, putting the first service function ip address as the last element in the list and the last service function ip address as the first element, setting of the NSH service index to the first element. The ingress NVE node, then, resolves the service first function ip address, to the NVE virtual tunnel endpoint node hosting or directly connected to the service function.

The Geneve tunnel destination is then set to the NVE tunnel endpoint hosting the first service function, and the service index is decremented to n-1 (where n is the number of elements in the SFL), and set on the SFL option TLV. An NSH metadata can also be set on the packet by the NVE ingress node.

The Geneve packet is sent out towards the first NVE.

HMAC optional sub-TLV may be set too.

5.2 Operation at each NVE along the service function path

The NVE node along the service function path corresponding to the Geneve tunnel destination of the packet, receives the packet, perform the service function forwarder function and identifies the SFL option, and locates the service function in the list based on the service index.

The Geneve tunnel header and option TLV(s) will be stripped and the packet will be delivered to the service function or virtual network function VNF or CNF. The NVE maintains state related to the association of the SFL option TLV and the NSH service path identifier. The packet passed to the service function encaped with the NSH header and NSH context, if the SF is NSH aware, other encapsulations like vlan or q- in-q encap may be used to pass the metadata and NSH SPI to the SF too.

When the packet comes back from the service function along with the service path identifier (SPI) context, based on SPI on the packet the NVE acting as the SFF will be able to locate the SFL option TLV.

If the metadata context indicate (1) that some service functions need to be bypassed the NVE should bypass in the SFL the service functions to be skipped and update the NSH service index accordingly. (2) A new

classification need to be performed on the packet, in that case the NVE can re-classify the packet or sent it to an NVE node capable of classification.

The NVE node, then, resolves the next service function ip address, to the NVE virtual tunnel endpoint node hosting or directly connected to the service function.

The NVE then sets the Geneve tunnel destination to the next NVE tunnel endpoint, and the NSH service index is decremented by 1 and set on the NSH Header, along with other NSH metadata option TLV.

The Geneve ip packet is sent out towards the next NVE.

5.3 Operation at Egress

At the last NVE node along the service function path, the NVE locates the service function in the SFL option TLV based on the NSH service index. The service index received at the last NVE node will be set to 1.

The Geneve tunnel header and option TLV(s) will be stripped and the packet will be delivered to the service function. The NVE maintains state related to the association of the SFL option TLV and the NSH service path identifier. The packet passed to the service function encaped with the NSH header and NSH context, if the SF is NSH aware, other encapsulations like vlan or q-in-q encap may be used to pass the metadata and NSH SPI to the SF too.

When the packet comes back from the service function, based on NSH SPI on the packet or based the NVE will be able to locate the SFL option TLV.

Given that the service index will be set to 1, the last NVE will now deliver the packet to the NVE hosting or directly connected to the inner packet destination.

A packet received with a service function index of 0 MUST be dropped.

6. Security Considerations

Only NVE(s) that are the destinations of the Geneve tunnel packet will be inspecting the List of Service Function next hops Option. A Source routing option has some well-known security issues as described in [RFC4942] and [RFC5095].

The main use case for the use of the Geneve List of Service Function next hops Option will be within a single NVO3 administrative domain

where only trusted NVE nodes are enabled and configured participate, this is the same model as in [RFC6554].

NVE nodes MUST ignore the Geneve List of Service Function next hops Option created by outsiders based on NVA or trusted control plane information.

There is a need to prevent non-participating NVE node from using the Geneve Service Function List option TLV, as described in [draft-ietf-6man-segment-routing-header], we will use a security sub-TLV in the Service Function List option TLV, the security sub-TLV will be based on a key-hashed message authentication code (HMAC).

HMAC sub-TLV will contain:

HMAC Key-id, 32 bits wide;

HMAC, 256 bits wide (optional, exists only if HMAC Key-id is not 0).

The HMAC field is the output of the HMAC computation (per RFC 2104 [RFC2104]) using a pre-shared key identified by HMAC Key-id and of the text which consists of the concatenation of:

The source IPv4/IPv6 Geneve tunnel address

Version and Flags

HMAC Key-id.

All addresses in the List.

The purpose of the HMAC optional sub-TLV is to verify the validity, the integrity and the authorization of the Geneve Service Function List option TLV itself.

The HMAC optional sub-TLV is located at the end of the Geneve Service Function List option TLV.

The HMAC Key-id field serves as an index to the right combination of pre-shared key and hash algorithm and except that a value of 0 means that there is no HMAC field.

The HMAC Selection of a hash algorithm and Pre-shared key management will follow the procedures described in [draft-ietf-6man-segment-routing-header] section 6.2.

7. Management Considerations

The Source NVE can receive its information through any form of north bound Orchestrator. These could be from any open networking automation platform (ONAP) or others. The ingress to egress tunnel is built and managed by the service function classifier and service function forwarder by each node in an NVO3 domain. Error handling, is handled by the classifier reporting to north bound management systems.

8. Acknowledgements

The authors would like to acknowledge Jim Guichard for his feedback and valuable comments to this document.

9. IANA Considerations

This document makes the following registrations in the "Geneve Option Class" registry maintained by IANA:

Suggested Value	Description	Reference
XX	Geneve List of Service Function next hops	This document

In addition, this document request IANA to create and maintain a new Registry: "Geneve List of Service Function next hops Type-Value Objects".

The following code-point are requested from the registry:

Registry: Geneve List of Service Function next hops Type-Value Objects

Suggested Value	Description	Reference
1	HMAC TLV	This document

10. References

10.1 Normative References

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

10.2 Informative References

[Geneve] "Generic Network Virtualization Encapsulation", [I-D.ietf-

nvo3-geneve]

[RFC8300] Quinn, P., Elzur, U., and C. Pignataro, "Network Service Header (NSH)", RFC 8300, January 2018, <http://www.rfc-editor.org/info/rfc8300>.

[RFC4942] Davies, E., Krishnan, S., and P. Savola, "IPv6 Transition/Co-existence Security Considerations", RFC 4942, DOI 10.17487/RFC4942, September 2007, <http://www.rfc-editor.org/info/rfc4942>.

[RFC6554] Hui, J., Vasseur, JP., Culler, D., and V. Manral, "An IPv6 Routing Header for Source Routes with the Routing Protocol for Low-Power and Lossy Networks (RPL)", RFC 6554, DOI 10.17487/RFC6554, March 2012, <http://www.rfc-editor.org/info/rfc6554>.

[draft-ietf-6man-segment-routing-header] Previdi, S., et all, "IPv6 Segment Routing Header (SRH)", July 20, 2017, draft-ietf-6man-segment-routing-header-07

[RFC5095] Abley, J., Savola, P., and G. Neville-Neil, "Deprecation of Type 0 Routing Headers in IPv6", RFC 5095, DOI 10.17487/RFC5095, December 2007, <http://www.rfc-editor.org/info/rfc5095>.

[RFC7665] Halpern, J., Ed. and C. Pignataro, Ed., "Service Function Chaining (SFC) Architecture", RFC 7665, DOI 10.17487/RFC7665, October 2015, <https://www.rfc-editor.org/info/rfc7665>.

Authors' Addresses

Sami Boutros
VMware
Email: boutross@vmware.com

Dharma Rajan
VMware
Email: drajan@vmware.com

Philip Kippen
VMware
Email: pkippen@vmware.com

Pierluigi Rolando
VMware
Email: prolando@vmware.com

Jim Guichard
Huawei
Email: james.n.guichard@huawei.com

Sam Aldrin
Google
Email:aldrin.ietf@gmail.com

ippm
Internet-Draft
Intended status: Standards Track
Expires: December 29, 2018

F. Brockners
S. Bhandari
V. Govindan
C. Pignataro
Cisco
H. Gredler
RtBrick Inc.
J. Leddy
Comcast
S. Youell
JMPC
T. Mizrahi
Marvell
P. Lapukhov
Facebook
B. Gafni
A. Kfir
Mellanox Technologies, Inc.
M. Spiegel
Barefoot Networks
June 27, 2018

Geneve encapsulation for In-situ OAM Data
draft-brockners-ippm-ioam-geneve-01

Abstract

In-situ Operations, Administration, and Maintenance (IOAM) records operational and telemetry information in the packet while the packet traverses a path between two points in the network. This document outlines how IOAM data fields are encapsulated in Geneve.

This Internet-Draft will expire on December 29, 2018.

Table of Contents

1. Introduction

In-situ OAM (IOAM) records OAM information within the packet while the packet traverses a particular network domain. The term "in-situ" refers to the fact that the IOAM data fields are added to the data packets rather than is being sent within packets specifically dedicated to OAM. This document defines how IOAM data fields are transported as part of the Geneve [I-D.ietf-nvo3-geneve] encapsulation. The IOAM data fields are defined in [I-D.ietf-ippm-ioam-data].

2. Conventions

2.1. Requirement Language

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 [RFC2119].

2.2. Abbreviations

Abbreviations used in this document:

IOAM: In-situ Operations, Administration, and Maintenance

OAM: Operations, Administration, and Maintenance

Geneve: Generic Network Virtualization Encapsulation

3. IOAM Data Field Encapsulation in Geneve

Geneve is defined in [I-D.ietf-nvo3-geneve]. IOAM data fields are carried in the Geneve header as a tunnel option, using a single Geneve Option Class TBD_IOAM. The different IOAM data fields defined in [I-D.ietf-ippm-ioam-data] are added as TLVs using that Geneve Option Class. In an administrative domain where IOAM is used, insertion of the IOAM header in Geneve is enabled at the Geneve tunnel endpoints, which also serve as IOAM encapsulating/ decapsulating nodes by means of configuration.

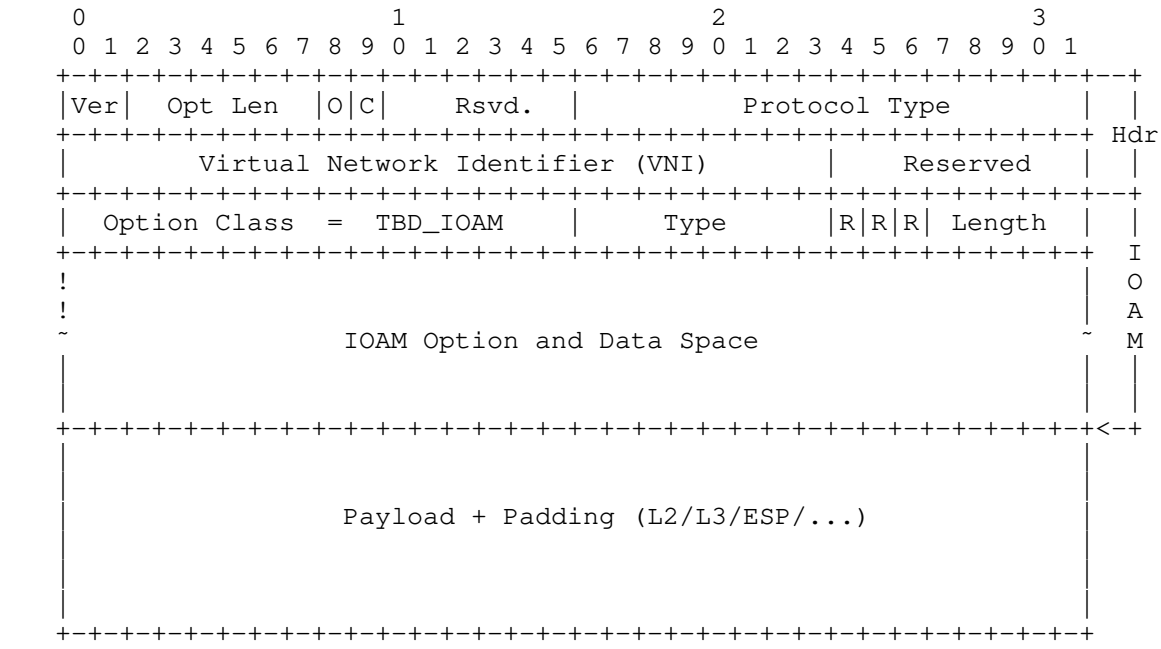


Figure 1: IOAM data encapsulation in Geneve

The Geneve header and fields are defined in [I-D.ietf-nvo3-geneve]. The Geneve Option Class value for use with IOAM is TBD_IOAM.

The fields related to the encapsulation of IOAM data fields in Geneve are defined as follows:

Option Class: 16-bit unsigned integer that determines the IOAM option class. The value is from the IANA registry setup for Geneve option classes as defined in [I-D.ietf-nvo3-geneve].

Type: 8-bit field defining the IOAM Option type, as defined in Section 7.2 of [I-D.ietf-ippm-ioam-data].

R (3 bits): Option control flags reserved for future use. MUST be zero on transmission and ignored on receipt.

Length: 5-bit unsigned integer. Length of the IOAM HDR in 4-octet units.

IOAM Option and Data Space: IOAM option header and data is present as defined by the Type field, and is defined in Section 4 of [I-D.ietf-ippm-ioam-data].

Multiple IOAM options MAY be included within the Geneve encapsulation. For example, if a Geneve encapsulation contains two IOAM options before a data payload, there would be two fields with TBD_IOAM Option Class each, differentiated by the Type field which specifies the type of the IOAM data included.

4. Considerations

This section summarizes a set of considerations on the overall approach taken for IOAM data encapsulation in Geneve, as well as deployment considerations.

4.1. Discussion of the encapsulation approach

This section is to support the working group discussion in selecting the most appropriate approach for encapsulating IOAM data fields in Geneve.

An encapsulation of IOAM data fields in Geneve should be friendly to an implementation in both hardware as well as software forwarders and support a wide range of deployment cases, including large networks that desire to leverage multiple IOAM data fields at the same time.

Hardware and software friendly implementation: Hardware forwarders benefit from an encapsulation that minimizes iterative look-ups of fields within the packet: Any operation which looks up the value of a field within the packet, based on which another lookup is performed, consumes additional gates and time in an implementation - both of which are desired to be kept to a minimum. This means that flat TLV structures are to be preferred over nested TLV structures. IOAM data fields are grouped into three option categories: Trace, proof-of-transit, and edge-to-edge. Each of these three options defines a TLV structure. A hardware-friendly encapsulation approach avoids grouping these three option categories into yet another TLV structure, but would rather carry the options as a serial sequence.

Total length of the IOAM data fields: The total length of IOAM data can grow quite large in case multiple different IOAM data fields are used and large path-lengths need to be considered. If for example an operator would consider using the IOAM trace option and capture node-id, app_data, egress/ingress interface-id, timestamp seconds, timestamps nanoseconds at every hop, then a total of 20 octets would be added to the packet at every hop. In case this particular deployment would have a maximum path length of 15 hops in the IOAM domain, then a maximum of 300 octets of IOAM data were to be encapsulated in the packet.

Concerns with the current encapsulation approach:

Hardware support: Using Geneve tunnel options to encapsulate IOAM data fields leads to a nested TLV structure. Each IOAM data field option (trace, proof-of-transit, and edge-to-edge) represents a type, with the different IOAM data fields being TLVs within this the particular option type. Nested TLVs require iterative look-ups, a fact that creates potential challenges for implementations in hardware. It would be desirable to offer a way to encapsulate IOAM in a way that keeps TLV nesting to a minimum.

Length: Geneve tunnel option length is a 5-bit field in the current specification [I-D.ietf-nvo3-geneve] resulting in a maximum option length of 128 (2^5 x 4) octets which constrains the use of IOAM to either small domains or a few IOAM data fields only. Support for large domains with a variety of IOAM data fields would be desirable.

4.2. IOAM and the use of the Geneve O-bit

[I-D.ietf-nvo3-geneve] defines an "O bit" for OAM packets. Per [I-D.ietf-nvo3-geneve] the O bit indicates that the packet contains a control message instead of data payload. Packets that carry IOAM data fields in addition to regular data payload / customer traffic must not set the O bit. Packets that carry only IOAM data fields without any payload must set the O bit.

4.3. Transit devices

If IOAM is deployed in domains where UDP port numbers are not controlled and do not have a domain-wide meaning, such as on the global Internet, transit devices MUST NOT attempt to modify the IOAM data contained in the IOAM option class. In case UDP port numbers are not controlled there might be UDP packets, which leverage the UDP port number that Geneve utilizes, i.e. 6081, but the payload of these packets isn't Geneve. The scenario and associated reasoning is discussed in [RFC7605] which states that "it is important to recognize that any interpretation of port numbers -- except at the endpoints -- may be incorrect, because port numbers are meaningful only at the endpoints."

5. IANA Considerations

IANA is requested to allocate a Geneve "option class" numbers for IOAM:

Option Class	Description	Reference
x	TBD_IOAM	This document

6. Security Considerations

The security considerations of Geneve are discussed in [I-D.ietf-nvo3-geneve], and the security considerations of IOAM in general are discussed in [I-D.ietf-ippm-ioam-data].

IOAM is considered a "per domain" feature, where one or several operators decide on leveraging and configuring IOAM according to their needs. Still, operators need to properly secure the IOAM domain to avoid malicious configuration and use, which could include injecting malicious IOAM packets into a domain.

7. Acknowledgements

The authors would like to thank Eric Vyncke, Nalini Elkins, Srihari Raghavan, Ranganathan T S, Karthik Babu Harichandra Babu, Akshaya Nadahalli, Stefano Previdi, Hemant Singh, Erik Nordmark, LJ Wobker, and Andrew Yourtchenko for the comments and advice.

8. Normative References

[I-D.ietf-ippm-ioam-data]
Brockners, F., Bhandari, S., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Mozes, D., Lapukhov, P., Chang, R., daniel.bernier@bell.ca, d., and J. Lemon, "Data Fields for In-situ OAM", draft-ietf-ippm-ioam-data-02 (work in progress), March 2018.

[I-D.ietf-nvo3-geneve]
Gross, J., Ganga, I., and T. Sridhar, "Geneve: Generic Network Virtualization Encapsulation", draft-ietf-nvo3-geneve-06 (work in progress), March 2018.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC2784] Farinacci, D., Li, T., Hanks, S., Meyer, D., and P. Traina, "Generic Routing Encapsulation (GRE)", RFC 2784, DOI 10.17487/RFC2784, March 2000, <https://www.rfc-editor.org/info/rfc2784>.

[RFC3232] Reynolds, J., Ed., "Assigned Numbers: RFC 1700 is Replaced by an On-line Database", RFC 3232, DOI 10.17487/RFC3232, January 2002, <https://www.rfc-editor.org/info/rfc3232>.

[RFC7605] Touch, J., "Recommendations on Using Assigned Transport Port Numbers", BCP 165, RFC 7605, DOI 10.17487/RFC7605, August 2015, <https://www.rfc-editor.org/info/rfc7605>.

Authors' Addresses

Frank Brockners
Cisco Systems, Inc.
Hansaallee 249, 3rd Floor
DUESSELDORF, NORDRHEIN-WESTFALEN 40549
Germany

Email: fbrockne@cisco.com

Shwetha Bhandari
Cisco Systems, Inc.
Cessna Business Park, Sarjapura Marathalli Outer Ring Road
Bangalore, KARNATAKA 560 087
India

Email: shwethab@cisco.com

Vengada Prasad Govindan
Cisco Systems, Inc.

Email: venggovi@cisco.com

Carlos Pignataro
Cisco Systems, Inc.
7200-11 Kit Creek Road
Research Triangle Park, NC 27709
United States

Email: cpignata@cisco.com

Hannes Gredler
RtBrick Inc.

Email: hannes@rtbrick.com

John Leddy
Comcast

Email: John_Leddy@cable.comcast.com

Stephen Youell
JP Morgan Chase
25 Bank Street
London E14 5JP
United Kingdom

Email: stephen.youell@jpmorgan.com

Tal Mizrahi
Marvell
6 Hamada St.
Yokneam 20692
Israel

Email: talmi@marvell.com

Petr Lapukhov
Facebook
1 Hacker Way
Menlo Park, CA 94025
US

Email: petr@fb.com

Barak Gafni
Mellanox Technologies, Inc.
350 Oakmead Parkway, Suite 100
Sunnyvale, CA 94085
U.S.A.

Email: gbarak@mellanox.com

Aviv Kfir
Mellanox Technologies, Inc.
350 Oakmead Parkway, Suite 100
Sunnyvale, CA 94085
U.S.A.

Email: avivk@mellanox.com

Mickey Spiegel
Barefoot Networks
2185 Park Boulevard
Palo Alto, CA 94306
US

Email: mspiegel@barefootnetworks.com

ippm
Internet-Draft
Intended status: Standards Track
Expires: May 23, 2021

F. Brockners, Ed.
Cisco
S. Bhandari
Thoughtspot
V. Govindan
C. Pignataro, Ed.
N. Nainar, Ed.
Cisco
H. Gredler
RtBrick Inc.
J. Leddy

S. Youell
JMPC
T. Mizrahi
Huawei Network.IO Innovation Lab
P. Lapukhov
Facebook
B. Gafni
A. Kfir
Mellanox Technologies, Inc.
M. Spiegel
Barefoot Networks, an Intel company

November 19, 2020

Geneve encapsulation for In-situ OAM Data
draft-brockners-ippm-ioam-geneve-05

Abstract

In-situ Operations, Administration, and Maintenance (IOAM) records operational and telemetry information in the packet while the packet traverses a path between two points in the network. This document proposes a new Geneve tunnel option and outlines how IOAM data fields are carried in the option data field.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Table of Contents

1. Introduction

In-situ OAM (IOAM) records OAM information within the packet while the packet traverses a particular network domain. The term "in-situ" refers to the fact that the IOAM data fields are added to the data packets rather than is being sent within packets specifically dedicated to OAM. This document proposes a new Geneve tunnel option and defines how IOAM data fields are transported as part of the

tunnel option in the Geneve [I-D.ietf-nvo3-geneve] encapsulation. The IOAM data fields are defined in [I-D.ietf-ippm-ioam-data].

2. Conventions

2.1. Requirement Language

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 [RFC2119].

2.2. Abbreviations

Abbreviations used in this document:

IOAM: In-situ Operations, Administration, and Maintenance

OAM: Operations, Administration, and Maintenance

Geneve: Generic Network Virtualization Encapsulation

3. IOAM Data Field Encapsulation in Geneve

Geneve is defined in [I-D.ietf-nvo3-geneve]. When Geneve encapsulation header is used, IOAM data fields can either be carried after the Geneve header, identified by the next protocol field or can be carried in the Geneve header itself as a tunnel option. The former approach is defined in [I-D.weis-ippm-ioam-eth] while the latter approach is defined in this document.

IOAM data fields are carried using a single Geneve Option Class TBD_IOAM. The different IOAM data fields defined in [I-D.ietf-ippm-ioam-data] are added as TLVs using that Geneve Option Class. In an administrative domain where IOAM is used, insertion of the IOAM header in Geneve is enabled at the Geneve tunnel endpoints, which also serve as IOAM encapsulating/decapsulating nodes by means of configuration.

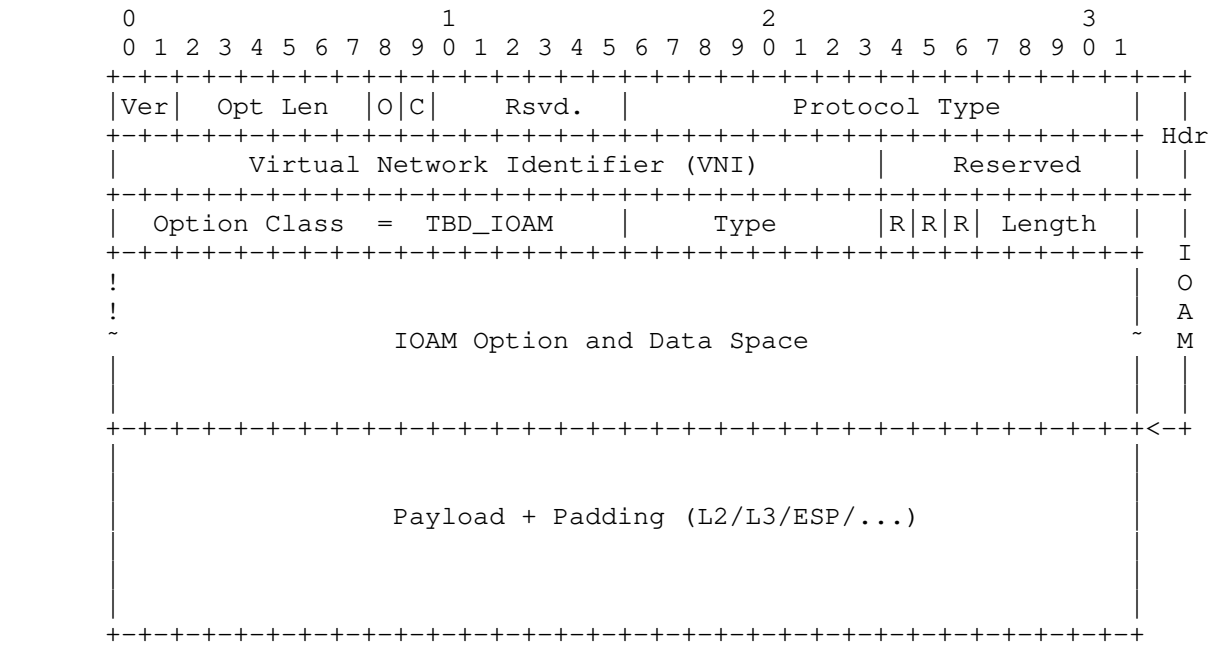


Figure 1: IOAM data encapsulation in Geneve

The Geneve header and fields are defined in [I-D.ietf-nvo3-geneve]. The Geneve Option Class value for use with IOAM is TBD_IOAM.

The fields related to the encapsulation of IOAM data fields in Geneve are defined as follows:

Option Class: 16-bit unsigned integer that determines the IOAM option class. The value is from the IANA registry setup for Geneve option classes as defined in [I-D.ietf-nvo3-geneve].

Type: 8-bit field defining the IOAM Option type, as defined in Section 7.2 of [I-D.ietf-ippm-ioam-data]. The 'C' bit MUST be set to zero.

R (3 bits): Option control flags reserved for future use. The flags MUST be set to zero on transmission and ignored on receipt.

Length: 5-bit unsigned integer. Length of the IOAM HDR in 4-octet units.

IOAM Option and Data Space: IOAM option header and data is present as defined by the Type field, and is defined in Section 4 of [I-D.ietf-ippm-ioam-data].

Multiple distinct IOAM Option-Types MAY be included within the same Geneve encapsulation. Each IOAM Option-Type MUST occur at most once within the same Geneve encapsulation. For example, if a Geneve encapsulation contains two IOAM Option-Types before a data payload, there would be two fields with TBD_IOAM Option Class each, differentiated by the Type field which specifies the type of the IOAM data included.

4. Considerations

This section summarizes a set of considerations on the overall approach taken for IOAM data encapsulation in Geneve, as well as deployment considerations.

4.1. Discussion of the encapsulation approach

This section is to support the working group discussion in selecting the most appropriate approach for encapsulating IOAM data fields in Geneve.

An encapsulation of IOAM data fields in Geneve should be friendly to an implementation in both hardware as well as software forwarders and support a wide range of deployment cases, including large networks that desire to leverage multiple IOAM data fields at the same time.

Hardware and software friendly implementation: Hardware forwarders benefit from an encapsulation that minimizes iterative look-ups of fields within the packet: Any operation which looks up the value of a field within the packet, based on which another lookup is performed, consumes additional gates and time in an implementation - both of which are desired to be kept to a minimum. This means that flat TLV structures are to be preferred over nested TLV structures. IOAM data fields are grouped into three option categories: Trace, proof-of-transit, and edge-to-edge. Each of these three options defines a TLV structure. A hardware-friendly encapsulation approach avoids grouping these three option categories into yet another TLV structure, but would rather carry the options as a serial sequence.

Total length of the IOAM data fields: The total length of IOAM data can grow quite large in case multiple different IOAM data fields are used and large path-lengths need to be considered. If for example an operator would consider using the IOAM trace option and capture node-id, app_data, egress/ingress interface-id, timestamp seconds, timestamps nanoseconds at every hop, then a total of 20 octets would be added to the packet at every hop. In case this particular deployment would have a maximum path length

of 15 hops in the IOAM domain, then a maximum of 300 octets of IOAM data were to be encapsulated in the packet.

Concerns with the current encapsulation approach:

Hardware support: Using Geneve tunnel options to encapsulate IOAM data fields leads to a nested TLV structure. Each IOAM data field option (trace, proof-of-transit, and edge-to-edge) represents a type, with the different IOAM data fields being TLVs within this the particular option type. Nested TLVs require iterative look-ups, a fact that creates potential challenges for implementations in hardware. It would be desirable to offer a way to encapsulate IOAM in a way that keeps TLV nesting to a minimum.

Length: Geneve tunnel option length is a 5-bit field in the current specification [I-D.ietf-nvo3-geneve] resulting in a maximum option length of 128 (2^5 x 4) octets which constrains the use of IOAM to either small domains or a few IOAM data fields only. Support for large domains with a variety of IOAM data fields would be desirable.

4.2. IOAM and the use of the Geneve O-bit

[I-D.ietf-nvo3-geneve] defines an "O bit" for Control packets. Per [I-D.ietf-nvo3-geneve] the O bit indicates that the packet contains a control message instead of data payload. Packets that carry IOAM data fields in addition to regular data payload / customer traffic must not set the O bit. Packets that carry only IOAM data fields without any payload must set the O bit.

4.3. Transit devices

If IOAM is deployed in domains where UDP port numbers are not controlled and do not have a domain-wide meaning, such as on the global Internet, transit devices MUST NOT attempt to modify the IOAM data contained in the IOAM option class. In case UDP port numbers are not controlled there might be UDP packets, which leverage the UDP port number that Geneve utilizes, i.e. 6081, but the payload of these packets isn't Geneve. The scenario and associated reasoning is discussed in [RFC7605] which states that "it is important to recognize that any interpretation of port numbers -- except at the endpoints -- may be incorrect, because port numbers are meaningful only at the endpoints."

4.4. Additional Encapsulation Consideration

Geneve encapsulation header supports carrying IOAM data fields either as part of the tunnel option or as the protocol data unit that follows the Geneve header. An operator may choose to enable either of the options but it is not recommended to include both in the same data packet.

5. IANA Considerations

IANA is requested to allocate a Geneve "option class" numbers for IOAM:

Option Class	Description	Reference
x	TBD_IOAM	This document

6. Security Considerations

The security considerations of Geneve are discussed in [I-D.ietf-nvo3-geneve], and the security considerations of IOAM in general are discussed in [I-D.ietf-ippm-ioam-data].

IOAM is considered a "per domain" feature, where one or several operators decide on leveraging and configuring IOAM according to their needs. Still, operators need to properly secure the IOAM domain to avoid malicious configuration and use, which could include injecting malicious IOAM packets into a domain.

7. Acknowledgements

The authors would like to thank Eric Vyncke, Nalini Elkins, Srihari Raghavan, Ranganathan T S, Karthik Babu Harichandra Babu, Akshaya Nadahalli, Stefano Previdi, Hemant Singh, Erik Nordmark, LJ Wobker, Andrew Yourtchenko and Nagendra Kumar Nainar for the comments and advice.

8. Normative References

[I-D.ietf-ippm-ioam-data]
Brockners, F., Bhandari, S., and T. Mizrahi, "Data Fields for In-situ OAM", draft-ietf-ippm-ioam-data-10 (work in progress), July 2020.

[I-D.ietf-nvo3-geneve]
Gross, J., Ganga, I., and T. Sridhar, "Geneve: Generic Network Virtualization Encapsulation", draft-ietf-nvo3-geneve-16 (work in progress), March 2020.

[I-D.weis-ippm-ioam-eth]
Weis, B., Brockners, F., Hill, C., Bhandari, S., Govindan, V., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Kfir, A., Gafni, B., Lapukhov, P., and M. Spiegel, "EtherType Protocol Identification of In-situ OAM Data", draft-weis-ippm-ioam-eth-04 (work in progress), May 2020.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC2784] Farinacci, D., Li, T., Hanks, S., Meyer, D., and P. Traina, "Generic Routing Encapsulation (GRE)", RFC 2784, DOI 10.17487/RFC2784, March 2000, <https://www.rfc-editor.org/info/rfc2784>.

[RFC3232] Reynolds, J., Ed., "Assigned Numbers: RFC 1700 is Replaced by an On-line Database", RFC 3232, DOI 10.17487/RFC3232, January 2002, <https://www.rfc-editor.org/info/rfc3232>.

[RFC7605] Touch, J., "Recommendations on Using Assigned Transport Port Numbers", BCP 165, RFC 7605, DOI 10.17487/RFC7605, August 2015, <https://www.rfc-editor.org/info/rfc7605>.

Authors' Addresses

Frank Brockners (editor)
Cisco Systems, Inc.
Hansaallee 249, 3rd Floor
DUESSELDORF, NORDRHEIN-WESTFALEN 40549
Germany

Email: fbrockne@cisco.com

Shwetha Bhandari
Thoughtspot
3rd Floor, Indiqube Orion, 24th Main Rd, HSR Layout
Bangalore, KARNATAKA 560 102
India

Email: shwetha.bhandari@thoughtspot.com

Vengada Prasad Govindan
Cisco Systems, Inc.

Email: venggovi@cisco.com

Carlos Pignataro (editor)
Cisco Systems, Inc.
7200-11 Kit Creek Road
Research Triangle Park, NC 27709
United States

Email: cpignata@cisco.com

Nagendra Kumar Nainar (editor)
Cisco Systems, Inc.
7200-11 Kit Creek Road
Research Triangle Park, NC 27709
United States

Email: naikumar@cisco.com

Hannes Gredler
RtBrick Inc.

Email: hannes@rtbrick.com

John Leddy
United States

Email: john@leddy.net

Stephen Youell
JP Morgan Chase
25 Bank Street
London E14 5JP
United Kingdom

Email: stephen.youell@jpmorgan.com

Tal Mizrahi
Huawei Network.IO Innovation Lab
Israel

Email: tal.mizrahi.phd@gmail.com

Petr Lapukhov
Facebook
1 Hacker Way
Menlo Park, CA 94025
US

Email: petr@fb.com

Barak Gafni
Mellanox Technologies, Inc.
350 Oakmead Parkway, Suite 100
Sunnyvale, CA 94085
U.S.A.

Email: gbarak@mellanox.com

Aviv Kfir
Mellanox Technologies, Inc.
350 Oakmead Parkway, Suite 100
Sunnyvale, CA 94085
U.S.A.

Email: avivk@mellanox.com

Mickey Spiegel
Barefoot Networks, an Intel company
4750 Patrick Henry Drive
Santa Clara, CA 95054
US

Email: mickey.spiegel@intel.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 10, 2019

J. Gross, Ed.
I. Ganga, Ed.
Intel
T. Sridhar, Ed.
VMware
October 07, 2018

Geneve: Generic Network Virtualization Encapsulation
draft-ietf-nvo3-geneve-08

Abstract

Network virtualization involves the cooperation of devices with a wide variety of capabilities such as software and hardware tunnel endpoints, transit fabrics, and centralized control clusters. As a result of their role in tying together different elements in the system, the requirements on tunnels are influenced by all of these components. Flexibility is therefore the most important aspect of a tunnel protocol if it is to keep pace with the evolution of the system. This draft describes Geneve, a protocol designed to recognize and accommodate these changing capabilities and needs.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 10, 2019.

Table of Contents

1. Introduction

Networking has long featured a variety of tunneling, tagging, and other encapsulation mechanisms. However, the advent of network virtualization has caused a surge of renewed interest and a corresponding increase in the introduction of new protocols. The large number of protocols in this space, ranging all the way from VLANs [IEEE.802.1Q_2014] and MPLS [RFC3031] through the more recent VXLAN [RFC7348], NVGRE [RFC7637], often leads to questions about the need for new encapsulation formats and what it is about network virtualization in particular that leads to their proliferation.

While many encapsulation protocols seek to simply partition the underlay network or bridge between two domains, network virtualization views the transit network as providing connectivity between multiple components of a distributed system. In many ways this system is similar to a chassis switch with the IP underlay network playing the role of the backplane and tunnel endpoints on the edge as line cards. When viewed in this light, the requirements placed on the tunnel protocol are significantly different in terms of the quantity of metadata necessary and the role of transit nodes.

Current work such as VL2 [VL2] and the NVO3 working group [I-D.ietf-nvo3-dataplane-requirements] have described some of the properties that the data plane must have to support network virtualization. However, one additional defining requirement is the need to carry system state along with the packet data. The use of some metadata is certainly not a foreign concept - nearly all protocols used for virtualization have at least 24 bits of identifier space as a way to partition between tenants. This is often described as overcoming the limits of 12-bit VLANs, and when seen in that context, or any context where it is a true tenant identifier, 16 million possible entries is a large number. However, the reality is that the metadata is not exclusively used to identify tenants and encoding other information quickly starts to crowd the space. In fact, when compared to the tags used to exchange metadata between line cards on a chassis switch, 24-bit identifiers start to look quite small. There are nearly endless uses for this metadata, ranging from storing input ports for simple security policies to service based context for interposing advanced middleboxes.

Existing tunnel protocols have each attempted to solve different aspects of these new requirements, only to be quickly rendered out of date by changing control plane implementations and advancements. Furthermore, software and hardware components and controllers all

have different advantages and rates of evolution - a fact that should be viewed as a benefit, not a liability or limitation. This draft describes Geneve, a protocol which seeks to avoid these problems by providing a framework for tunneling for network virtualization rather than being prescriptive about the entire system.

1.1. Requirements Language

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 [RFC2119].

In this document, these words will appear with that interpretation only when in ALL CAPS. Lower case uses of these words are not to be interpreted as carrying RFC-2119 significance.

1.2. Terminology

The NVO3 framework [RFC7365] defines many of the concepts commonly used in network virtualization. In addition, the following terms are specifically meaningful in this document:

Checksum offload. An optimization implemented by many NICs which enables computation and verification of upper layer protocol checksums in hardware on transmit and receive, respectively. This typically includes IP and TCP/UDP checksums which would otherwise be computed by the protocol stack in software.

Clos network. A technique for composing network fabrics larger than a single switch while maintaining non-blocking bandwidth across connection points. ECMP is used to divide traffic across the multiple links and switches that constitute the fabric. Sometimes termed "leaf and spine" or "fat tree" topologies.

ECMP. Equal Cost Multipath. A routing mechanism for selecting from among multiple best next hop paths by hashing packet headers in order to better utilize network bandwidth while avoiding reordering a single stream.

Geneve. Generic Network Virtualization Encapsulation. The tunnel protocol described in this draft.

LRO. Large Receive Offload. The receive-side equivalent function of LSO, in which multiple protocol segments (primarily TCP) are coalesced into larger data units.

NIC. Network Interface Card. A NIC could be part of a tunnel endpoint or transit device and can either process Geneve packets or aid in the processing of Geneve packets.

OAM. Operations, Administration, and Management. A suite of tools used to monitor and troubleshoot network problems.

Transit device. A forwarding element along the path of the tunnel making up part of the Underlay Network. A transit device MAY be capable of understanding the Geneve packet format but does not originate or terminate Geneve packets.

LSO. Large Segmentation Offload. A function provided by many commercial NICs that allows data units larger than the MTU to be passed to the NIC to improve performance, the NIC being responsible for creating smaller segments of size less than or equal to the MTU with correct protocol headers. When referring specifically to TCP/IP, this feature is often known as TSO (TCP Segmentation Offload).

Tunnel endpoint. A component performing encapsulation and decapsulation of packets, such as Ethernet frames or IP datagrams, in Geneve headers. As the ultimate consumer of any tunnel metadata, endpoints have the highest level of requirements for parsing and interpreting tunnel headers. Tunnel endpoints may consist of either software or hardware implementations or a combination of the two. Endpoints are frequently a component of an NVE but may also be found in middleboxes or other elements making up an NVO3 Network.

VM. Virtual Machine.

2. Design Requirements

Geneve is designed to support network virtualization use cases, where tunnels are typically established to act as a backplane between the virtual switches residing in hypervisors, physical switches, or middleboxes or other appliances. An arbitrary IP network can be used as an underlay although Clos networks composed using ECMP links are a common choice to provide consistent bisectional bandwidth across all connection points. Figure 1 shows an example of a hypervisor, top of rack switch for connectivity to physical servers, and a WAN uplink connected using Geneve tunnels over a simplified Clos network. These tunnels are used to encapsulate and forward frames from the attached components such as VMs or physical links.

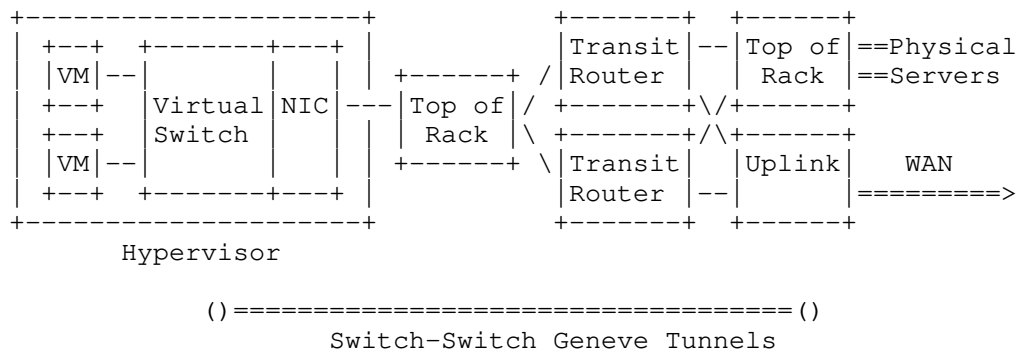


Figure 1: Sample Geneve Deployment

To support the needs of network virtualization, the tunnel protocol should be able to take advantage of the differing (and evolving) capabilities of each type of device in both the underlay and overlay networks. This results in the following requirements being placed on the data plane tunneling protocol:

- The data plane is generic and extensible enough to support current and future control planes.

- Tunnel components are efficiently implementable in both hardware and software without restricting capabilities to the lowest common denominator.

- High performance over existing IP fabrics.

These requirements are described further in the following subsections.

2.1. Control Plane Independence

Although some protocols for network virtualization have included a control plane as part of the tunnel format specification (most notably, the original VXLAN spec prescribed a multicast learning-based control plane), these specifications have largely been treated as describing only the data format. The VXLAN packet format has actually seen a wide variety of control planes built on top of it.

There is a clear advantage in settling on a data format: most of the protocols are only superficially different and there is little advantage in duplicating effort. However, the same cannot be said of control planes, which are diverse in very fundamental ways. The case for standardization is also less clear given the wide variety in requirements, goals, and deployment scenarios.

As a result of this reality, Geneve aims to be a pure tunnel format specification that is capable of fulfilling the needs of many control planes by explicitly not selecting any one of them. This simultaneously promotes a shared data format and increases the chances that it will not be obsoleted by future control plane enhancements.

2.2. Data Plane Extensibility

Achieving the level of flexibility needed to support current and future control planes effectively requires an options infrastructure to allow new metadata types to be defined, deployed, and either finalized or retired. Options also allow for differentiation of products by encouraging independent development in each vendor's core specialty, leading to an overall faster pace of advancement. By far the most common mechanism for implementing options is Type-Length-Value (TLV) format.

It should be noted that while options can be used to support non-wirespeed control packets, they are equally important on data packets as well to segregate and direct forwarding (for instance, the examples given before of input port based security policies and service interposition both require tags to be placed on data packets). Therefore, while it would be desirable to limit the extensibility to only control packets for the purposes of simplifying the datapath, that would not satisfy the design requirements.

2.2.1. Efficient Implementation

There is often a conflict between software flexibility and hardware performance that is difficult to resolve. For a given set of functionality, it is obviously desirable to maximize performance. However, that does not mean new features that cannot be run at that speed today should be disallowed. Therefore, for a protocol to be efficiently implementable means that a set of common capabilities can be reasonably handled across platforms along with a graceful mechanism to handle more advanced features in the appropriate situations.

The use of a variable length header and options in a protocol often raises questions about whether it is truly efficiently implementable in hardware. To answer this question in the context of Geneve, it is important to first divide "hardware" into two categories: tunnel endpoints and transit devices.

Endpoints must be able to parse the variable header, including any options, and take action. Since these devices are actively participating in the protocol, they are the most affected by Geneve.

However, as endpoints are the ultimate consumers of the data, transmitters can tailor their output to the capabilities of the recipient. As new functionality becomes sufficiently well defined to add to endpoints, supporting options can be designed using ordering restrictions and other techniques to ease parsing.

Transit devices MAY be able to interpret the options, however, as non-terminating devices, transit devices do not originate or terminate the Geneve packet, hence MUST NOT insert or delete options, which is the responsibility of Geneve endpoints. The participation of transit devices in interpreting options is OPTIONAL.

Further, either tunnel endpoints or transit devices MAY use offload capabilities of NICs such as checksum offload to improve the performance of Geneve packet processing. The presence of a Geneve variable length header SHOULD NOT prevent the tunnel endpoints and transit devices from using such offload capabilities.

2.3. Use of Standard IP Fabrics

IP has clearly cemented its place as the dominant transport mechanism and many techniques have evolved over time to make it robust, efficient, and inexpensive. As a result, it is natural to use IP fabrics as a transit network for Geneve. Fortunately, the use of IP encapsulation and addressing is enough to achieve the primary goal of delivering packets to the correct point in the network through standard switching and routing.

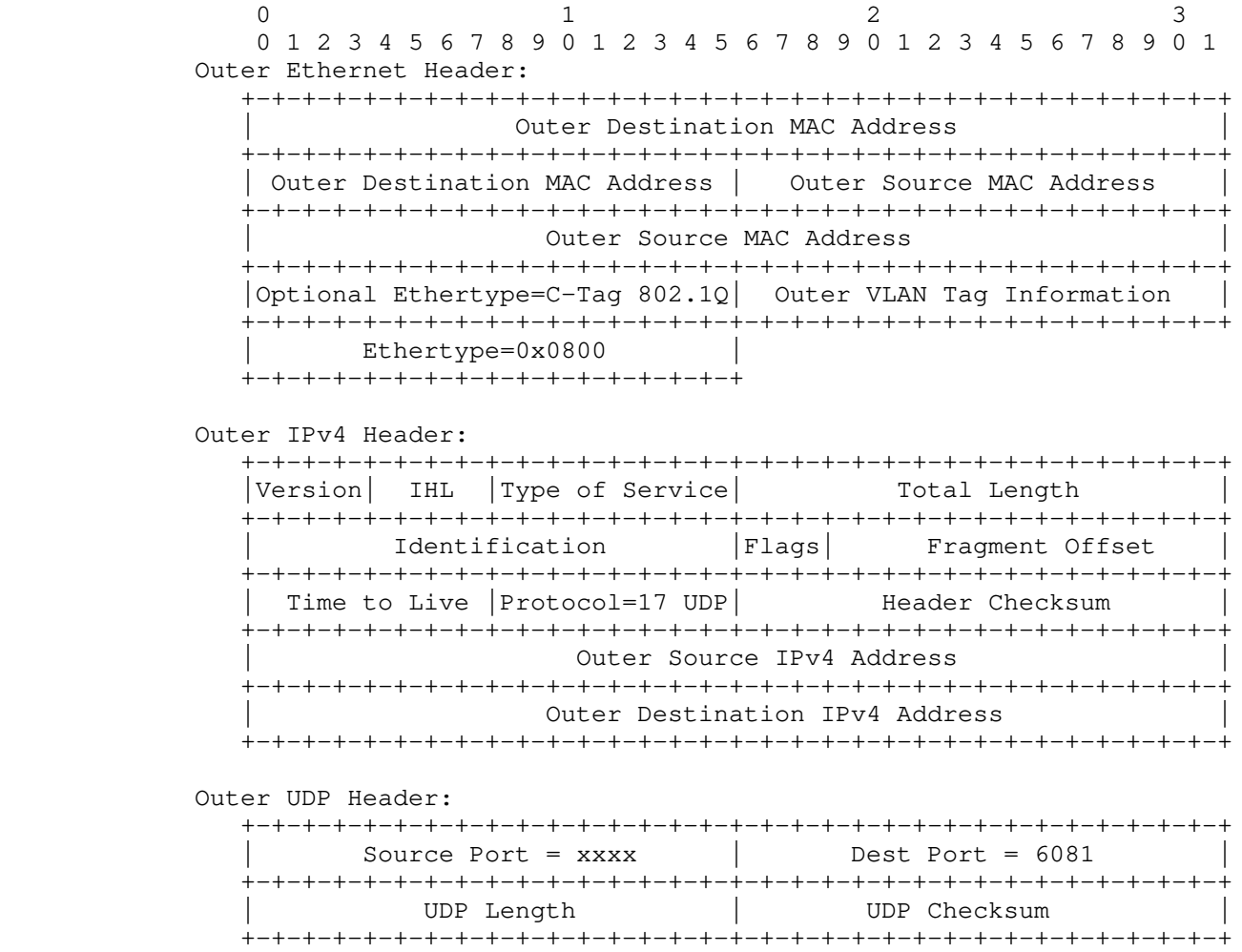
In addition, nearly all underlay fabrics are designed to exploit parallelism in traffic to spread load across multiple links without introducing reordering in individual flows. These equal cost multipathing (ECMP) techniques typically involve parsing and hashing the addresses and port numbers from the packet to select an outgoing link. However, the use of tunnels often results in poor ECMP performance without additional knowledge of the protocol as the encapsulated traffic is hidden from the fabric by design and only endpoint addresses are available for hashing.

Since it is desirable for Geneve to perform well on these existing fabrics, it is necessary for entropy from encapsulated packets to be exposed in the tunnel header. The most common technique for this is to use the UDP source port, which is discussed further in Section 3.3.

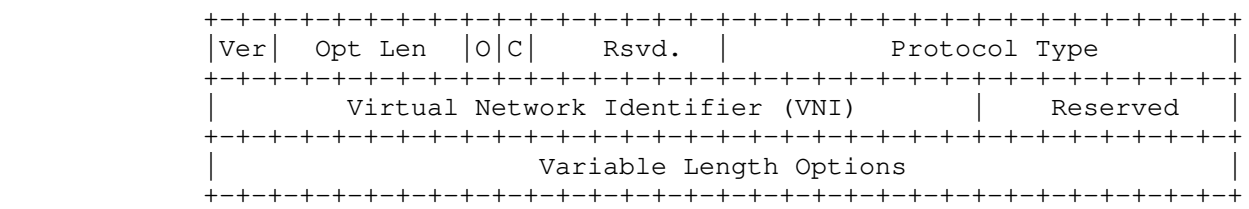
3. Geneve Encapsulation Details

The Geneve packet format consists of a compact tunnel header encapsulated in UDP over either IPv4 or IPv6. A small fixed tunnel header provides control information plus a base level of functionality and interoperability with a focus on simplicity. This header is then followed by a set of variable options to allow for future innovation. Finally, the payload consists of a protocol data unit of the indicated type, such as an Ethernet frame. Section 3.1 and Section 3.2 illustrate the Geneve packet format transported (for example) over Ethernet along with an Ethernet payload.

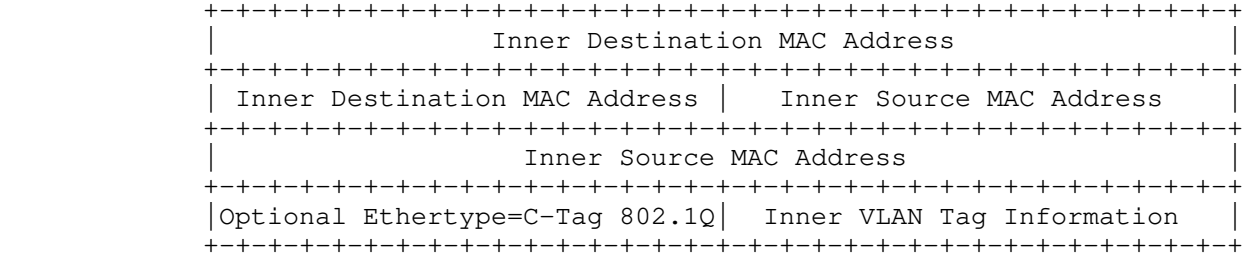
3.1. Geneve Packet Format Over IPv4



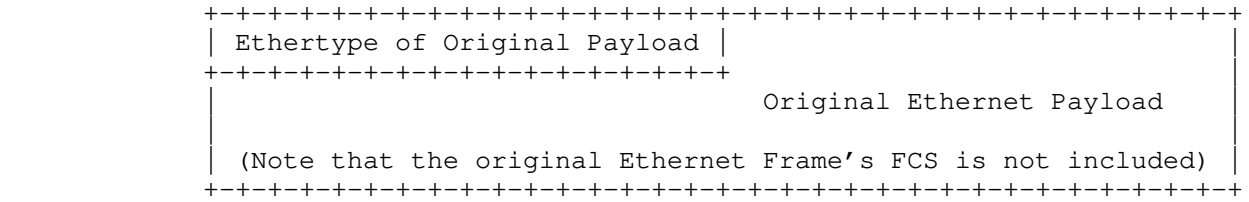
Geneve Header:



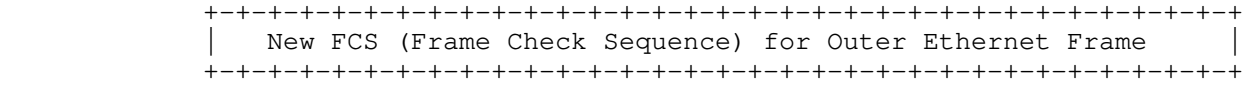
Inner Ethernet Header (example payload):



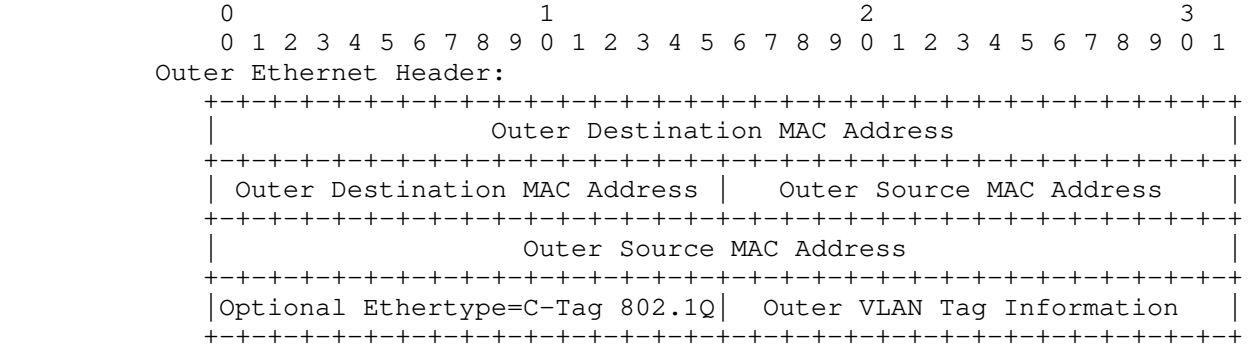
Payload:

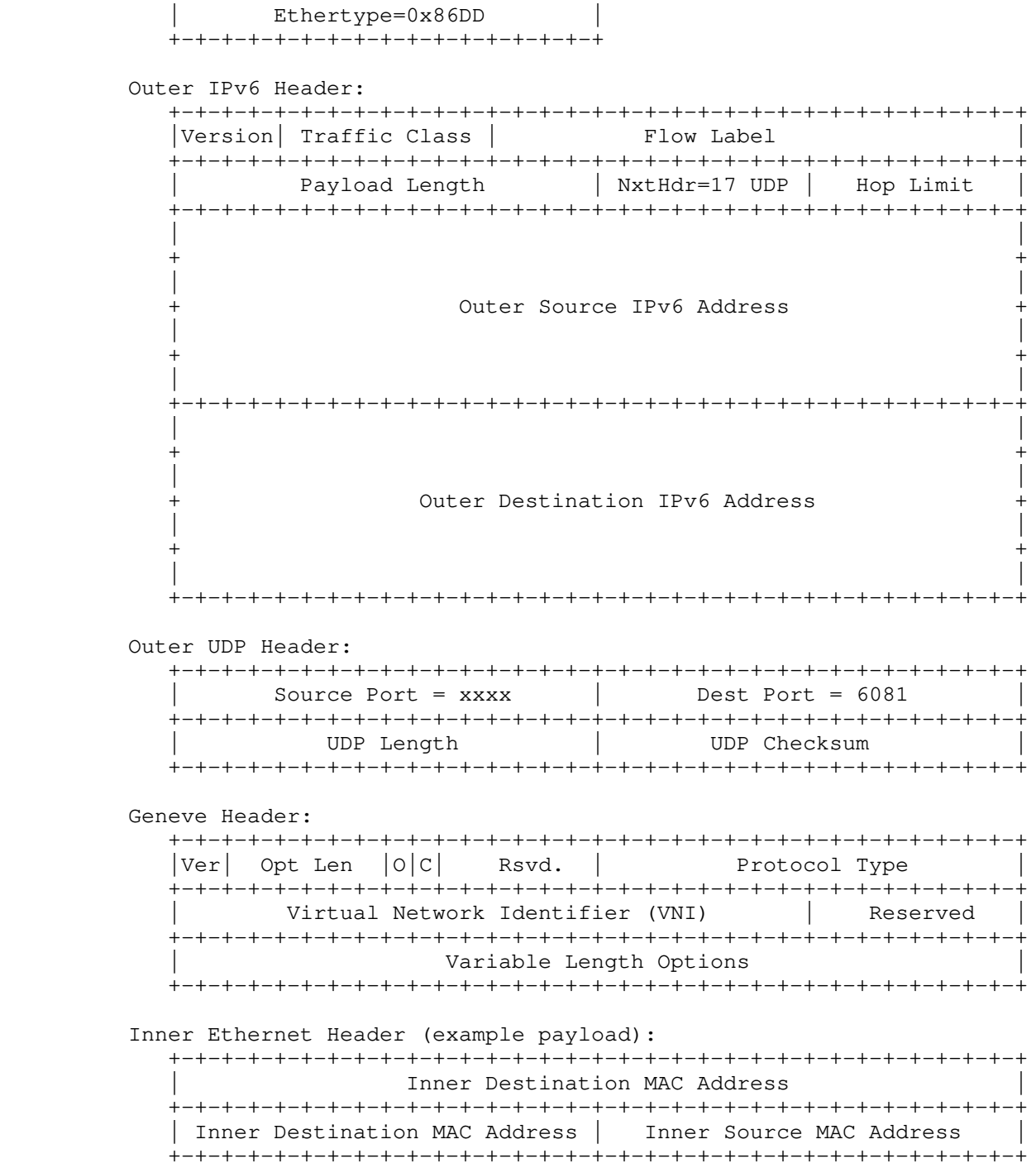


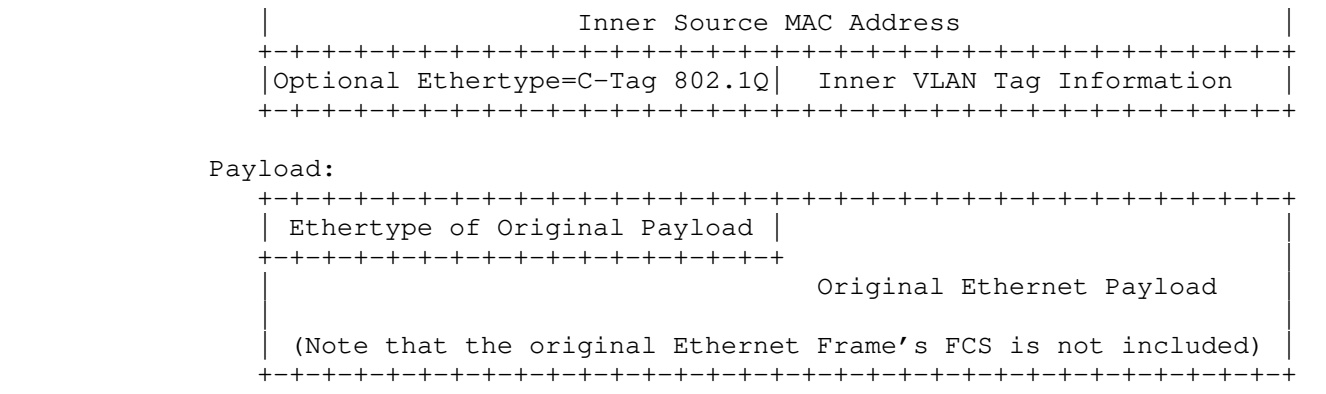
Frame Check Sequence:



3.2. Geneve Packet Format Over IPv6




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 = 6081
UDP Length
UDP Checksum
Geneve Header:
Ver
Opt Len
O
C
Rsvd.
Protocol Type
Virtual Network Identifier (VNI)
Reserved
Variable Length Options
Inner Ethernet Header (example payload):
Inner Destination MAC Address
Inner Destination MAC Address
Inner Source MAC Address



3.3. UDP Header

The use of an encapsulating UDP [RFC0768] header follows the connectionless semantics of Ethernet and IP in addition to providing entropy to routers performing ECMP. The header fields are therefore interpreted as follows:

Source port: A source port selected by the originating tunnel endpoint. This source port SHOULD be the same for all packets belonging to a single encapsulated flow to prevent reordering due to the use of different paths. To encourage an even distribution of flows across multiple links, the source port SHOULD be calculated using a hash of the encapsulated packet headers using, for example, a traditional 5-tuple. Since the port represents a flow identifier rather than a true UDP connection, the entire 16-bit range MAY be used to maximize entropy.

Dest port: IANA has assigned port 6081 as the fixed well-known destination port for Geneve. Although the well-known value should be used by default, it is RECOMMENDED that implementations make this configurable. The chosen port is used for identification of Geneve packets and MUST NOT be reversed for different ends of a connection as is done with TCP.

UDP length: The length of the UDP packet including the UDP header.

UDP checksum: The checksum MAY be set to zero on transmit for

packets encapsulated in both IPv4 and IPv6 [RFC6935]. When a packet is received with a UDP checksum of zero it MUST be accepted and decapsulated. If the originating tunnel endpoint optionally encapsulates a packet with a non-zero checksum, it MUST be a correctly computed UDP checksum. Upon receiving such a packet, the egress endpoint MUST validate the checksum. If the checksum is not correct, the packet MUST be dropped, otherwise the packet MUST be accepted for decapsulation. It is RECOMMENDED that the UDP checksum be computed to protect the Geneve header and options in situations where the network reliability is not high and the packet is not protected by another checksum or CRC.

3.4. Tunnel Header Fields

Ver (2 bits): The current version number is 0. Packets received by an endpoint with an unknown version MUST be dropped. Non-terminating devices processing Geneve packets with an unknown version number MUST treat them as UDP packets with an unknown payload.

Opt Len (6 bits): The length of the options fields, expressed in four byte multiples, not including the eight byte fixed tunnel header. This results in a minimum total Geneve header size of 8 bytes and a maximum of 260 bytes. The start of the payload headers can be found using this offset from the end of the base Geneve header.

O (1 bit): OAM packet. This packet contains a control message instead of a data payload. Control messages are sent between Geneve endpoints. Endpoints MUST NOT forward the payload and transit devices MUST NOT attempt to interpret or process it. Since these are infrequent control messages, it is RECOMMENDED that endpoints direct these packets to a high priority control queue (for example, to direct the packet to a general purpose CPU from a forwarding ASIC or to separate out control traffic on a NIC). Transit devices MUST NOT alter forwarding behavior on the basis of this bit, such as ECMP link selection.

C (1 bit): Critical options present. One or more options has the critical bit set (see Section 3.5). If this bit is set then tunnel endpoints MUST parse the options list to interpret any critical options. On endpoints where option parsing is not supported the packet MUST be dropped on the basis of the 'C' bit in the base header. If the bit is not set tunnel endpoints MAY strip all options using 'Opt Len' and forward the decapsulated packet. Transit devices MUST NOT drop packets on the basis of this bit.

The critical bit allows hardware implementations the flexibility to handle options processing in the hardware fastpath or in the exception (slow) path without the need to process all the options. For example, a critical option such as secure hash to provide Geneve header integrity check must be processed by tunnel endpoints and typically processed in the hardware fastpath.

Rsvd. (6 bits): Reserved field which MUST be zero on transmission and ignored on receipt.

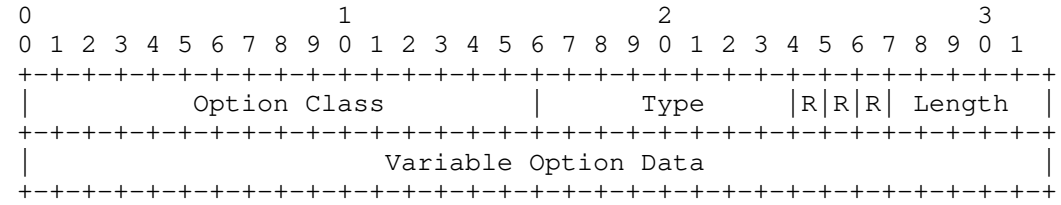
Protocol Type (16 bits): The type of the protocol data unit appearing after the Geneve header. This follows the EtherType [ETYPES] convention with Ethernet itself being represented by the value 0x6558.

Virtual Network Identifier (VNI) (24 bits): An identifier for a unique element of a virtual network. In many situations this may represent an L2 segment, however, the control plane defines the forwarding semantics of decapsulated packets. The VNI MAY be used as part of ECMP forwarding decisions or MAY be used as a mechanism to distinguish between overlapping address spaces contained in the encapsulated packet when load balancing across CPUs.

Reserved (8 bits): Reserved field which MUST be zero on transmission and ignored on receipt.

Transit devices MUST maintain consistent forwarding behavior irrespective of the value of 'Opt Len', including ECMP link selection. These devices SHOULD be able to forward packets containing options without resorting to a slow path.

3.5. Tunnel Options



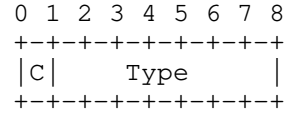
Geneve Option

The base Geneve header is followed by zero or more options in Type-Length-Value format. Each option consists of a four byte option header and a variable amount of option data interpreted according to the type.

Option Class (16 bits): Namespace for the 'Type' field. IANA will be requested to create a "Geneve Option Class" registry to allocate identifiers for organizations, technologies, and vendors that have an interest in creating types for options. Each organization may allocate types independently to allow experimentation and rapid innovation. It is expected that over time certain options will become well known and a given implementation may use option types from a variety of sources. In addition, IANA will be requested to reserve specific ranges for standardized and experimental options.

Type (8 bits): Type indicating the format of the data contained in this option. Options are primarily designed to encourage future extensibility and innovation and so standardized forms of these options will be defined in a separate document.

The high order bit of the option type indicates that this is a critical option. If the receiving endpoint does not recognize this option and this bit is set then the packet MUST be dropped. If the critical bit is set in any option then the 'C' bit in the Geneve base header MUST also be set. Transit devices MUST NOT drop packets on the basis of this bit. The following figure shows the location of the 'C' bit in the 'Type' field:



The requirement to drop a packet with an unknown critical option applies to the entire tunnel endpoint system and not a particular component of the implementation. For example, in a system comprised of a forwarding ASIC and a general purpose CPU, this does not mean that the packet must be dropped in the ASIC. An implementation may send the packet to the CPU using a rate-limited control channel for slow-path exception handling.

R (3 bits): Option control flags reserved for future use. MUST be zero on transmission and ignored on receipt.

Length (5 bits): Length of the option, expressed in four byte multiples excluding the option header. The total length of each option may be between 4 and 128 bytes. A value of 0 in the Length field implies an option with only the option header without the variable option data. Packets in which the total length of all options is not equal to the 'Opt Len' in the base header are invalid and MUST be silently dropped if received by an endpoint.

Variable Option Data: Option data interpreted according to 'Type'.

3.5.1. Options Processing

Geneve options are intended to be originated and processed by tunnel endpoints. However, options MAY be interpreted by transit devices along the tunnel path. Transit devices not processing Geneve headers SHOULD process Geneve packets as any other UDP packet and maintain consistent forwarding behavior.

In tunnel endpoints, the generation and interpretation of options is determined by the control plane, which is out of the scope of this document. However, to ensure interoperability between heterogeneous devices some requirements are imposed on options and the devices that process them:

- Receiving endpoints MUST drop packets containing unknown options with the 'C' bit set in the option type. Conversely, transit devices MUST NOT drop packets as a result of encountering unknown options, including those with the 'C' bit set.

- Some options may be defined in such a way that the position in the option list is significant. Options or their ordering, MUST NOT be changed by transit devices.

- An option MUST NOT affect the parsing or interpretation of any other option.

When designing a Geneve option, it is important to consider how the option will evolve in the future. Once an option is defined it is reasonable to expect that implementations may come to depend on a specific behavior. As a result, the scope of any future changes must be carefully described upfront.

Unexpectedly significant interoperability issues may result from changing the length of an option that was defined to be a certain size. A particular option is specified to have either a fixed length, which is constant, or a variable length, which may change over time or for different use cases. This property is part of the definition of the option and conveyed by the 'Type'. For fixed length options, some implementations may choose to ignore the length field in the option header and instead parse based on the well known length associated with the type. In this case, redefining the length will impact not only parsing of the option in question but also any options that follow. Therefore, options that are defined to be fixed length in size MUST NOT be redefined to a different length. Instead, a new 'Type' should be allocated.

4. Implementation and Deployment Considerations

4.1. Encapsulation of Geneve in IP

As an IP-based tunnel protocol, Geneve shares many properties and techniques with existing protocols. The application of some of these are described in further detail, although in general most concepts applicable to the IP layer or to IP tunnels generally also function in the context of Geneve.

4.1.1. IP Fragmentation

To prevent fragmentation and maximize performance, the best practice when using Geneve is to ensure that the MTU of the physical network is greater than or equal to the MTU of the encapsulated network plus tunnel headers. Manual or upper layer (such as TCP MSS clamping) configuration can be used to ensure that fragmentation never takes place, however, in some situations this may not be feasible.

It is strongly RECOMMENDED that Path MTU Discovery ([RFC1191], [RFC1981]) be used by setting the DF bit in the IP header when Geneve packets are transmitted over IPv4 (this is the default with IPv6). The use of Path MTU Discovery on the transit network provides the encapsulating endpoint with soft-state about the link that it may use to prevent or minimize fragmentation depending on its role in the virtualized network. For example, recommendations/guidance for handling fragmenation in similar overlay encapsulation services like PWE3 are provided in section 5.3 of [RFC3985].

Note that some implementations may not be capable of supporting fragmentation or other less common features of the IP header, such as options and extension headers.

4.1.2. DSCP and ECN

When encapsulating IP (including over Ethernet) packets in Geneve, there are several considerations for propagating DSCP and ECN bits from the inner header to the tunnel on transmission and the reverse on reception.

[RFC2983] provides guidance for mapping DSCP between inner and outer IP headers. Network virtualization is typically more closely aligned with the Pipe model described, where the DSCP value on the tunnel header is set based on a policy (which may be a fixed value, one based on the inner traffic class, or some other mechanism for grouping traffic). Aspects of the Uniform model (which treats the inner and outer DSCP value as a single field by copying on ingress and egress) may also apply, such as the ability to remark the inner

header on tunnel egress based on transit marking. However, the Uniform model is not conceptually consistent with network virtualization, which seeks to provide strong isolation between encapsulated traffic and the physical network.

[RFC6040] describes the mechanism for exposing ECN capabilities on IP tunnels and propagating congestion markers to the inner packets. This behavior MUST be followed for IP packets encapsulated in Geneve.

4.1.3. Broadcast and Multicast

Geneve tunnels may either be point-to-point unicast between two endpoints or may utilize broadcast or multicast addressing. It is not required that inner and outer addressing match in this respect. For example, in physical networks that do not support multicast, encapsulated multicast traffic may be replicated into multiple unicast tunnels or forwarded by policy to a unicast location (possibly to be replicated there).

With physical networks that do support multicast it may be desirable to use this capability to take advantage of hardware replication for encapsulated packets. In this case, multicast addresses may be allocated in the physical network corresponding to tenants, encapsulated multicast groups, or some other factor. The allocation of these groups is a component of the control plane and therefore outside of the scope of this document. When physical multicast is in use, the 'C' bit in the Geneve header may be used with groups of devices with heterogeneous capabilities as each device can interpret only the options that are significant to it if they are not critical.

4.1.4. Unidirectional Tunnels

Generally speaking, a Geneve tunnel is a unidirectional concept. IP is not a connection oriented protocol and it is possible for two endpoints to communicate with each other using different paths or to have one side not transmit anything at all. As Geneve is an IP-based protocol, the tunnel layer inherits these same characteristics.

It is possible for a tunnel to encapsulate a protocol, such as TCP, which is connection oriented and maintains session state at that layer. In addition, implementations MAY model Geneve tunnels as connected, bidirectional links, such as to provide the abstraction of a virtual port. In both of these cases, bidirectionality of the tunnel is handled at a higher layer and does not affect the operation of Geneve itself.

4.2. Constraints on Protocol Features

Geneve is intended to be flexible to a wide range of current and future applications. As a result, certain constraints may be placed on the use of metadata or other aspects of the protocol in order to optimize for a particular use case. For example, some applications may limit the types of options which are supported or enforce a maximum number or length of options. Other applications may only handle certain encapsulated payload types, such as Ethernet or IP. This could be either globally throughout the system or, for example, restricted to certain classes of devices or network paths.

These constraints may be communicated to tunnel endpoints either explicitly through a control plane or implicitly by the nature of the application. As Geneve is defined as a data plane protocol that is control plane agnostic, the exact mechanism is not defined in this document.

4.2.1. Constraints on Options

While Geneve options are more flexible, a control plane may restrict the number of option TLVs as well as the order and size of the TLVs, between tunnel endpoints, to make it simpler for a data plane implementation in software or hardware to handle [I-D.ietf-nvo3-encap]. For example, there may be some critical information such as a secure hash that must be processed in a certain order to provide lowest latency.

A control plane may negotiate a subset of option TLVs and certain TLV ordering, as well may limit the total number of option TLVs present in the packet, for example, to accommodate hardware capable of processing fewer options [I-D.ietf-nvo3-encap]. Hence, a control plane needs to have the ability to describe the supported TLVs subset and their order to the tunnel end points. In the absence of a control plane, alternative configuration mechanisms may be used for this purpose. The exact mechanism is not defined in this document.

4.3. NIC Offloads

Modern NICs currently provide a variety of offloads to enable the efficient processing of packets. The implementation of many of these offloads requires only that the encapsulated packet be easily parsed (for example, checksum offload). However, optimizations such as LSO and LRO involve some processing of the options themselves since they must be replicated/merged across multiple packets. In these situations, it is desirable to not require changes to the offload logic to handle the introduction of new options. To enable this,

some constraints are placed on the definitions of options to allow for simple processing rules:

- When performing LSO, a NIC MUST replicate the entire Geneve header and all options, including those unknown to the device, onto each resulting segment. However, a given option definition may override this rule and specify different behavior in supporting devices. Conversely, when performing LRO, a NIC MAY assume that a binary comparison of the options (including unknown options) is sufficient to ensure equality and MAY merge packets with equal Geneve headers.

- Options MUST NOT be reordered during the course of offload processing, including when merging packets for the purpose of LRO.

- NICs performing offloads MUST NOT drop packets with unknown options, including those marked as critical.

There is no requirement that a given implementation of Geneve employ the offloads listed as examples above. However, as these offloads are currently widely deployed in commercially available NICs, the rules described here are intended to enable efficient handling of current and future options across a variety of devices.

4.4. Inner VLAN Handling

Geneve is capable of encapsulating a wide range of protocols and therefore a given implementation is likely to support only a small subset of the possibilities. However, as Ethernet is expected to be widely deployed, it is useful to describe the behavior of VLANs inside encapsulated Ethernet frames.

As with any protocol, support for inner VLAN headers is OPTIONAL. In many cases, the use of encapsulated VLANs may be disallowed due to security or implementation considerations. However, in other cases trunking of VLAN frames across a Geneve tunnel can prove useful. As a result, the processing of inner VLAN tags upon ingress or egress from a tunnel endpoint is based upon the configuration of the endpoint and/or control plane and not explicitly defined as part of the data format.

5. Interoperability Issues

Viewed exclusively from the data plane, Geneve does not introduce any interoperability issues as it appears to most devices as UDP packets. However, as there are already a number of tunnel protocols deployed in network virtualization environments, there is a practical question of transition and coexistence.

Since Geneve is a superset of the functionality of the most common protocols used for network virtualization (VXLAN, NVGRE) it should be straightforward to port an existing control plane to run on top of it with minimal effort. With both the old and new packet formats supporting the same set of capabilities, there is no need for a hard transition - endpoints directly communicating with each other use any common protocol, which may be different even within a single overall system. As transit devices are primarily forwarding packets on the basis of the IP header, all protocols appear similar and these devices do not introduce additional interoperability concerns.

To assist with this transition, it is strongly suggested that implementations support simultaneous operation of both Geneve and existing tunnel protocols as it is expected to be common for a single node to communicate with a mixture of other nodes. Eventually, older protocols may be phased out as they are no longer in use.

6. Security Considerations

As encapsulated within an UDP/IP packet, Geneve does not have any inherent security mechanisms. As a result, an attacker with access to the underlay network transporting the IP packets has the ability to snoop or inject packets. Legitimate but malicious tunnel endpoints may also spoof identifiers in the tunnel header to gain access to networks owned by other tenants.

Within a particular security domain, such as a data center operated by a single service provider, the most common and highest performing security mechanism is isolation of trusted components. Tunnel traffic can be carried over a separate VLAN and filtered at any untrusted boundaries. In addition, tunnel endpoints should only be operated in environments controlled by the service provider, such as the hypervisor itself rather than within a customer VM.

When crossing an untrusted link, such as the public Internet, IPsec [RFC4301] may be used to provide authentication and/or encryption of the IP packets formed as part of Geneve encapsulation.

Geneve does not otherwise affect the security of the encapsulated packets. As per the guidelines of BCP72 [RFC3552], the following sections describe potential security risks that may be applicable to Geneve deployments and approaches to mitigate such risks. It is also noted that not all such risks are applicable to all Geneve deployment scenarios, i.e., only a subset may be applicable to certain deployments. So an operator has to make an assessment based on their network environment and determine the risks that are applicable to their specific environment and use appropriate mitigation approaches as applicable.

6.1. Data Confidentiality

Geneve is a network virtualization overlay encapsulation protocol designed to establish tunnels between network virtualization end points (NVE) over an existing IP network. It can be used to deploy multi-tenant overlay networks over an existing IP underlay network in a public or private data center. The overlay service is typically provided by a service provider, for example a cloud services provider or a private data center operator. Due to the nature of multi-tenancy in such environments, a tenant system may expect data confidentiality to ensure its packet data is not tampered with (active attack) in transit or a target of unauthorized monitoring (passive attack). A tenant may expect the overlay service provider to provide data confidentiality as part of the service or a tenant may bring its own data confidentiality mechanisms like IPsec or TLS to protect the data end to end between its tenant systems.

If an operator determines data confidentiality is necessary in their environment based on their risk analysis, for example as in multi-tenant environments, then an encryption mechanism SHOULD be used to encrypt the tenant data end to end between the NVEs. The NVEs may use existing well established encryption mechanisms such as IPsec, DTLs, etc., The operator may choose not to enable the encryption if, for example, the packet data is already encrypted by the tenant system.

6.1.1. Inter-data center traffic

A tenant system in a customer premises (private data center) may want to connect to tenant systems on their tenant overlay network in a public cloud data center or a tenant may want to have its tenant systems located in multiple geographically separated data centers for high availability. Geneve data traffic between tenant systems across such separated networks should be protected from threats when traversing public networks. Any Geneve overlay data leaving the data center network beyond the operator's security domain, for example over the public Internet, SHOULD be secured by encryption mechanisms such as IPsec or other VPN mechanisms to protect the communications between the NVEs when they are geographically separated over untrusted network links. Implementation of specific data protection mechanisms employed between data centers is beyond the scope of this document.

6.2. Data Integrity

Geneve encapsulation is used between NVEs to establish overlay tunnels over an existing IP underlay network. In a multi-tenant data center, a rogue or compromised tenant system may try to launch a

passive attack such as monitoring the traffic of other tenants, or an active attack such as spoofing or trying to inject unauthorized Geneve encapsulated traffic into the network. To prevent such attacks, an NVE MUST not propagate Geneve packets beyond the NVE to tenant systems and SHOULD employ packet filtering mechanisms so as not to forward unauthorized traffic between TSs in different tenant networks.

A compromised network node or a transit device within a data center may launch an active attack trying to tamper with the Geneve packet data between NVEs. Malicious tampering of Geneve header fields may cause the packet from one tenant to be forwarded to a different tenant network. If an operator determines the possibility of such threat in their environment, the operator may choose to employ data integrity mechanisms between NVEs. In order to prevent such risks, a data integrity mechanism SHOULD be used in such environments to protect the integrity of Geneve packets including packet headers, options and payload on communications between NVE pairs. A cryptographic data protection mechanism such as IPsec may be used to provide data integrity protection. A data center operator may choose to deploy any other data integrity mechanisms as applicable and supported in their underlay networks.

Geneve supports Geneve Options, so an operator may choose to use a Geneve option TLV to provide a cryptographic data protection mechanism, to verify the data integrity of the Geneve header, Geneve options or the entire Geneve packet including the payload. Implementation of such a mechanism is beyond the scope of this document.

6.3. Authentication of NVE peers

A rogue network device or a compromised NVE in a data center environment might be able to spoof Geneve packets as if it came from a legitimate NVE. In order to mitigate such a risk, an operator SHOULD use an Authentication mechanism, such as IPsec to ensure that the Geneve packet originated from the intended NVE peer, in environments where the operator determines spoofing or rogue devices is a potential threat. Other simpler source checks such as ingress filtering for VLAN/MAC/IP address, reverse path forwarding checks, etc., may be used in certain trusted environments to ensure Geneve packets originated from the intended NVE peer.

6.4. Multicast/Broadcast

In typical data center networks where IP multicasting is not supported in the underlay network, multicasting may be supported using multiple unicast tunnels. The same security requirements as

described in the above sections can be used to protect Geneve communications between NVE peers. If IP multicasting is supported in the underlay network and the operator chooses to use it for multicast traffic among Geneve endpoints, then the operator in such environments may use data protection mechanisms such as IPsec with Multicast extensions [RFC5374] to protect multicast traffic among Geneve NVE groups.

6.5. Control plane communications

A Network Virtualization Authority (NVA) as outlined in [RFC8014] may be used as a control plane for configuring and managing the Geneve NVEs. The data center operator is expected to use security mechanisms to protect the communications between the NVA to NVEs and use authentication mechanisms to detect any rogue or compromised NVEs within their administrative domain. Data protection mechanisms for control plane communication or authentication mechanisms between the NVA and the NVEs is beyond the scope of this document.

7. IANA Considerations

IANA has allocated UDP port 6081 as the well-known destination port for Geneve. Upon publication, the registry should be updated to cite this document. The original request was:

Service Name: geneve
Transport Protocol(s): UDP
Assignee: Jesse Gross <jgross@vmware.com>
Contact: Jesse Gross <jgross@vmware.com>
Description: Generic Network Virtualization Encapsulation (Geneve)
Reference: This document
Port Number: 6081

In addition, IANA is requested to create a "Geneve Option Class" registry to allocate Option Classes. This shall be a registry of 16-bit hexadecimal values along with descriptive strings. The identifiers 0x0-0xFF are to be reserved for standardized options for allocation by IETF Review [RFC5226] and 0xFFF0-0xFFFF for Experimental Use. Otherwise, identifiers are to be assigned to any organization with an interest in creating Geneve options on a First Come First Served basis. The registry is to be populated with the following initial values:

Option Class	Description
0x0000..0x00FF	Unassigned - IETF Review
0x0100	Linux
0x0101	Open vSwitch
0x0102	Open Virtual Networking (OVN)
0x0103	In-band Network Telemetry (INT)
0x0104	VMware
0x0105	Amazon
0x0106	Cisco
0x0107..0xFFEF	Unassigned - First Come First Served
0xFFF0..FFFF	Experimental

8. Contributors

The following individuals were authors of an earlier version of this document and made significant contributions:

Pankaj Garg
Microsoft Corporation
1 Microsoft Way
Redmond, WA 98052
USA

Email: pankajg@microsoft.com

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

Email: chrisw@redhat.com

Puneet Agarwal
Innovium, Inc.
6001 America Center Drive
San Jose, CA 95002
USA

Email: puneet@innovium.com

Kenneth Duda
Arista Networks
5453 Great America Parkway
Santa Clara, CA 95054

USA

Email: kduda@arista.com

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

Email: ddutt@cumulusnetworks.com

Jon Hudson
Independent

Email: jon.hudson@gmail.com

Ariel Hendel
Facebook, Inc.
1 Hacker Way
Menlo Park, CA 94025
USA

Email: ahendel@fb.com

9. Acknowledgements

The authors wish to thank Martin Casado, Bruce Davie and Dave Thaler for their input, feedback, and helpful suggestions.

10. References

10.1. Normative References

[RFC0768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, DOI 10.17487/RFC0768, August 1980, <https://www.rfc-editor.org/info/rfc768>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", RFC 5226, DOI 10.17487/RFC5226, May 2008, <https://www.rfc-editor.org/info/rfc5226>.

10.2. Informative References

[ETYPES] The IEEE Registration Authority, "IEEE 802 Numbers", 2013, <http://www.iana.org/assignments/ieee-802-numbers/ieee-802-numbers.xml>.

[I-D.ietf-nvo3-dataplane-requirements] Bitar, N., Lasserre, M., Balus, F., Morin, T., Jin, L., and B. Khasnabish, "NVO3 Data Plane Requirements", draft-ietf-nvo3-dataplane-requirements-03 (work in progress), April 2014.

[I-D.ietf-nvo3-encap] Boutros, S., Ganga, I., Garg, P., Manur, R., Mizrahi, T., Mozes, D., Nordmark, E., Smith, M., Aldrin, S., and I. Bagdonas, "NVO3 Encapsulation Considerations", draft-ietf-nvo3-encap-01 (work in progress), October 2017.

[IEEE.802.1Q_2014] IEEE, "IEEE Standard for Local and metropolitan area networks--Bridges and Bridged Networks", IEEE 802.1Q-2014, DOI 10.1109/ieeestd.2014.6991462, December 2014, <http://ieeexplore.ieee.org/servlet/opac?punumber=6991460>.

[RFC1191] Mogul, J. and S. Deering, "Path MTU discovery", RFC 1191, DOI 10.17487/RFC1191, November 1990, <https://www.rfc-editor.org/info/rfc1191>.

[RFC1981] McCann, J., Deering, S., and J. Mogul, "Path MTU Discovery for IP version 6", RFC 1981, DOI 10.17487/RFC1981, August 1996, <https://www.rfc-editor.org/info/rfc1981>.

[RFC2983] Black, D., "Differentiated Services and Tunnels", RFC 2983, DOI 10.17487/RFC2983, October 2000, <https://www.rfc-editor.org/info/rfc2983>.

[RFC3031] Rosen, E., Viswanathan, A., and R. Callon, "Multiprotocol Label Switching Architecture", RFC 3031, DOI 10.17487/RFC3031, January 2001, <https://www.rfc-editor.org/info/rfc3031>.

[RFC3552] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, DOI 10.17487/RFC3552, July 2003, <https://www.rfc-editor.org/info/rfc3552>.

[RFC3985] Bryant, S., Ed. and P. Pate, Ed., "Pseudo Wire Emulation Edge-to-Edge (PWE3) Architecture", RFC 3985, DOI 10.17487/RFC3985, March 2005, <https://www.rfc-editor.org/info/rfc3985>.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, DOI 10.17487/RFC4301, December 2005, <https://www.rfc-editor.org/info/rfc4301>.

[RFC5374] Weis, B., Gross, G., and D. Ignjatic, "Multicast Extensions to the Security Architecture for the Internet Protocol", RFC 5374, DOI 10.17487/RFC5374, November 2008, <https://www.rfc-editor.org/info/rfc5374>.

[RFC6040] Briscoe, B., "Tunnelling of Explicit Congestion Notification", RFC 6040, DOI 10.17487/RFC6040, November 2010, <https://www.rfc-editor.org/info/rfc6040>.

[RFC6935] Eubanks, M., Chimento, P., and M. Westerlund, "IPv6 and UDP Checksums for Tunneled Packets", RFC 6935, DOI 10.17487/RFC6935, April 2013, <https://www.rfc-editor.org/info/rfc6935>.

[RFC7348] Mahalingam, M., Dutt, D., Duda, K., Agarwal, P., Kreeger, L., Sridhar, T., Bursell, M., and C. Wright, "Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks", RFC 7348, DOI 10.17487/RFC7348, August 2014, <https://www.rfc-editor.org/info/rfc7348>.

[RFC7365] Lasserre, M., Balus, F., Morin, T., Bitar, N., and Y. Rekhter, "Framework for Data Center (DC) Network Virtualization", RFC 7365, DOI 10.17487/RFC7365, October 2014, <https://www.rfc-editor.org/info/rfc7365>.

[RFC7637] Garg, P., Ed. and Y. Wang, Ed., "NVGRE: Network Virtualization Using Generic Routing Encapsulation", RFC 7637, DOI 10.17487/RFC7637, September 2015, <https://www.rfc-editor.org/info/rfc7637>.

[RFC8014] Black, D., Hudson, J., Kreeger, L., Lasserre, M., and T. Narten, "An Architecture for Data-Center Network Virtualization over Layer 3 (NVO3)", RFC 8014, DOI 10.17487/RFC8014, December 2016, <https://www.rfc-editor.org/info/rfc8014>.

[VL2] Greenberg, A., et al., "VL2: A Scalable and Flexible Data Center Network", ACM SIGCOMM Computer Communication Review, DOI 10.1145/1594977.1592576, 2009, <http://www.sigcomm.org/sites/default/files/ccr/papers/2009/October/1594977-1592576.pdf>.

Authors' Addresses

Jesse Gross (editor)

Email: jesse@kernel.org

Ilango Ganga (editor)
Intel Corporation
2200 Mission College Blvd.
Santa Clara, CA 95054
USA

Email: ilango.s.ganga@intel.com

T. Sridhar (editor)
VMware, Inc.
3401 Hillview Ave.
Palo Alto, CA 94304
USA

Email: tsridhar@vmware.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 8, 2020

J. Gross, Ed.

I. Ganga, Ed.
Intel

T. Sridhar, Ed.
VMware

March 07, 2020

Geneve: Generic Network Virtualization Encapsulation
draft-ietf-nvo3-geneve-16

Abstract

Network virtualization involves the cooperation of devices with a wide variety of capabilities such as software and hardware tunnel endpoints, transit fabrics, and centralized control clusters. As a result of their role in tying together different elements in the system, the requirements on tunnels are influenced by all of these components. Flexibility is therefore the most important aspect of a tunnel protocol if it is to keep pace with the evolution of the system. This document describes Geneve, an encapsulation protocol designed to recognize and accommodate these changing capabilities and needs.

Table of Contents

1. Introduction

Networking has long featured a variety of tunneling, tagging, and other encapsulation mechanisms. However, the advent of network virtualization has caused a surge of renewed interest and a corresponding increase in the introduction of new protocols. The large number of protocols in this space, for example, ranging all the way from VLANs [IEEE.802.1Q_2018] and MPLS [RFC3031] through the more recent VXLAN [RFC7348] (Virtual eXtensible Local Area Network) and NVGRE [RFC7637] (Network Virtualization Using Generic Routing Encapsulation), often leads to questions about the need for new encapsulation formats and what it is about network virtualization in particular that leads to their proliferation. Note that the list of protocols presented above is non-exhaustive.

While many encapsulation protocols seek to simply partition the underlay network or bridge between two domains, network virtualization views the transit network as providing connectivity between multiple components of a distributed system. In many ways this system is similar to a chassis switch with the IP underlay network playing the role of the backplane and tunnel endpoints on the edge as line cards. When viewed in this light, the requirements placed on the tunnel protocol are significantly different in terms of the quantity of metadata necessary and the role of transit nodes.

Work such as [VL2] (A Scalable and Flexible Data Center Network) and the NVO3 Data Plane Requirements [I-D.ietf-nvo3-dataplane-requirements] have described some of the properties that the data plane must have to support network virtualization. However, one additional defining requirement is the need to carry metadata (e.g. system state) along with the packet data; example use cases of metadata are noted below. The use of some metadata is certainly not a foreign concept - nearly all protocols used for network virtualization have at least 24 bits of identifier space as a way to partition between tenants. This is often described as overcoming the limits of 12-bit VLANs, and when seen in that context, or any context where it is a true tenant identifier, 16 million possible entries is a large number. However, the reality is

that the metadata is not exclusively used to identify tenants and encoding other information quickly starts to crowd the space. In fact, when compared to the tags used to exchange metadata between line cards on a chassis switch, 24-bit identifiers start to look quite small. There are nearly endless uses for this metadata, ranging from storing input port identifiers for simple security policies to sending service based context for advanced middlebox applications that terminate and re-encapsulate Geneve traffic.

Existing tunnel protocols have each attempted to solve different aspects of these new requirements, only to be quickly rendered out of date by changing control plane implementations and advancements. Furthermore, software and hardware components and controllers all have different advantages and rates of evolution - a fact that should be viewed as a benefit, not a liability or limitation. This draft describes Geneve, a protocol which seeks to avoid these problems by providing a framework for tunneling for network virtualization rather than being prescriptive about the entire system.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

1.2. Terminology

The NVO3 Framework [RFC7365] defines many of the concepts commonly used in network virtualization. In addition, the following terms are specifically meaningful in this document:

Checksum offload. An optimization implemented by many NICs (Network Interface Controller) which enables computation and verification of upper layer protocol checksums in hardware on transmit and receive, respectively. This typically includes IP and TCP/UDP checksums which would otherwise be computed by the protocol stack in software.

Clos network. A technique for composing network fabrics larger than a single switch while maintaining non-blocking bandwidth across connection points. ECMP is used to divide traffic across the multiple links and switches that constitute the fabric. Sometimes termed "leaf and spine" or "fat tree" topologies.

ECMP. Equal Cost Multipath. A routing mechanism for selecting from among multiple best next hop paths by hashing packet headers in order

to better utilize network bandwidth while avoiding reordering of packets within a flow.

Geneve. Generic Network Virtualization Encapsulation. The tunnel protocol described in this document.

LRO. Large Receive Offload. The receive-side equivalent function of LSO, in which multiple protocol segments (primarily TCP) are coalesced into larger data units.

LSO. Large Segmentation Offload. A function provided by many commercial NICs that allows data units larger than the MTU to be passed to the NIC to improve performance, the NIC being responsible for creating smaller segments of size less than or equal to the MTU with correct protocol headers. When referring specifically to TCP/IP, this feature is often known as TSO (TCP Segmentation Offload).

Middlebox. The term middlebox in the context of this document refers to network service functions or appliances for service interposition that would typically implement NVE functionality, which terminate or re-encapsulate Geneve traffic.

NIC. Network Interface Controller. Also called as Network Interface Card or Network Adapter. A NIC could be part of a tunnel endpoint or transit device and can either process Geneve packets or aid in the processing of Geneve packets.

Transit device. A forwarding element (e.g. router or switch) along the path of the tunnel making up part of the Underlay Network. A transit device may be capable of understanding the Geneve packet format but does not originate or terminate Geneve packets.

Tunnel endpoint. A component performing encapsulation and decapsulation of packets, such as Ethernet frames or IP datagrams, in Geneve headers. As the ultimate consumer of any tunnel metadata, tunnel endpoints have the highest level of requirements for parsing and interpreting tunnel headers. Tunnel endpoints may consist of either software or hardware implementations or a combination of the two. Tunnel endpoints are frequently a component of an NVE (Network Virtualization Edge) but may also be found in middleboxes or other elements making up an NVO3 Network.

VM. Virtual Machine.

2. Design Requirements

Geneve is designed to support network virtualization use cases for data center environments, where tunnels are typically established to act as a backplane between the virtual switches residing in hypervisors, physical switches, or middleboxes or other appliances. An arbitrary IP network can be used as an underlay although Clos networks composed using ECMP links are a common choice to provide consistent bisectional bandwidth across all connection points. Many of the concepts of network virtualization overlays over Layer 3 IP networks are described in the NVO3 Framework [RFC7365]. Figure 1 shows an example of a hypervisor, top of rack switch for connectivity to physical servers, and a WAN uplink connected using Geneve tunnels over a simplified Clos network. These tunnels are used to encapsulate and forward frames from the attached components such as VMs or physical links.

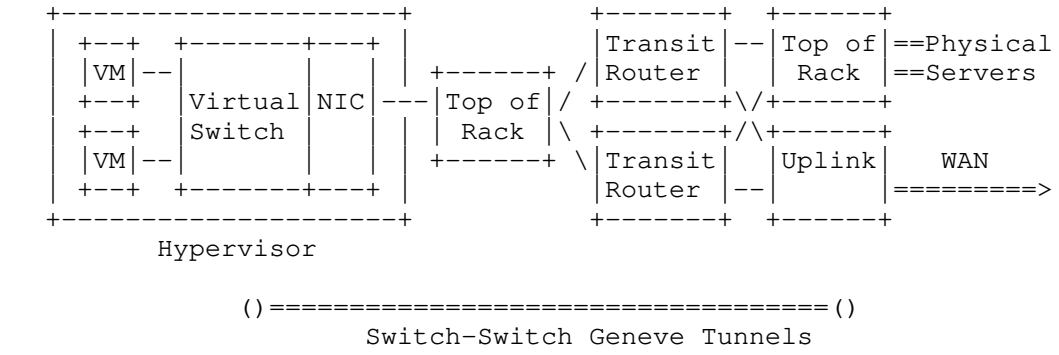


Figure 1: Sample Geneve Deployment

To support the needs of network virtualization, the tunnel protocol should be able to take advantage of the differing (and evolving) capabilities of each type of device in both the underlay and overlay networks. This results in the following requirements being placed on the data plane tunneling protocol:

- The data plane is generic and extensible enough to support current and future control planes.
- Tunnel components are efficiently implementable in both hardware and software without restricting capabilities to the lowest common denominator.
- High performance over existing IP fabrics.

These requirements are described further in the following subsections.

2.1. Control Plane Independence

Although some protocols for network virtualization have included a control plane as part of the tunnel format specification (most notably, VXLAN [RFC7348] prescribed a multicast learning- based control plane), these specifications have largely been treated as describing only the data format. The VXLAN packet format has actually seen a wide variety of control planes built on top of it.

There is a clear advantage in settling on a data format: most of the protocols are only superficially different and there is little advantage in duplicating effort. However, the same cannot be said of control planes, which are diverse in very fundamental ways. The case for standardization is also less clear given the wide variety in requirements, goals, and deployment scenarios.

As a result of this reality, Geneve is a pure tunnel format specification that is capable of fulfilling the needs of many control planes by explicitly not selecting any one of them. This simultaneously promotes a shared data format and reduces the chance of obsolescence by future control plane enhancements.

2.2. Data Plane Extensibility

Achieving the level of flexibility needed to support current and future control planes effectively requires an options infrastructure to allow new metadata types to be defined, deployed, and either finalized or retired. Options also allow for differentiation of products by encouraging independent development in each vendor's core specialty, leading to an overall faster pace of advancement. By far the most common mechanism for implementing options is Type-Length-Value (TLV) format.

It should be noted that while options can be used to support non-wirespeed control packets, they are equally important on data packets as well to segregate and direct forwarding (for instance, the examples given before of input port based security policies and terminating/re-encapsulating service interposition both require tags to be placed on data packets). Therefore, while it would be desirable to limit the extensibility to only control packets for the purposes of simplifying the datapath, that would not satisfy the design requirements.

2.2.1. Efficient Implementation

There is often a conflict between software flexibility and hardware performance that is difficult to resolve. For a given set of functionality, it is obviously desirable to maximize performance. However, that does not mean new features that cannot be run at a desired speed today should be disallowed. Therefore, for a protocol to be efficiently implementable means that a set of common capabilities can be reasonably handled across platforms along with a graceful mechanism to handle more advanced features in the appropriate situations.

The use of a variable length header and options in a protocol often raises questions about whether it is truly efficiently implementable in hardware. To answer this question in the context of Geneve, it is important to first divide "hardware" into two categories: tunnel endpoints and transit devices.

Tunnel endpoints must be able to parse the variable header, including any options, and take action. Since these devices are actively participating in the protocol, they are the most affected by Geneve. However, as tunnel endpoints are the ultimate consumers of the data, transmitters can tailor their output to the capabilities of the recipient.

Transit devices may be able to interpret the options, however, as non-terminating devices, transit devices do not originate or terminate the Geneve packet, hence MUST NOT modify Geneve headers and MUST NOT insert or delete options, which is the responsibility of tunnel endpoints. Options, if present in the packet, MUST only be generated and terminated by tunnel endpoints. The participation of transit devices in interpreting options is OPTIONAL.

Further, either tunnel endpoints or transit devices MAY use offload capabilities of NICs such as checksum offload to improve the performance of Geneve packet processing. The presence of a Geneve variable length header should not prevent the tunnel endpoints and transit devices from using such offload capabilities.

2.3. Use of Standard IP Fabrics

IP has clearly cemented its place as the dominant transport mechanism and many techniques have evolved over time to make it robust, efficient, and inexpensive. As a result, it is natural to use IP fabrics as a transit network for Geneve. Fortunately, the use of IP encapsulation and addressing is enough to achieve the primary goal of delivering packets to the correct point in the network through standard switching and routing.

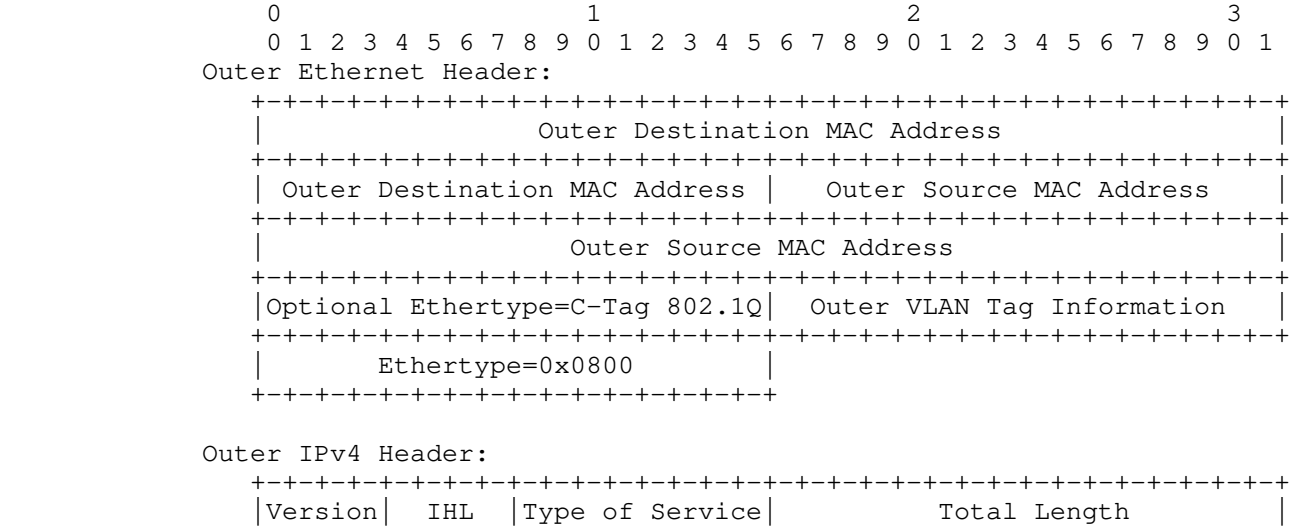
In addition, nearly all underlay fabrics are designed to exploit parallelism in traffic to spread load across multiple links without introducing reordering in individual flows. These equal cost multipathing (ECMP) techniques typically involve parsing and hashing the addresses and port numbers from the packet to select an outgoing link. However, the use of tunnels often results in poor ECMP performance without additional knowledge of the protocol as the encapsulated traffic is hidden from the fabric by design and only tunnel endpoint addresses are available for hashing.

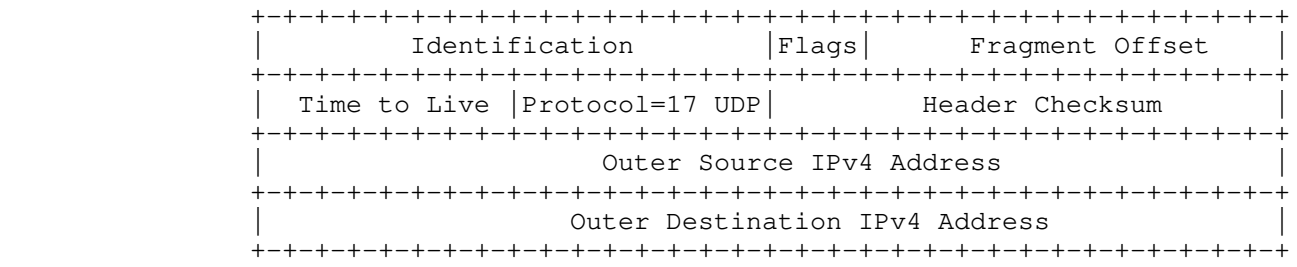
Since it is desirable for Geneve to perform well on these existing fabrics, it is necessary for entropy from encapsulated packets to be exposed in the tunnel header. The most common technique for this is to use the UDP source port, which is discussed further in Section 3.3.

3. Geneve Encapsulation Details

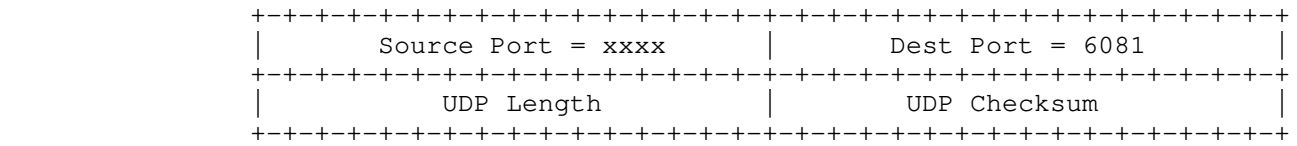
The Geneve packet format consists of a compact tunnel header encapsulated in UDP over either IPv4 or IPv6. A small fixed tunnel header provides control information plus a base level of functionality and interoperability with a focus on simplicity. This header is then followed by a set of variable options to allow for future innovation. Finally, the payload consists of a protocol data unit of the indicated type, such as an Ethernet frame. Section 3.1 and Section 3.2 illustrate the Geneve packet format transported (for example) over Ethernet along with an Ethernet payload.

3.1. Geneve Packet Format Over IPv4

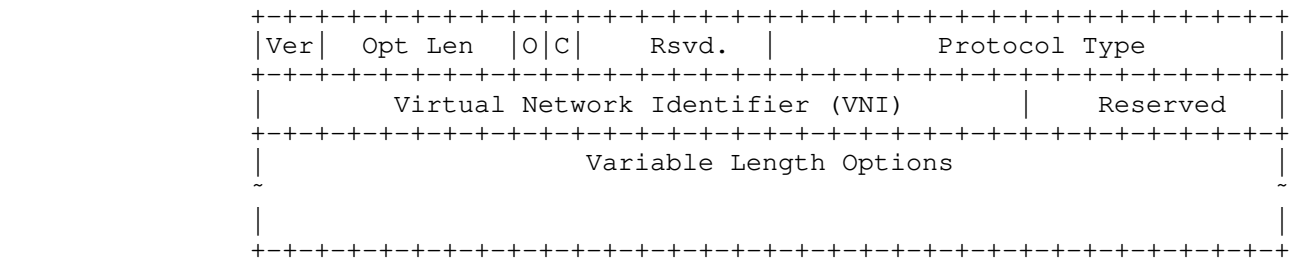




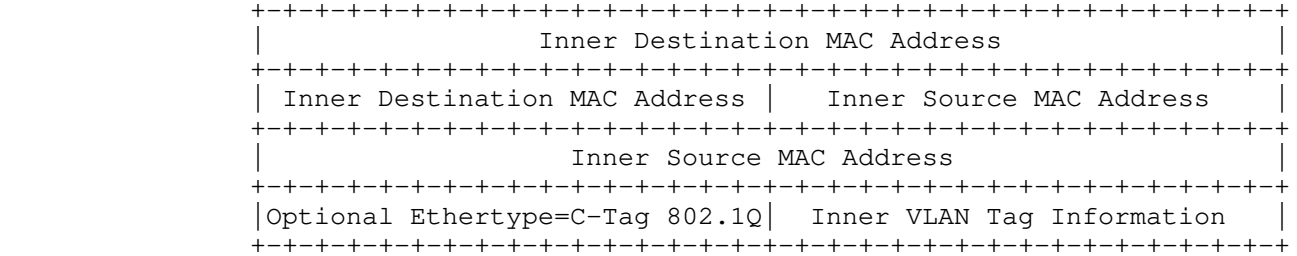
Outer UDP Header:



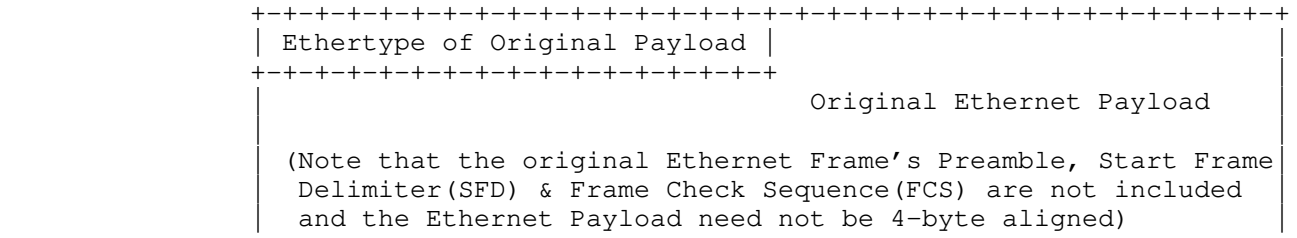
Geneve Header:



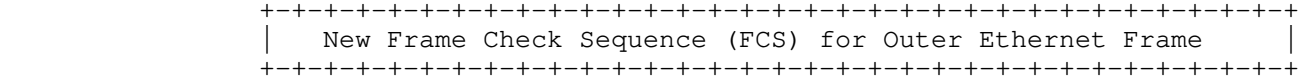
Inner Ethernet Header (example payload):



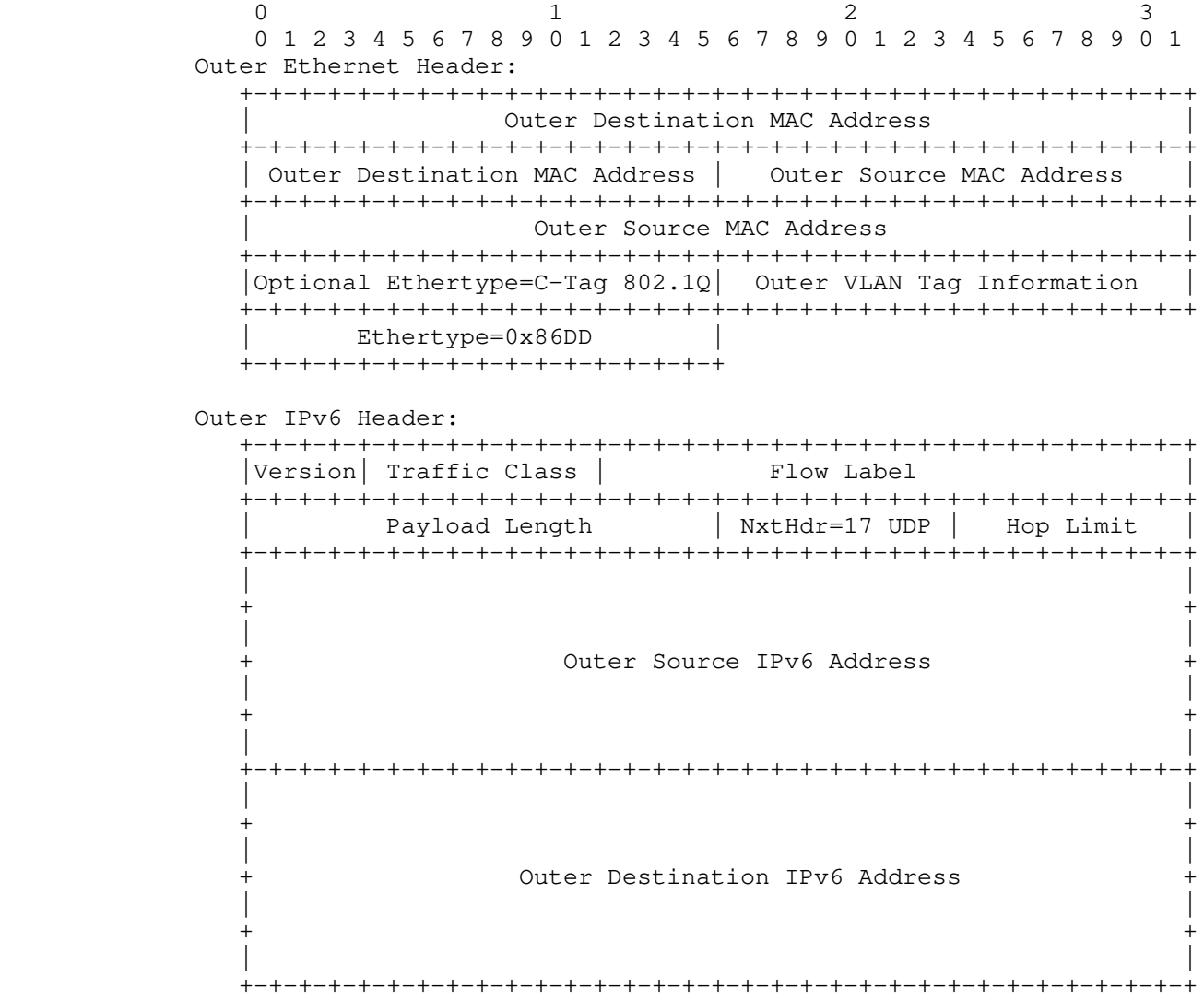
Payload:



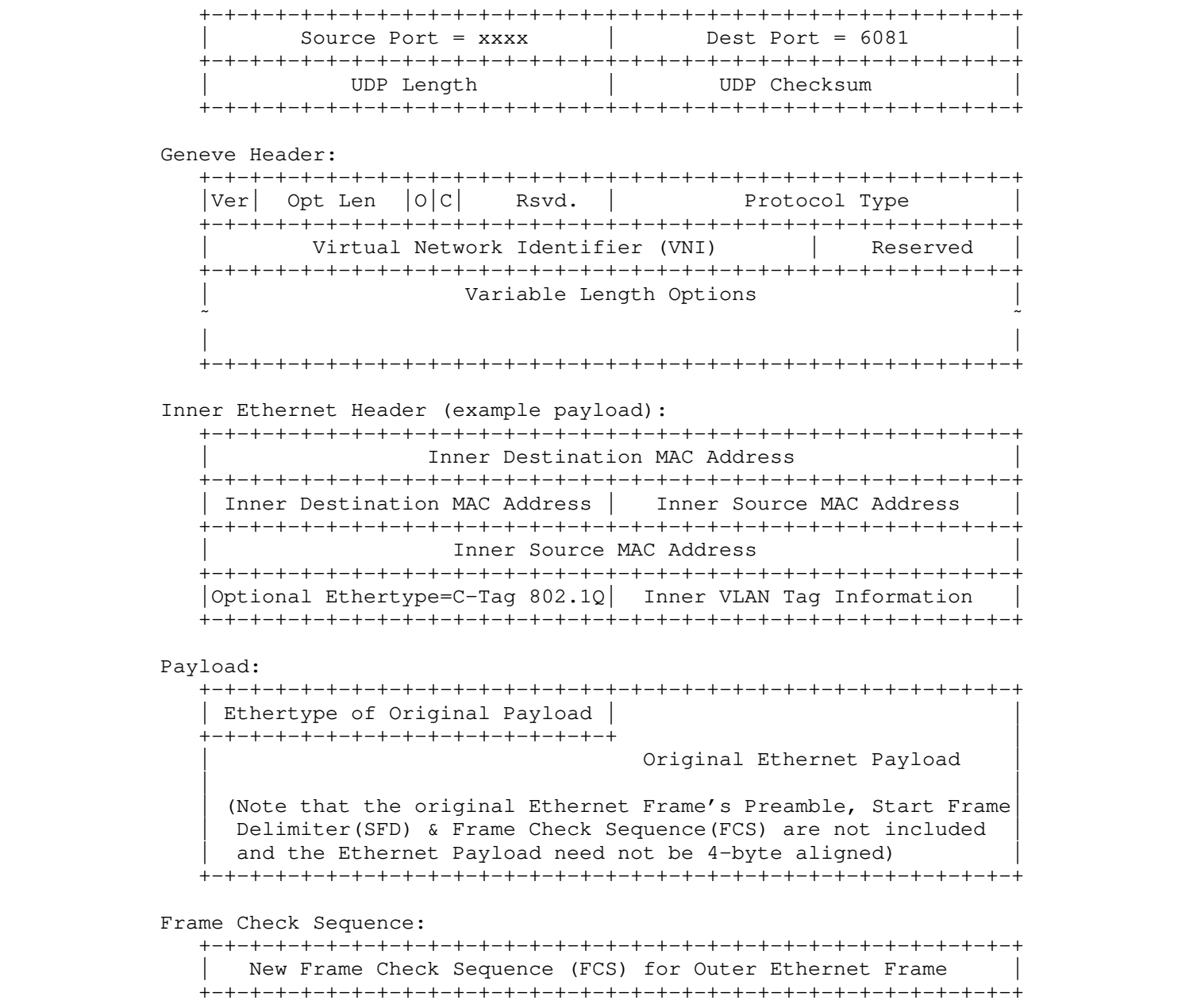
Frame Check Sequence:



3.2. Geneve Packet Format Over IPv6



Outer UDP Header:


Source Port = xxxx
Dest Port = 6081
UDP Length
UDP Checksum
Geneve Header:
Ver
Opt Len
O
C
Rsvd.
Protocol Type
Virtual Network Identifier (VNI)
Reserved
Variable Length Options
Inner Ethernet Header (example payload):
Inner Destination MAC Address
Inner Destination MAC Address
Inner Source MAC Address
Inner Source MAC Address
Optional Ethertype=C-Tag 802.1Q
Inner VLAN Tag Information
Payload:
Ethertype of Original Payload
Original Ethernet Payload
(Note that the original Ethernet Frame's Preamble, Start Frame Delimiter(SFD) & Frame Check Sequence(FCS) are not included and the Ethernet Payload need not be 4-byte aligned)
Frame Check Sequence:
New Frame Check Sequence (FCS) for Outer Ethernet Frame

3.3. UDP Header

The use of an encapsulating UDP [RFC0768] header follows the connectionless semantics of Ethernet and IP in addition to providing entropy to routers performing ECMP. The header fields are therefore interpreted as follows:

Source port: A source port selected by the originating tunnel endpoint. This source port SHOULD be the same for all packets belonging to a single encapsulated flow to prevent reordering due to the use of different paths. To encourage an even distribution of flows across multiple links, the source port SHOULD be calculated using a hash of the encapsulated packet headers using, for example, a traditional 5-tuple. Since the port represents a flow identifier rather than a true UDP connection, the entire 16-bit range MAY be used to maximize entropy. In addition to setting the source port, for IPv6, flow label MAY also be used for providing entropy. For an example of using IPv6 flow label for tunnel use cases, see [RFC6438].

If Geneve traffic is shared with other UDP listeners on the same IP address, tunnel endpoints SHOULD implement a mechanism to ensure ICMP return traffic arising from network errors is directed to the correct listener. The definition of such a mechanism is beyond the scope of this document.

Dest port: IANA has assigned port 6081 as the fixed well-known destination port for Geneve. Although the well-known value should be used by default, it is RECOMMENDED that implementations make this configurable. The chosen port is used for identification of Geneve packets and MUST NOT be reversed for different ends of a connection as is done with TCP. It is the responsibility of the control plane for any reconfiguration of the assigned port and its interpretation by respective devices. The definition of the control plane is beyond the scope of this document.

UDP length: The length of the UDP packet including the UDP header.

UDP checksum: In order to protect the Geneve header, options and payload from potential data corruption, UDP checksum SHOULD be generated as specified in [RFC0768] and [RFC1112] when Geneve is encapsulated in IPv4. To protect the IP header, Geneve header, options and payload from potential data corruption, the UDP checksum MUST be generated by default as specified in [RFC0768] and [RFC8200] when Geneve is encapsulated in IPv6, except for certain conditions, which are outlined in the next paragraph. Upon receiving such packets with non-zero UDP checksum, the

receiving tunnel endpoints MUST validate the checksum. If the checksum is not correct, the packet MUST be dropped, otherwise the packet MUST be accepted for decapsulation.

Under certain conditions, the UDP checksum MAY be set to zero on transmit for packets encapsulated in both IPv4 and IPv6 [RFC8200]. See Section 4.3 for additional requirements that apply when using zero UDP checksum with IPv4 and IPv6. Disabling the use of UDP checksums is an operational consideration that should take into account the risks and effects of packet corruption.

3.4. Tunnel Header Fields

Ver (2 bits): The current version number is 0. Packets received by a tunnel endpoint with an unknown version MUST be dropped. Transit devices interpreting Geneve packets with an unknown version number MUST treat them as UDP packets with an unknown payload.

Opt Len (6 bits): The length of the options fields, expressed in four byte multiples, not including the eight byte fixed tunnel header. This results in a minimum total Geneve header size of 8 bytes and a maximum of 260 bytes. The start of the payload headers can be found using this offset from the end of the base Geneve header.

Transit devices MUST maintain consistent forwarding behavior irrespective of the value of 'Opt Len', including ECMP link selection.

O (1 bit): Control packet. This packet contains a control message. Control messages are sent between tunnel endpoints. Tunnel endpoints MUST NOT forward the payload and transit devices MUST NOT attempt to interpret it. Since control messages are less frequent, it is RECOMMENDED that tunnel endpoints direct these packets to a high priority control queue (for example, to direct the packet to a general purpose CPU from a forwarding ASIC or to separate out control traffic on a NIC). Transit devices MUST NOT alter forwarding behavior on the basis of this bit, such as ECMP link selection.

C (1 bit): Critical options present. One or more options has the critical bit set (see Section 3.5). If this bit is set then tunnel endpoints MUST parse the options list to interpret any critical options. On tunnel endpoints where option parsing is not supported the packet MUST be dropped on the basis of the 'C' bit in the base header. If the bit is not set tunnel endpoints MAY strip all options using 'Opt Len' and forward the decapsulated

packet. Transit devices MUST NOT drop packets on the basis of this bit.

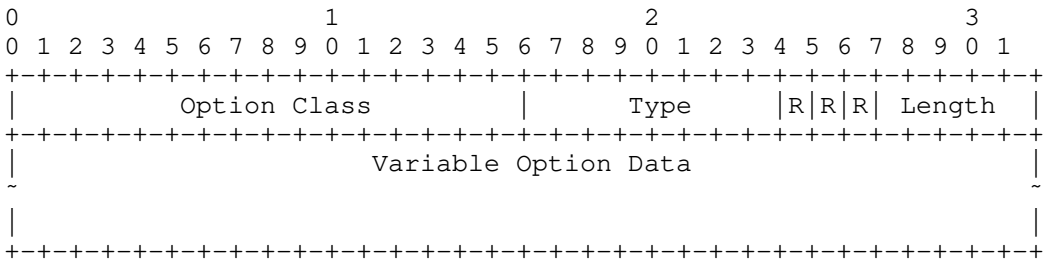
Rsvd. (6 bits): Reserved field, which MUST be zero on transmission and MUST be ignored on receipt.

Protocol Type (16 bits): The type of the protocol data unit appearing after the Geneve header. This follows the EtherType [ETYPES] convention; with Ethernet itself being represented by the value 0x6558.

Virtual Network Identifier (VNI) (24 bits): An identifier for a unique element of a virtual network. In many situations this may represent an L2 segment, however, the control plane defines the forwarding semantics of decapsulated packets. The VNI MAY be used as part of ECMP forwarding decisions or MAY be used as a mechanism to distinguish between overlapping address spaces contained in the encapsulated packet when load balancing across CPUs.

Reserved (8 bits): Reserved field which MUST be zero on transmission and ignored on receipt.

3.5. Tunnel Options



Geneve Option

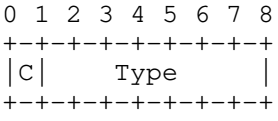
The base Geneve header is followed by zero or more options in Type-Length-Value format. Each option consists of a four byte option header and a variable amount of option data interpreted according to the type.

Option Class (16 bits): Namespace for the 'Type' field. IANA will be requested to create a "Geneve Option Class" registry to allocate identifiers for organizations, technologies, and vendors that have an interest in creating types for options. Each organization may allocate types independently to allow experimentation and rapid innovation. It is expected that over time certain options will become well known and a given

implementation may use option types from a variety of sources. In addition, IANA will be requested to reserve specific ranges for allocation by IETF Review and for Experimental Use (see Section 7).

Type (8 bits): Type indicating the format of the data contained in this option. Options are primarily designed to encourage future extensibility and innovation and so standardized forms of these options will be defined in separate documents.

The high order bit of the option type indicates that this is a critical option. If the receiving tunnel endpoint does not recognize this option and this bit is set then the packet MUST be dropped. If this bit is set in any option then the 'C' bit in the Geneve base header MUST also be set. Transit devices MUST NOT drop packets on the basis of this bit. The following figure shows the location of the 'C' bit in the 'Type' field:



The requirement to drop a packet with an unknown option with the 'C' bit set applies to the entire tunnel endpoint system and not a particular component of the implementation. For example, in a system comprised of a forwarding ASIC and a general purpose CPU, this does not mean that the packet must be dropped in the ASIC. An implementation may send the packet to the CPU using a rate-limited control channel for slow-path exception handling.

R (3 bits): Option control flags reserved for future use. These bits MUST be zero on transmission and MUST be ignored on receipt.

Length (5 bits): Length of the option, expressed in four byte multiples excluding the option header. The total length of each option may be between 4 and 128 bytes. A value of 0 in the Length field implies an option with only an option header and no variable option data. Packets in which the total length of all options is not equal to the 'Opt Len' in the base header are invalid and MUST be silently dropped if received by a tunnel endpoint that processes the options.

Variable Option Data: Option data interpreted according to 'Type'.

3.5.1. Options Processing

Geneve options are intended to be originated and processed by tunnel endpoints. However, options MAY be interpreted by transit devices along the tunnel path. Transit devices not interpreting Geneve headers (which may or may not include options) MUST handle Geneve packets as any other UDP packet and maintain consistent forwarding behavior.

In tunnel endpoints, the generation and interpretation of options is determined by the control plane, which is beyond the the scope of this document. However, to ensure interoperability between heterogeneous devices some requirements are imposed on options and the devices that process them:

- Receiving tunnel endpoints MUST drop packets containing unknown options with the 'C' bit set in the option type. Conversely, transit devices MUST NOT drop packets as a result of encountering unknown options, including those with the 'C' bit set.

- The contents of the options and their ordering MUST NOT be modified by transit devices.

- If a tunnel endpoint receives a Geneve packet with 'Opt Len' (total length of all options) that exceeds the options processing capability of the tunnel endpoint then the tunnel endpoint MUST drop such packets. An implementation may raise an exception to the control plane of such an event. It is the responsibility of the control plane to ensure the communicating peer tunnel endpoints have the processing capability to handle the total length of options. The definition of the control plane is beyond the scope of this document.

When designing a Geneve option, it is important to consider how the option will evolve in the future. Once an option is defined it is reasonable to expect that implementations may come to depend on a specific behavior. As a result, the scope of any future changes must be carefully described upfront.

Architecturally, options are intended to be self-descriptive and independent. This enables parallelism in option processing and reduces implementation complexity. However, the control plane may impose certain ordering restrictions as described in Section 4.5.1.

Unexpectedly significant interoperability issues may result from changing the length of an option that was defined to be a certain size. A particular option is specified to have either a fixed length, which is constant, or a variable length, which may change

over time or for different use cases. This property is part of the definition of the option and conveyed by the 'Type'. For fixed length options, some implementations may choose to ignore the length field in the option header and instead parse based on the well known length associated with the type. In this case, redefining the length will impact not only parsing of the option in question but also any options that follow. Therefore, options that are defined to be fixed length in size MUST NOT be redefined to a different length. Instead, a new 'Type' should be allocated. Actual definition of the option type is beyond the scope of this document. The option type and its interpretation should be defined by the entity that owns the option class.

Options may be processed by NIC hardware utilizing offloads (e.g. LSO and LRO) as described in Section 4.6. Careful consideration should be given to how the offload capabilities outlined in Section 4.6 impact an option's design.

4. Implementation and Deployment Considerations

4.1. Applicability Statement

Geneve is a network virtualization overlay encapsulation protocol designed to establish tunnels between NVEs over an existing IP network. It is intended for use in public or private data center environments, for deploying multi-tenant overlay networks over an existing IP underlay network.

Geneve is a UDP based encapsulation protocol transported over existing IPv4 and IPv6 networks. Hence, as a UDP based protocol, Geneve adheres to the UDP usage guidelines as specified in [RFC8085]. The applicability of these guidelines are dependent on the underlay IP network and the nature of Geneve payload protocol (example TCP/IP, IP/Ethernet).

Geneve is intended to be deployed in a data center network environment operated by a single operator or adjacent set of cooperating network operators that fits with the definition of controlled environments in [RFC8085]. A network in a controlled environment can be managed to operate under certain conditions whereas in the general Internet this cannot be done. Hence requirements for a tunnel protocol operating under a controlled environment can be less restrictive than the requirements of the general Internet.

For the purpose of this document, a traffic-managed controlled environment (TMCE) is defined as an IP network that is traffic-engineered and/or otherwise managed (e.g., via use of traffic rate

limiters) to avoid congestion. The concept of TMCE is outlined in [RFC8086]. Significant portions of the text in Section 4.1 through Section 4.3 are based on [RFC8086] as applicable to Geneve.

It is the responsibility of the operator to ensure that the guidelines/requirements in this section are followed as applicable to their Geneve deployment(s).

4.2. Congestion Control Functionality

Geneve does not natively provide congestion control functionality and relies on the payload protocol traffic for congestion control. As such Geneve MUST be used with congestion controlled traffic or within a network that is traffic managed to avoid congestion (TMCE). An operator of a traffic managed network (TMCE) may avoid congestion by careful provisioning of their networks, rate-limiting of user data traffic and traffic engineering according to path capacity.

4.3. UDP Checksum

In order to provide integrity of Geneve headers, options and payload, (for example to avoid misdelivery of payload to different tenant systems) in case of data corruption, the outer UDP checksum SHOULD be used with Geneve when transported over IPv4. The UDP checksum provides a statistical guarantee that a payload was not corrupted in transit. These integrity checks are not strong from a coding or cryptographic perspective and are not designed to detect physical-layer errors or malicious modification of the datagram (see Section 3.4 of [RFC8085]). In deployments where such a risk exists, an operator SHOULD use additional data integrity mechanisms such as offered by IPsec (see Section 6.2).

An operator MAY choose to disable UDP checksums and use zero checksums if Geneve packet integrity is provided by other data integrity mechanisms such as IPsec or additional checksums or if one of the conditions in Section 4.3.1 a, b, c are met.

By default, UDP checksums MUST be used when Geneve is transported over IPv6. A tunnel endpoint MAY be configured for use with zero UDP checksum if additional requirements in Section 4.3.1 are met.

4.3.1. UDP Zero Checksum Handling with IPv6

When Geneve is used over IPv6, the UDP checksum is used to protect IPv6 headers, UDP headers and Geneve headers, options and payload from potential data corruption. As such by default Geneve MUST use UDP checksums when transported over IPv6. An operator MAY choose to configure to operate with zero UDP checksum if operating in a traffic

managed controlled environment as stated in Section 4.1 if one of the following conditions are met.

a. It is known that the packet corruption is exceptionally unlikely (perhaps based on knowledge of equipment types in their underlay network) and the operator is willing to take a risk of undetected packet corruption

b. It is judged through observational measurements (perhaps through historic or current traffic flows that use non zero checksum) that the level of packet corruption is tolerably low and where the operator is willing to take the risk of undetected corruption.

c. Geneve payload is carrying applications that are tolerant of misdelivered or corrupted packets (perhaps through higher layer checksum validation and/or reliability through retransmission)

In addition Geneve tunnel implementations using zero UDP checksum MUST meet the following requirements:

1. Use of UDP checksum over IPv6 MUST be the default configuration for all Geneve tunnels.

2. If Geneve is used with zero UDP checksum over IPv6 then such tunnel endpoint implementation MUST meet all the requirements specified in Section 4 of [RFC6936] and requirement 1 as specified in Section 5 of [RFC6936] as that is relevant to Geneve.

3. The Geneve tunnel endpoint that decapsulates the tunnel SHOULD check the source and destination IPv6 addresses are valid for the Geneve tunnel that is configured to receive zero UDP checksum and discard other packets for which such check fails.

4. The Geneve tunnel endpoint that encapsulates the tunnel MAY use different IPv6 source addresses for each Geneve tunnel that uses zero UDP checksum mode in order to strengthen the decapsulator's check of the IPv6 source address (i.e the same IPv6 source address is not to be used with more than one IPv6 destination address, irrespective of whether that destination address is a unicast or multicast address). When this is not possible, it is RECOMMENDED to use each source address for as few Geneve tunnels that use zero UDP checksum as is feasible.

   Note that (for requirements 3 and 4) the receiving tunnel endpoint can apply these checks only if it has out-of-band knowledge that the encapsulating tunnel endpoint is applying the

indicated behavior. One possibility to obtain this out-of-band knowledge is through signaling by the control plane. The definition of the control plane is beyond the scope of this document.

5. Measures SHOULD be taken to prevent Geneve traffic over IPv6 with zero UDP checksum from escaping into the general Internet. Examples of such measures include employing packet filters at the gateways or edge of Geneve network and/or keeping logical or physical separation of the Geneve network from networks carrying the general Internet traffic.

The above requirements do not change either the requirements specified in [RFC8200] or the requirements specified in [RFC6936].

The use of the source IPv6 address in addition to the destination IPv6 address, plus the recommendation against reuse of source IPv6 addresses among Geneve tunnels collectively provide some mitigation for the absence of UDP checksum coverage of the IPv6 header. A traffic-managed controlled environment that satisfies at least one of three conditions listed at the beginning of this section provides additional assurance.

Editorial Note (The following paragraph to be removed by the RFC Editor before publication)

It was discussed during TSVART early review if the level of requirement for using different IPv6 source addresses for different tunnel destinations would need to be "MAY" or "SHOULD". The discussion concluded that it was appropriate to keep this as "MAY", since it was considered not realistic for control planes having to maintain a high level of state on a per tunnel destination basis. In addition, the text above provides sufficient guidance to operators and implementors on possible mitigations.

4.4. Encapsulation of Geneve in IP

As an IP-based tunnel protocol, Geneve shares many properties and techniques with existing protocols. The application of some of these are described in further detail, although in general most concepts applicable to the IP layer or to IP tunnels generally also function in the context of Geneve.

4.4.1. IP Fragmentation

It is strongly RECOMMENDED that Path MTU Discovery ([RFC1191], [RFC8201]) be used to prevent or minimize fragmentation. The use of Path MTU Discovery on the transit network provides the encapsulating

tunnel endpoint with soft-state about the link that it may use to prevent or minimize fragmentation depending on its role in the virtualized network. The NVE can maintain this state (the MTU size of the tunnel link(s) associated with the tunnel endpoint), so if a tenant system sends large packets that when encapsulated exceed the MTU size of the tunnel link, the tunnel endpoint can discard such packets and send exception messages to the tenant system(s). If the tunnel endpoint is associated with a routing or forwarding function and/or has the capability to send ICMP messages, the encapsulating tunnel endpoint MAY send ICMP fragmentation needed [RFC0792] or Packet Too Big [RFC4443] messages to the tenant system(s). When determining the MTU size of a tunnel link, maximum length of options MUST be assumed as options may vary on a per-packet basis. For example, recommendations/guidance for handling fragmentation in similar overlay encapsulation services like PWE3 are provided in Section 5.3 of [RFC3985].

Note that some implementations may not be capable of supporting fragmentation or other less common features of the IP header, such as options and extension headers. For example, some of the issues associated with MTU size and fragmentation in IP tunneling and use of ICMP messages is outlined in Section 4.2 of [I-D.ietf-intarea-tunnels].

4.4.2. DSCP, ECN and TTL

When encapsulating IP (including over Ethernet) packets in Geneve, there are several considerations for propagating DSCP and ECN bits from the inner header to the tunnel on transmission and the reverse on reception.

[RFC2983] provides guidance for mapping DSCP between inner and outer IP headers. Network virtualization is typically more closely aligned with the Pipe model described, where the DSCP value on the tunnel header is set based on a policy (which may be a fixed value, one based on the inner traffic class, or some other mechanism for grouping traffic). Aspects of the Uniform model (which treats the inner and outer DSCP value as a single field by copying on ingress and egress) may also apply, such as the ability to remark the inner header on tunnel egress based on transit marking. However, the Uniform model is not conceptually consistent with network virtualization, which seeks to provide strong isolation between encapsulated traffic and the physical network.

[RFC6040] describes the mechanism for exposing ECN capabilities on IP tunnels and propagating congestion markers to the inner packets. This behavior MUST be followed for IP packets encapsulated in Geneve.

Though Uniform or Pipe models could be used for TTL (or Hop Limit in case of IPv6) handling when tunneling IP packets, the Pipe model is more aligned with network virtualization. [RFC2003] provides guidance on handling TTL between inner IP header and outer IP tunnels; this model is more aligned with the Pipe model and is RECOMMENDED for use with Geneve for network virtualization applications.

4.4.3. Broadcast and Multicast

Geneve tunnels may either be point-to-point unicast between two tunnel endpoints or may utilize broadcast or multicast addressing. It is not required that inner and outer addressing match in this respect. For example, in physical networks that do not support multicast, encapsulated multicast traffic may be replicated into multiple unicast tunnels or forwarded by policy to a unicast location (possibly to be replicated there).

With physical networks that do support multicast it may be desirable to use this capability to take advantage of hardware replication for encapsulated packets. In this case, multicast addresses may be allocated in the physical network corresponding to tenants, encapsulated multicast groups, or some other factor. The allocation of these groups is a component of the control plane and therefore is beyond the scope of this document.

When physical multicast is in use, devices with heterogeneous capabilities may be present in the same group. Some options may only be interpretable by a subset of the devices in the group. Other devices can safely ignore such options unless the 'C' bit is set to mark the unknown option as critical. Requirements outlined in Section 3.4 apply for critical options.

In addition, [RFC8293] provides examples of various mechanisms that can be used for multicast handling in network virtualization overlay networks.

4.4.4. Unidirectional Tunnels

Generally speaking, a Geneve tunnel is a unidirectional concept. IP is not a connection oriented protocol and it is possible for two tunnel endpoints to communicate with each other using different paths or to have one side not transmit anything at all. As Geneve is an IP-based protocol, the tunnel layer inherits these same characteristics.

It is possible for a tunnel to encapsulate a protocol, such as TCP, which is connection oriented and maintains session state at that

layer. In addition, implementations MAY model Geneve tunnels as connected, bidirectional links, such as to provide the abstraction of a virtual port. In both of these cases, bidirectionality of the tunnel is handled at a higher layer and does not affect the operation of Geneve itself.

4.5. Constraints on Protocol Features

Geneve is intended to be flexible to a wide range of current and future applications. As a result, certain constraints may be placed on the use of metadata or other aspects of the protocol in order to optimize for a particular use case. For example, some applications may limit the types of options which are supported or enforce a maximum number or length of options. Other applications may only handle certain encapsulated payload types, such as Ethernet or IP. This could be either globally throughout the system or, for example, restricted to certain classes of devices or network paths.

These constraints may be communicated to tunnel endpoints either explicitly through a control plane or implicitly by the nature of the application. As Geneve is defined as a data plane protocol that is control plane agnostic, definition of such mechanisms are beyond the scope of this document.

4.5.1. Constraints on Options

While Geneve options are flexible, a control plane may restrict the number of option TLVs as well as the order and size of the TLVs between tunnel endpoints to make it simpler for a data plane implementation in software or hardware to handle [I-D.ietf-nvo3-encap]. For example, there may be some critical information such as a secure hash that must be processed in a certain order to provide lowest latency or there may be other scenarios where the options must be processed in a certain order due to protocol semantics.

A control plane may negotiate a subset of option TLVs and certain TLV ordering, as well may limit the total number of option TLVs present in the packet, for example, to accommodate hardware capable of processing fewer options [I-D.ietf-nvo3-encap]. Hence, a control plane needs to have the ability to describe the supported TLVs subset and their order to the tunnel endpoints. In the absence of a control plane, alternative configuration mechanisms may be used for this purpose. Such mechanisms are beyond the scope of this document.

4.6. NIC Offloads

Modern NICs currently provide a variety of offloads to enable the efficient processing of packets. The implementation of many of these offloads requires only that the encapsulated packet be easily parsed (for example, checksum offload). However, optimizations such as LSO and LRO involve some processing of the options themselves since they must be replicated/merged across multiple packets. In these situations, it is desirable to not require changes to the offload logic to handle the introduction of new options. To enable this, some constraints are placed on the definitions of options to allow for simple processing rules:

- When performing LSO, a NIC MUST replicate the entire Geneve header and all options, including those unknown to the device, onto each resulting segment unless an option allows an exception. Conversely, when performing LRO, a NIC may assume that a binary comparison of the options (including unknown options) is sufficient to ensure equality and MAY merge packets with equal Geneve headers.

- Options MUST NOT be reordered during the course of offload processing, including when merging packets for the purpose of LRO.

- NICs performing offloads MUST NOT drop packets with unknown options, including those marked as critical, unless explicitly configured.

There is no requirement that a given implementation of Geneve employ the offloads listed as examples above. However, as these offloads are currently widely deployed in commercially available NICs, the rules described here are intended to enable efficient handling of current and future options across a variety of devices.

4.7. Inner VLAN Handling

Geneve is capable of encapsulating a wide range of protocols and therefore a given implementation is likely to support only a small subset of the possibilities. However, as Ethernet is expected to be widely deployed, it is useful to describe the behavior of VLANs inside encapsulated Ethernet frames.

As with any protocol, support for inner VLAN headers is OPTIONAL. In many cases, the use of encapsulated VLANs may be disallowed due to security or implementation considerations. However, in other cases trunking of VLAN frames across a Geneve tunnel can prove useful. As a result, the processing of inner VLAN tags upon ingress or egress from a tunnel endpoint is based upon the configuration of the tunnel

endpoint and/or control plane and not explicitly defined as part of the data format.

5. Transition Considerations

Viewed exclusively from the data plane, Geneve is compatible with existing IP networks as it appears to most devices as UDP packets. However, as there are already a number of tunnel protocols deployed in network virtualization environments, there is a practical question of transition and coexistence.

Since Geneve builds on the base data plane functionality provided by the most common protocols used for network virtualization (VXLAN, NVGRE) it should be straightforward to port an existing control plane to run on top of it with minimal effort. With both the old and new packet formats supporting the same set of capabilities, there is no need for a hard transition - tunnel endpoints directly communicating with each other can use any common protocol, which may be different even within a single overall system. As transit devices are primarily forwarding packets on the basis of the IP header, all protocols appear similar and these devices do not introduce additional interoperability concerns.

To assist with this transition, it is strongly suggested that implementations support simultaneous operation of both Geneve and existing tunnel protocols as it is expected to be common for a single node to communicate with a mixture of other nodes. Eventually, older protocols may be phased out as they are no longer in use.

6. Security Considerations

As encapsulated within a UDP/IP packet, Geneve does not have any inherent security mechanisms. As a result, an attacker with access to the underlay network transporting the IP packets has the ability to snoop, alter or inject packets. Compromised tunnel endpoints or transit devices may also spoof identifiers in the tunnel header to gain access to networks owned by other tenants.

Within a particular security domain, such as a data center operated by a single service provider, the most common and highest performing security mechanism is isolation of trusted components. Tunnel traffic can be carried over a separate VLAN and filtered at any untrusted boundaries.

When crossing an untrusted link, such as the general Internet, VPN technologies such as IPsec [RFC4301] should be used to provide authentication and/or encryption of the IP packets formed as part of Geneve encapsulation (See Section 6.1.1).

Geneve does not otherwise affect the security of the encapsulated packets. As per the guidelines of BCP 72 [RFC3552], the following sections describe potential security risks that may be applicable to Geneve deployments and approaches to mitigate such risks. It is also noted that not all such risks are applicable to all Geneve deployment scenarios, i.e., only a subset may be applicable to certain deployments. So an operator has to make an assessment based on their network environment and determine the risks that are applicable to their specific environment and use appropriate mitigation approaches as applicable.

6.1. Data Confidentiality

Geneve is a network virtualization overlay encapsulation protocol designed to establish tunnels between NVEs over an existing IP network. It can be used to deploy multi-tenant overlay networks over an existing IP underlay network in a public or private data center. The overlay service is typically provided by a service provider, for example a cloud services provider or a private data center operator, this may or not may be the same provider as an underlay service provider. Due to the nature of multi-tenancy in such environments, a tenant system may expect data confidentiality to ensure its packet data is not tampered with (active attack) in transit or a target of unauthorized monitoring (passive attack) for example by other tenant systems or underlay service provider. A compromised network node or a transit device within a data center may passively monitor Geneve packet data between NVEs; or route traffic for further inspection. A tenant may expect the overlay service provider to provide data confidentiality as part of the service or a tenant may bring its own data confidentiality mechanisms like IPsec or TLS to protect the data end to end between its tenant systems. The overlay provider is expected to provide cryptographic protection in cases where the underlay provider is not the same as the overlay provider to ensure the payload is not exposed to the underlay.

If an operator determines data confidentiality is necessary in their environment based on their risk analysis, for example as in multi-tenant environments, then an encryption mechanism SHOULD be used to encrypt the tenant data end to end between the NVEs. The NVEs may use existing well established encryption mechanisms such as IPsec, DTLS, etc.

6.1.1. Inter-Data Center Traffic

A tenant system in a customer premises (private data center) may want to connect to tenant systems on their tenant overlay network in a public cloud data center or a tenant may want to have its tenant systems located in multiple geographically separated data centers for

high availability. Geneve data traffic between tenant systems across such separated networks should be protected from threats when traversing public networks. Any Geneve overlay data leaving the data center network beyond the operator's security domain SHOULD be secured by encryption mechanisms such as IPsec or other VPN technologies to protect the communications between the NVEs when they are geographically separated over untrusted network links. Specification of data protection mechanisms employed between data centers is beyond the scope of this document.

The principles described in Section 4 regarding controlled environments still apply to the geographically separated data center usage outlined in this section.

6.2. Data Integrity

Geneve encapsulation is used between NVEs to establish overlay tunnels over an existing IP underlay network. In a multi-tenant data center, a rogue or compromised tenant system may try to launch a passive attack such as monitoring the traffic of other tenants, or an active attack such as trying to inject unauthorized Geneve encapsulated traffic such as spoofing, replay, etc., into the network. To prevent such attacks, an NVE MUST NOT propagate Geneve packets beyond the NVE to tenant systems and SHOULD employ packet filtering mechanisms so as not to forward unauthorized traffic between tenant systems in different tenant networks. An NVE MUST NOT interpret Geneve packets from tenant systems other than as frames to be encapsulated.

A compromised network node or a transit device within a data center may launch an active attack trying to tamper with the Geneve packet data between NVEs. Malicious tampering of Geneve header fields may cause the packet from one tenant to be forwarded to a different tenant network. If an operator determines the possibility of such threat in their environment, the operator may choose to employ data integrity mechanisms between NVEs. In order to prevent such risks, a data integrity mechanism SHOULD be used in such environments to protect the integrity of Geneve packets including packet headers, options and payload on communications between NVE pairs. A cryptographic data protection mechanism such as IPsec may be used to provide data integrity protection. A data center operator may choose to deploy any other data integrity mechanisms as applicable and supported in their underlay networks, although non-cryptographic mechanisms may not protect the Geneve portion of the packet from tampering.

6.3. Authentication of NVE peers

A rogue network device or a compromised NVE in a data center environment might be able to spoof Geneve packets as if it came from a legitimate NVE. In order to mitigate such a risk, an operator SHOULD use an authentication mechanism, such as IPsec to ensure that the Geneve packet originated from the intended NVE peer, in environments where the operator determines spoofing or rogue devices is a potential threat. Other simpler source checks such as ingress filtering for VLAN/MAC/IP address, reverse path forwarding checks, etc., may be used in certain trusted environments to ensure Geneve packets originated from the intended NVE peer.

6.4. Options Interpretation by Transit Devices

Options, if present in the packet, are generated and terminated by tunnel endpoints. As indicated in Section 2.2.1, transit devices may interpret the options. However, if the packet is protected by tunnel endpoint to tunnel endpoint encryption, for example through IPsec, transit devices will not have visibility into the Geneve header or options in the packet. In such cases transit devices MUST handle Geneve packets as any other IP packet and maintain consistent forwarding behavior. In cases where options are interpreted by transit devices, the operator MUST ensure that transit devices are trusted and not compromised. The definition of a mechanism to ensure this trust is beyond the scope of this document.

6.5. Multicast/Broadcast

In typical data center networks where IP multicasting is not supported in the underlay network, multicasting may be supported using multiple unicast tunnels. The same security requirements as described in the above sections can be used to protect Geneve communications between NVE peers. If IP multicasting is supported in the underlay network and the operator chooses to use it for multicast traffic among tunnel endpoints, then the operator in such environments may use data protection mechanisms such as IPsec with multicast extensions [RFC5374] to protect multicast traffic among Geneve NVE groups.

6.6. Control Plane Communications

A Network Virtualization Authority (NVA) as outlined in [RFC8014] may be used as a control plane for configuring and managing the Geneve NVEs. The data center operator is expected to use security mechanisms to protect the communications between the NVA to NVEs and use authentication mechanisms to detect any rogue or compromised NVEs within their administrative domain. Data protection mechanisms for

control plane communication or authentication mechanisms between the NVA and the NVEs are beyond the scope of this document.

7. IANA Considerations

IANA has allocated UDP port 6081 in the Service Name and Transport Protocol Port Number Registry [IANA-SN] as the well-known destination port for Geneve based on early registration.

Upon publication of this document, this registration will have its reference changed to cite this document [RFC-to-be] and inline with [RFC6335] the assignee and contact of the port entry should be changed to IESG <iesg@ietf.org> and IETF Chair <chair@ietf.org> respectively:

Service Name: geneve
Transport Protocol(s): UDP
Assignee: IESG <iesg@ietf.org>
Contact: IETF Chair <chair@ietf.org>
Description: Generic Network Virtualization Encapsulation (Geneve)
Reference: [RFC-to-be]
Port Number: 6081

In addition, IANA is requested to create a new "Geneve Option Class" registry to allocate Option Classes. This registry is to be placed under a new Network Virtualization Overlay (NVO3) protocols page (to be created) in IANA protocol registries [IANA-PR]. The Geneve Option Class registry shall consist of 16-bit hexadecimal values along with descriptive strings, assignee/contact information and references. The registration rules for the new registry are (as defined by [RFC8126]):

Range	Registration Procedures
0x0000..0x00FF	IETF Review
0x0100..0xFEFF	First Come First Served
0xFF00..0xFFFF	Experimental Use

Initial registrations in the new registry are as follows:

Option Class	Description	Assignee/Contact	References
0x0100	Linux		
0x0101	Open vSwitch (OVS)		
0x0102	Open Virtual Networking (OVN)		
0x0103	In-band Network Telemetry (INT)		
0x0104	VMware, Inc.		
0x0105	Amazon.com, Inc.		
0x0106	Cisco Systems, Inc.		
0x0107	Oracle Corporation		
0x0108..0x0110	Amazon.com, Inc.		

8. Contributors

The following individuals were authors of an earlier version of this document and made significant contributions:

Pankaj Garg
Microsoft Corporation
1 Microsoft Way
Redmond, WA 98052
USA

Email: pankajg@microsoft.com

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

Email: chrisw@redhat.com

Kenneth Duda
Arista Networks
5453 Great America Parkway
Santa Clara, CA 95054
USA

Email: kduda@arista.com

Dinesh G. Dutt
Independent

Email: didutt@gmail.com

Jon Hudson
Independent

Email: jon.hudson@gmail.com

Ariel Hendel
Facebook, Inc.
1 Hacker Way
Menlo Park, CA 94025
USA

Email: ahendel@fb.com

9. Acknowledgements

The authors wish to acknowledge Puneet Agarwal, David Black, Sami Boutros, Scott Bradner, Martin Casado, Alissa Cooper, Roman Danyliw, Bruce Davie, Anoop Ghanwani, Benjamin Kaduk, Suresh Krishnan, Mirja Kuhlewind, Barry Leiba, Daniel Migault, Greg Mirksy, Tal Mizrahi,

Kathleen Moriarty, Magnus Nystrom, Adam Roach, Sabrina Tanamal, Dave Thaler, Eric Vyncke, Magnus Westerlund and many other members of the NVO3 WG for their reviews, comments and suggestions.

The authors would like to thank Sam Aldrin, Alia Atlas, Matthew Bocci, Benson Schliesser, and Martin Vigoureux for their guidance throughout the process.

10. References

10.1. Normative References

[RFC0768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, DOI 10.17487/RFC0768, August 1980, <https://www.rfc-editor.org/info/rfc768>.

[RFC0792] Postel, J., "Internet Control Message Protocol", STD 5, RFC 792, DOI 10.17487/RFC0792, September 1981, <https://www.rfc-editor.org/info/rfc792>.

[RFC1112] Deering, S., "Host extensions for IP multicasting", STD 5, RFC 1112, DOI 10.17487/RFC1112, August 1989, <https://www.rfc-editor.org/info/rfc1112>.

[RFC1191] Mogul, J. and S. Deering, "Path MTU discovery", RFC 1191, DOI 10.17487/RFC1191, November 1990, <https://www.rfc-editor.org/info/rfc1191>.

[RFC2003] Perkins, C., "IP Encapsulation within IP", RFC 2003, DOI 10.17487/RFC2003, October 1996, <https://www.rfc-editor.org/info/rfc2003>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC4443] Conta, A., Deering, S., and M. Gupta, Ed., "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", STD 89, RFC 4443, DOI 10.17487/RFC4443, March 2006, <https://www.rfc-editor.org/info/rfc4443>.

[RFC6040] Briscoe, B., "Tunnelling of Explicit Congestion Notification", RFC 6040, DOI 10.17487/RFC6040, November 2010, <https://www.rfc-editor.org/info/rfc6040>.

[RFC6936] Fairhurst, G. and M. Westerlund, "Applicability Statement for the Use of IPv6 UDP Datagrams with Zero Checksums", RFC 6936, DOI 10.17487/RFC6936, April 2013, <https://www.rfc-editor.org/info/rfc6936>.

[RFC7365] Lasserre, M., Balus, F., Morin, T., Bitar, N., and Y. Rekhter, "Framework for Data Center (DC) Network Virtualization", RFC 7365, DOI 10.17487/RFC7365, October 2014, <https://www.rfc-editor.org/info/rfc7365>.

[RFC8085] Eggert, L., Fairhurst, G., and G. Shepherd, "UDP Usage Guidelines", BCP 145, RFC 8085, DOI 10.17487/RFC8085, March 2017, <https://www.rfc-editor.org/info/rfc8085>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <https://www.rfc-editor.org/info/rfc8200>.

[RFC8201] McCann, J., Deering, S., Mogul, J., and R. Hinden, Ed., "Path MTU Discovery for IP version 6", STD 87, RFC 8201, DOI 10.17487/RFC8201, July 2017, <https://www.rfc-editor.org/info/rfc8201>.

10.2. Informative References

[ETYPES] The IEEE Registration Authority, "IEEE 802 Numbers", <https://www.iana.org/assignments/ieee-802-numbers>.

[I-D.ietf-intarea-tunnels] Touch, J. and M. Townsley, "IP Tunnels in the Internet Architecture", draft-ietf-intarea-tunnels-10 (work in progress), September 2019.

[I-D.ietf-nvo3-dataplane-requirements] Bitar, N., Lasserre, M., Balus, F., Morin, T., Jin, L., and B. Khasnabish, "NVO3 Data Plane Requirements", draft-ietf-nvo3-dataplane-requirements-03 (work in progress), April 2014.

[I-D.ietf-nvo3-encap]
Boutros, S., "NVO3 Encapsulation Considerations", draft-ietf-nvo3-encap-05 (work in progress), February 2020.

[IANA-PR] IANA, "Protocol Registries", <https://www.iana.org/protocols>.

[IANA-SN] IANA, "Service Name and Transport Protocol Port Number Registry", <https://www.iana.org/assignments/service-names-port-numbers>.

[IEEE.802.1Q_2018]
IEEE, "IEEE Standard for Local and Metropolitan Area Networks--Bridges and Bridged Networks", IEEE 802.1Q-2018, DOI 10.1109/ieeestd.2018.8403927, July 2018, <http://ieeexplore.ieee.org/servlet/opac?punumber=8403925>.

[RFC2983] Black, D., "Differentiated Services and Tunnels", RFC 2983, DOI 10.17487/RFC2983, October 2000, <https://www.rfc-editor.org/info/rfc2983>.

[RFC3031] Rosen, E., Viswanathan, A., and R. Callon, "Multiprotocol Label Switching Architecture", RFC 3031, DOI 10.17487/RFC3031, January 2001, <https://www.rfc-editor.org/info/rfc3031>.

[RFC3552] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, DOI 10.17487/RFC3552, July 2003, <https://www.rfc-editor.org/info/rfc3552>.

[RFC3985] Bryant, S., Ed. and P. Pate, Ed., "Pseudo Wire Emulation Edge-to-Edge (PWE3) Architecture", RFC 3985, DOI 10.17487/RFC3985, March 2005, <https://www.rfc-editor.org/info/rfc3985>.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, DOI 10.17487/RFC4301, December 2005, <https://www.rfc-editor.org/info/rfc4301>.

[RFC5374] Weis, B., Gross, G., and D. Ignjatic, "Multicast Extensions to the Security Architecture for the Internet Protocol", RFC 5374, DOI 10.17487/RFC5374, November 2008, <https://www.rfc-editor.org/info/rfc5374>.

[RFC6335] Cotton, M., Eggert, L., Touch, J., Westerlund, M., and S. Cheshire, "Internet Assigned Numbers Authority (IANA) Procedures for the Management of the Service Name and Transport Protocol Port Number Registry", BCP 165, RFC 6335, DOI 10.17487/RFC6335, August 2011, <https://www.rfc-editor.org/info/rfc6335>.

[RFC6438] Carpenter, B. and S. Amante, "Using the IPv6 Flow Label for Equal Cost Multipath Routing and Link Aggregation in Tunnels", RFC 6438, DOI 10.17487/RFC6438, November 2011, <https://www.rfc-editor.org/info/rfc6438>.

[RFC7348] Mahalingam, M., Dutt, D., Duda, K., Agarwal, P., Kreeger, L., Sridhar, T., Bursell, M., and C. Wright, "Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks", RFC 7348, DOI 10.17487/RFC7348, August 2014, <https://www.rfc-editor.org/info/rfc7348>.

[RFC7637] Garg, P., Ed. and Y. Wang, Ed., "NVGRE: Network Virtualization Using Generic Routing Encapsulation", RFC 7637, DOI 10.17487/RFC7637, September 2015, <https://www.rfc-editor.org/info/rfc7637>.

[RFC8014] Black, D., Hudson, J., Kreeger, L., Lasserre, M., and T. Narten, "An Architecture for Data-Center Network Virtualization over Layer 3 (NVO3)", RFC 8014, DOI 10.17487/RFC8014, December 2016, <https://www.rfc-editor.org/info/rfc8014>.

[RFC8086] Yong, L., Ed., Crabbe, E., Xu, X., and T. Herbert, "GRE-in-UDP Encapsulation", RFC 8086, DOI 10.17487/RFC8086, March 2017, <https://www.rfc-editor.org/info/rfc8086>.

[RFC8293] Ghanwani, A., Dunbar, L., McBride, M., Bannai, V., and R. Krishnan, "A Framework for Multicast in Network Virtualization over Layer 3", RFC 8293, DOI 10.17487/RFC8293, January 2018, <https://www.rfc-editor.org/info/rfc8293>.

[VL2] "VL2: A Scalable and Flexible Data Center Network", ACM SIGCOMM Computer Communication Review, DOI 10.1145/1594977.1592576, 2009, <https://www.sigcomm.org/sites/default/files/ccr/papers/2009/October/1594977-1592576.pdf>.

Authors' Addresses

Jesse Gross (editor)

Email: jesse@kernel.org

Ilango Ganga (editor)
Intel Corporation
2200 Mission College Blvd.
Santa Clara, CA 95054
USA

Email: ilango.s.ganga@intel.com

T. Sridhar (editor)
VMware, Inc.
3401 Hillview Ave.
Palo Alto, CA 94304
USA

Email: tsridhar@vmware.com

NVO3
Internet-Draft
Intended status: Informational
Expires: April 15, 2019

D. Migault
Ericsson
S. Boutros
D. Wings
VMware, Inc.
S. Krishnan
Kaloom

October 12, 2018

Geneve Security Requirements
draft-mglt-nvo3-geneve-security-requirements-04

Abstract

The document defines the security requirements to protect tenants overlay traffic against security threats from the NVO3 network components that are interconnected with tunnels implemented using Generic Network Virtualization Encapsulation (Geneve).

The document provides two sets of security requirements: 1. requirements to evaluate the data plane security of a given deployment of Geneve overlay. Such requirements are intended to Geneve overlay provider to evaluate a given deployment.
2. requirement a security mechanism need to fulfill to secure any deployment of Geneve overlay deployment

Table of Contents

1. Requirements Notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. Introduction

The network virtualization overlay over Layer 3 (NVO3) as depicted in Figure 1, allows an overlay cloud provider to provide a logical L2/L3 interconnect for the Tenant Systems TSes that belong to a specific tenant network. A packet received from a TS is encapsulated by the ingress Network Virtualization Edge (NVE). The encapsulated packet is then sent to the remote NVE through a tunnel. When reaching the egress NVE of the tunnel, the packet is decapsulated and forwarded to the target TS. The L2/L3 address mappings to the remote NVE(s) are distributed to the NVEs by a logically centralized Network Virtualization Authority (NVA) or using a distributed control plane such as Ethernet-VPN. In a datacenter, the NVO3 tunnels can be implemented using Generic Network Virtualization Encapsulation (Geneve) [I-D.ietf-nvo3-geneve]. Such Geneve tunnels establish NVE-to-NVE communications, may transit within the data center via Transit device. The Geneve tunnels overlay network enable multiple Virtual Networks to coexist over a shared underlay infrastructure, and a Virtual Network may span a single data center or multiple data centers.

The underlay infrastructure on which the multi-tenancy overlay networks are hosted, can be owned and provided by an underlay provider who may be different from the overlay cloud provider.

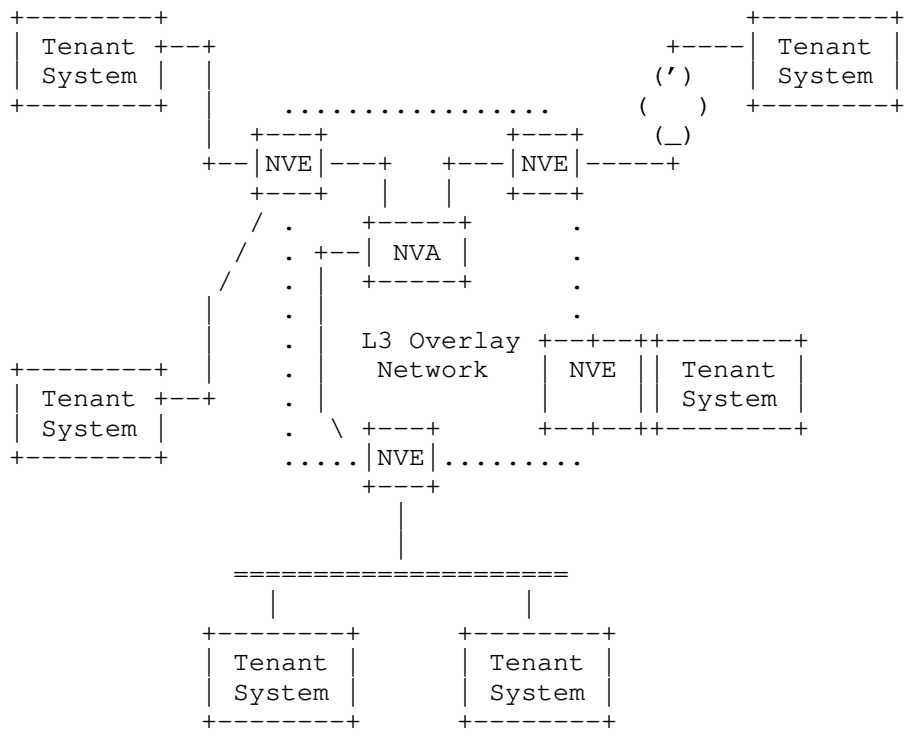


Figure 1: Generic Reference Model for Network Virtualization Overlays [RFC7365]

This document discusses the security risks that a Geneve based NVO3 network may encounter. In addition, this document lists the requirements to protect the Geneve packet components defined in [I-D.ietf-nvo3-geneve] that include the Geneve tunnel IP and UDP header, the Geneve Header, Geneve options, and inner payload.

The document provides two sets of security requirements:

1. SEC-OP: requirements to evaluate a given deployment of Geneve overlay. Such requirements are intended to Geneve overlay provider to evaluate a given deployment. Security of the Geneve packet may be achieved using various mechanisms. Typically, some deployments may use a limited subset of the capabilities provided by Geneve and rely on specific assumptions. Given these specificities, the secure deployment of a given Geneve deployment may be achieved reusing specific mechanisms such as for example DTLS [RFC6347] or IPsec [RFC4301]. On the other hand, the definition of a security mechanisms that enables to secure any Geneve deployment requires the design of a Geneve specific

mechanism. Note that the security s limited to the security of the data plane only. Additional requirements for the control plan MAY be considered in [I-D.ietf-nvo3-security-requirements].

2. SEC-GEN: requirements a security mechanism need to fulfill to secure any deployment of Geneve overlay deployment. Such mechanism may require the design of a specific solution. In the case new protocol needs to be design, the document strongly recommend to re-use existing security protocols like IP Security (IPsec) [RFC4301] and Datagram Transport Layer Security (DTLS) [RFC6347], and existing encryption algorithms (such as [RFC8221]), and authentication protocols.

This document assumes the following roles are involved: - Tenant: designates the entity that connects various systems within a single virtualized network. The various system can typically be containers, VMs implementing a single or various functions.
- Geneve Overlay Provider: provides the Geneve overlay that seamlessly connect the various Tenant Systems over a given virtualized network.
- Infrastructure Provider: provides the infrastructure that runs the Geneve overlay network as well as the Tenant System. A given deployment may consider different infrastructure provider with different level of trust. Typically the Geneve overlay network may use a public cloud to extend the resource of a private cloud. Similarly, a edge computing may extend its resources using resource of the core network.

Tenant, Geneve Overlay Provider and Infrastructure Provider can be implemented by a single or various different entities with different level of trust between each other. The simplest deployment may consists in a single entity running its systems in its data center and using Geneve in order to manage its internal resources. A more complex use case may consider that a Tenant subscribe to the Geneve Overlay Provider which manage the virtualized network over various type of infrastructure. The trust between the Tenant, Geneve Overlay Provider and Infrastructure Provider may be limited.

Given the different relations between Tenant, Geneve Overlay Provider and Infrastructure Provider, this document aims providing requirements to ensure: 1. The Geneve Overlay Provider delivers tenant payload traffic (Geneve inner payload) and ensuring privacy and integrity. 2. The Geneve Overlay Provider provides the necessary means to prevent injection or redirection of the Tenant traffic from a rogue node in the Geneve overlay network or a rogue node from the infrastructure. 3. The Geneve Overlay Provider can rely on the Geneve overlay in term of robustness and reliability of the signaling associated to the Geneve packets (Geneve tunnel header,

Geneve header and Geneve options) in order to appropriately manage its overlay.

3. Terminology

This document uses the terminology of [RFC8014], [RFC7365] and [I-D.ietf-nvo3-geneve].

4. Security Threats

Attacks from compromised NVO3 and underlay network devices, and attacks from compromised tenant systems defined in [I-D.ietf-nvo3-security-requirements]. This document considers these attacks in the scope of Geneve, that is when the attackers knowing the details of the Geneve packets can perform their attacks by changing fields in the Geneve tunnel header, base header, Geneve options and Geneve inner payload. The scope of Geneve excludes security requirements related to the control plane.

Threats include traffic analysis, sniffing, injection, redirection, and replay. Based on these threats, this document enumerates the security requirements.

Threats are divided into two categories: passive attack and active attack.

Threats are always associated with risks and the evaluation of these risks depend among other things on the environment.

4.1. Passive Attacks

Passive attacks include traffic analysis (noticing which workloads are communicating with which other workloads, how much traffic, and when those communications occur) and sniffing (examining traffic for useful information such as personally-identifyable information or protocol information (e.g., TLS certificate, overlay routing protocols).

A rogue element of the overlay Geneve network under the control of an attacker may leak and redirect the traffic from a virtual network to the attacker for passive monitoring [RFC7258].

Avoiding leaking information is hard to enforced and the security requirements expect to mitigate such attacks by lowering the consequences, typically making leaked data unusable to an attacker..

4.2. Active Attacks

Active attacks involve modifying packets, injecting packets, or interfering with packet delivery (such as by corrupting packet checksum). Active attack may target the Tenant System or the Geneve overlay.

There are multiple motivations to inject illegitimate traffic into a tenants network. When the rogue element is on the path of the TS traffic, it may be able to inject and receive the corresponding messages back. On the other hand, if the attacker is not on the path of the TS traffic it may be limited to only inject traffic to a TS without receiving any response back. When rogue element have access to the traffic in both directions, the possibilities are only limited by the capabilities of the other on path elements - Transit device, NVE or TS - to detect and protect against the illegitimate traffic. On the other hand, when the rogue element is not on path, the surface for such attacks remains still quite large. For example, an attacker may target a specific TS or application by crafting a specific packet that can either generate load on the system or crash the system or application. TCP syn flood typically overload the TS while not requiring the ability to receive responses. Note that udp application are privileged target as they do not require the establishment of a session and are expected to treat any incoming packets.

Traffic injection may also be used to flood the virtual network to disrupt the communications between the TS or to introduce additional cost for the tenant, for example when pricing considers the traffic inside the virtual network. The two latest attacks may also take advantage of applications with a large factor of amplification for their responses as well as applications that upon receiving a packet interact with multiple TS. Similarly, applications running on top of UDP are privileged targets.

Note also that an attacker that is not able to receive the response traffic, may use other channels to evaluate or measure the impact of the attack. Typically, in the case of a service, the attacker may have access, for example, to a user interface that provides indication on the level of disruption and the success of an attack, Such feed backs may also be used by the attacker to discover or scan the network.

Preventing traffic to cross virtual networks, reduce the surface of attack, but rogue element main still perform attacks within a given virtual network by replaying a legitimate packet. Some variant of such attack also includes modification of unprotected parts when available in order for example to increase the payload size.

5. Requirements for Security Mitigations

The document assumes that Security protocols, algorithms, and implementations provide the security properties for which they are designed, an attack caused by a weakness in a cryptographic algorithm is out of scope.

Protecting network connecting TSes and NVEs which could be accessible to outside attackers is out of scope.

An attacker controlling an underlying network device may break the communication of the overlays by discarding or delaying the delivery of the packets passing through it. The security consideration to prevent this type of attack is out of scope of this document.

Securing communication between NVAs and NVEs is out of scope.

Selectively providing integrity / authentication, confidentiality / encryption of only portions of the Geneve packet is in scope. This will be the case if the Tenant Systems uses security protocol to protect its communications.

5.1. Protection Against Traffic Sniffing

Passive attacks consists in inferring information about a virtualized network or some Tenant System from observing the traffic. This could also involve the correlation between observed traffic and additional information. For example, a passive network observer can determine two virtual machines are communicating by manipulating activity or network activity of other virtual machines on that same host. For example, the attacker could control (or be otherwise aware of) network activity of the other VMs running on the same host, and deduce other network activity is due to a victim VM.

The inner payload, unless protection is provided by the Tenant System reveals the content of the communication. This may mitigate by the Tenant using application level security such as, for example JSON Web Encryption [RFC7516] or transport layer security such as DTLS [RFC6347] or TLS [RFC8446] or IPsec/ESP [RFC4303]. However none of these security protocols are sufficient to protect the entire inner payload. IPsec/ESP still leave in clear the optional L2 layer information as well as the IP addresses and some IP options. In addition to these pieces of information, the use of TLS or DTLS reveals the transport layer protocol as well as ports.

A secure deployment of a Geneve overlay must fulfill the requirement below:

1. SEC-OP-1: A secure deployment of a Geneve overlay SHOULD by default encrypt the inner payload. A Geneve overlay provider MAY disable this capability for example when encryption is performed by the Tenant System and that level of confidentiality is believed to be sufficient. In order to provide additional protection to traffic already encrypted by the Tenant the Geneve network operator MAY partially encrypt the clear part of the inner payload.

A Geneve security mechanism must fulfill the requirements below:

- SEC-GEN-1: Geneve security mechanism MUST provide the capability to encrypt the inner payload.

- SEC-GEN-2: Geneve security mechanism SHOULD provide the capability to partially encrypt the inner payload header.

The Geneve Header and Geneve Options contains metadata information related to the communications. Note that a Geneve packet may have a combination of Geneve options that needs to be read by transit device, in which case this option needs to be read by the transit device while other options MAY only be accessed by the tunnel endpoint. Information revealed as well as correlation with traffic volumetry may reveal pattern traffic within a given virtualized network as well as any information revealed by the current and future Geneve Option.

A secure deployment of a Geneve overlay must fulfill the requirement below:

- SEC-OP-2: A secure deployment of a Geneve overlay MUST evaluate the information associated to the leakage of the Geneve Outer Header, Geneve Header and Geneve Option. When those information are likely to carry sensitive information. they MUST NOT be transmit in clear text.

- SEC-OP-3: A secure deployment of a Geneve overlay MUST evaluate the risk associated to traffic pattern recognition. When a risk has been identified, traffic pattern recognition MUST be addressed with padding policies as well as generation of dummy packets.

A Geneve security mechanism must fulfill the requirements below:

- SEC-GEN-3: Geneve security mechanism MUST provide the capability to encrypt a single or a set of options while leave other Geneve Option in clear. Reversely, a Geneve security mechanism MUST be able to leave a Geneve option in clear, while encrypting the others.

- SEC-GEN-4: Geneve security mechanism MUST provide means to encrypt the information of Geneve Header. Reversely, a Geneve security mechanism MUST be able to leave in clear header information while encrypting the other.

- SEC-GEN-5: Geneve security mechanism MUST provide the ability to pad a Geneve packet.

- SEC-GEN-6: Geneve security mechanism MUST provide the ability to send dummy packets.

5.2. Protecting Against Traffic Injection

Traffic injection from a rogue non legitimate NVO3 Geneve overlay device or a rogue underlay transit device can target an NVE, a transit underlay device or a Tenant System. Targeting a Tenant's System requires a valid MAC and IP addresses of the Tenant's System.

Tenant's System may protect their communications using IPsec or TLS. Such protection protects the Tenants from receiving spoofed packets, as any injected packet is expected to be discarded by the destination Tenant's System. Such protection does not protect the tenant system from receiving illegitimate packets that may disrupt the Tenant's System performance. The Geneve overlay network MAY still need to prevent such spoofed Tenant's system packets from being steered to the Tenant's system. When the Tenant's Systems are not protecting their communications, the Geneve overlay network SHOULD be able to to prevent a rogue device from injecting traffic into the overlay network.

In order to prevent traffic injection to one virtual network, the destination legitimate Geneve NVE MUST be able to authenticate the incoming Geneve packets from the source NVE. The Geneve architecture considers transit devices that MAY process some Geneve Option without affecting the Geneve packet. These transit device MAY Authenticate the Geneve packet as part of the Geneve packet processing but MAY also process other Geneve options. As a result, integrity protection and authentication SHOULD be performed by transit device, prior to any processing.

A secure deployment of a Geneve overlay must fulfill the requirement below:

- SEC-OP-4: A secure deployment of a Geneve overlay SHOULD authenticate communications between NVE to protect the Geneve Overlay infrastructure as well as the Tenants System's communications (Geneve Packet). A Geneve overlay provider MAY disable authentication of the inner packet and delegates it to the

Tenant Systems when communications between Tenant's System is secured. This is NOT RECOMMENDED. To prevent injection between virtualized network, it is strongly RECOMMENDED that at least the Geneve Header is authenticated.

- SEC-OP-5: A secure deployment of a Geneve overlay SHOULD NOT process data prior authentication. If that is not possible, the Geneve overlay provider SHOULD evaluate its impact.

A Geneve security mechanism must fulfill the requirements below:

- SEC-GEN-8: Geneve Security mechanism MUST provide means for a tunnel endpoint (NVE) to authenticate data prior it is being processed. A tunnel endpoint (NVE) MUST be able to authenticate at least:
  - the Geneve Header and a subset of Geneve Options
  - the Geneve Header, a subset of Geneve options and the Geneve inner payload
  - the Geneve Header, a subset of Geneve options and the Geneve inner payload or the portion of the inner payload in case the Tenant's System provides some authentication mechanism.
- SEC-GEN-9: Geneve Security mechanism SHOULD provide means for a transit device to authenticate the Geneve Option prior processing it. Authentication MAY concern the whole Geneve packet, but MAY be limited to the Geneve Option.

5.3. Protecting Against Traffic Redirection

A rogue device of the NVO3 overlay Geneve network or the underlay network may redirect the traffic from a virtual network to the attacker for passive or active attacks. If the rogue device is in charge of the securing the Geneve packet, then Geneve security mechanisms are not intended to address this threat. More specifically, a rogue source NVE will still be able to redirect the traffic in clear text before protecting (and encrypting the packet). A rogue destination NVE will still be able to redirect the traffic in clear text after decrypting the Geneve packets. The same occurs with a rogue transit that is in charge of encrypting and decrypting a Geneve Option, Geneve Option or any information. The security mechanisms are intended to protect a Geneve information from any on path node. Note that modern cryptography recommend the use of authenticated encryption. This section assumes such algorithms are used, and as such encrypted packets are also authenticated.

To prevent an attacker located in the middle between the NVEs and modifying the tunnel address information in the data packet header to redirect the data traffic, the solution need to provide confidentiality protection for data traffics exchanged between NVEs.

Requirements are similar as those provided in section Section 5.1 to mitigate sniffing attacks and those provided in section Section 5.2 to mitigate traffic injection attacks.

5.4. Protecting Against Traffic Replay

A rogue device of the NVO3 overlay Geneve network or the underlay network may replay a Geneve packet, to load the network and/or a specific Tenant System with a modified Geneve payload. In some cases, such attacks may target an increase of the tenants costs.

When traffic between tenants is not protected, the rogue device may forward the modified packet over a valid (authenticated) Geneve Header. The crafted packet may for example, include a specifically crafted application payload for a specific Tenant Systems application, with the intention to load the tenant specific application.

Updating the Geneve header and option parameters such as setting an OAM bit, adding bogus option TLVs, or setting a critical bit, may result in different processing behavior, that could greatly impact performance of the overlay network and the underlay infrastructure and thus affect the tenants traffic delivery.

The NVO3 overlay network and underlay network nodes that may address such attacks MUST provide means to authenticate the Geneve packet components.

A secure deployment of a Geneve overlay must fulfill the requirement below:

- SEC-OP-6: A secure deployment of a Geneve overlay MUST evaluate the flows subject to replay attacks. Flows that are subject to this attacks MUST be authenticated with an anti replay mechanism. Note that when partial authentication is provided, the part not covered by the authentication remains a surface of attack. It is strongly RECOMMENDED that the Geneve Header is both authenticated with anti replay protection.

A Geneve security mechanism must fulfill the requirements below:

- SEC-GEN-10: Geneve Security mechanism MUST provide means for a tunnel endpoint (NVE) to validate the Geneve Header corresponds to the Geneve payload, and discard such packets.

5.5. Security Management

A secure deployment of a Geneve overlay must fulfill the requirement below:

- SEC-OP-7: A secure deployment of a Geneve overlay MUST define the security policies that associates the encryption, and authentication associated to each flow between NVEs.

- SEC-OP-8: A secure deployment of a Geneve overlay SHOULD define distinct material for each flow. The cryptographic depends on the nature of the flow (multicast, unicast) as well as on the security mechanism enabled to protect the flow.

A Geneve security mechanism must fulfill the requirements below:

- SEC-GEN-11: A Geneve security mechanism MUST be managed via security policies associated for each traffic flow to be protected. Geneve overlay provider MUST be able to configure NVEs with different security policies for different flows. A flow MUST be identified at minimum by the Geneve virtual network identifier and the inner IP and transport headers, and optionally additional fields which define a flow (e.g., inner IP DSCP, IPv6 flow id, Geneve options).

- SEC-GEN-12: A Geneve security mechanism MUST be able to assign different cryptographic keys to protect the unicast tunnels between NVEs respectively.

- SEC-GEN-13: A Geneve security mechanisms, when multicast is used, packets,MUST be able to assign distinct cryptographic group keys to protect the multicast packets exchanged among the NVEs within different multicast groups. Upon receiving a data packet, an egress Geneve NVE MUST be able to verify whether the packet is sent from a proper ingress NVE which is authorized to forward that packet.

6. IANA Considerations

There are no IANA consideration for this document.

7. Security Considerations

The whole document is about security.

Limiting the coverage of the authentication / encryption provides some means for an attack to craft special packets.

The current document details security requirements that are related to the Geneve protocol. Instead, [I-D.ietf-nvo3-security-requirements] provides generic architecture security requirement upon the deployment of an NVO3 overlay network. It is strongly recommended to read that document as architecture requirements also apply here. In addition, architecture security requirements go beyond the scope of Geneve communications, and as such are more likely to address the security needs upon deploying an Geneve overlay network.

7.1. TLS

This section compares how NVE communications using TLS meet the security requirements for a secure Geneve overlay deployment. In this example TLS is used over the Geneve Outer Header and secured the Geneve Header, Geneve Options and the inner payload.

The use of TLS MAY fill the security requirements for a secure Geneve deployment. However TLS cannot be considered as the Geneve security mechanism enabling all Geneve deployments.

The use of to secure a Geneve overlay deployment TLS meets SEC-OP-1 as it protects the inner payload of the tenant. It meets SEC-OP-2 as except from the UDP port, no information concerning Geneve is leaked. SEC-OP-3 is not met as TLS does not provide the ability to send dummy traffic, nor to pad. SEC-OP-4 is met as the communication is authenticated, including the Geneve Header. SEC-OP-5 is met as the Geneve Packet is processed once it has been authenticated. SEC-OP-6 is met as TLS comes with anti replay protection. SEC-OP-7 and SEC-OP-8 may also be met with security policies established per UDP destination port where only unicast is considered.

The use of TLS as a generic Geneve Security mechanism meets SEC-GEN-1 as it encrypts the inner payload. However, TLS, but does not enable partial encryption of the inner payload. TLS does not meet SEC-GEN3 or SEC-GEN-4 that requires the ability to encrypt of a subset of the Geneve Options or the Geneve Header information. In addition, TLS does not enable that some Geneve option of Header information remain in clear text while other are encrypted. Typically TLS would not be compatible with transit device. In addition is make the Geneve option visible to the transit device, TLS does not provide the

ability for a transit device to authenticate the option before processing it. SEC-GEN-5 and SEC-GEN-6 are not met as TLS does not provide padding nor the ability to generate dummy packets. TLS does not meet SEC-GEN-8 that requires the ability to authenticate some combination of Geneve Header, Geneve Options, (partial) inner payload. TLS does not meet SEC-GEN-9 that requires the ability to authenticate a single Geneve Option. TLS meets SEC-GEN-10 as it provides anti replay mechanism to the authentication. SEC-GEN-11 is not natively supported as TLS security is established by UDP destination ports, rather than by flow. If more than one security policy or flow needs to be considered a binding between flow and ports needs to be established. SEC-GEN-13 is not met for mutlicast traffic.

7.2. IPsec

The use of IPsec/ESP or IPsec/AH share most of the analysis performed for TLS. The main advantages of using IPsec would be that IPsec supports multicast communications and natively supports flow based security policies. However, the use of these security policies in a context of Geneve is not natively supported.

8. Acknowledgments

We would like to thank Ilango S Ganaga for its useful reviews and clarifications as well as Matthew Bocci, Sam Aldrin and Ignas Bagdona for moving the work forward.

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, DOI 10.17487/RFC4301, December 2005, <https://www.rfc-editor.org/info/rfc4301>.

[RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, DOI 10.17487/RFC4303, December 2005, <https://www.rfc-editor.org/info/rfc4303>.

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012, <https://www.rfc-editor.org/info/rfc6347>.

[RFC7258] Farrell, S. and H. Tschofenig, "Pervasive Monitoring Is an Attack", BCP 188, RFC 7258, DOI 10.17487/RFC7258, May 2014, <https://www.rfc-editor.org/info/rfc7258>.

[RFC7365] Lasserre, M., Balus, F., Morin, T., Bitar, N., and Y. Rekhter, "Framework for Data Center (DC) Network Virtualization", RFC 7365, DOI 10.17487/RFC7365, October 2014, <https://www.rfc-editor.org/info/rfc7365>.

[RFC7516] Jones, M. and J. Hildebrand, "JSON Web Encryption (JWE)", RFC 7516, DOI 10.17487/RFC7516, May 2015, <https://www.rfc-editor.org/info/rfc7516>.

[RFC8014] Black, D., Hudson, J., Kreeger, L., Lasserre, M., and T. Narten, "An Architecture for Data-Center Network Virtualization over Layer 3 (NVO3)", RFC 8014, DOI 10.17487/RFC8014, December 2016, <https://www.rfc-editor.org/info/rfc8014>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8221] Wouters, P., Migault, D., Mattsson, J., Nir, Y., and T. Kivinen, "Cryptographic Algorithm Implementation Requirements and Usage Guidance for Encapsulating Security Payload (ESP) and Authentication Header (AH)", RFC 8221, DOI 10.17487/RFC8221, October 2017, <https://www.rfc-editor.org/info/rfc8221>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

9.2. Informative References

[I-D.ietf-nvo3-geneve]
Gross, J., Ganga, I., and T. Sridhar, "Geneve: Generic Network Virtualization Encapsulation", draft-ietf-nvo3-geneve-08 (work in progress), October 2018.

[I-D.ietf-nvo3-security-requirements]
Hartman, S., Zhang, D., Wasserman, M., Qiang, Z., and M. Zhang, "Security Requirements of NVO3", draft-ietf-nvo3-security-requirements-07 (work in progress), June 2016.

Authors' Addresses

Daniel Migault
Ericsson
8275 Trans Canada Route
Saint Laurent, QC 4S 0B6
Canada

EMail: daniel.migault@ericsson.com

Sami Boutros
VMware, Inc.

EMail: boutros@vmware.com<

Dan Wings
VMware, Inc.

EMail: dwing@vmware.com

Suresh Krishnan
Kaloom

EMail: suresh@kaloom.com

NVO3
Internet-Draft
Intended status: Informational
Expires: September 1, 2019

D. Migault
Ericsson
S. Boutros
D. Wings
VMware, Inc.
S. Krishnan
Kaloom

February 28, 2019

Geneve Security Requirements
draft-mglt-nvo3-geneve-security-requirements-06

Abstract

The document defines the security requirements to protect tenants overlay traffic against security threats from the NVO3 network components that are interconnected with tunnels implemented using Generic Network Virtualization Encapsulation (Geneve).

The document provides two sets of security requirements: 1. requirements to evaluate the data plane security of a given deployment of Geneve overlay. Such requirements are intended to Geneve overlay provider to evaluate a given deployment.
2. requirement a security mechanism need to fulfill to secure any deployment of Geneve overlay deployment

Table of Contents

1. Requirements Notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. Introduction

The network virtualization overlay over Layer 3 (NVO3) as depicted in Figure 1, allows an overlay cloud provider to provide a logical L2/L3 interconnect for the Tenant Systems TSes that belong to a specific tenant network. A packet received from a TS is encapsulated by the ingress Network Virtualization Edge (NVE). The encapsulated packet is then sent to the remote NVE through a tunnel. When reaching the egress NVE of the tunnel, the packet is decapsulated and forwarded to the target TS. The L2/L3 address mappings to the remote NVE(s) are distributed to the NVEs by a logically centralized Network Virtualization Authority (NVA) or using a distributed control plane such as Ethernet-VPN. In a datacenter, the NVO3 tunnels can be implemented using Generic Network Virtualization Encapsulation (Geneve) [I-D.ietf-nvo3-geneve]. Such Geneve tunnels establish NVE-to-NVE communications, may transit within the data center via Transit device. The Geneve tunnels overlay network enable multiple Virtual Networks to coexist over a shared underlay infrastructure, and a Virtual Network may span a single data center or multiple data centers.

The underlay infrastructure on which the multi-tenancy overlay networks are hosted, can be owned and provided by an underlay provider who may be different from the overlay cloud provider.

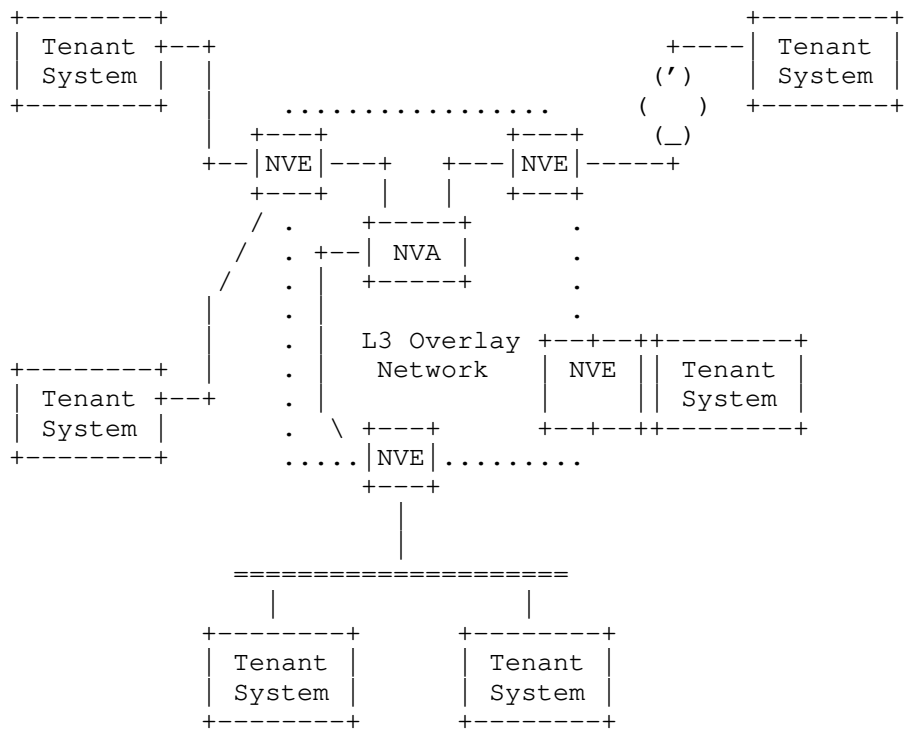


Figure 1: Generic Reference Model for Network Virtualization Overlays [RFC7365]

This document discusses the security risks that a Geneve based NVO3 network may encounter. In addition, this document lists the requirements to protect the Geneve packet components defined in [I-D.ietf-nvo3-geneve] that include the Geneve tunnel IP and UDP header, the Geneve Header, Geneve options, and inner payload.

The document provides two sets of security requirements:

1. SEC-OP: requirements to evaluate a given deployment of Geneve overlay. Such requirements are intended to Geneve overlay provider to evaluate a given deployment. Security of the Geneve packet may be achieved using various mechanisms. Typically, some deployments may use a limited subset of the capabilities provided by Geneve and rely on specific assumptions. Given these specificities, the secure deployment of a given Geneve deployment may be achieved reusing specific mechanisms such as for example DTLS [RFC6347] or IPsec [RFC4301]. On the other hand, the definition of a security mechanisms that enables to secure any Geneve deployment requires the design of a Geneve specific

mechanism. Note that the security s limited to the security of the data plane only. Additional requirements for the control plan MAY be considered in [I-D.ietf-nvo3-security-requirements]. A given Geneve deployment will be considered secured when matching with all SEC-OP requirements does not raises any concern. As such the given deployment will be considered passing SEC-OP requirements that are not applicable.

2. SEC-GEN: requirements a security mechanism need to fulfill to secure any deployment of Geneve overlay deployment. Such mechanism may require the design of a specific solution. In the case new protocol needs to be design, the document strongly recommend to re-use existing security protocols like IP Security (IPsec) [RFC4301] and Datagram Transport Layer Security (DTLS) [RFC6347], and existing encryption algorithms (such as [RFC8221]), and authentication protocols. A given candidate for a security mechanism will be considered as valid when matching with all SEC-GEN requirements does not raise any concern. In other words, at least all MUST status are met.

This document assumes the following roles are involved: - Tenant: designates the entity that connects various systems within a single virtualized network. The various system can typically be containers, VMs implementing a single or various functions.
- Geneve Overlay Provider: provides the Geneve overlay that seamlessly connect the various Tenant Systems over a given virtualized network.
- Infrastructure Provider: provides the infrastructure that runs the Geneve overlay network as well as the Tenant System. A given deployment may consider different infrastructure provider with different level of trust. Typically the Geneve overlay network may use a public cloud to extend the resource of a private cloud. Similarly, a edge computing may extend its resources using resource of the core network.

Tenant, Geneve Overlay Provider and Infrastructure Provider can be implemented by a single or various different entities with different level of trust between each other. The simplest deployment may consists in a single entity running its systems in its data center and using Geneve in order to manage its internal resources. A more complex use case may consider that a Tenant subscribe to the Geneve Overlay Provider which manage the virtualized network over various type of infrastructure. The trust between the Tenant, Geneve Overlay Provider and Infrastructure Provider may be limited.

Given the different relations between Tenant, Geneve Overlay Provider and Infrastructure Provider, this document aims providing requirements to ensure: 1. The Geneve Overlay Provider delivers

tenant payload traffic (Geneve inner payload) and ensuring privacy and integrity. 2. The Geneve Overlay Provider provides the necessary means to prevent injection or redirection of the Tenant traffic from a rogue node in the Geneve overlay network or a rogue node from the infrastructure. 3. The Geneve Overlay Provider can rely on the Geneve overlay in term of robustness and reliability of the signaling associated to the Geneve packets (Geneve tunnel header, Geneve header and Geneve options) in order to appropriately manage its overlay.

3. Terminology

This document uses the terminology of [RFC8014], [RFC7365] and [I-D.ietf-nvo3-geneve].

4. Security Threats

This section considers attacks performed by NVE, network devices or any other devices using Geneve, that is when the attackers knowing the details of the Geneve packets can perform their attacks by changing fields in the Geneve tunnel header, base header, Geneve options and Geneve inner payload. Attacks related to the control plane are outside the scope of this document. The reader is encouraged to read [I-D.ietf-nvo3-security-requirements] for a similar threat analysis of NVO3 overlay networks.

Threats include traffic analysis, sniffing, injection, redirection, and replay. Based on these threats, this document enumerates the security requirements.

Threats are divided into two categories: passive attack and active attack.

Threats are always associated with risks and the evaluation of these risks depend among other things on the environment.

4.1. Passive Attacks

Passive attacks include traffic analysis (noticing which workloads are communicating with which other workloads, how much traffic, and when those communications occur) and sniffing (examining traffic for useful information such as personally-identifyable information or protocol information (e.g., TLS certificate, overlay routing protocols).

Passive attacks may also consist in inferring information about a virtualized network or some Tenant System from observing the Geneve traffic. This could also involve the correlation between observed

traffic and additional information. For example, a passive network observer can determine two virtual machines are communicating by manipulating activity or network activity of other virtual machines on that same host. For example, the attacker could control (or be otherwise aware of) network activity of the other VMs running on the same host, and deduce other network activity is due to a victim VM.

A rogue element of the overlay Geneve network under the control of an attacker may leak and redirect the traffic from a virtual network to the attacker for passive monitoring [RFC7258].

Avoiding leaking information is hard to enforced. The security requirements provided in section {{sniffing} expect to mitigate such attacks by lowering the consequences, typically making leaked data unusable to an attacker.

4.2. Active Attacks

Active attacks involve modifying Geneve packets, injecting Geneve packets, or interfering with Geneve packet delivery (such as by corrupting packet checksum). Active attack may target the Tenant System or the Geneve overlay.

There are multiple motivations to inject illegitimate traffic into a tenants network. When the rogue element is on the path of the TS traffic, it may be able to inject and receive the corresponding messages back. On the other hand, if the attacker is not on the path of the TS traffic it may be limited to only inject traffic to a TS without receiving any response back. When rogue element have access to the traffic in both directions, the possibilities are only limited by the capabilities of the other on path elements - Transit device, NVE or TS - to detect and protect against the illegitimate traffic. On the other hand, when the rogue element is not on path, the surface for such attacks remains still quite large. For example, an attacker may target a specific TS or application by crafting a specific packet that can either generate load on the system or crash the system or application. TCP syn flood typically overload the TS while not requiring the ability to receive responses. Note that udp application are privileged target as they do not require the establishment of a session and are expected to treat any incoming packets.

Traffic injection may also be used to flood the virtual network to disrupt the communications between the TS or to introduce additional cost for the tenant, for example when pricing considers the traffic inside the virtual network. The two latest attacks may also take advantage of applications with a large factor of amplification for their responses as well as applications that upon receiving a packet

interact with multiple TS. Similarly, applications running on top of UDP are privileged targets.

Note also that an attacker that is not able to receive the response traffic, may use other channels to evaluate or measure the impact of the attack. Typically, in the case of a service, the attacker may have access, for example, to a user interface that provides indication on the level of disruption and the success of an attack, Such feed backs may also be used by the attacker to discover or scan the network.

Preventing traffic to cross virtual networks, reduce the surface of attack, but rogue element main still perform attacks within a given virtual network by replaying a legitimate packet. Some variant of such attack also includes modification of unprotected parts when available in order for example to increase the payload size.

5. Requirements for Security Mitigations

The document assumes that Security protocols, algorithms, and implementations provide the security properties for which they are designed, an attack caused by a weakness in a cryptographic algorithm is out of scope. The algorithm used MUST follow the cryptographic guidance such as [RFC8247], [RFC8221] or [RFC7525]. In this context, when the document mentions encryption, it assumes authenticated encryption.

Protecting network connecting TSes and NVEs which could be accessible to outside attackers is out of scope.

An attacker controlling an underlying network device may break the communication of the overlays by discarding or delaying the delivery of the packets passing through it. The security consideration to prevent this type of attack is out of scope of this document.

Securing communication between NVAs and NVEs is out of scope.

Selectively providing integrity / authentication, confidentiality / encryption of only portions of the Geneve packet is in scope. This will be the case if the Tenant Systems uses security protocol to protect its communications.

5.1. Protection Against Traffic Sniffing

The inner payload, unless protection is provided by the Tenant System reveals the content of the communication. This may be mitigate by the Tenant using application level security such as, for example JSON Web Encryption [RFC7516] or transport layer security such as DTLS

[RFC6347] or TLS [RFC8446] or IPsec/ESP [RFC4303]. However none of these security protocols are sufficient to protect the entire inner payload. IPsec/ESP still leave in clear the optional L2 layer information as well as the IP addresses and some IP options. In addition to these pieces of information, the use of TLS or DTLS reveals the transport layer protocol as well as ports. As a result, the confidentiality protection of the inner packet may be handled either entirely by the Geneve Overlay Provider, or partially by the Tenant or handled by both the Tenant and the Geneve Overlay Provider.

The Geneve Header contains information related to the Geneve communications or metadata designated as Geneve Information. Geneve Information is carried on the Geneve Outer Header, the Geneve Header (excluding Geneve Options) as well as in the Geneve Options. Geneve Information needs to be accessed solely by a NVE or transit device while other Geneve Information may need to be accessed by other transit devices. More specifically, a subset of the information contained in the Geneve Header (excluding Geneve Options) as well as a subset of (none, one or multiple Geneve Option) may be accessed by a transit device or the NVE while the others needs to be accessed by other transit devices. The confidentiality protection of the Geneve Information is handled by the Geneve Overlay Provider.

In addition to Geneve Information, the traffic generated for the Geneve overlay may be exposed to traffic volumetry and pattern analysis within a virtualized network. Confidentiality protection against traffic pattern recognition is handled by the Geneve Overlay Provider.

5.1.1. Operational Security Requirements

A secure deployment of a Geneve overlay must fulfill the requirement below:

- SEC-OP-1: A secure deployment of a Geneve overlay SHOULD by default encrypt the inner payload. A Geneve overlay provider MAY disable this capability for example when encryption is performed by the Tenant System and that level of confidentiality is believed to be sufficient. In order to provide additional protection to traffic already encrypted by the Tenant the Geneve network operator MAY partially encrypt the clear part of the inner payload.

- SEC-OP-2: A secure deployment of a Geneve overlay MUST evaluate the information associated to the leakage of Geneve Information carried by the Geneve Packet. When a risk analysis concludes that the risk of leaking sensitive information is too high, such Geneve Information MUST NOT be transmit in clear text.

- SEC-OP-3: A secure deployment of a Geneve overlay MUST evaluate the risk associated to traffic pattern recognition. When a risk has been identified, traffic pattern recognition MUST be addressed with padding policies as well as generation of dummy packets.

5.1.2. Geneve Security Requirements

A Geneve security mechanism must fulfill the requirements below:

- SEC-GEN-1: Geneve security mechanism MUST provide the capability to encrypt the inner payload.

- SEC-GEN-2: Geneve security mechanism SHOULD provide the capability to partially encrypt the inner payload header.

- SEC-GEN-3: Geneve security mechanism MUST provide means to encrypt a single or a set of zero, one or multiple Geneve Options while leave other Geneve Options in clear. Reversely, a Geneve security mechanism MUST be able to leave a Geneve option in clear, while encrypting the others.

- SEC-GEN-4: Geneve security mechanism MUST provide means to encrypt the information of Geneve Header (excluding Geneve Options). Reversely, a Geneve security mechanism MUST be able to leave in clear Geneve Header information (Geneve Options excluded) while encrypting the other.

- SEC-GEN-5: Geneve security mechanisms MUST provide the ability to provide confidentiality protection between multiple nodes, i.e. multiple transit devices and a NVE.

- SEC-GEN-6: Geneve security mechanism MUST provide the ability to pad a Geneve packet.

- SEC-GEN-7: Geneve security mechanism MUST provide the ability to send dummy packets.

5.2. Protecting Against Traffic Injection

Traffic injection from a rogue non legitimate NVO3 Geneve overlay device or a rogue underlay transit device can target an NVE, a transit underlay device or a Tenant System. Targeting a Tenant's System requires a valid MAC and IP addresses of the Tenant's System.

When traffic between tenants is not protected, the rogue device may forward the modified packet over a valid (authenticated) Geneve Header. The crafted packet may for example, include a specifically crafted application payload for a specific Tenant Systems

application, with the intention to load the tenant specific application. Tenant's System may provide integrity protection of the inner payload by protect their communications using for example IPsec/ESP, IPsec/AH [RFC4302], TLS or DTLS. Such protection protects at various layers the Tenants from receiving spoofed packets, as any injected packet is expected to be discarded by the destination Tenant's System. Note IPsec/ESP with NULL encryption may be used to authenticate-only the layers above IP in which case the IP header remains unprotected. However IPsec/AH enables the protection of the entire IP packet, including the IP header. As a result, when Geneve encapsulates IP packets the Tenant has the ability to integrity protect the IP packet on its own, without relying on the Geneve overlay network. On the other hand, L2 layers remains unprotected. As encryption is using authenticated encryption, authentication may also be provided via encryption. At the time of writing the document DTLS 1.3 [I-D.ietf-tls-dtls13] is still a draft document and TLS 1.3 does not yet provide the ability for authenticate only the traffic. As such it is likely that the use of DTLS1.3 may not involve authentication-only cipher suites. Similarly to confidentiality protection, integrity protection may be handled either entirely by the Geneve Overlay Provider, or partially by the Tenant or handled by both the Tenant and the Geneve Overlay Provider.

In addition to confidentiality protection of the inner payload, integrity protection also prevents the Tenant System from receiving illegitimate packets that may disrupt the Tenant's System performance. The Geneve overlay network need to prevent the overlay to be used as a vector to spoof packets being steered to the Tenant's system. As a result, the Overlay Network Provider needs to ensure that inner packets steered to the Tenant's network are only originating from one Tenant System and not from an outsider using the Geneve Overlay to inject packets to one virtual network. As such, the destination NVE MUST be able to authenticate the incoming Geneve packets from the source NVE. This may be performed by the NVE authenticating the full Geneve Packet. When the Geneve Overlay wants to take advantage of the authentication performed by the Tenant System, the NVE shoudl be able to perform some checks between the Geneve Header and the inner payload. Suppose two Geneve packets are composed of a Geneve Header (H1, and H2) and a inner payload (P1 and P2). Suppose H1, H2, P1 and P2 are authenticated. The replacement of P2 by P1 by an attacker will be detected by the NVE only if there is a binding between H2 and P2. Such integrity protection is handled by the Geneve Overlay Provider.

While traffic injection may target the Tenant's virtual network or a specific Tenant System, traffic injection may also target the Geneve Overlay Network by injecting Geneve Options that will affect the processing of the Geneve Packet. Updating the Geneve header and

option parameters such as setting an OAM bit, adding bogus option TLVs, or setting a critical bit, may result in different processing behavior, that could greatly impact performance of the overlay network and the underlay infrastructure and thus affect the tenants traffic delivery. As such, the Geneve Overlay should provide integrity protection of the Geneve Information present in the Geneve Header to guarantee Geneve processing is not altered.

The Geneve architecture considers transit devices that may process some Geneve Options. More specifically, a Geneve packet may have A subset of Geneve Information of the Geneve Header (excluding Geneve Options) as well as a set of zero, one or multiple of Geneve Options accessed by one or more transit devices. This information needs to be authenticated by a transit device while other options may be authenticated by other transit devices or the tunnel endpoint. The integrity protection is handled by the Geneve Overlay Provider and authentication MUST be performed prior any processing.

5.2.1. Operational Security Requirements

A secure deployment of a Geneve overlay must fulfill the requirement below:

- SEC-OP-4: A secure deployment of a Geneve overlay MUST provide the capability authenticate the inner payload when encryption is not provided. A Geneve overlay provider MAY disable this capability for example when this is performed by the Tenant System and that level of integrity is believed to be sufficient. In order to provide additional protection to traffic already protected by the Tenant the Geneve network operator MAY partially protect the unprotected part of the inner payload.

- SEC-OP-5: A secure deployment of a Geneve overlay MUST evaluate the risk associated to a change of the Geneve Outer Header, Geneve Header (excluding Geneve Options) and Geneve Option. When a risk analysis concludes that the risk is too high, this piece of information MUST be authenticated.

- SEC-OP-6: A secure deployment of a Geneve overlay SHOULD authenticate communications between NVE to protect the Geneve Overlay infrastructure as well as the Tenants System's communications (Geneve Packet). A Geneve overlay provider MAY disable authentication of the inner packet and delegates it to the Tenant Systems when communications between Tenant's System is secured. This is NOT RECOMMENDED. Instead, it is RECOMMENDED that mechanisms binds the inner payload to the Geneve Header. To prevent injection between virtualized network, it is strongly

RECOMMENDED that at least the Geneve Header without Geneve Options is authenticated.

- SEC-OP-7: A secure deployment of a Geneve overlay SHOULD NOT process data prior authentication. If that is not possible, the Geneve overlay provider SHOULD evaluate its impact.

5.2.2. Geneve Security Requirements

A Geneve security mechanism must fulfill the requirements below:

- SEC-GEN-8: Geneve security mechanism MUST provide the capability to authenticate the inner payload.

- SEC-GEN-9: Geneve security mechanism SHOULD provide the capability to partially authenticate the inner payload header.

- SEC-GEN-10: Geneve security mechanism MUST provide the capability to authenticate a single or a set of options while leave other Geneve Option unauthenticated. Reversely, a Geneve security mechanism MUST be able to leave a Geneve option unauthenticated, while encrypting the others.

- SEC-GEN-11: Geneve security mechanism MUST provide means to authenticate the information of Geneve Header (Geneve Option excluded). Reversely, a Geneve security mechanism MUST be able to leave unauthenticated Geneve header information (Geneve Options excluded) while authenticating the other.

- SEC-GEN-12: Geneve Security mechanism MUST provide means for a tunnel endpoint (NVE) to authenticate data prior it is being processed.

- SEC-GEN-13: Geneve Security mechanism MUST provide means for a transit device to authenticate data prior it is being processed.

5.3. Protecting Against Traffic Redirection

A rogue device of the NVO3 overlay Geneve network or the underlay network may redirect the traffic from a virtual network to the attacker for passive or active attacks. If the rogue device is in charge of securing the Geneve packet, then Geneve security mechanisms are not intended to address this threat. More specifically, a rogue source NVE will still be able to redirect the traffic in clear text before protecting (and encrypting the packet). A rogue destination NVE will still be able to redirect the traffic in clear text after decrypting the Geneve packets. The same occurs with a rogue transit that is in charge of encrypting and decrypting a Geneve Option,

Geneve Option or any information. The security mechanisms are intended to protect a Geneve information from any on path node. Note that modern cryptography recommend the use of authenticated encryption. This section assumes such algorithms are used, and as such encrypted packets are also authenticated.

To prevent an attacker located in the middle between the NVEs and modifying the tunnel address information in the data packet header to redirect the data traffic, the solution needs to provide confidentiality protection for data traffic exchanged between NVEs.

Requirements are similar as those provided in section Section 5.1 to mitigate sniffing attacks and those provided in section Section 5.2 to mitigate traffic injection attacks.

5.4. Protecting Against Traffic Replay

A rogue device of the NVO3 overlay Geneve network or the underlay network may replay a Geneve packet, to load the network and/or a specific Tenant System with a modified Geneve payload. In some cases, such attacks may target an increase of the tenants costs.

When traffic between Tenant System is not protected against anti-replay. A packet even authenticated can be replayed. DTLS and IPsec provides anti replay mechanisms, so it is unlikely that authenticated Tenant's traffic is subject to replay attacks.

Similarly to integrity protection, the Geneve Overlay Provider should prevent the overlay to be used to replay packet to the Tenant's System. In addition, similarly to integrity protection, the Geneve Overlay network may also be a target of a replay attack, and NVE as well as transit devices should benefit from the same protection.

Given the proximity between authentication and anti-replay mechanisms and that most authentication mechanisms integrates anti-replay attacks, we RECOMMEND that authentication contains an anti-replay mechanisms.

5.4.1. Geneve Security Requirements

A secure deployment of a Geneve overlay must fulfill the requirement below:

- SEC-OP-8: A secure deployment of a Geneve overlay MUST evaluate the communications subject to replay attacks. Communications that are subject to this attacks MUST be authenticated with an anti replay mechanism. Note that when partial authentication is provided, the part not covered by the authentication remains a

surface of attack. It is strongly RECOMMENDED that the Geneve Header is authenticated with anti replay protection.

5.4.2. Geneve Security Requirements

A Geneve security mechanism must fulfill the requirements below:

- SEC-GEN-14: Geneve Security mechanism MUST provide authentication with anti-replay protection.

5.5. Security Management

5.5.1. Operational Security Requirements

A secure deployment of a Geneve overlay must fulfill the requirement below:

- SEC-OP-9: A secure deployment of a Geneve overlay MUST define the security policies that associates the encryption, and authentication associated to each flow between NVEs.

- SEC-OP-10: A secure deployment of a Geneve overlay SHOULD define distinct material for each flow. The cryptographic depends on the nature of the flow (multicast, unicast) as well as on the security mechanism enabled to protect the flow.

5.5.2. Geneve Security Requirements

A Geneve security mechanism must fulfill the requirements below:

- SEC-GEN-15: A Geneve security mechanism MUST be managed via security policies associated for each traffic flow to be protected. Geneve overlay provider MUST be able to configure NVEs with different security policies for different flows. A flow MUST be identified at minimum by the Geneve virtual network identifier and the inner IP and transport headers, and optionally additional fields which define a flow (e.g., inner IP DSCP, IPv6 flow id, Geneve options).

- SEC-GEN-16: A Geneve security mechanism MUST be able to assign different cryptographic keys to protect the unicast tunnels between NVEs respectively.

- SEC-GEN-17: A Geneve security mechanisms, when multicast is used, packets,MUST be able to assign distinct cryptographic group keys to protect the multicast packets exchanged among the NVEs within different multicast groups. Upon receiving a data packet, an egress Geneve NVE MUST be able to verify whether the packet is

sent from a proper ingress NVE which is authorized to forward that packet.

6. IANA Considerations

There are no IANA consideration for this document.

7. Security Considerations

The whole document is about security.

Limiting the coverage of the authentication / encryption provides some means for an attack to craft special packets.

The current document details security requirements that are related to the Geneve protocol. Instead, [I-D.ietf-nvo3-security-requirements] provides generic architecture security requirement upon the deployment of an NVO3 overlay network. It is strongly recommended to read that document as architecture requirements also apply here. In addition, architecture security requirements go beyond the scope of Geneve communications, and as such are more likely to address the security needs upon deploying an Geneve overlay network.

8. Appendix

8.1. DTLS

This section compares how NVE communications using DTLS meet the security requirements for a secure Geneve overlay deployment. In this example DTLS is used over the Geneve Outer Header and secures the Geneve Header including the Geneve Options and the inner payload.

The use of DTLS MAY fill the security requirements for a secure Geneve deployment. However DTLS cannot be considered as the Geneve security mechanism enabling all Geneve deployments. To ease the reading of the Requirements met by DTLS or IPsec, the requirements list indicates with Y (Yes) when the requirement and N (No) when the requirement is not met. In addition, an explanation is provided on the reasoning. This section is not normative and its purpose is limited to illustrative purpose.

8.1.1. Operational Security Requirements

This section shows how DTLS may secure some Geneve deployments. Some Geneve deployments may not be secured by DTLS, but that does not exclude DTLS from being used.

- SEC-OP-1 (Y): A deployment using DTLS between NVEs with an non NULL encryption cipher suite will provide confidentiality to the full Geneve Packet which contains the inner payload. As such the use of DTLS meets SEC-OP-1. Note that DTLS does not provide partial encryption and as such the Geneve Overlay Provider may not benefit from the encryption performed by the Tenant if performed, which may result in some portion of the payload being encrypted twice.

- SEC-OP-2 (Y): A deployment using DTLS between NVEs with an non NULL encryption cipher suite encrypt the Geneve Packet which includes the Geneve Header and associated metadata. Only the UDP port is leaked which could be acceptable. As such, the use of DTLS meets SEC-OP-2.

- SEC-OP-3 (Y/N): A deployment using DTLS between NVEs will not be able to send dummy packets or pad Geneve Packet unless this is managed by the Geneve packet itself. DTLS does not provide the ability to send dummy traffic, nor to pad. As a result DTLS itself does not meet this requirement. This requirement may be met if handled by the Geneve protocol. As such SEC-OP-3 may not be met for some the deployment. However, it is not a mandatory requirement and as such it is likely that the use of DTLS SEC-OP-3 is met.

- SEC-OP-4 (Y): Similarly to SEC-OP-1, A deployment using DTLS between NVEs provides integrity protection to the full Geneve Packet which includes the inner payload. As such the use of DTLS meets SEC-OP-4. Note that DTLS 1.2 provides integrity-only cipher suites while DTLS 1.3 does not yet. As a result, the use of DTLS 1.3 may provide integrity protection using authenticated encryption.

- SEC-OP-5 (Y): Similarly to SEC-OP-2, A deployment using DTLS between NVE authenticates the full Geneve Packet which includes the Geneve Header. Only the UDP port is left unauthenticated. As such, the use of DTLS meets SEC-OP-5.

- SEC-OP-6 (Y): A deployment using DTLS between NVE authenticates NVE-to-NVE communications and the use of DTLS meets SEC-OP-6.

- SEC-OP-7 (Y/N): A deployment using DTLS between NVEs is not compatible with a Geneve architecture that includes transit devices. When the DTLS session uses a non NULL encryption cipher suite, the transit device will not be able to access it. When the NULL encryption cipher suite is used, the transit device may be able to access the data, but will not be able to authenticate it prior to processing the packet. As such the use of DTLS only

meets SEC-OP-7 for deployment that do not include any transit devices.

- SEC-OP-8 (Y): A deployment using DTLS between NVEs provides anti-replay protection and so, the use of DTLS meets SEC-OP-8.

- SEC-OP-9 (Y/N): DTLS does not define any policies. Instead DTLS process is bound to an UDP socket. As such handling of flow policies is handled outside the scope of DTLS. As such SEC-OP-9 is met outside the scope of DTLS.

- SEC-OP-10 (N): DTLS session may be established with specific material, as such it is possible to assign different material for each flow. However, the binding between flow and session is performed outside the scope of DTLS. In addition, DTLS does not support multicast. As such, the use of DTLS may only meets SEC-OP-10 in the case of unicast communications.

8.1.2. Geneve Security Requirements

This section shows that DTLS cannot be used as a generic Geneve security mechanism to secure Geneve deployments. A Geneve security mechanism woudl need to meet all SEC-GEN requirements.

- SEC-GEN-1 (Y): A deployment using DTLS between NVEs with an non NULL encryption cipher suite will provide confidentiality to the full Geneve Packet which contains the inner payload. As such the use of DTLS meets SEC-GEN-1.

- SEC-GEN-2 (Y): A deployment using DTLS between NVEs with an non NULL encryption cipher suite will not be able to partially encrypt the inner payload header. However such requirement is not set a mandatory so the use of DTLS meets SEC-GEN-2

- SEC-GEN-3 (N): A deployment using DTLS between NVEs with an non NULL encryption cipher suite encrypt the Geneve Packet which includes the Geneve Header and all Geneve Options. However DTLS does not provides any means to selectively encrypt or leave in clear text a subset of Geneve Options. As a result the use of DTLS does not meet SEC-GEN-3.

- SEC-GEN-4 (N): A deployment using DTLS between NVEs with an non NULL encryption cipher suite encrypt the Geneve Packet which includes the Geneve Header and all Geneve Options. However, DTLS does not provides means to selectively encrypt some information of the Geneve Header. As such the use of DTLS does not meet SEC-GEN-5.

- SEC-GEN-5 (N): A deployment using DTLS between NVEs with an non NULL encryption cipher suite provides end-to-end security between the NVEs and as such does not permit the interaction of one or multiple on-path transit devices. As such the use of DTLS does not meet SEC-GEN-5.

- SEC-GEN-6 (N): A deployment using DTLS between NVEs with an non NULL encryption cipher suite does not provide padding facilities. This requirements is not met by DTLS itself and needs to be handled by Geneve and specific options. As a result, the use of DTLS does not meet SEC-GEN-6

- SEC-GEN-7 (N): A deployment using DTLS between NVEs with an non NULL encryption cipher suite does not provide the ability to send dummy packets. This requirements is not met by DTLS itself and needs to be handled by Geneve and specific options. As a result, the use of DTLS does not meet SEC-GEN-7.

- SEC-GEN-8 (Y): A deployment using DTLS between NVEs with an non NULL encryption cipher suite or a NULL encryption cipher suite provide authentication of the inner payload. As such the use of DTLS meets SEC-GEN-8.

- SEC-GEN-9 (Y): A deployment using DTLS between NVEs does not provide the ability to partially authenticate the inner payload header. However such requirement is not set a mandatory so the use of DTLS meets SEC-GEN-9

- SEC-GEN-10 (N): A deployment using DTLS between NVEs authenticates the Geneve Packet which includes the Geneve Header and all Geneve Options. However, DTLS does not provides means to selectively encrypt some information of the Geneve Header. As such the use of DTLS meets SEC-GEN-10.

- SEC-GEN-11 (N): A deployment using DTLS between NVEs authenticates the Geneve Packet which includes the Geneve Header and all Geneve Options. However, DTLS does not provides means to selectively authenticate some information of the Geneve Header. As such the use of DTLS does not meet SEC-GEN-11.

- SEC-GEN-12 (Y): A deployment using DTLS between NVEs authenticates the data prior the data is processed by the NVE. As such, the use of DTLS meets SEC-GEN-12.

- SEC-GEN-13 (N): A deployment using DTLS between NVEs authenticates the data when the tunnel reaches the NVE. As a result the transit device is not able to authenticate the data prior accessing it and the use of DTLS does not meet SEC-GEN-13.

- SEC-GEN-14 (Y): DTLS provides anti-replay mechanism as such, the use of DTLS meets SEC-GEN-14.

- SEC-GEN-15 (N): DTLS itself does not have a policy base mechanism. As a result, the classification of the flows needs to be handled by a module outside DTLS. In order to meet SEC-GEN-15 further integration is needed and DTLS in itself cannot be considered as meeting SEC-GEN-15.

- SEC-GEN-16 (Y): DTLS is able to assign various material to each flows, as such the use of DTLS meets SEC-GEN-16.

- SEC-GEN-17 (N): DTLS does not handle mutlicast communications. As such the use of DTLS does not meet SEC-GEN-17.

8.2. IPsec

This section compares how NVE communications using IPsec/ESP or IPsec/AH meet the security requirements for a secure Geneve overlay deployment. In this example secures the Geneve IP packet including Outer IP header, the Geneve Outer Header, the Geneve Header including Geneve Options and the inner payload.

The use of IPsec/ESP or IPsec/AH share most of the analysis performed for DTLS. The main advantages of using IPsec would be that IPsec supports multicast communications and natively supports flow based security policies. However, the use of these security policies in a context of Geneve is not natively supported.

As a result, the use of IPsec MAY fill the security requirements for a secure Geneve deployment. However IPsec cannot be considered as the Geneve security mechanism enabling all Geneve deployments.

8.2.1. Operational Security Requirements

This section shows how IPsec may secure some Geneve deployments. Some Geneve deployments may not be secured by IPsec, but that does not exclude IPsec from being used.

- SEC-OP-1 (Y): A deployment using IPsec/ESP between NVEs with an non NULL encryption will provide confidentiality to the full Outer IP payload of the Geneve Packet which contains the inner payload. As a result, such deployments meet SEC-OP-1. Note that IPsec/ESP does not provide partial encryption and as such the Geneve Overlay Provider may not benefit from the encryption performed by the Tenant if performed, which may result in some portion of the payload being encrypted twice.

- SEC-OP-2 (Y): A deployment using IPsec/ESP between NVEs with an non NULL encryption encrypts the Outer IP payload Geneve IP Packet which includes the Geneve Header and associated information. As such SEC-OP-2 is met.

- SEC-OP-3 (Y): A deployment using IPsec/ESP between NVEs will be able to send dummy packets or pad Geneve Packet. As such OP-SEC-3 is met.

- SEC-OP-4 (Y): Similarly to SEC-OP-1, A deployment using IPsec/ESP or IPsec/AH between NVEs provides integrity protection to the full Geneve Packet which includes the inner payload. As such SEC-OP-4 is met.

- SEC-OP-5 (Y): Similarly to SEC-OP-2, A deployment using IPsec/ESP or IPsec/AH between NVE authenticates the full Geneve Packet which includes the Geneve Header. As such SEC-OP-5 is met as well.

- SEC-OP-6 (Y): A deployment using IPsec/ESP or IPsec/AH between NVE authenticates NVE-to-NVE communications and SEC-OP-6 is met.

- SEC-OP-7 (Y/N): A deployment using IPsec between NVEs is not compatible with a Geneve architecture that includes transit devices. When IPsec/ESP with a non NULL encryption is used, the transit device will not be able to access it. When IPsec/AH or IPsec/ESP with the NULL encryption is used, the transit device may be able to access the data, but will not be able to authenticate it prior to processing the packet. As SEC-OP-7 is only met for deployment that do not include any transit devices.

- SEC-OP-8 (Y): A deployment using IPsec between NVEs provides anti-replay protection and so meets SEC-OP-8.

- SEC-OP-9 (Y/N): IPsec enables the definition of security policies. As such IPsec is likely to handle a per flow security. However the traffic selector required for Geneve flows may not be provided natively by IPsec. As such Sec-OP-9 is only partialy met.

- SEC-OP-10 (Y): IPsec session may be established with specific material, as such it is possible to assign different material for each flow. In addition IPsec supports multicats communications. As such SEC-OP-10 is met.

8.2.2. Geneve Security Requirements

This section shows that IPsec cannot be used as a generic Geneve security mechanism to secure Geneve deployments. A Geneve security mechanism would need to meet all SEC-GEN requirements.

- SEC-GEN-1 (Y): A deployment using IPsec/ESP between NVEs with an non NULL encryption provide confidentiality to the full Geneve Packet which contains the inner payload. As such IPsec/ESP meets SEC-GEN-1.

- SEC-GEN-2 (Y): A deployment using IPsec/ESP between NVEs with an non NULL encryption will not be able to partially encrypt the inner payload header. However such requirement is not set a mandatory so IPsec/ESP meets SEC-GEN-2

- SEC-GEN-3 (N): A deployment using IPsec between NVEs with an non NULL encryption encrypts the Outer IP payload of the Geneve Packet which includes the Geneve Header and all Geneve Options. However IPsec/ESP does not provides any means to selectively encrypt or leave in clear text a subset of Geneve Options. As a result SEC-GEN-3 is not met.

- SEC-GEN-4 (N): A deployment using IPsec/ESP between NVEs with an non NULL encryption encrypts the Geneve Packet which includes the Geneve Header and all Geneve Options. However, IPsec/ESP does not provides means to selectively encrypt some information of the Geneve Header. As such SEC-GEN-5 is not met.

- SEC-GEN-5 (N): A deployment using IPsec between NVEs with an non NULL encryption provides end-to-end security between the NVEs and as such does not permit the interaction of one or multiple on-path transit devices. As such IPsec/ESP does not meet SEC-GEN-5.

- SEC-GEN-6 (Y): A deployment using IPsec/ESP between NVEs with an non NULL encryption provides padding facilities and as such IPsec/ESP meets SEC-GEN-6.

- SEC-GEN-7 (Y): A deployment using IPsec between NVEs with an non NULL encryption cipher provides the ability to send dummy packets. As such IPsec/ESP meets SEC-GEN-7.

- SEC-GEN-8 (Y): A deployment using IPsec/ESP or IPsec/AH authenticates the inner payload. As such SEC-GEN-8 is met.

- SEC-GEN-9 (Y): A deployment using IPsec/AH or IPsec/ESP between NVEs does not provide the ability to partially authenticate the inner payload header. However such requirement is not set a mandatory so IPsec meets SEC-GEN-9

- SEC-GEN-10 (N): A deployment using IPsec/ESP or IPsec/AH between NVEs authenticates the Geneve Packet which includes the Geneve Header and all Geneve Options. However, IPsec does not provides

means to selectively encrypt some information of the Geneve Header. As such SEC-GEN-10 is not met.

- SEC-GEN-11 (N): A deployment using IPsec/ESP or IPsec/AH between NVEs authenticates the Geneve Packet which includes the Geneve Header and all Geneve Options. However, IPsec does not provides means to selectively authenticate some information of the Geneve Header. As such SEC-GEN-11 is not met.

- SEC-GEN-12 (Y): A deployment using IPsec/ESP or IPsec/AH between NVEs authenticates the data prior the data is processed by the NVE. As such SEC-GEN-12 is met.

- SEC-GEN-13 (N): A deployment using IPsec/ESP or IPsec/AH between NVEs authenticates the data when the tunnel reaches the NVE. As a result the transit device is not able to authenticate the data prior accessing it and SEC-GEN-13 is not met.

- SEC-GEN-14 (Y): IPsec/ESP and IPsec/AH provides anti-replay mechanism as such SEC-GEN-14 is met.

- SEC-GEN-15 (N): IPsec is a policy base architecture. As a result, the classification of the flows needs to be handled by IPsec. However, the traffic selector available are probably not those required by Geneve and further integration is needed. As such SEC-GEN-15 is not met.

- SEC-GEN-16 (Y): IPsec is able to assign various material to each flows, as such SEC-GEN-16 is met.

- SEC-GEN-17 (Y): IPsec handles mutlicast communications. As such SEC-GEN-17 is met.

9. Acknowledgments

We would like to thank Ilango S Ganaga, Magnus Nystroem for their useful reviews and clarifications as well as Matthew Bocci, Sam Aldrin and Ignas Bagdona for moving the work forward.

10. References

10.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, DOI 10.17487/RFC4301, December 2005, <https://www.rfc-editor.org/info/rfc4301>.

[RFC4302] Kent, S., "IP Authentication Header", RFC 4302, DOI 10.17487/RFC4302, December 2005, <https://www.rfc-editor.org/info/rfc4302>.

[RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, DOI 10.17487/RFC4303, December 2005, <https://www.rfc-editor.org/info/rfc4303>.

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012, <https://www.rfc-editor.org/info/rfc6347>.

[RFC7258] Farrell, S. and H. Tschofenig, "Pervasive Monitoring Is an Attack", BCP 188, RFC 7258, DOI 10.17487/RFC7258, May 2014, <https://www.rfc-editor.org/info/rfc7258>.

[RFC7365] Lasserre, M., Balus, F., Morin, T., Bitar, N., and Y. Rekhter, "Framework for Data Center (DC) Network Virtualization", RFC 7365, DOI 10.17487/RFC7365, October 2014, <https://www.rfc-editor.org/info/rfc7365>.

[RFC7516] Jones, M. and J. Hildebrand, "JSON Web Encryption (JWE)", RFC 7516, DOI 10.17487/RFC7516, May 2015, <https://www.rfc-editor.org/info/rfc7516>.

[RFC7525] Sheffer, Y., Holz, R., and P. Saint-Andre, "Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", BCP 195, RFC 7525, DOI 10.17487/RFC7525, May 2015, <https://www.rfc-editor.org/info/rfc7525>.

[RFC8014] Black, D., Hudson, J., Kreeger, L., Lasserre, M., and T. Narten, "An Architecture for Data-Center Network Virtualization over Layer 3 (NVO3)", RFC 8014, DOI 10.17487/RFC8014, December 2016, <https://www.rfc-editor.org/info/rfc8014>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8221] Wouters, P., Migault, D., Mattsson, J., Nir, Y., and T. Kivinen, "Cryptographic Algorithm Implementation Requirements and Usage Guidance for Encapsulating Security Payload (ESP) and Authentication Header (AH)", RFC 8221, DOI 10.17487/RFC8221, October 2017, <https://www.rfc-editor.org/info/rfc8221>.

[RFC8247] Nir, Y., Kivinen, T., Wouters, P., and D. Migault, "Algorithm Implementation Requirements and Usage Guidance for the Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 8247, DOI 10.17487/RFC8247, September 2017, <https://www.rfc-editor.org/info/rfc8247>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

10.2. Informative References

[I-D.ietf-nvo3-geneve]
Gross, J., Ganga, I., and T. Sridhar, "Geneve: Generic Network Virtualization Encapsulation", draft-ietf-nvo3-geneve-09 (work in progress), February 2019.

[I-D.ietf-nvo3-security-requirements]
Hartman, S., Zhang, D., Wasserman, M., Qiang, Z., and M. Zhang, "Security Requirements of NVO3", draft-ietf-nvo3-security-requirements-07 (work in progress), June 2016.

[I-D.ietf-tls-dtls13]
Rescorla, E., Tschofenig, H., and N. Modadugu, "The Datagram Transport Layer Security (DTLS) Protocol Version 1.3", draft-ietf-tls-dtls13-30 (work in progress), November 2018.

Authors' Addresses

Daniel Migault
Ericsson
8275 Trans Canada Route
Saint Laurent, QC 4S 0B6
Canada

EMail: daniel.migault@ericsson.com

Sami Boutros
VMware, Inc.

EMail: boutros@vmware.com<

Dan Wings
VMware, Inc.

EMail: dwing@vmware.com

Suresh Krishnan
Kaloom

EMail: suresh@kaloom.com

ippm
Internet-Draft
Intended status: Standards Track
Expires: April 25, 2019

B. Weis
F. Brockners
C. Hill
S. Bhandari
V. Govindan
C. Pignataro
Cisco
H. Gredler
RtBrick Inc.
J. Leddy
Comcast
S. Youell
JMPC
T. Mizrahi
Marvell
A. Kfir
B. Gafni
Mellanox Technologies, Inc.
P. Lapukhov
Facebook
M. Spiegel
Barefoot Networks

October 22, 2018

Ethernet Encapsulation for In-situ OAM Data
draft-weis-ippm-ioam-eth-00

Abstract

In-situ Operations, Administration, and Maintenance (IOAM) records operational and telemetry information in the packet while the packet traverses a path between two points in the network. This document outlines how encapsulations using an EtherType to identify IOAM data fields as the next header in a packet.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any

time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 25, 2019.

Table of Contents

1. Introduction

In-situ Operations, Administration, and Maintenance (IOAM) records operational and telemetry information in the packet while the packet traverses a particular network domain. The term "in-situ" refers to the fact that the IOAM data fields are added to the data packets rather than being sent within packets specifically dedicated to OAM. This document defines how IOAM data fields are carried as part of encapsulations where the IOAM data follows a header that uses an EtherType to denote the next protocol in the packet. Examples of these protocols are GRE [RFC2784] and Geneve [I-D.ietf-nvo3-geneve]).

This document outlines how IOAM data fields are encoded in these protocols.

2. Conventions

2.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2.2. Abbreviations

Abbreviations used in this document:

E2E: Edge-to-Edge

Geneve: Generic Network Virtualization Encapsulation

GRE: Generic Routing Encapsulation

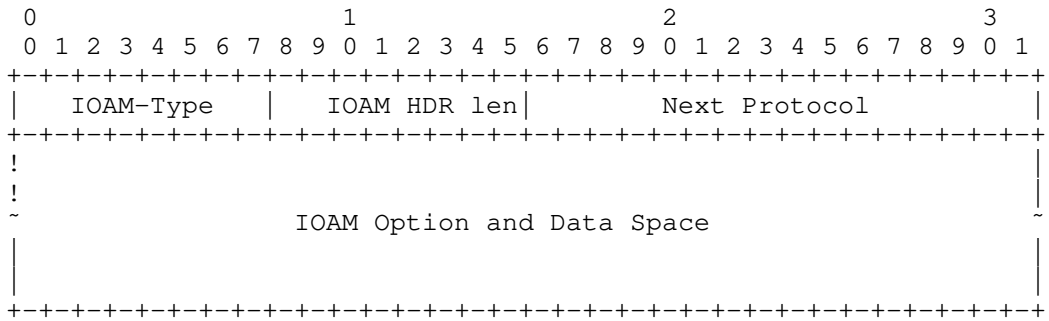
IOAM: In-situ Operations, Administration, and Maintenance

OAM: Operations, Administration, and Maintenance

POT: Proof of Transit

3. IOAM Ethertype

When the IOAM data fields are included within an encapsulation that identifies the next protocol using an EtherType (e.g., GRE or Geneve) the presence of IOAM data fields are identified with TBD_IOAM. When the Ethernet Encapsulation for In-situ OAM Data is used, an additional IOAM header is also included. This header indicates the type of IOAM data that follows, and the next protocol that follows the IOAM data.



The IOAM encapsulation is defined as follows.

IOAM Type: 8-bit field defining the IOAM Option type, as defined in Section 7.2 of [I-D.ietf-ippm-ioam-data].

IOAM HDR Len: 8 bits Length field contains the length of the variable IOAM data octets in 4-octet units.

Next Protocol: 16 bits Next Protocol Type field contains the protocol type of the packet following IOAM protocol header. When the most significant octet is 0x00, the Protocol Type is taken to be an IP Protocol Number as defined in [IP-PROT]. Otherwise, the Protocol Type is defined to be an EtherType value from [ETYPES]. An implementation receiving a packet containing a Protocol Type which is not listed in one of those registries SHOULD discard the packet.

IOAM Option and Data Space: IOAM option header and data is present as specified by the IOAM-Type field, and is defined in Section 4 of [I-D.ietf-ippm-ioam-data].

Multiple IOAM options MAY be included within the IOAM Option and Data Space. For example, if two IOAM options are included, the Next Protocol field of the first IOAM option will contain the value of TBD_IOAM, while the Next Protocol field of the second IOAM option will contain the Ethertype or IP protocol Number indicating the type of the data packet.

4. Usage Examples of the IOAM Ethertype

The Ethernet Encapsulation for In-situ OAM Data can be used with many encapsulations. The following sections show how it can be used with GRE and Geneve.

4.1. Example: GRE Encapsulation of In-situ OAM Date Fields

When IOAM data fields are carried in GRE, the IOAM encapsulation defined above follows the GRE header, as shown in Figure 1.

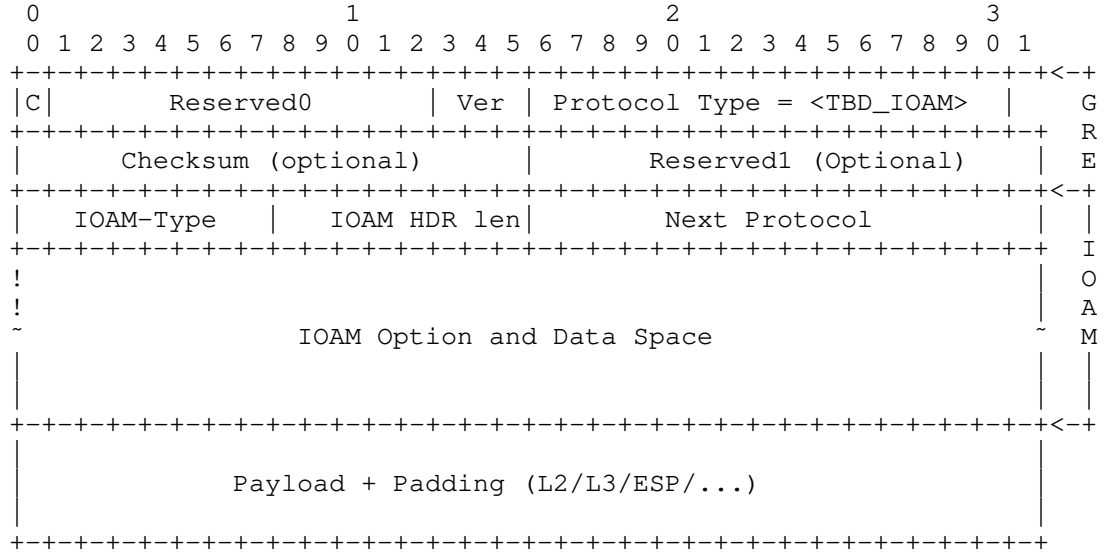


Figure 1: GRE Encapsulation Example

The GRE header and fields are defined in [RFC2784]. The GRE Protocol Type value is TBD_IOAM.

4.2. Example: Geneve Encapsulation of IOAM Data Fields

When IOAM data fields are carried in Geneve, the IOAM encapsulation defined above follows the Geneve header, as shown in Figure 2.

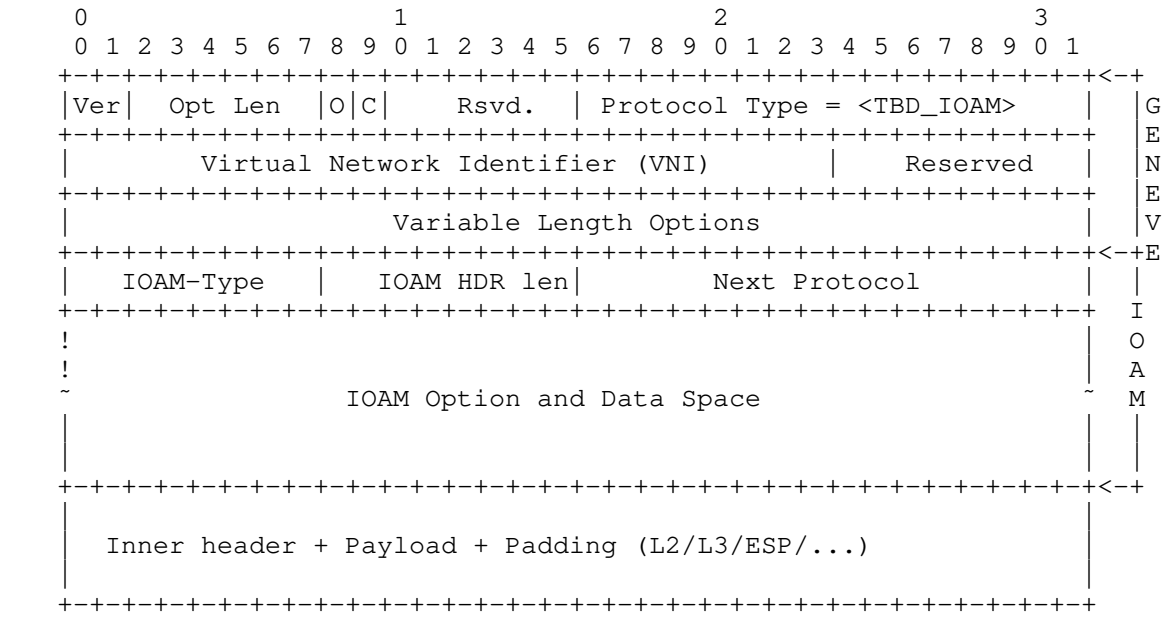


Figure 2: Geneve Encapsulation Example

The GENEVE header and fields are defined in [I-D.ietf-nvo3-geneve]. The Geneve Protocol Type value is TBD_IOAM.

5. Security Considerations

This document describes the encapsulation of IOAM data fields in GRE. Security considerations of the specific IOAM data fields for each case (i.e., Trace, Proof of Transit, and E2E) are described in defined in [I-D.ietf-ippm-ioam-data].

As this document describes new protocol fields within the existing GRE encapsulation, these are similar to the security considerations of [RFC2784].

IOAM data transported in an OAM E2E header SHOULD be integrity protected (e.g., with IPsec ESP [RFC4303]) to detect changes made by a device between the sending and receiving OAM endpoints.

6. IANA Considerations

A new EtherType value is requested to be added to the [ETYPES] IANA registry. The description should be "In-situ OAM (IOAM)".

7. References

7.1. Normative References

[ETYPES] "IANA Ethernet Numbers", <https://www.iana.org/assignments/ieee-802-numbers/ieee-802-numbers.xhtml>.

[I-D.ietf-ippm-ioam-data] Brockners, F., Bhandari, S., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Mozes, D., Lapukhov, P., Chang, R., daniel.bernier@bell.ca, d., and J. Lemon, "Data Fields for In-situ OAM", draft-ietf-ippm-ioam-data-03 (work in progress), June 2018.

[I-D.ietf-nvo3-geneve] Gross, J., Ganga, I., and T. Sridhar, "Geneve: Generic Network Virtualization Encapsulation", draft-ietf-nvo3-geneve-08 (work in progress), October 2018.

[IP-PROT] "IANA Protocol Numbers", <https://www.iana.org/assignments/protocol-numbers/protocol-numbers.xhtml>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC2784] Farinacci, D., Li, T., Hanks, S., Meyer, D., and P. Traina, "Generic Routing Encapsulation (GRE)", RFC 2784, DOI 10.17487/RFC2784, March 2000, <https://www.rfc-editor.org/info/rfc2784>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

7.2. Informative References

[RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, DOI 10.17487/RFC4303, December 2005, <https://www.rfc-editor.org/info/rfc4303>.

Authors' Addresses

Brian Weis
Cisco Systems, Inc.
170 W. Tasman Drive
San Jose, California 95134-1706
USA

Phone: +1-408-526-4796
Email: bew@cisco.com

Frank Brockners
Cisco Systems, Inc.
Hansaallee 249, 3rd Floor
DUESSELDORF, NORDRHEIN-WESTFALEN 40549
Germany

Email: fbrockne@cisco.com

Craig Hill
Cisco Systems, Inc.
13600 Dulles Technology Drive
Herndon, Virginia 20171
United States

Email: crhill@cisco.com

Shwetha Bhandari
Cisco Systems, Inc.
Cessna Business Park, Sarjapura Marathalli Outer Ring Road
Bangalore, KARNATAKA 560 087
India

Email: shwethab@cisco.com

Vengada Prasad Govindan
Cisco Systems, Inc.

Email: venggovi@cisco.com

Carlos Pignataro
Cisco Systems, Inc.
7200-11 Kit Creek Road
Research Triangle Park, NC 27709
United States

Email: cpignata@cisco.com

Hannes Gredler
RtBrick Inc.

Email: hannes@rtbrick.com

John Leddy
Comcast

Email: John_Leddy@cable.comcast.com

Stephen Youell
JP Morgan Chase
25 Bank Street
London E14 5JP
United Kingdom

Email: stephen.youell@jpmorgan.com

Tal Mizrahi
Marvell
6 Hamada St.
Yokneam 20692
Israel

Email: talmi@marvell.com

Aviv Kfir
Mellanox Technologies, Inc.
350 Oakmead Parkway, Suite 100
Sunnyvale, CA 94085
U.S.A.

Email: avivk@mellanox.com

Barak Gafni
Mellanox Technologies, Inc.
350 Oakmead Parkway, Suite 100
Sunnyvale, CA 94085
U.S.A.

Email: gbarak@mellanox.com

Petr Lapukhov
Facebook
1 Hacker Way
Menlo Park, CA 94025
US

Email: petr@fb.com

Mickey Spiegel
Barefoot Networks
2185 Park Boulevard
Palo Alto, CA 94306
US

Email: mspiegel@barefootnetworks.com

ippm
Internet-Draft
Intended status: Standards Track
Expires: October 30, 2020

B. Weis
Independent
F. Brockners
C. Hill
S. Bhandari
V. Govindan
C. Pignataro
Cisco
H. Gredler
RtBrick Inc.
J. Leddy

S. Youell
JMPC
T. Mizrahi
Huawei Network.IO Innovation Lab
A. Kfir
B. Gafni
Mellanox Technologies, Inc.
P. Lapukhov
Facebook
M. Spiegel
Barefoot Networks, an Intel company
May 13, 2020

EtherType Protocol Identification of In-situ OAM Data draft-weis-ippm-ioam-eth-04

Abstract

In-situ Operations, Administration, and Maintenance (IOAM) records operational and telemetry information in the packet while the packet traverses a path between two points in the network. This document defines an EtherType that identifies IOAM data fields as being the next protocol in a packet, and a header that encapsulates the IOAM data fields.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Copyright Notice

Table of Contents

1. Introduction

In-situ Operations, Administration, and Maintenance (IOAM) records operational and telemetry information in the packet while the packet traverses a particular network domain. The term "in-situ" refers to the fact that the IOAM data fields are added to the data packets rather than being sent within packets specifically dedicated to OAM. This document proposes a new Ethertype for IOAM and defines how IOAM

data fields are carried as part of encapsulations where the IOAM data fields follows an encapsulation header that uses an EtherType to denote the next protocol in the packet. Examples of these protocols are GRE [RFC2890] and Geneve [I-D.ietf-nvo3-geneve]). This document outlines how IOAM data fields are encoded in these protocols.

2. Conventions

2.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2.2. Abbreviations

Abbreviations used in this document:

E2E: Edge-to-Edge

Geneve: Generic Network Virtualization Encapsulation

GRE: Generic Routing Encapsulation

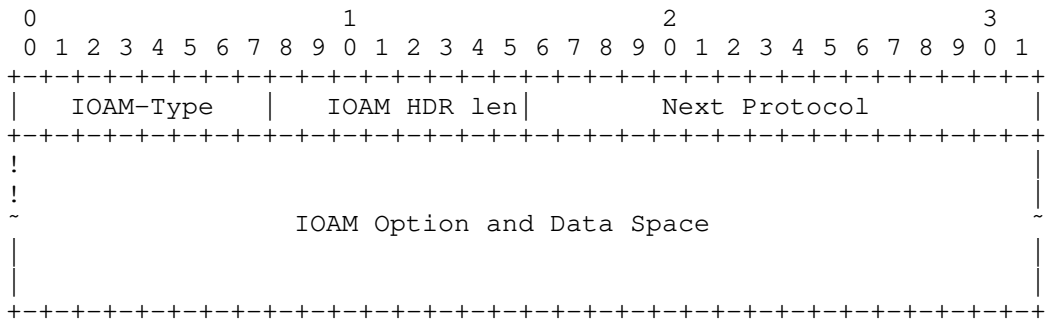
IOAM: In-situ Operations, Administration, and Maintenance

OAM: Operations, Administration, and Maintenance

POT: Proof of Transit

3. IOAM EtherType

When the IOAM data fields are included within an encapsulation that identifies the next protocol using an EtherType (e.g., GRE or Geneve) the presence of IOAM data fields are identified with TBD_IOAM. When this EtherType is used, an additional IOAM header is also included. This header indicates the type of IOAM data fields that follows, and the next protocol that follows the IOAM data fields.



The IOAM encapsulation is defined as follows.

IOAM Type: 8-bit field defining the IOAM Option type, as defined in Section 7.2 of [I-D.ietf-ippm-ioam-data].

IOAM HDR Len: 8 bit Length field contains the length of the IOAM header in 4-octet units.

Next Protocol: 16 bits Next Protocol Type field contains the protocol type of the packet following IOAM protocol header. Protocol Type is defined to be an EtherType value from [ETYPES]. An implementation receiving a packet containing a Protocol Type which is not listed in one of those registries SHOULD discard the packet.

IOAM Option and Data Space: IOAM option header and data is present as specified by the IOAM-Type field, and is defined in Section 4 of [I-D.ietf-ippm-ioam-data].

Multiple IOAM options MAY be included within the IOAM Option and Data Space. For example, if two IOAM options are included, the Next Protocol field of the first IOAM option will contain the value of TBD_IOAM, while the Next Protocol field of the second IOAM option will contain the EtherType indicating the type of the data packet.

4. Usage Examples of the IOAM EtherType

The IOAM EtherType can be used with many encapsulations. The following sections show how it can be used with GRE and Geneve.

4.1. Example: GRE Encapsulation of IOAM Data Fields

When IOAM data fields are carried in GRE, the IOAM encapsulation defined above follows the GRE header, as shown in Figure 1.

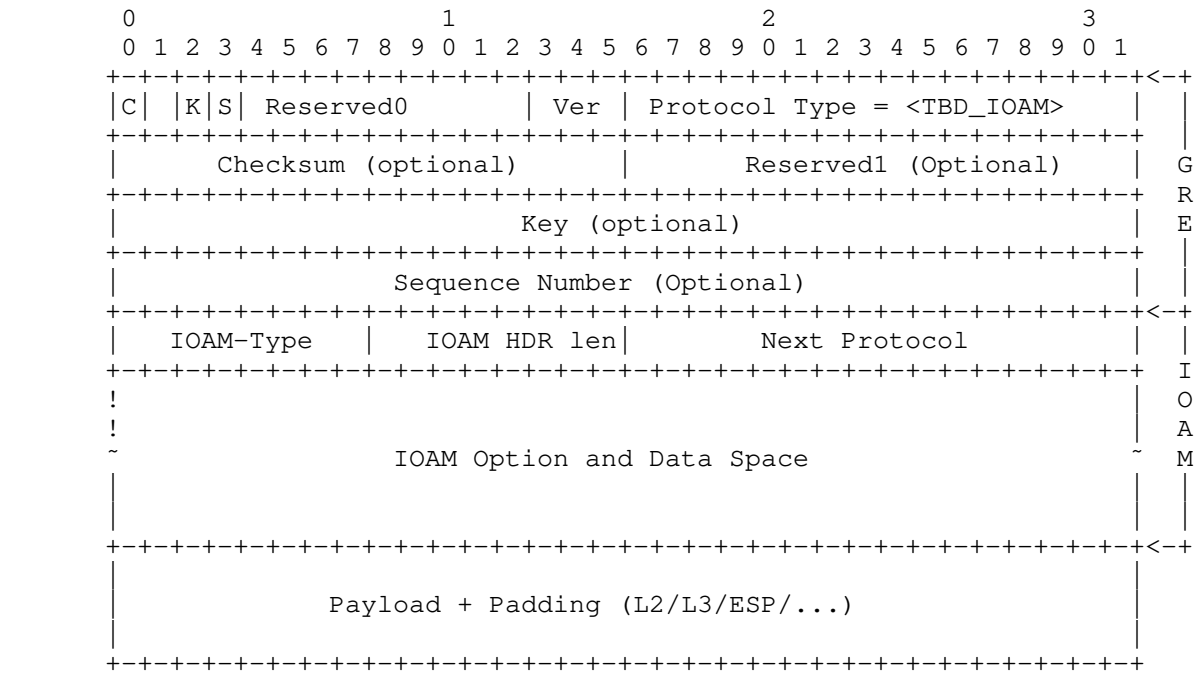


Figure 1: GRE Encapsulation Example

The GRE header and fields are defined in [RFC2890]. The GRE Protocol Type value is TBD_IOAM.

Figure 2 shows two example protocol header stacks that use GRE along with IOAM. IOAM Option-Types (the below diagram uses "IOAM" as shorthand for IOAM Option-Types) are sequenced in behind the GRE header that follows the "outer" IP header.

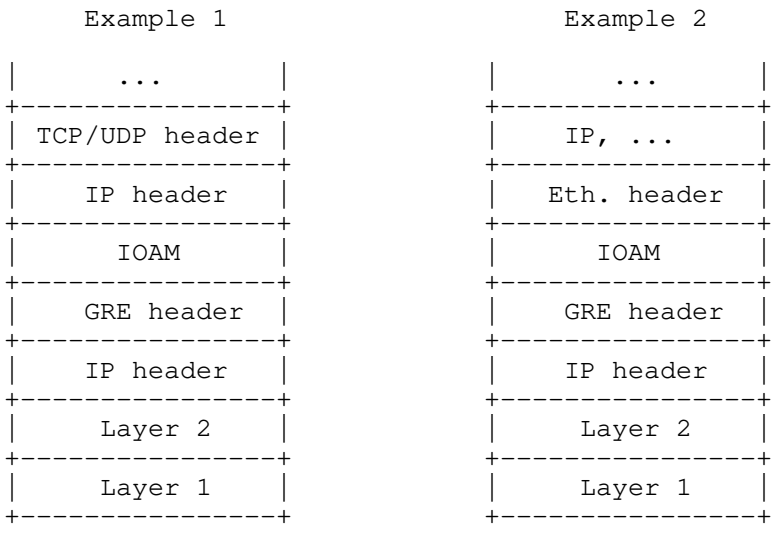


Figure 2: GRE with IOAM examples

4.2. Example: Geneve Encapsulation of IOAM Data Fields

When IOAM data fields are carried in Geneve, the IOAM encapsulation defined above follows the Geneve header, as shown in Figure 3.

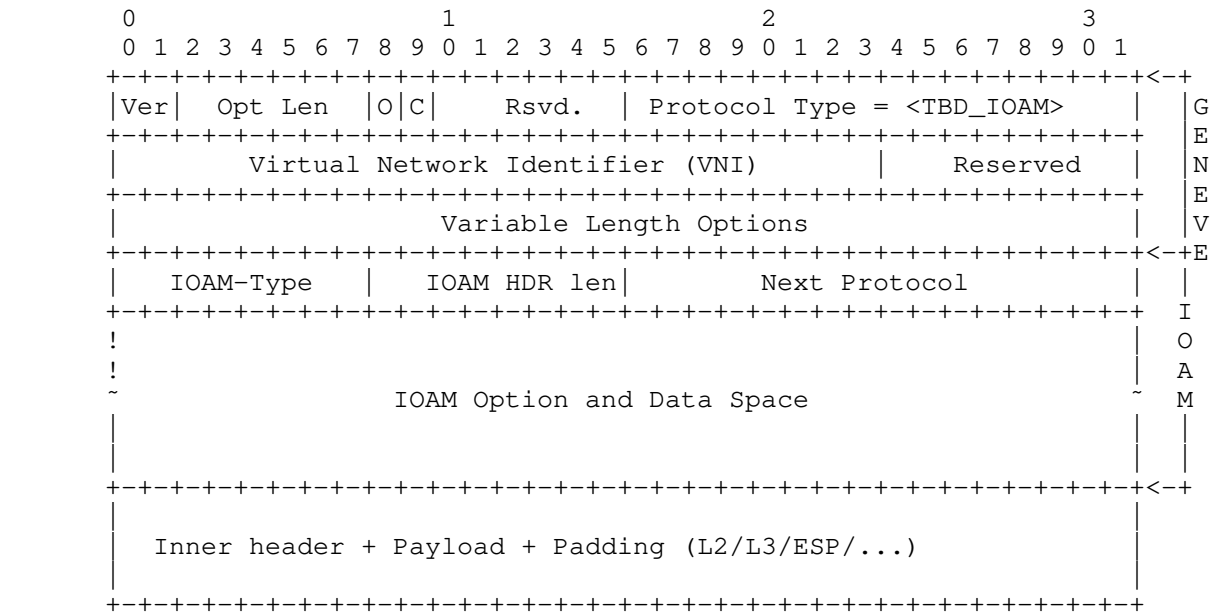


Figure 3: Geneve Encapsulation Example

The GENEVE header and fields are defined in [I-D.ietf-nvo3-geneve]. The Geneve Protocol Type value is TBD_IOAM.

5. Security Considerations

This document describes the encapsulation of IOAM data fields in GRE. Security considerations of the specific IOAM data fields for each case (i.e., Trace, Proof of Transit, and E2E) are described in defined in [I-D.ietf-ippm-ioam-data].

As this document describes new protocol fields within the existing GRE encapsulation, these are similar to the security considerations of [RFC2890].

IOAM data transported in an OAM E2E header SHOULD be integrity protected (e.g., with IPsec ESP [RFC4303]) to detect changes made by a device between the sending and receiving OAM endpoints.

6. IANA Considerations

A new EtherType value is requested to be added to the [ETYPES] IANA registry by IEEE Registration Authority. The description should be "In-situ OAM (IOAM)".

7. Acknowledgements

We would like to thank Nagendra Kumar Nainar for the contribution.

8. References

8.1. Normative References

[ETYPES] "IANA Ethernet Numbers", <https://www.iana.org/assignments/ieee-802-numbers/ieee-802-numbers.xhtml>.

[I-D.ietf-ippm-ioam-data] Brockners, F., Bhandari, S., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Mozes, D., Lapukhov, P., remy@barefootnetworks.com, r., daniel.bernier@bell.ca, d., and J. Lemon, "Data Fields for In-situ OAM", draft-ietf-ippm-ioam-data-09 (work in progress), March 2020.

[I-D.ietf-nvo3-geneve] Gross, J., Ganga, I., and T. Sridhar, "Geneve: Generic Network Virtualization Encapsulation", draft-ietf-nvo3-geneve-16 (work in progress), March 2020.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC2890] Dommety, G., "Key and Sequence Number Extensions to GRE", RFC 2890, DOI 10.17487/RFC2890, September 2000, <https://www.rfc-editor.org/info/rfc2890>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

8.2. Informative References

[RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, DOI 10.17487/RFC4303, December 2005, <https://www.rfc-editor.org/info/rfc4303>.

Authors' Addresses

Brian Weis
Independent
USA

Email: bew.stds@gmail.com

Frank Brockners
Cisco Systems, Inc.
Hansaallee 249, 3rd Floor
DUESSELDORF, NORDRHEIN-WESTFALEN 40549
Germany

Email: fbrockne@cisco.com

Craig Hill
Cisco Systems, Inc.
13600 Dulles Technology Drive
Herndon, Virginia 20171
United States

Email: crhill@cisco.com

Shwetha Bhandari
Cisco Systems, Inc.
Cessna Business Park, Sarjapura Marathalli Outer Ring Road
Bangalore, KARNATAKA 560 087
India

Email: shwethab@cisco.com

Vengada Prasad Govindan
Cisco Systems, Inc.

Email: venggovi@cisco.com

Carlos Pignataro
Cisco Systems, Inc.
7200-11 Kit Creek Road
Research Triangle Park, NC 27709
United States

Email: cpignata@cisco.com

Hannes Gredler
RtBrick Inc.

Email: hannes@rtbrick.com

John Leddy
United States

Email: john@leddy.net

Stephen Youell
JP Morgan Chase
25 Bank Street
London E14 5JP
United Kingdom

Email: stephen.youell@jpmorgan.com

Tal Mizrahi
Huawei Network.IO Innovation Lab
Israel

Email: tal.mizrahi.phd@gmail.com

Aviv Kfir
Mellanox Technologies, Inc.
350 Oakmead Parkway, Suite 100
Sunnyvale, CA 94085
U.S.A.

Email: avivk@mellanox.com

Barak Gafni
Mellanox Technologies, Inc.
350 Oakmead Parkway, Suite 100
Sunnyvale, CA 94085
U.S.A.

Email: gbarak@mellanox.com

Petr Lapukhov
Facebook
1 Hacker Way
Menlo Park, CA 94025
US

Email: petr@fb.com

Mickey Spiegel
Barefoot Networks, an Intel company
4750 Patrick Henry Drive
Santa Clara, CA 95054
US

Email: mickey.spiegel@intel.com