

INTERNET-DRAFT
Intended Status: Standard Track

Sami Boutros(Ed.)
VMware

Dan Wing
Calvin Qian
VMware

Expires: January 1, 2018

June 30, 2017

IPSec over Geneve encapsulation
draft-boutros-nvo3-ipsec-over-geneve-00

Abstract

This document specifies how Generic Network Virtualization Encapsulation (Geneve) can be used to carry IP Encapsulating Security Payload (ESP) and IP Authentication Header (AH) to provide secure transport over IP networks. Using IPSec ESP and AH will provide both Geneve header integrity protection and Geneve payload encryption.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at
<http://www.ietf.org/lid-abstracts.html>

The list of Internet-Draft Shadow Directories can be accessed at
<http://www.ietf.org/shadow.html>

Copyright and License Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Terminology	3
3. Abbreviations	3
4.0 Encapsulation Security Payload (ESP) over Geneve tunnel . .	4
5.0 IP Authentication header (AH) over Geneve tunnel	5
6. Control Plane Considerations	6
7. Acknowledgements	7
8. Security Considerations	7
9. IANA Considerations	7
10. References	7
10.1 Normative References	7
10.2 Informative References	7
Authors' Addresses	8

1. Introduction

The Network Virtualization over Layer 3 (NVO3) develop solutions for network virtualization within a data center (DC) environment that assumes an IP-based underlay. An NVO3 solution provides layer 2 and/or layer 3 overlay services for virtual networks enabling multi-tenancy and workload mobility. The Generic Network Virtualization Encapsulation [GENEVE] have been recently recommended to be the proposed standard for network virtualization overlay encapsulation.

Generic Network Virtualization Encapsulation (Geneve) does not have any inherent security mechanisms. An attacker with access to the underlay network transporting the IP packets has the ability to snoop or inject packets.

Within a particular security domain, such as a data center operated by a single 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. If the remote tunnel endpoint is not completely trusted, for example it resides on a customer premises, then it may also be necessary to sanitize any tunnel metadata to prevent tenant-hopping attacks.

This document describes how Geneve tunnel encapsulation [GENEVE] can be used to carry both the IP Encapsulation security payload (ESP) [RFC4303] and the IP authentication header (AH) [RFC4302] to provide secure transport over IP networks. Using IPsec ESP and AH will provide both Geneve header integrity protection and Geneve payload encryption.

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

IPsec IP Security

ESP IP Encapsulating Security Payload

AH IP Authentication Header

GRE Generic Routing Encapsulation (GRE)

EtherIP Tunneling Ethernet Frames in IP Datagrams

Transit device Underlay network devices between NVE(s).

4.0 Encapsulation Security Payload (ESP) over Geneve tunnel

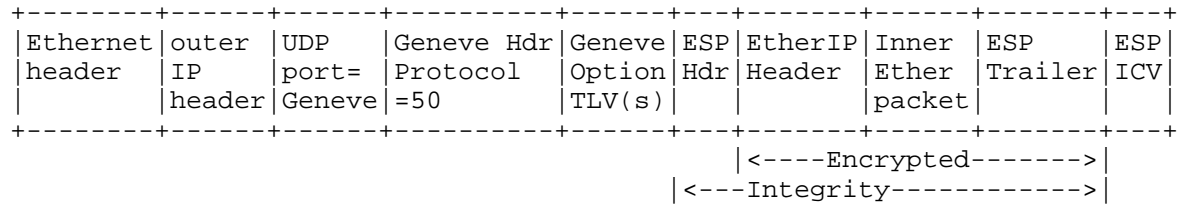
The Geneve packet is encapsulated in UDP over either IPv4 or IPv6. The Geneve packet consists of a Geneve header which is then followed by a set of variable options. The Geneve header protocol type will indicate that the Geneve payload is the ESP protocol (IP Protocol 50), and the Geneve payload will consist of the ESP protocol data. The ESP next header can carry only an IP protocol so can't carry the inner Ethernet frame, so given that (1) Generic Routing Encapsulation (GRE) [RFC2784] and EtherIP [RFC3378] are IP protocols and (2) GRE/EtherIP can carry Ethernet Frames, hence the need of EtherIP/GRE encapsulation for the inner Ethernet payload.

The ESP Next Header field will be set to inner payload protocol which can be either EtherIP (97) or to the Generic Routing Encapsulation (GRE) (47). The GRE protocol type will be set to the Ethernet protocol type.

Inner Ethernet packet, as sent/received by the virtual machine:

```
+-----+-----+
| Ethernet | Ethernet |
| header   | Payload  |
+-----+-----+
```

After applying Geneve and ESP, with ETH-IP header.



After applying Geneve and ESP, with GRE header.

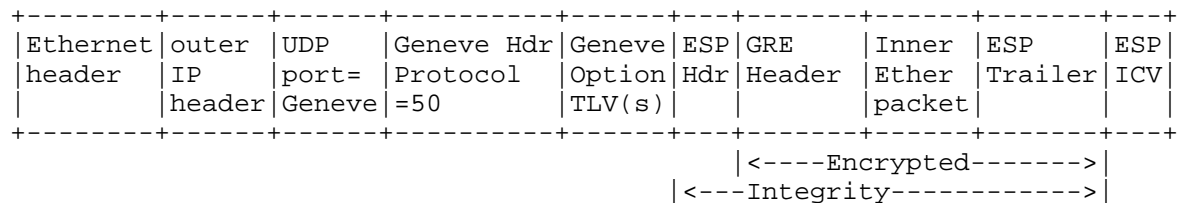


Figure 1: ESP over Geneve Packet Diagram

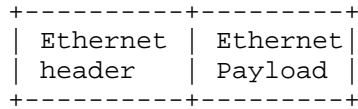
5.0 IP Authentication header (AH) over Geneve tunnel

The Geneve packet is encapsulated in UDP over either IPv4 or IPv6. The Geneve packet consists of a Geneve header which is then followed by a set of variable options. The Geneve header protocol type will indicate that the Geneve payload is the AH protocol (IP Protocol 51), and the Geneve payload will consist of the Authentication header (AH). The AH next header can carry only an IP protocol so can't carry the inner Ethernet frame, so given that (1) Generic Routing Encapsulation (GRE) [RFC2784] and EtherIP [RFC3378] are IP protocols and (2) GRE/EtherIP can carry Ethernet Frames, hence the need of EtherIP/GRE encapsulation for the inner Ethernet payload.

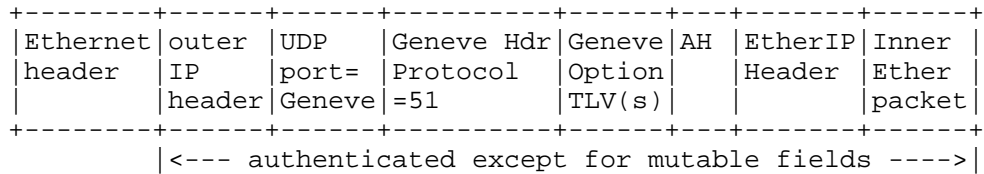
The AH Next Header field will be set to inner payload protocol which can be either EtherIP (97) or to the Generic Routing Encapsulation (GRE) (47). The GRE protocol type will be set to the Ethernet protocol type.

It is to be noted that some of the option TLV(s) in the Geneve header SHOULD be treated as mutable fields and not included in the AH authentication.

Inner Ethernet packet, as sent/received by the virtual machine:



After applying Geneve and AH, with ETH-IP header.



After applying Geneve and AH, with GRE header.

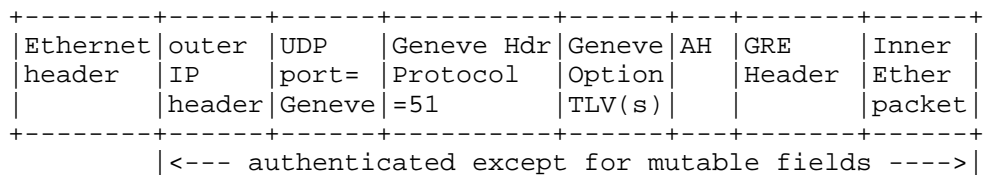


Figure 2: AH over Geneve Packet Diagram

6. Control Plane Considerations

A control plane extension could allow a Network Virtualization Endpoint (NVE) to express the next protocol that can be carried by Geneve to its peers.

In the datapath, a transmitting NVE MUST NOT encapsulate a packet destined to another NVE with any protocol the receiving NVE is not capable of processing.

In this document the next protocol signaled in control plane by NVE(s) can be ESP or AH.

Once 2 NVE(s) agree to carry ESP or AH as next protocol, Security Association and Key Management Protocol defined in [RFC2408] can be used to negotiate, establish, modify and delete Security Associations. As well, mechanisms to perform key exchange defined in [RFC2409] can be used.

7. Acknowledgements

The authors would like to thank T. Sridhar for his valuable comments.

8. Security Considerations

This document does not introduce any additional security constraints.

9. IANA Considerations

TBD

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]

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

[RFC4302] Kent, S. "IP Authentication Header", RFC 4302, December 2005, <<http://www.rfc-editor.org/info/rfc4302>>.

[RFC4303] Kent, S. "IP Encapsulating Security Payload", RFC 4303, December 2005, <<http://www.rfc-editor.org/info/rfc4303>>.

[RFC3378] R. Housley, et al. "EtherIP: Tunneling Ethernet Frames in IP Datagrams", RFC 3378, September 2002, <<http://www.rfc-editor.org/info/rfc3378>>.

[RFC2784] T. Li, et al. "Generic Routing Encapsulation (GRE)", RFC 2784, March 2000, <<http://www.rfc-editor.org/info/rfc2784>>.

[RFC2408] D. Maughan, et al. "Internet Security Association and Key Management Protocol (ISAKMP)", RFC 2408, November 1998, <<http://www.rfc-editor.org/info/rfc2408>>.

[RFC2409] D. Harkins, et al. "The Internet Key Exchange (IKE)", RFC 2409, November 1998, <<http://www.rfc-editor.org/info/rfc2409>>.

Authors' Addresses

Sami Boutros
VMware
Email: sboutros@vmware.com

Dan Wing
VMware, Inc.
Email: dwing@vmware.com

Calvin Qian
VMware, Inc.
Email: calvinq@vmware.com

INTERNET-DRAFT
Intended Status: Informational

Sami Boutros(Ed.)
VMware

Ilango Ganga
Intel

Pankaj Garg
Microsoft

Rajeev Manur
Broadcom

Tal Mizrahi
Marvell

David Mozes
Mellanox

Erik Nordmark

Michael Smith
Cisco

Expires: December 9, 2017

June 7, 2017

NVO3 Encapsulation Considerations
draft-ietf-nvo3-encap-00

Abstract

As communicated by WG Chairs, the IETF NVO3 chairs and Routing Area director have chartered a design team to take forward the encapsulation discussion and see if there is potential to design a common encapsulation that addresses the various technical concerns.

There are implications of different encapsulations in real environments consisting of both software and hardware implementations and spanning multiple data centers. For example, OAM functions such as path MTU discovery become challenging with multiple encapsulations along the data path.

The design team recommend Geneve with few modifications as the common encapsulation, more details are described in section 7.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at <http://www.ietf.org/lid-abstracts.html>

The list of Internet-Draft Shadow Directories can be accessed at <http://www.ietf.org/shadow.html>

Copyright and License Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Problem Statement	4
2. Design Team Goals	4
3. Terminology	4
4. Abbreviations	4
5. Issues with current Encapsulations	5
5.1 Geneve	5
5.2 GUE	5

5.3 VXLAN-GPE	5
6. Common Encapsulation Considerations	6
6.1 Current Encapsulations	6
6.2 Useful Extensions Use cases	6
6.2.1. Telemetry extensions.	6
6.2.2. Security/Integrity extensions	7
6.2.3. Group Base Policy	7
6.3 Hardware Considerations	7
6.4 Extension Size	8
6.5 Extension Ordering	9
6.6 TLV vs Bit Fields	9
6.7 Control Plane Considerations	10
6.8 Split NVE	11
6.9 Larger VNI Considerations	11
7. Design team recommendations	11
8. Acknowledgements	14
9. Security Considerations	14
10. References	14
10.1 Normative References	14
10.2 Informative References	15
11. Appendix A	15
11.1. Overview	15
11.2. Extensibility	15
11.2.1. Native Extensibility Support	15
11.2.2. Extension Parsing	15
11.2.3. Critical Extensions	16
11.2.4. Maximal Header Length	16
11.3. Encapsulation Header	16
11.3.1. Virtual Network Identifier (VNI)	16
11.3.2. Next Protocol	16
11.3.3. Other Header Fields	17
11.4. Comparison Summary	17
Authors' Addresses (In alphabetical order)	18

1. Problem Statement

As communicated by WG Chairs, the NVO3 WG charter states that it may produce requirements for network virtualization data planes based on encapsulation of virtual network traffic over an IP-based underlay data plane. Such requirements should consider OAM and security. Based on these requirements the WG will select, extend, and/or develop one or more data plane encapsulation format(s).

This has led to drafts describing three encapsulations being adopted by the working group:

- draft-ietf-nvo3-geneve-03
- draft-ietf-nvo3-gue-04
- draft-ietf-nvo3-vxlan-gpe-02

Discussion on the list and in face-to-face meetings has identified a number of technical problems with each of these encapsulations. Furthermore, there was clear consensus at the IETF meeting in Berlin that it is undesirable for the working group to progress more than one data plane encapsulation. Although consensus could not be reached on the list, the overall consensus was for a single encapsulation (RFC2418, Section 3.3). Nonetheless there has been resistance to converging on a single encapsulation format.

2. Design Team Goals

As communicated by WG Chairs, the design team should take one of the proposed encapsulations and enhance it to address the technical concerns. The simple evolution of deployed networks as well as applicability to all locations in the NVO3 architecture are goals. The DT should specifically avoid a design that is burdensome on hardware implementations, but should allow future extensibility. The chosen design should also operate well with ICMP and in ECMP environments. If further extensibility is required, then it should be done in such a manner that it does not require the consent of an entity outside of the IETF.

3. 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].

4. 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

Transit device Underlay network devices between NVE(s).

5. Issues with current Encapsulations

As summarized by WG Chairs.

5.1 Geneve

- Can't be implemented cost-effectively in all use cases because variable length header and order of the TLVs makes is costly (in terms of number of gates) to implement in hardware
- Header doesn't fit into largest commonly available parse buffer (256 bytes in NIC). Cannot justify doubling buffer size unless it is mandatory for hardware to process additional option fields.

5.2 GUE

- There were a significant number of objections related to the complexity of implementation in hardware, similar to those noted for Geneve above.

5.3 VXLAN-GPE

- GPE is not day-1 backwards compatible with VXLAN. Although the frame format is similar, it uses a different UDP port, so would require changes to existing implementations even if the rest of the GPE frame is the same.
- GPE is insufficiently extensible. Numerous extensions and options have been designed for GUE and Geneve. Note that these have not yet been validated by the WG.

- Security e.g. of the VNI has not been addressed by GPE. Although a shim header could be used for security and other extensions, this has not been defined yet and its implications on offloading in NICs are not understood.

6. Common Encapsulation Considerations

6.1 Current Encapsulations

Appendix A includes a detailed comparison between the three proposed encapsulations. The comparison indicates several common properties, but also three major differences among the encapsulations:

- Extensibility: Geneve and GUE were defined with built-in extensibility, while VXLAN-GPE is not inherently extensible. Note that any of the three encapsulations can be extended using the Network Service Header (NSH).
- Extension method: Geneve is extensible using Type/Length/Value (TLV) fields, while GUE uses a small set of possible extensions, and a set of flags that indicate which extension is present.
- Length field: Geneve and GUE include a Length field, indicating the length of the encapsulation header, while VXLAN-GPE does not include such a field.

6.2 Useful Extensions Use cases

Non vendor specific TLV MUST follow the standardization process. The following use cases for extensions shows that there is a strong requirement to support variable length extensions with possible different subtypes.

6.2.1. Telemetry extensions.

In several scenarios it is beneficial to make information about the path a packet took through the network or through a network device as well as associated telemetry information available to the operator.

This includes not only tasks like debugging, troubleshooting, as well as network planning and network optimization but also policy or service level agreement compliance checks.

Packet scheduling algorithms, especially for balancing traffic across equal cost paths or links, often leverage information contained within the packet, such as protocol number, IP-address or MAC-address. Probe packets would thus either need to be sent from the

exact same endpoints with the exact same parameters, or probe packets would need to be artificially constructed as "fake" packets and inserted along the path. Both approaches are often not feasible from an operational perspective, be it that access to the end-system is not feasible, or that the diversity of parameters and associated probe packets to be created is simply too large. An in-band telemetry mechanism in extensions is an alternative in those cases.

6.2.2. Security/Integrity extensions

Since the currently proposed NVO3 encapsulations do not protect their headers a single bit corruption in the VNI field could deliver a packet to the wrong tenant. Extensions are needed to use any sophisticated security.

The possibility of VNI spoofing with an NVO3 protocol is exacerbated by the use of UDP. Systems typically have no restrictions on applications being able to send to any UDP port so an unprivileged application can trivially spoof for instance, VXLAN packets, including using arbitrary VNIs.

One can envision HMAC-like support in some NVO3 extension to authenticate the header and the outer IP addresses, thereby preventing attackers from injecting packets with spoofed VNIs.

An other aspect of security is payload security. Essentially this is to make packets that look like IP|UDP|NVO3 Encap|DTLS/IPSEC-ESP Extension|payload. This is nice since we still have the UDP header for ECMP, the NVO3 header is in plain text so it can be read by network elements, and different security or other payload transforms can be supported on a single UDP port (we don't need a separate UDP for DTLS/IPSEC).

6.2.3. Group Base Policy

Another use case would be to carry the Group Based Policy (GBP) source group information within a NVO3 header extension in a similar manner as has been implemented for VXLAN [VXLAN-GBP]. This allows various forms of policy such as access control and QoS to be applied between abstract groups rather than coupled to specific endpoint addresses.

6.3 Hardware Considerations

Hardware restrictions should be taken into consideration along with future hardware enhancements that may provide more flexible metadata processing. However, the set of options that need to and will be

implemented in hardware will be a subset of what is implemented in software, since software NVEs are likely to grow features, and hence option support, at a more rapid rate.

We note that it is hard to predict which options will be implemented in which piece of hardware and when. That depends on whether the hardware will be in the form of a NIC providing increasing offload capabilities to software NVEs, or a switch chip being used as an NVE gateway towards non-NVO3 parts of the network, or even an transit devices that participates in the NVO3 dataplane e.g. for OAM purposes.

A result of this is that it doesn't look useful to prescribe some order of the option so that the ones that are likely to be implemented in hardware come first; we can't decide such an order when we define the options, however a control plane can enforce such order for some hardware implementations.

We do know that hardware needs to initially be able to efficiently skip over the NVO3 header to find the inner payload. That is needed for both NICs doing e.g. TCP offload and transit devices and NVEs applying policy/ACLs to the inner payload.

6.4 Extension Size

Extension header length has a significant impact to hardware and software implementations. A total header length that is too small will unnecessarily constrained software flexibility. A total header length that is too large will place a nontrivial cost on hardware implementations. Thus, the design team recommends that there be a minimum and maximum total extension header length selected. The maximum total header length is determined by the bits allocated for the total extension header length field. The risk with this approach is that it may be difficult to extend the total header size in the future. The minimum total header length is determined by a requirement in the specifications that all implementations must meet. The risk with this approach is that all implementations will only implement the minimum total header length which would then become the de facto maximum total header length. The recommended minimum total header length is 64 bytes.

Single Extension size should always be 4 bytes aligned.

The maximum length of a single option should be large enough to meet the different extension use case requirements e.g. in-band telemetry and future use.

6.5 Extension Ordering

In order to support hardware nodes at the tunnel endpoint or at the transit that can process one or few extensions TLVs in TCAM. A control plane in such a deployment can signal a capability to ensure a specific TLV will always appear in a specific order for example the first one in the packet.

The order of the TLVs should be HW friendly for both the sender and the receiver and possibly the transit node too.

Transit nodes doesn't participate in control plane communication between the end points and are not required to process the options however, if they do, they need to process only a small subset of options that will be consumed by tunnel endpoints.

6.6 TLV vs Bit Fields

If there is a well-known initial set of options that are likely to be implemented in software and in hardware, it can be efficient to use the bit-field approach as in GUE. However, as described in section 6.3, if options are added over time and different subsets of options are likely to be implemented in different pieces of hardware, then it would be hard for the IETF to specify which options should get the early bit fields. TLVs are a lot more flexible, which avoids the need to determine the relative importance different options. However, general TLV of arbitrary order, size, and repetition of the same order is difficult to implement in hardware. A middle ground is to use TLV with restrictions on the size and alignment, observing that individual TLVs can have a fixed length, and support in the control plane such that an NVE will only receive options that to needs and implements. The control plane approach can potentially be used to control the order of the TLVs sent to a particular NVE. Note that transit devices are not likely to participate in the control plane hence to the extent that they need to participate in option processing they need more effort, But transit devices would have issues with future GUE bits being defined for future options as well.

A benefit of TLVs from a HW perspective is that they are self describing i.e., all the information is in the TLV. In a Bit fields approach the hardware needs to look up the bit to determine the length of the data associated with the bit through some separate table, which would add hardware complexity.

There are use cases where multiple modules of software are running on NVE. This can be modules such as a diagnostic module by one vendor that does packet sampling and another module from a different vendor

that does a firewall. Using a TLV format, it is easier to have different software modules process different TLVs, which could be standard extensions or vendor specific extensions defined by the different vendors, without conflicting with each other. This can help with hardware modularity as well. There are some implementations with options that allows different software like mac learning and security handle different options.

6.7 Control Plane Considerations

Given that we want to allow large flexibility and extensibility for e.g. software NVEs, yet be able to support key extensions in less flexible e.g. hardware NVEs, it is useful to consider the control plane. By control plane in this context we mean both protocols such as EVPN and others, and also deployment specific configuration.

If each NVE can express in the control plane that they only care about particular extensions (could be a single extension, or a few), and the source NVEs only include requested extensions in the NVO3 packets, then the target NVE can both use a simpler parser (e.g., a TCAM might be usable to look for a single NVO3 extension) and the depth of the inner payload in the NVO3 packet will be minimized. Furthermore, if the target NVE cares about a few extensions and can express in the control plane the desired order of those extensions in the NVO3 packets, then it can provide useful functionality with minimal hardware requirements.

Note that transit devices that are not aware of the NVO3 extensions somewhat benefit from such an approach, since the inner payload is less deep in the packet if no extraneous extensions are included in the packet. However, in general a transit device is not likely to participate in the NVO3 control plane. (However, configuration mechanisms can take into account limitations of the transit devices used in particular deployments.)

Note that in this approach different NVEs could desire different (sets of) extensions, which means that the source NVE needs to be able to place different sets of extensions in different NVO3 packets, and perhaps in different order. It also assumes that underlay multicast or replication servers are not used together with NVO3 extensions.

There is a need to consider mandatory extensions versus optional extensions. Mandatory extensions require the receiver to drop the packet if the extension is unknown. A control plane mechanism can prevent the need for dropping unknown extensions, since they would not be included to targets that do not support them.

The control planes defined today need to add the ability to describe the different encapsulations. Thus perhaps EVPN, and any other control plane protocol that the IETF defines, should have a way to enumerate the supported NVO3 extensions and their order.

The WG should consider developing a separate draft on guidance for option processing and control plane participation. This should provide examples/guidance on range of usage models and deployments scenarios for specific options and ordering that are relevant for that specific deployment. This includes end points and middle boxes using the options. So, having the control plane negotiate the constraints is most appropriate and flexible way to address these requirements.

6.8 Split NVE

If the working group sees a need for having the hosts send and receive options in a split NVE case, this is possible using any of the existing extensible encapsulations (Geneve, GUE, GPE+NSH) by defining a way to carry those over other transports. NSH can already be used over different transports.

If we need to do this with other encapsulations it can be done by defining an Ether type for other encapsulations so that it can be carried over Ethernet and 802.1Q.

If we need to carry other encapsulations over MPLS, it would require an EVPN control plane to signal that other encapsulation header + options will be present in front of the L2 packet. The VNI can be ignored in the header, and the MPLS label will be the one used to identify the EVPN L2 instance.

6.9 Larger VNI Considerations

We discussed whether we should make VNI 32-bits or larger. The benefit of 24-bit VNI would be to avoid unnecessary changes with existing proposals and implementations that are almost all, if not all, are using 24-bit VNI. If we need a larger VNI, an extension can be used to support that.

7. Design team recommendations

We concluded that Geneve is most suitable as a starting point for proposed standard for network virtualization, for the following reasons:

1. We studied whether VNI should be in base header or in extensions and whether it should be 24-bit or 32-bit. The design team agreed that VNI is critical information for network virtualization and MUST be present in all packets. Design team also agreed that 24-bit VNI matches the existing widely used encapsulation format i.e. VxLAN and NVGRE and hence more suitable to use going forward.

2. Geneve has the total options length that allow skipping over the options for NIC offload operations, and will allow transit devices to view flow information in the inner payload.

3. We considered the option of using NSH with VxLAN-GPE but given that NSH is targeted at service chaining and contains service chaining information, it is less suitable for the network virtualization use case. The other downside for VxLAN-GPE was lack of header length in VxLAN-GPE and hence makes skipping over the headers to process inner payload more difficult. Total Option Length is present in Geneve. It is not possible to skip any options in the middle with VxLAN-GPE. In principle a split between a base header and a header with options is interesting (whether that options header is NSH or some new header without ties to a service path). We explored whether it would make sense to either use NSH for this, or define a new NVO3 options header. However, we observed that this makes it slightly harder to find the inner payload since the length field is not in the NVO3 header itself. Thus one more field would have to be extracted to compute the start of the inner payload. Also, if the experience with IPv6 extension headers is a guidance, there would be a risk that key pieces of hardware might not implement the options header, resulting in future calls to deprecate its use. Making the options part of the base NVO3 header has less of those issues. Even though the implementation of any particular option can not be predicted ahead of time, the option mechanism and ability to skip the options is likely to be broadly implemented.

4. We compared the TLV vs Bit-fields style extension and it was deemed that parsing both TLV and bit-fields is expensive and while bit-fields may be simpler to parse, it is also more restrictive and requires guessing which extensions will be widely implemented so they can get early bit assignments, given that half the bits are already assigned in GUE, a widely deployed extension may appear in a flag extension, and this will require extra processing, to dig the flag from the flag extension and then look for the extension itself. As well Bit-fields are not flexible enough to address the requirements from OAM, Telemetry and security extensions, for variable length option and different subtypes of the same option. While TLV are more flexible, a control plane can restrict the number of option TLVs as well the order and size of the TLVs to make it simpler for a dataplane implementation to handle.

5. We briefly discussed multi-vendor NVE case, and the need to allow vendors to put their own extensions in the NVE header. This is possible with TLVs.

6. We also agreed that the C bit in Geneve is helpful to allow receiver NVE to easily decide whether to process options or not. For example a UUID based packet trace and how an optional extension such as that can be ignored by receiver NVE and thus make it easy for NVE to skip over the options. Thus the C-bit remains as defined in Geneve.

7. There are already some extensions that are being discussed (see section 6.2) of varying sizes, by using Geneve option it is possible to get in band parameters like: switch id, ingress port, egress port, internal delay, and queue in telemetry defined extension TLV from switches. It is also possible to add Security extension TLVs like HMAC and DTLS/IPSEC to authenticate the Geneve packet header and secure the Geneve packet payload by software or hardware tunnel endpoints. As well, a Group Based Policy extension TLV can be carried.

8. There are implemented Geneve options today in production. There are as well new HW supporting Geneve TLV parsing. In addition In-band Telemetry (INT) specification being developed by P4.org illustrates the option of INT meta data carried over Geneve. OVN/OVS have also defined some option TLV(s) for Geneve.

9. The DT has addressed the usage models while considering the requirements and implementations in general that includes software and hardware.

There seems to be interest to standardize some well known secure option TLVs to secure the header and payload to guarantee encapsulation header integrity and tenant data privacy. The design team recommends that the working group consider standardizing such option(s).

We recommend the following enhancements to Geneve to make it more suitable to hardware and yet provide the flexibility for software:

We would propose a text such as, while TLV are more flexible, a control plane can restrict the number of option TLVs as well the order and size of the TLVs to make it simpler for a data plane implementation in software or hardware to handle. For example, there may be some critical information such as secure hash that must be processed in certain order at lowest latency.

A control plane can negotiate a subset of option TLVs and certain TLV

ordering, as well can limit the total number of option TLVs present in the packet, for example, to allow hardware capable of processing fewer options. Hence, the control planes need to have the ability to describe the supported TLVs subset and their order.

The Geneve draft could specify that the subset and order of option TLVs should be configurable for each remote NVE in the absence of a protocol control plane.

We recommend Geneve to follow fragmentation recommendations in overlay services like PWE3, and L2/L3 VPN recommendation to guarantee larger MTU for the tunnel overhead
<https://tools.ietf.org/html/rfc3985#section-5.3>

We request Geneve to provide a recommendation for critical bit processing - text could look like how critical bits can be used with control plane specifying the critical options.

Given that there is a telemetry option use case for a length of 256 bytes, we recommend Geneve to increase the Single TLV option length to 256.

We request Geneve to address Requirements for OAM considerations for alternate marking and for performance measurements that need 2 bits in the header. And clarify the need of the current OAM bit in the Geneve Header.

We recommend the WG to work on security options for Geneve.

8. Acknowledgements

The authors would like to thank Tom Herbert for providing the motivation for the Security/Integrity extension, and for his valuable comments, and would like to thank T. Sridhar for his valuable comments and feedback.

9. Security Considerations

This document does not introduce any additional security constraints.

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]
[GUE] Generic UDP Encapsulation [I-D.ietf-nvo3-gue]
[NSH] Network Service Header [I-D.ietf-sfc-nsh]
[VXLAN-GPE] Virtual eXtensible Local Area Network - Generic Protocol Extension [I-D.ietf-nvo3-vxlan-gpe]

[VXLAN-GBP] VXLAN Group Policy Option - [I-D.draft-smith-vxlan-group-policy-03]

11. Appendix A

11.1. Overview

This section presents a comparison of the three NVO3 encapsulation proposals, Geneve, GUE, and VXLAN-GPE. The three encapsulations use an outer UDP/IP transport. Geneve and VXLAN-GPE use an 8-octet header, while GUE uses a 4-octet header. In addition to the base header, optional extensions may be included in the encapsulation, as discussed in Section 3.2 below.

11.2. Extensibility

11.2.1. Native Extensibility Support

The Geneve and GUE encapsulations both enable optional headers to be incorporated at the end of the base encapsulation header.

VXLAN-GPE does not provide native support for header extensions. However, as discussed in [I-D.ietf-nvo3-vxlan-gpe], extensibility can be attained to some extent if the Network Service Header (NSH) [I-D.ietf-sfc-nsh] is used immediately following the VXLAN-GPE header. NSH supports either a fixed-size extension (MD Type 1), or a variable-size TLV-based extension (MD Type 2). It should be noted that NSH-over-VXLAN-GPE implies an additional overhead of the 8-octets NSH header, in addition to the VXLAN-GPE header.

11.2.2. Extension Parsing

The Geneve Variable Length Options are defined as Type/Length/Value(TLV) extensions. Similarly, VXLAN-GPE, when using NSH, can include NSH TLV-based extensions. In contrast, GUE defines a small set of possible extension fields (proposed in [I-D.herbert-

gue-extensions]), and a set of flags in the GUE header that indicate for each extension type whether it is present or not.

TLV-based extensions, as defined in Geneve, provide the flexibility for a large number of possible extension types. Similar behavior can be supported in NSH-over-VXLAN-GPE when using MD Type 2. The flag-based approach taken in GUE strives to simplify implementations by defining a small number of possible extensions, used in a fixed order.

The Geneve and GUE headers both include a length field, defining the total length of the encapsulation, including the optional extensions.

The length field simplifies the parsing of transit devices that skip the encapsulation header without parsing its extensions.

11.2.3. Critical Extensions

The Geneve encapsulation header includes the 'C' field, which indicates whether the current Geneve header includes critical options, which must be parsed by the tunnel endpoint. If the endpoint is not able to process the critical option, the packet is discarded.

11.2.4. Maximal Header Length

The maximal header length in Geneve, including options, is 260 octets. GUE defines the maximal header to be 128 octets. VXLAN-GPE uses a fixed-length header of 8 octets, unless NSH-over-VXLAN-GPE is used, yielding an encapsulation header of up to 264 octets.

11.3. Encapsulation Header

11.3.1. Virtual Network Identifier (VNI)

The Geneve and VXLAN-GPE headers both include a 24-bit VNI field. GUE, on the other hand, enables the use of a 32-bit field called VNID; this field is not included in the GUE header, but was defined as an optional extension in [I-D.herbert-gue-extensions].

The VXLAN-GPE header includes the 'I' bit, indicating that the VNI field is valid in the current header. A similar indicator is defined as a flag in the GUE header [I-D.herbert-gue-extensions].

11.3.2. Next Protocol

The three encapsulation headers include a field that specifies the

type of the next protocol header, which resides after the NVO3 encapsulation header. The Geneve header includes a 16-bit field that uses the IEEE Ethertype convention. GUE uses an 8-bit field, which uses the IANA Internet protocol numbering. The VXLAN-GPE header incorporates an 8-bit Next Protocol field, using a VXLAN-GPE-specific registry, defined in [I-D.ietf-nvo3-vxlan-gpe].

The VXLAN-GPE header also includes the 'P' bit, which explicitly indicates whether the Next Protocol field is present in the current header.

11.3.3. Other Header Fields

The OAM bit, which is defined in Geneve and in VXLAN-GPE, indicates whether the current packet is an OAM packet. The GUE header includes a similar field, but uses different terminology; the GUE 'C-bit' specifies whether the current packet is a control packet. Note that the GUE control bit can potentially be used in a large set of protocols that are not OAM protocols. However, the control packet examples discussed in [I-D.ietf-nvo3-gue] are OAM-related.

Each of the three NVO3 encapsulation headers includes a 2-bit Version field, which is currently defined to be zero.

The Geneve and VXLAN-GPE headers include reserved fields; 14 bits in the Geneve header, and 27 bits in the VXLAN-GPE header are reserved.

11.4. Comparison Summary

The following table summarizes the comparison between the three NVO3 encapsulations.

	Geneve	GUE	VXLAN-GPE
Outer transport	UDP/IP	UDP/IP	UDP/IP
Base header length	8 octets	4 octets	8 octets (16 octets using NSH)
Extensibility	Variable length options	Extension fields	No native extensibility. Extensible using NSH.

Extension parsing method	TLV-based	Flag-based	TLV-based (using NSH with MD Type 2)
Extension order	Variable	Fixed	Variable (using NSH)
Length field	+	+	-
Max Header Length	260 octets	128 octets	8 octets (264 using NSH)
Critical extension bit	+	-	-
VNI field size	24 bits	32 bits (extension)	24 bits
Next protocol field	16 bits Ethertype registry	8 bits Internet protocol registry	8 bits New registry
Next protocol indicator	-	-	+
OAM / control field	OAM bit	Control bit	OAM bit
Version field	2 bits	2 bits	2 bits
Reserved bits	14 bits	-	27 bits

Figure 1: NVO3 Encapsulation Comparison

Authors' Addresses (In alphabetical order)

Sami Boutros
 VMware
 Email: sboutros@vmware.com

Ilango Ganga
 Intel
 Email: ilango.s.ganga@intel.com

Pankaj Garg
 Microsoft

Email: pankajg@microsoft.com

Rajeev Manur
Broadcom
Email: rajeev.manur@broadcom.com

Tal Mizrahi
Marvell
Email: talmi@marvell.com

David Mozes
Mellanox
Email: davidm@mellanox.com

Erik Nordmark
Arista Networks
Email: nordmark@sonic.net

Michael Smith
Cisco
Email: michsmit@cisco.com

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

Ignas Bagdonas
Equinix
Email: ibagdona.ietf@gmail.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 14, 2017

J. Gross, Ed.
I. Ganga, Ed.
Intel
T. Sridhar, Ed.
VMware
March 13, 2017

Geneve: Generic Network Virtualization Encapsulation
draft-ietf-nvo3-geneve-04

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 <http://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 September 14, 2017.

Copyright Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents

(<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
1.1. Requirements Language	4
1.2. Terminology	4
2. Design Requirements	5
2.1. Control Plane Independence	6
2.2. Data Plane Extensibility	7
2.2.1. Efficient Implementation	7
2.3. Use of Standard IP Fabrics	8
3. Geneve Encapsulation Details	9
3.1. Geneve Packet Format Over IPv4	9
3.2. Geneve Packet Format Over IPv6	10
3.3. UDP Header	12
3.4. Tunnel Header Fields	13
3.5. Tunnel Options	14
3.5.1. Options Processing	16
4. Implementation and Deployment Considerations	17
4.1. Encapsulation of Geneve in IP	17
4.1.1. IP Fragmentation	17
4.1.2. DSCP and ECN	17
4.1.3. Broadcast and Multicast	18
4.1.4. Unidirectional Tunnels	18
4.2. Constraints on Protocol Features	19
4.2.1. Constraints on Options	19
4.3. NIC Offloads	19
4.4. Inner VLAN Handling	20
5. Interoperability Issues	20
6. Security Considerations	21
7. IANA Considerations	22
8. Contributors	22
9. Acknowledgements	24
10. References	24
10.1. Normative References	24
10.2. Informative References	24
Authors' Addresses	26

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], and STT [I-D.davie-stt], 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] 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.

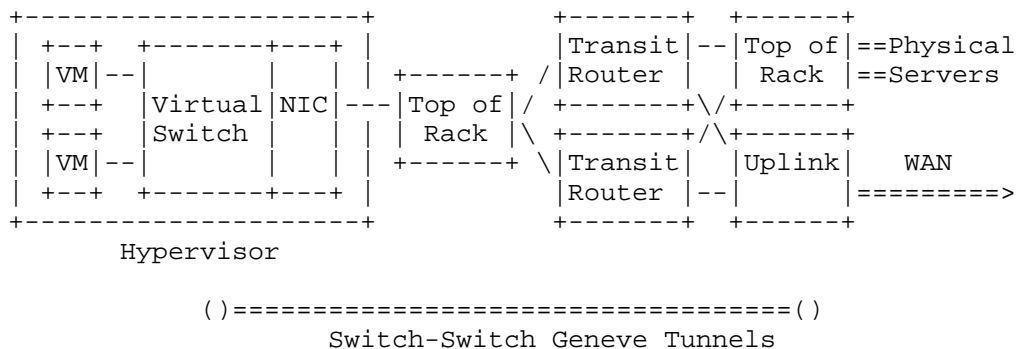


Figure 1: Sample Geneve Deployment

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:

- o The data plane is generic and extensible enough to support current and future control planes.
- o Tunnel components are efficiently implementable in both hardware and software without restricting capabilities to the lowest common denominator.
- o 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 and participate in Geneve packet processing. However, as non-terminating devices, they do not originate or terminate the Geneve packet. The participation of transit devices in Geneve packet processing 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

```

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

```

Outer Ethernet Header:

```

+++++
|                                     Outer Destination MAC Address                                     |
+++++
| Outer Destination MAC Address | Outer Source MAC Address |
+++++
|                                     Outer Source MAC Address                                     |
+++++
| Optional Ethertype=C-Tag 802.1Q | Outer VLAN Tag Information |
+++++
|                                     Ethertype=0x0800                                     |
+++++

```

Outer IPv4 Header:

```

+++++
| Version | IHL | Type of Service | Total Length |
+++++
| Identification | Flags | Fragment Offset |
+++++
| Time to Live | Protocol=17 UDP | Header Checksum |
+++++
|                                     Outer Source IPv4 Address                                     |
+++++
|                                     Outer Destination IPv4 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 |
+-----+-----+-----+-----+-----+-----+-----+-----+
|           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 FCS is not included) |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

Frame Check Sequence:

```

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

```

3.2. Geneve Packet Format Over IPv6

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

Outer Ethernet Header:

```

+-----+
|                Outer Destination MAC Address                |
+-----+
| Outer Destination MAC Address | Outer Source MAC Address |
+-----+
|                Outer Source MAC Address                |
+-----+
|Optional Ethertype=C-Tag 802.1Q| Outer VLAN Tag Information |
+-----+
|                Ethertype=0x86DD                |
+-----+

```

Outer IPv6 Header:

```

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

```

Outer UDP Header:

Source Port = xxxx	Dest Port = 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 FCS is not included) |
+-----+

```

Frame Check Sequence:

```

+-----+
| New FCS (Frame Check Sequence) 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.

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. 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 or modify packets on the basis of this bit.

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

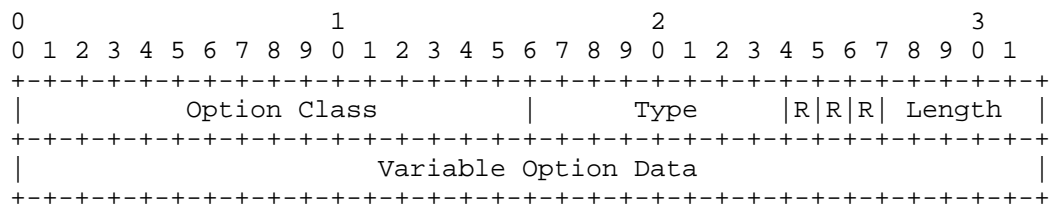
Protocol Type (16 bits): The type of the protocol data unit appearing after the Geneve header. This follows the EtherType [ETYPES] convention with Ethernet itself being represented by the value 0x6558.

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:

```

0 1 2 3 4 5 6 7 8
+-----+
|C|      Type      |
+-----+
```

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. 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 primarily intended to be originated and processed by tunnel endpoints. However, options MAY be processed by transit devices along the tunnel path as well. 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:

- o 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.
- o Some options may be defined in such a way that the position in the option list is significant. Therefore, options MUST NOT be reordered by transit devices.
- o 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.

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.dt-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.dt-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:

- o 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.
- o Options MUST NOT be reordered during the course of offload processing, including when merging packets for the purpose of LRO.
- o 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 three most common protocols used for network virtualization (VXLAN, NVGRE, and STT) 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 UDP/IP packets, 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 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. If the remote tunnel endpoint is not completely trusted, for example it resides on a customer premises, then it may also be necessary to sanitize any tunnel metadata to prevent tenant-hopping attacks.

Geneve does not otherwise affect the security of the encapsulated packets.

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..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
Brocade Communications Systems, Inc.
130 Holger Way
San Jose, CA 95134
USA

Email: jon.hudson@gmail.com

Ariel Hendel
Broadcom Limited
3151 Zanker Road
San Jose, CA 95134
USA

Email: ariel.hendel@broadcom.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, <<http://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, <<http://www.rfc-editor.org/info/rfc2119>>.
- [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, DOI 10.17487/RFC5226, May 2008, <<http://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.davie-stt]
Davie, B. and J. Gross, "A Stateless Transport Tunneling Protocol for Network Virtualization (STT)", draft-davie-stt-08 (work in progress), April 2016.
- [I-D.dt-nvo3-encap]
Boutros, S., Aldrin, S., Elzur, U., Ganga, I., Manur, R., Mozes, D., and M. Smith, "NVO3 Encapsulation Considerations", draft-dt-nvo3-encap-01 (work in progress), March 2017.

- [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.
- [IEEE.802.1Q-2014]
IEEE, "IEEE Standard for Local and metropolitan area
networks -- Bridges and Bridged Networks", IEEE
Std 802.1Q, 2014.
- [RFC1191] Mogul, J. and S. Deering, "Path MTU discovery", RFC 1191,
DOI 10.17487/RFC1191, November 1990,
<<http://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, <<http://www.rfc-editor.org/info/rfc1981>>.
- [RFC2983] Black, D., "Differentiated Services and Tunnels",
RFC 2983, DOI 10.17487/RFC2983, October 2000,
<<http://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,
<<http://www.rfc-editor.org/info/rfc3031>>.
- [RFC4301] Kent, S. and K. Seo, "Security Architecture for the
Internet Protocol", RFC 4301, DOI 10.17487/RFC4301,
December 2005, <<http://www.rfc-editor.org/info/rfc4301>>.
- [RFC6040] Briscoe, B., "Tunnelling of Explicit Congestion
Notification", RFC 6040, DOI 10.17487/RFC6040, November
2010, <<http://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,
<<http://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,
<<http://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, <<http://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, <<http://www.rfc-editor.org/info/rfc7637>>.
- [VL2] Greenberg et al, , "VL2: A Scalable and Flexible Data Center Network", 2009.
Proc. ACM SIGCOMM 2009

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 Working Group
INTERNET-DRAFT
Intended Status: Informational

Yizhou Li
Lucy Yong
Huawei Technologies
Lawrence Kreeger
Cisco
Thomas Narten
IBM
David Black
EMC
February 28, 2017

Expires: September 1, 2017

Split-NVE Control Plane Requirements
draft-ietf-nvo3-hpvr2nve-cp-req-06

Abstract

In a Split-NVE architecture, the functions of the NVE are split across a server and an external network equipment which is called an external NVE. The server-resident control plane functionality resides in control software, which may be part of a hypervisor or container management software; for simplicity, this draft refers to the hypervisor as the location of this software.

A control plane protocol(s) between a hypervisor and its associated external NVE(s) is used for the hypervisor to distribute its virtual machine networking state to the external NVE(s) for further handling. This document illustrates the functionality required by this type of control plane signaling protocol and outlines the high level requirements. Virtual machine states as well as state transitioning are summarized to help clarifying the needed protocol requirements.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at
<http://www.ietf.org/lid-abstracts.html>

The list of Internet-Draft Shadow Directories can be accessed at
<http://www.ietf.org/shadow.html>

Copyright and License Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	4
1.1 Terminology	5
1.2 Target Scenarios	6
2. VM Lifecycle	8
2.1 VM Creation Event	8
2.2 VM Live Migration Event	9
2.3 VM Termination Event	10
2.4 VM Pause, Suspension and Resumption Events	10
3. Hypervisor-to-NVE Control Plane Protocol Functionality	10
3.1 VN connect and Disconnect	11
3.2 TSI Associate and Activate	12
3.3 TSI Disassociate and Deactivate	15
4. Hypervisor-to-NVE Control Plane Protocol Requirements	16
5. VDP Applicability and Enhancement Needs	17
6. Security Considerations	19
7. IANA Considerations	19
8. Acknowledgements	19
8. References	20
8.1 Normative References	20
8.2 Informative References	20

Appendix A. IEEE 802.1Qbg VDP Illustration (For information only)	20
Authors' Addresses	23

1. Introduction

In the Split-NVE architecture shown in Figure 1, the functionality of the NVE is split across an end device supporting virtualization and an external network device which is called an external NVE. The portion of the NVE functionality located on the end device is called the tNVE and the portion located on the external NVE is called the nNVE in this document. Overlay encapsulation/decapsulation functions are normally off-loaded to the nNVE on the external NVE.

The tNVE is normally implemented as a part of hypervisor or container and/or virtual switch in an virtualized end device. This document uses the term "hypervisor" throughout when describing the Split-NVE scenario where part of the NVE functionality is off-loaded to a separate device from the "hypervisor" that contains a VM connected to a VN. In this context, the term "hypervisor" is meant to cover any device type where part of the NVE functionality is off-loaded in this fashion, e.g., a Network Service Appliance, Linux Container.

The problem statement [RFC7364], discusses the needs for a control plane protocol (or protocols) to populate each NVE with the state needed to perform the required functions. In one scenario, an NVE provides overlay encapsulation/decapsulation packet forwarding services to Tenant Systems (TSs) that are co-resident within the NVE on the same End Device (e.g. when the NVE is embedded within a hypervisor or a Network Service Appliance). In such cases, there is no need for a standardized protocol between the hypervisor and NVE, as the interaction is implemented via software on a single device. While in the Split-NVE architecture scenarios, as shown in figure 2 to figure 4, a control plane protocol(s) between a hypervisor and its associated external NVE(s) is required for the hypervisor to distribute the virtual machines networking states to the NVE(s) for further handling. The protocol indeed is an NVE-internal protocol and runs between tNVE and nNVE logical entities. This protocol is mentioned in NVO3 problem statement [RFC7364] and appears as the third work item.

Virtual machine states and state transitioning are summarized in this document to show events where the NVE needs to take specific actions. Such events might correspond to actions the control plane signaling protocols between the hypervisor and external NVE will need to take. Then the high level requirements to be fulfilled are outlined.

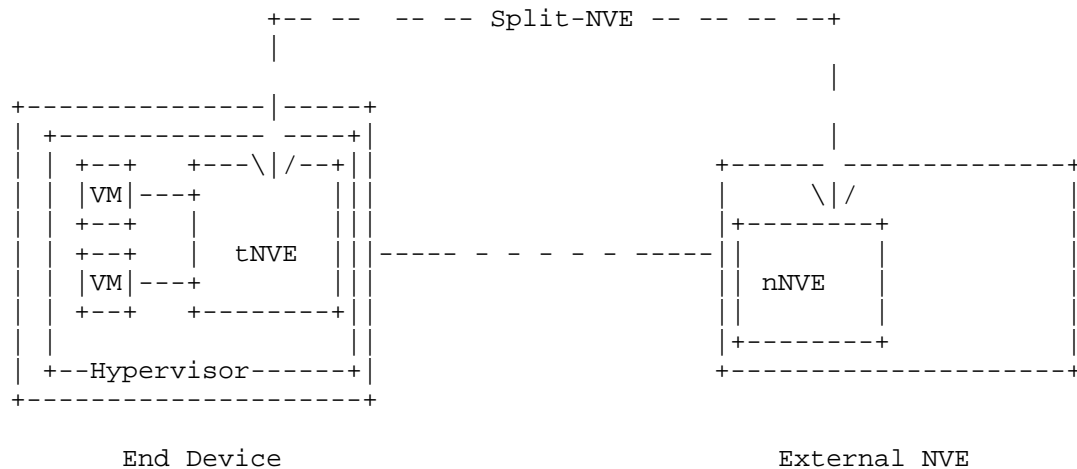


Figure 1 Split-NVE structure

This document uses VMs as an example of Tenant Systems (TSs) in order to describe the requirements, even though a VM is just one type of Tenant System that may connect to a VN. For example, a service instance within a Network Service Appliance is another type of TS, as are systems running on an OS-level virtualization technologies like containers. The fact that VMs have lifecycles (e.g., can be created and destroyed), can be moved, and can be started or stopped results in a general set of protocol requirements, most of which are applicable to other forms of TSs. It should also be noted that not all of the requirements are applicable to all forms of TSs.

Section 2 describes VM states and state transitioning in its lifecycle. Section 3 introduces Hypervisor-to-NVE control plane protocol functionality derived from VM operations and network events. Section 4 outlines the requirements of the control plane protocol to achieve the required functionality.

1.1 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].

This document uses the same terminology as found in [RFC7365] and [I-D.ietf-nvo3-nve-nva-cp-req]. This section defines additional terminology used by this document.

Split-NVE: a type of NVE that the functionalities of it are split across an end device supporting virtualization and an external network device.

tNVE: the portion of Split-NVE functionalities located on the end device supporting virtualization. It interacts with tenant system by internal interface in end device.

nNVE: the portion of Split-NVE functionalities located on the network device which is directly or indirectly connects to the end device holding the corresponding tNVE. nNVE normally performs encapsulation and decapsulation to the overlay network.

External NVE: the physical network device holding nNVE

Hypervisor/Container: the logical collection of software, firmware and/or hardware that allows the creation and running of server or service appliance virtualization. tNVE is located on Hypervisor/Container. It is loosely used in this document to refer to the end device supporting the virtualization. For simplicity, we also use Hypervisor in this document to represent both hypervisor and container.

VN Profile: Meta data associated with a VN that is applied to any attachment point to the VN. That is, VAP properties that are applied to all VAPs associated with a given VN and used by an NVE when ingressing/egressing packets to/from a specific VN. Meta data could include such information as ACLs, QoS settings, etc. The VN Profile contains parameters that apply to the VN as a whole. Control protocols between the NVE and NVA could use the VN ID or VN Name to obtain the VN Profile.

VSI: Virtual Station Interface. [IEEE 802.1Qbg]

VDP: VSI Discovery and Configuration Protocol [IEEE 802.1Qbg]

1.2 Target Scenarios

In the Split-NVE architecture, an external NVE can provide an offload of the encapsulation / decapsulation function, network policy enforcement, as well as the VN Overlay protocol overhead. This offloading may provide performance improvements and/or resource savings to the End Device (e.g. hypervisor) making use of the external NVE.

The following figures give example scenarios of a Split-NVE architecture.

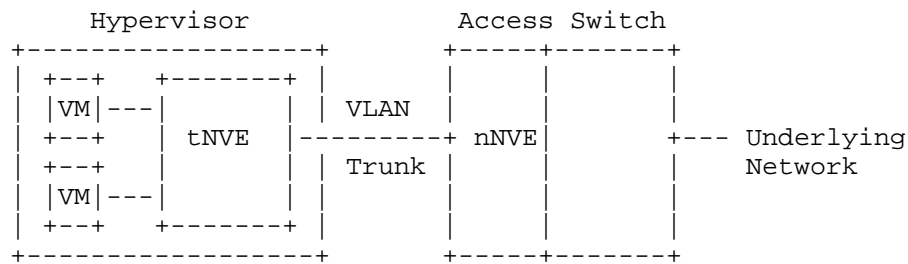


Figure 2 Hypervisor with an External NVE

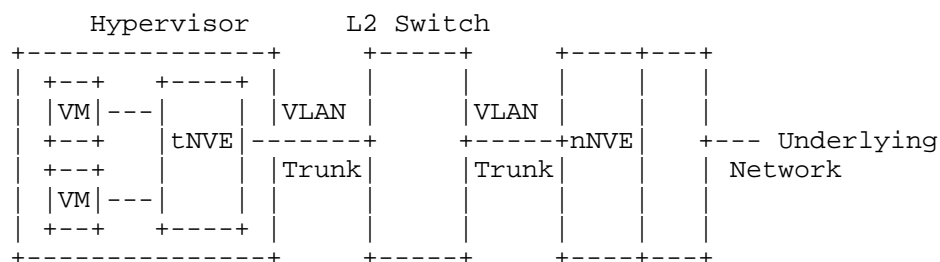
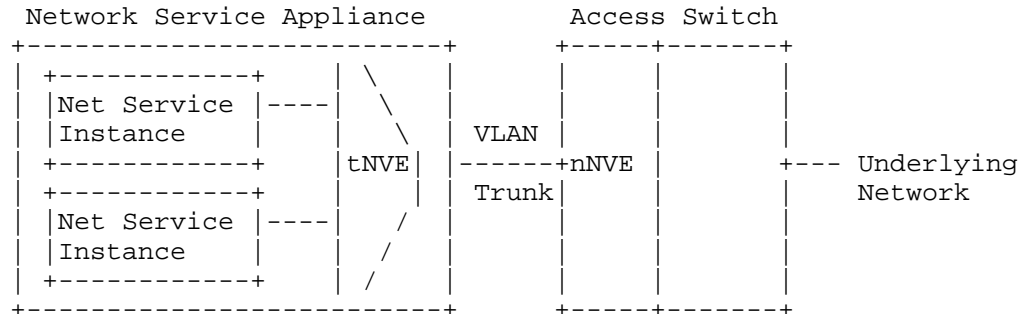
Figure 3 Hypervisor with an External NVE
across an Ethernet Access Switch

Figure 4 Physical Network Service Appliance with an External NVE

Tenant Systems connect to external NVEs via a Tenant System Interface (TSI). The TSI logically connects to the external NVE via a Virtual Access Point (VAP) [I-D.ietf-nvo3-arch]. The external NVE may provide Layer 2 or Layer 3 forwarding. In the Split-NVE architecture, the external NVE may be able to reach multiple MAC and IP addresses via a TSI. For example, Tenant Systems that are providing network services (such as transparent firewall, load balancer, VPN gateway) are likely

to have complex address hierarchy. This implies that if a given TSI disassociates from one VN, all the MAC and/or IP addresses are also disassociated. There is no need to signal the deletion of every MAC or IP when the TSI is brought down or deleted. In the majority of cases, a VM will be acting as a simple host that will have a single TSI and single MAC and IP visible to the external NVE.

Figures 2-4 show the use of VLANs to separate traffic for multiple VNs between the tNVE and nNVE; VLANs are not strictly necessary if only one VN is involved, but multiple VNs are expected in most cases, and hence this draft assumes their presence.

2. VM Lifecycle

Figure 2 of [I-D.ietf-opsawg-vmm-mib] shows the state transition of a VM. Some of the VM states are of interest to the external NVE. This section illustrates the relevant phases and events in the VM lifecycle. It should be noted that the following subsections do not give an exhaustive traversal of VM lifecycle state. They are intended as the illustrative examples which are relevant to Split-NVE architecture, not as prescriptive text; the goal is to capture sufficient detail to set a context for the signaling protocol functionality and requirements described in the following sections.

2.1 VM Creation Event

VM creation event makes the VM state transiting from Preparing to Shutdown and then to Running [I-D.ietf-opsawg-vmm-mib]. The end device allocates and initializes local virtual resources like storage in the VM Preparing state. In Shutdown state, the VM has everything ready except that CPU execution is not scheduled by the hypervisor and VM's memory is not resident in the hypervisor. From the Shutdown state to Running state, normally it requires the human execution or system triggered event. Running state indicates the VM is in the normal execution state. As part of transitioning the VM to the Running state, the hypervisor must also provision network connectivity for the VM's TSI(s) so that Ethernet frames can be sent and received correctly. No ongoing migration, suspension or shutdown is in process.

In the VM creation phase, the VM's TSI has to be associated with the external NVE. Association here indicates that hypervisor and the external NVE have signaled each other and reached some agreement. Relevant networking parameters or information have been provisioned properly. The External NVE should be informed of the VM's TSI MAC address and/or IP address. In addition to external network

connectivity, the hypervisor may provide local network connectivity between the VM's TSI and other VM's TSI that are co-resident on the same hypervisor. When the intra or inter-hypervisor connectivity is extended to the external NVE, a locally significant tag, e.g. VLAN ID, should be used between the hypervisor and the external NVE to differentiate each VN's traffic. Both the hypervisor and external NVE sides must agree on that tag value for traffic identification, isolation and forwarding.

The external NVE may need to do some preparation work before it signals successful association with TSI. Such preparation work may include locally saving the states and binding information of the tenant system interface and its VN, communicating with the NVA for network provisioning, etc.

Tenant System interface association should be performed before the VM enters running state, preferably in Shutdown state. If association with external NVE fails, the VM should not go into running state.

2.2 VM Live Migration Event

Live migration is sometimes referred to as "hot" migration, in that from an external viewpoint, the VM appears to continue to run while being migrated to another server (e.g., TCP connections generally survive this class of migration). In contrast, "cold" migration consists of shutdown VM execution on one server and restart it on another. For simplicity, the following abstract summary about live migration assumes shared storage, so that the VM's storage is accessible to the source and destination servers. Assume VM live migrates from hypervisor 1 to hypervisor 2. Such migration event involves the state transition on both hypervisors, source hypervisor 1 and destination hypervisor 2. VM state on source hypervisor 1 transits from Running to Migrating and then to Shutdown [I-D.ietf-opsawg-vmm-mib]. VM state on destination hypervisor 2 transits from Shutdown to Migrating and then Running.

The external NVE connected to destination hypervisor 2 has to associate the migrating VM's TSI with it by discovering the TSI's MAC and/or IP addresses, its VN, locally significant VID if any, and provisioning other network related parameters of the TSI. The external NVE may be informed about the VM's peer VMs, storage devices and other network appliances with which the VM needs to communicate or is communicating. The migrated VM on destination hypervisor 2 SHOULD not go to Running state before all the network provisioning and binding has been done.

The migrating VM SHOULD not be in Running state at the same time on

the source hypervisor and destination hypervisor during migration. The VM on the source hypervisor does not transition into Shutdown state until the VM successfully enters the Running state on the destination hypervisor. It is possible that VM on the source hypervisor stays in Migrating state for a while after VM on the destination hypervisor is in Running state.

2.3 VM Termination Event

VM termination event is also referred to as "powering off" a VM. VM termination event leads to its state going to Shutdown. There are two possible causes to terminate a VM [I-D.ietf-opsawg-vmm-mib], one is the normal "power off" of a running VM; the other is that VM has been migrated to another hypervisor and the VM image on the source hypervisor has to stop executing and to be shutdown.

In VM termination, the external NVE connecting to that VM needs to deprovision the VM, i.e. delete the network parameters associated with that VM. In other words, the external NVE has to de-associate the VM's TSI.

2.4 VM Pause, Suspension and Resumption Events

The VM pause event leads to the VM transiting from Running state to Paused state. The Paused state indicates that the VM is resident in memory but no longer scheduled to execute by the hypervisor [I-D.ietf-opsawg-vmm-mib]. The VM can be easily re-activated from Paused state to Running state.

The VM suspension event leads to the VM transiting from Running state to Suspended state. The VM resumption event leads to the VM transiting state from Suspended state to Running state. Suspended state means the memory and CPU execution state of the virtual machine are saved to persistent store. During this state, the virtual machine is not scheduled to execute by the hypervisor [I-D.ietf-opsawg-vmm-mib].

In the Split-NVE architecture, the external NVE should keep any paused or suspended VM in association as the VM can return to Running state at any time.

3. Hypervisor-to-NVE Control Plane Protocol Functionality

The following subsections show the illustrative examples of the state transitions on external NVE which are relevant to Hypervisor-to-NVE Signaling protocol functionality. It should be noted they are not prescriptive text for full state machines.

3.1 VN connect and Disconnect

In Split-NVE scenario, a protocol is needed between the End Device(e.g. Hypervisor) making use of the external NVE and the external NVE in order to make the external NVE aware of the changing VN membership requirements of the Tenant Systems within the End Device.

A key driver for using a protocol rather than using static configuration of the external NVE is because the VN connectivity requirements can change frequently as VMs are brought up, moved and brought down on various hypervisors throughout the data center or external cloud.

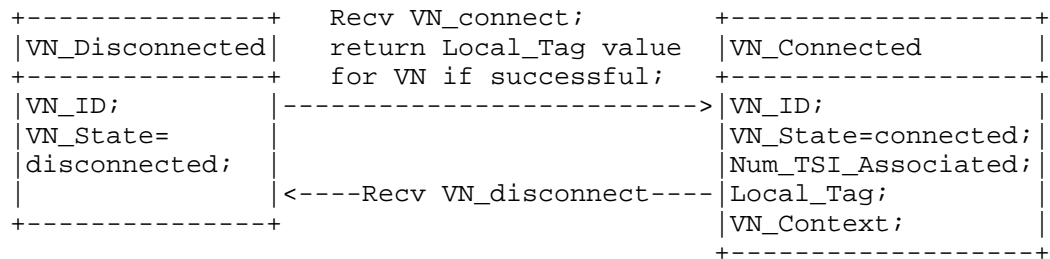


Figure 5 State Transition Example of a VAP Instance
on an External NVE

Figure 5 shows the state transition for a VAP on the external NVE. An NVE that supports the hypervisor to NVE control plane protocol should support one instance of the state machine for each active VN. The state transition on the external NVE is normally triggered by the hypervisor-facing side events and behaviors. Some of the interleaved interaction between NVE and NVA will be illustrated for better understanding of the whole procedure; while others of them may not be shown. More detailed information regarding that is available in [I-D.ietf-nvo3-nve-nva-cp-req].

The external NVE must be notified when an End Device requires connection to a particular VN and when it no longer requires connection. In addition, the external NVE must provide a local tag value for each connected VN to the End Device to use for exchange of packets between the End Device and the external NVE (e.g. a locally significant 802.1Q tag value). How "local" the significance is depends on whether the Hypervisor has a direct physical connection to

the external NVE (in which case the significance is local to the physical link), or whether there is an Ethernet switch (e.g. a blade switch) connecting the Hypervisor to the NVE (in which case the significance is local to the intervening switch and all the links connected to it).

These VLAN tags are used to differentiate between different VNs as packets cross the shared access network to the external NVE. When the external NVE receives packets, it uses the VLAN tag to identify the VN of packets coming from a given TSI, strips the tag, and adds the appropriate overlay encapsulation for that VN and sends it towards the corresponding remote NVE across the underlying IP network.

The Identification of the VN in this protocol could either be through a VN Name or a VN ID. A globally unique VN Name facilitates portability of a Tenant's Virtual Data Center. Once an external NVE receives a VN connect indication, the NVE needs a way to get a VN Context allocated (or receive the already allocated VN Context) for a given VN Name or ID (as well as any other information needed to transmit encapsulated packets). How this is done is the subject of the NVE-to-NVA protocol which are part of work items 1 and 2 in [RFC7364].

VN_connect message can be explicit or implicit. Explicit means the hypervisor sending a message explicitly to request for the connection to a VN. Implicit means the external NVE receives other messages, e.g. very first TSI associate message (see the next subsection) for a given VN, to implicitly indicate its interest to connect to a VN.

A VN_disconnect message will indicate that the NVE can release all the resources for that disconnected VN and transit to VN_disconnected state. The local tag assigned for that VN can possibly be reclaimed by other VN.

3.2 TSI Associate and Activate

Typically, a TSI is assigned a single MAC address and all frames transmitted and received on that TSI use that single MAC address. As mentioned earlier, it is also possible for a Tenant System to exchange frames using multiple MAC addresses or packets with multiple IP addresses.

Particularly in the case of a TS that is forwarding frames or packets from other TSs, the external NVE will need to communicate the mapping between the NVE's IP address (on the underlying network) and ALL the addresses the TS is forwarding on behalf of for the corresponding VN to the NVA.

The NVE has two ways in which it can discover the tenant addresses for which frames must be forwarded to a given End Device (and ultimately to the TS within that End Device).

1. It can glean the addresses by inspecting the source addresses in packets it receives from the End Device.
2. The hypervisor can explicitly signal the address associations of a TSI to the external NVE. The address association includes all the MAC and/or IP addresses possibly used as source addresses in a packet sent from the hypervisor to external NVE. The external NVE may further use this information to filter the future traffic from the hypervisor.

To perform the second approach above, the "hypervisor-to-NVE" protocol requires a means to allow End Devices to communicate new tenant addresses associations for a given TSI within a given VN.

Figure 6 shows the example of a state transition for a TSI connecting to a VAP on the external NVE. An NVE that supports the hypervisor to NVE control plane protocol may support one instance of the state machine for each TSI connecting to a given VN.

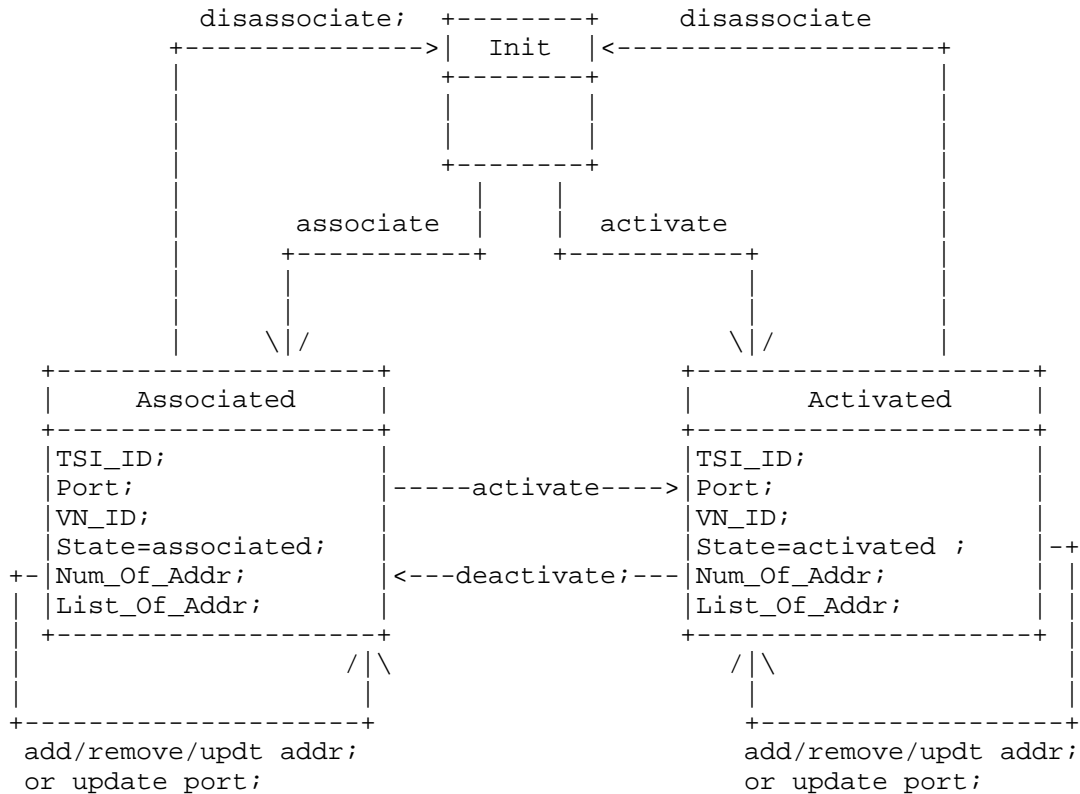


Figure 6 State Transition Example of a TSI Instance
on an External NVE

Associated state of a TSI instance on an external NVE indicates all the addresses for that TSI have already associated with the VAP of the external NVE on port p for a given VN but no real traffic to and from the TSI is expected and allowed to pass through. An NVE has reserved all the necessary resources for that TSI. An external NVE may report the mappings of its' underlay IP address and the associated TSI addresses to NVA and relevant network nodes may save such information to its mapping table but not forwarding table. A NVE may create ACL or filter rules based on the associated TSI addresses on the attached port p but not enable them yet. Local tag for the VN corresponding to the TSI instance should be provisioned on port p to receive packets.

VM migration event (discussed section 2) may cause the hypervisor to send an associate message to the NVE connected to the destination hypervisor the VM migrates to. VM creation event may also lead to the

same practice.

The Activated state of a TSI instance on an external NVE indicates that all the addresses for that TSI functioning correctly on port p and traffic can be received from and sent to that TSI via the NVE. The mappings of the NVE's underlay IP address and the associated TSI addresses should be put into the forwarding table rather than the mapping table on relevant network nodes. ACL or filter rules based on the associated TSI addresses on the attached port p in NVE are enabled. Local tag for the VN corresponding to the TSI instance MUST be provisioned on port p to receive packets.

The Activate message makes the state transit from Init or Associated to Activated. VM creation, VM migration and VM resumption events discussed in section 4 may trigger the Activate message to be sent from the hypervisor to the external NVE.

TSI information may get updated either in Associated or Activated state. The following are considered updates to the TSI information: add or remove the associated addresses, update current associated addresses (for example updating IP for a given MAC), update NVE port information based on where the NVE receives messages. Such updates do not change the state of TSI. When any address associated to a given TSI changes, the NVE should inform the NVA to update the mapping information on NVE's underlying address and the associated TSI addresses. The NVE should also change its local ACL or filter settings accordingly for the relevant addresses. Port information update will cause the local tag for the VN corresponding to the TSI instance to be provisioned on new port p and removed from the old port.

3.3 TSI Disassociate and Deactivate

Disassociate and deactivate conceptually are the reverse behaviors of associate and activate. From Activated state to Associated state, the external NVE needs to make sure the resources are still reserved but the addresses associated to the TSI are not functioning and no traffic to and from the TSI is expected and allowed to pass through. For example, the NVE needs to inform the NVA to remove the relevant addresses mapping information from forwarding or routing table. ACL or filtering rules regarding the relevant addresses should be disabled. From Associated or Activated state to the Init state, the NVE will release all the resources relevant to TSI instances. The NVE should also inform the NVA to remove the relevant entries from mapping table. ACL or filtering rules regarding the relevant addresses should be removed. Local tag provisioning on the connecting port on NVE should be cleared.

A VM suspension event (discussed in section 2) may cause the relevant TSI instance(s) on the NVE to transit from Activated to Associated state. A VM pause event normally does not affect the state of the relevant TSI instance(s) on the NVE as the VM is expected to run again soon. The VM shutdown event will normally cause the relevant TSI instance(s) on NVE transit to Init state from Activated state. All resources should be released.

A VM migration will lead the TSI instance on the source NVE to leave Activated state. When a VM migrates to another hypervisor connecting to the same NVE, i.e. source and destination NVE are the same, NVE should use TSI_ID and incoming port to differentiate two TSI instance.

Although the triggering messages for state transition shown in Figure 6 does not indicate the difference between VM creation/shutdown event and VM migration arrival/departure event, the external NVE can make optimizations if it is notified of such information. For example, if the NVE knows the incoming activate message is caused by migration rather than VM creation, some mechanisms may be employed or triggered to make sure the dynamic configurations or provisionings on the destination NVE are the same as those on the source NVE for the migrated VM. For example IGMP query [RFC2236] can be triggered by the destination external NVE to the migrated VM on destination hypervisor so that the VM is forced to answer an IGMP report to the multicast router. Then multicast router can correctly send the multicast traffic to the new external NVE for those multicast groups the VM had joined before the migration.

4. Hypervisor-to-NVE Control Plane Protocol Requirements

Req-1: The protocol MUST support a bridged network connecting End Devices to External NVE.

Req-2: The protocol MUST support multiple End Devices sharing the same External NVE via the same physical port across a bridged network.

Req-3: The protocol MAY support an End Device using multiple external NVEs simultaneously, but only one external NVE for each VN.

Req-4: The protocol MAY support an End Device using multiple external NVEs simultaneously for the same VN.

Req-5: The protocol MUST allow the End Device initiating a request to its associated External NVE to be connected/disconnected to a given VN.

Req-6: The protocol MUST allow an External NVE initiating a request to its connected End Devices to be disconnected to a given VN.

Req-7: When a TS attaches to a VN, the protocol MUST allow for an End Device and its external NVE to negotiate one or more locally-significant tag(s) for carrying traffic associated with a specific VN (e.g., 802.1Q tags).

Req-8: The protocol MUST allow an End Device initiating a request to associate/disassociate and/or activate/deactive address(es) of a TSI instance to a VN on an NVE port.

Req-9: The protocol MUST allow the External NVE initiating a request to disassociate and/or deactivate address(es) of a TSI instance to a VN on an NVE port.

Req-10: The protocol MUST allow an End Device initiating a request to add, remove or update address(es) associated with a TSI instance on the external NVE. Addresses can be expressed in different formats, for example, MAC, IP or pair of IP and MAC.

Req-11: The protocol MUST allow the External NVE to authenticate the End Device connected.

Req-12: The protocol MUST be able to run over L2 links between the End Device and its External NVE.

Req-13: The protocol SHOULD support the End Device indicating if an associate or activate request from it results from a VM hot migration event.

5. VDP Applicability and Enhancement Needs

Virtual Station Interface (VSI) Discovery and Configuration Protocol (VDP) [IEEE 802.1Qbg] can be the control plane protocol running between the hypervisor and the external NVE. Appendix A illustrates VDP for reader's information.

VDP facilitates the automatic discovery and configuration for Edge Virtual Bridging (EVB) station and Edge Virtual Bridging (EVB) bridge. EVB station is normally an end station running multiple VMs. It is conceptually equivalent to hypervisor in this document. And EVB bridge is conceptually equivalent to the external NVE.

VDP is able to pre-associate/associate/de-associate a VSI on EVB station to a port on the EVB bridge. VSI is approximately the concept

of a virtual port a VM connects to the hypervisor in this document context. The EVB station and the EVB bridge can reach the agreement on VLAN ID(s) assigned to a VSI via VDP message exchange. Other configuration parameters can be exchanged via VDP as well. VDP is carried over Edge Control Protocol(ECP) [IEEE8021Qbg] which provides a reliable transportation over a layer 2 network.

VDP protocol needs some extensions to fulfill the requirements listed in this document. Table 1 shows the needed extensions and/or clarifications in NVO3 context.

Req	VDP supported?	remarks
Req-1	Partially	Needs extension. Must be able to send to a specific unicast MAC and should be able to send to a non-reserved well known multicast address other than the nearest customer bridge address
Req-2		
Req-3		
Req-4		
Req-5	Yes	VN is indicated by GroupID
Req-6	Yes	Bridge sends De-Associate
Req-7	Yes	VID=NULL in request and bridge returns the assigned value in response or specify GroupID in request and get VID assigned in returning response. Multiple VLANs per group is allowed
Req-8	Partially	requirements
		associate/disassociate
		activate/deactivate
Req-9	Yes	VDP bridge initiates de-associate
Req-10	Partially	Needs extension for IPv4/IPv6 address. Add a new "filter info format" type
Req-11	No	Out-of-band mechanism is preferred, e.g. MACSec or 802.1x.

Req-12	Yes	L2 protocol naturally
Req-13	Partially	M bit for migrated VM on destination hypervisor and S bit for that on source hypervisor. It is indistinguishable when M/S is 0 between no guidance and events not caused by migration where NVE may act differently. Needs new New bits for migration indication in new "filter info format" type

Table 1 Compare VDP with the requirements

Simply adding the ability to carry layer 3 addresses, VDP can serve the Hypervisor-to-NVE control plane functions pretty well. Other extensions are the improvement of the protocol capabilities for better fit in NVO3 network.

6. Security Considerations

NVEs must ensure that only properly authorized Tenant Systems are allowed to join and become a part of any specific Virtual Network. In addition, NVEs will need appropriate mechanisms to ensure that any hypervisor wishing to use the services of an NVE are properly authorized to do so. One design point is whether the hypervisor should supply the NVE with necessary information (e.g., VM addresses, VN information, or other parameters) that the NVE uses directly, or whether the hypervisor should only supply a VN ID and an identifier for the associated VM (e.g., its MAC address), with the NVE using that information to obtain the information needed to validate the hypervisor-provided parameters or obtain related parameters in a secure manner.

7. IANA Considerations

No IANA action is required. RFC Editor: please delete this section before publication.

8. Acknowledgements

This document was initiated and merged from the drafts draft-kreeger-nvo3-hypervisor-nve-cp, draft-gu-nvo3-tes-nve-mechanism and draft-kompella-nvo3-server2nve. Thanks to all the co-authors and contributing members of those drafts.

The authors would like to specially thank Jon Hudson for his generous

help in improving the readability of this document.

8. References

8.1 Normative References

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

8.2 Informative References

- [RFC7364] Narten, T., Gray, E., Black, D., Fang, L., Kreeger, L., and M. Napierala, "Problem Statement: Overlays for Network Virtualization", October 2014.
- [RFC7365] Lasserre, M., Balus, F., Morin, T., Bitar, N., and Y. Rekhter, "Framework for DC Network Virtualization", October 2014.
- [I-D.ietf-nvo3-nve-nva-cp-req] Kreeger, L., Dutt, D., Narten, T., and D. Black, "Network Virtualization NVE to NVA Control Protocol Requirements", draft-ietf-nvo3-nve-nva-cp-req-01 (work in progress), October 2013.
- [I-D.ietf-nvo3-arch] Black, D., Narten, T., et al, "An Architecture for Overlay Networks (NVO3)", draft-narten-nvo3-arch, work in progress.
- [I-D.ietf-opsawg-vmm-mib] Asai H., MacFaden M., Schoenwaelder J., Shima K., Tsou T., "Management Information Base for Virtual Machines Controlled by a Hypervisor", draft-ietf-opsawg-vmm-mib-00 (work in progress), February 2014.
- [IEEE 802.1Qbg] IEEE, "Media Access Control (MAC) Bridges and Virtual Bridged Local Area Networks - Amendment 21: Edge Virtual Bridging", IEEE Std 802.1Qbg, 2012
- [8021Q] IEEE, "Media Access Control (MAC) Bridges and Virtual Bridged Local Area Networks", IEEE Std 802.1Q-2011, August, 2011

Appendix A. IEEE 802.1Qbg VDP Illustration (For information only)

VDP has the format shown in Figure A.1. Virtual Station Interface (VSI) is an interface to a virtual station that is attached to a downlink port

of an internal bridging function in server. VSI's VDP packet will be handled by an external bridge. VDP is the controlling protocol running between the hypervisor and the external bridge.

TLV type	TLV info	Status	VSI	VSI	VSIID	VSIID	Filter	Filter Info
7b	str len 9b	1oct	Type ID 3oct	Type Ver 1oct	Format 1oct	16oct	Info format 1oct	M oct
<---TLV header--->			<---VSI type&instance---> <-----VSI attributes----->					
<---TLV header--->			<-----TLV info string = 23 + M octets----->					

Figure A.1: VDP TLV definitions

There are basically four TLV types.

1. Pre-Associate: Pre-Associate is used to pre-associate a VSI instance with a bridge port. The bridge validates the request and returns a failure Status in case of errors. Successful pre-association does not imply that the indicated VSI Type or provisioning will be applied to any traffic flowing through the VSI. The pre-associate enables faster response to an associate, by allowing the bridge to obtain the VSI Type prior to an association.

2. Pre-Associate with resource reservation: Pre-Associate with Resource Reservation involves the same steps as Pre-Associate, but on successful pre-association also reserves resources in the Bridge to prepare for a subsequent Associate request.

3. Associate: The Associate creates and activates an association between a VSI instance and a bridge port. The Bridge allocates any required bridge resources for the referenced VSI. The Bridge activates the configuration for the VSI Type ID. This association is then applied to the traffic flow to/from the VSI instance.

4. Deassociate: The de-associate is used to remove an association between a VSI instance and a bridge port. Pre-Associated and Associated VSIs can be de-associated. De-associate releases any resources that were reserved as a result of prior Associate or Pre-Associate operations for that VSI instance.

Deassociate can be initiated by either side and the rest types of messages can only be initiated by the server side.

Some important flag values in VDP Status field:

1. M-bit (Bit 5): Indicates that the user of the VSI (e.g., the VM) is migrating (M-bit = 1) or provides no guidance on the migration of the user of the VSI (M-bit = 0). The M-bit is used as an indicator relative to the VSI that the user is migrating to.

2. S-bit (Bit 6): Indicates that the VSI user (e.g., the VM) is suspended (S-bit = 1) or provides no guidance as to whether the user of the VSI is suspended (S-bit = 0). A keep-alive Associate request with S-bit = 1 can be sent when the VSI user is suspended. The S-bit is used as an indicator relative to the VSI that the user is migrating from.

The filter information format currently supports 4 types as the following.

1. VID Filter Info format

#of entries (2octets)	PS (1bit)	PCP (3bits)	VID (12bits)
<--Repeated per entry-->			

Figure A.2 VID Filter Info format

2. MAC/VID filter format

#of entries (2octets)	MAC address (6 octets)	PS (1bit)	PCP (3bits)	VID (12bits)
<-----Repeated per entry----->				

Figure A.3 MAC/VID filter format

3. GroupID/VID filter format

#of entries (2octets)	GroupID (4 octets)	PS (1bit)	PCP (3bits)	VID (12bits)
<-----Repeated per entry----->				

Figure A.4 GroupID/VID filter format

4. GroupID/MAC/VID filter format

#of entries (2 octets)	GroupID (4 octets)	MAC address (6 octets)	PS (1bit)	PCP (3b)	VID (12bits)
<-----Repeated per entry----->					

Figure A.5 GroupID/MAC/VID filter format

The null VID can be used in the VDP Request sent from the hypervisor to the external bridge. Use of the null VID indicates that the set of VID values associated with the VSI is expected to be supplied by the Bridge. The Bridge can obtain VID values from the VSI Type whose identity is specified by the VSI Type information in the VDP Request. The set of VID values is returned to the station via the VDP Response. The returned VID value can be a locally significant value. When GroupID is used, it is equivalent to the VN ID in NVO3. GroupID will be provided by the hypervisor to the bridge. The bridge will map GroupID to a locally significant VLAN ID.

The VSIID in VDP request that identify a VM can be one of the following format: IPV4 address, IPV6 address, MAC address, UUID or locally defined.

Authors' Addresses

Yizhou Li
Huawei Technologies
101 Software Avenue,
Nanjing 210012
China

Phone: +86-25-56625409
EMail: liyizhou@huawei.com

Lucy Yong
Huawei Technologies, USA

Email: lucy.yong@huawei.com

Lawrence Kreeger
Cisco

Email: kreeger@cisco.com

Thomas Narten
IBM

Email: narten@us.ibm.com
David Black
EMC

Email: david.black@emc.com

NVO3
Internet-Draft
Intended status: Standards Track
Expires: December 29, 2017

D. Migault
June 27, 2017

Geneve Header Authentication Option (GAO)
draft-mglt-nvo3-geneve-authentication-option-00

Abstract

This document describes the Geneve Header Authentication Option (GAO). This option enables a Geneve element to authenticate the Geneve Header with selected associated Geneve Options as well as a portion of the Geneve Payload.

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 <http://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 29, 2017.

Copyright Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Requirements notation	2
2. Introduction	2
3. Position versus DTLS/IPsec	3
4. Scope of the GAO	5
5. Terminology	6
6. GAO Description	6
7. GAO Processing	7
7.1. GAO Placement	7
7.2. GSA Parameters	8
7.3. GAO Outbound Processing	10
7.3.1. Generating the Sequence Number	10
7.3.2. Generating a Covered Length	11
7.3.3. GAO Inbound Processing	12
8. IANA Considerations	15
9. Security Considerations	15
10. Acknowledgment	15
11. References	15
11.1. Normative References	15
11.2. Informative References	16
Author's Address	16

1. Requirements notation

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

2. Introduction

Geneve [I-D.ietf-nvo3-geneve] defines an overlay network that enables communications between tenants within a given virtual network. The Geneve overlay network enables these tenants to be distributed over a data center or multiple data centers. As multiple virtual networks share a common infrastructure, Geneve needs to isolate both communications between virtual networks as well as each virtual network address space [RFC7364].

The Geneve Header indicates the virtual network a communication belongs to with a Virtual Network Identifier (VNI). Geneve packets may be steered to the appropriated destination tenant through the destination switch based on that NVI value. In addition to the NVI, the Geneve Header may carry additional metadata that impacts how the traffic could be steered to the destination tenant.

As stated by [I-D.ietf-nvo3-encap] and [I-D.mglt-nvo3-security-requirements], it is crucial that the

information of the Geneve Header remains protected and authenticated in order to prevent that traffic be delivered to the wrong tenant. Typically, without integrity check mechanisms, a one bit switch in the NVI results in such a wrong delivery. Such vulnerability is further increased by the use of UDP encapsulation that makes any application able to spoof packets.

This document addresses these issues by proposing a GAO which enables to authenticate the Geneve Header with a set of selected Geneve Options as well as a portion of inner packet (Geneve Payload) carried by the Geneve overlay network (Geneve Payload). In addition, GAO also prevent a Geneve Packet to be replayed by introducing an anti-replay mechanism. GAO does not provides encryption which is instead provided by [I-D.mglt-nvo3-geneve-encryption-option].

3. Position versus DTLS/IPsec

This section exposes the motivations for designing GAO rather than re-using existing security mechanisms such as DTLS or IPsec.

GAO provides integrity protection of a Geneve Packet, i.e. the Geneve Header, including a set of Geneve Options as well as a portion of the Geneve Payload.

As Geneve is encapsulated in UDP packet, DTLS is a natural candidate. Similarly IPsec/AH [RFC4302] defines an protocol to authenticate an IP packet. However relying on DTLS (or IPsec)/AH instead of a specific extension designed for Geneve comes with the following drawbacks:

- o Modern versions of DTLS [I-D.ietf-tls-dtls13] currently do not consider authentication-only. Instead the traffic is always encrypted. Encrypting the Geneve Header prevents on path Geneve elements to manage secured intra NVI communications. Typically when multiple intra NVI communications are multiplexed into a DTLS tunnel, a Geneve on path element will not be able to re-route some traffic nor to appropriately prioritize flows or load balance them according to their NVI. On the other hand, DTLS1.2 [RFC6347] enabled authentication-only protection, and further cipher suite could be defined for DTLS1.3 in case there were a significant advantage in using DTLS to secure the Geneve communications. In case such cipher will not be available, currently defined end to end encryption would prevent providing information useful to manage the various intra-NVI communications. This information might be carried by lower layer such as UDP using port number for example. However, such alternatives clearly makes secure intra NVI communication unnecessarily too hard to manage, and so does

not encourage a secure deployment of these communications.
Typically:

- * Management of secure Geneve communications are reduces to management of UDP tunnel which ignores all motivations for designing Geneve. That is the ability to tag flows, as well as to carry states or metadata.
 - * Management complexity is increased with an additional binding between Geneve Header and port number for example. Not only a new binding is introduced, but as Geneve Headers and UDP source ports / destination ports have different spaces ranges, this makes such correspondence not straight forward to manage. Typically NVI are 24 bit long while source port are 16 bit long, this means that additional destination port may be used in order to benefit from the full NVI space.
 - * Increases the number of tunnels and the number of keying material as different Geneve Header needs to be transported in different UDP tunnel. The number of UDP tunnels may reach the number of different Geneve communications.
- o DTLS comes with a key exchange agreement, included as part of the DTLS protocol. In most cases, DTLS or TLS is used without any configurations by a (D)TLS client while the (D)TLS server has all the necessary authentication information, so the (D)TLS client can appropriately authenticate the (D)TLS server. In this case, for end-to-end authentication, authentication is performed by both Geneve NVEs which requires all of them to be appropriately provisioned with the necessary authentication credentials. Management of these authentication credential is not trivial and is expected to handled in addition of the security policies. In addition, the presence of such handshake protocol may introduce some latencies in a forwarding plan usually managed by a orchestrator. As a result, if DTLS would be used, a variant of DTLS without key exchange may rather be considered.
 - o Geneve does not provide any standard way to inform whether a packet is authenticated or not. The current assigned port number for Geneve is 6081. In order to make possible for the receiving node to distinguish an unprotected Geneve Packet from a protected traffic, a new port should be defined.
 - o The current use of TLS is usually based on a TLS client wishing to access a resource using TLS. In that case, the TLS client uses a specific port number. A server may also redirect the requests from a client that is non protected to a specific port which defines the protected version of that service. Such redirection

is usually performed when the service defines that resource has to be accessed using a secure channel. In addition, the redirection is performed by the application protocol. As a result, the security policies are usually quite simple that is, 1) security initiated by the client or 2) server enforces that all requested are secured. The case of Geneve overlay network considers instead the coexistence of protected and non protected traffic which would require some mechanisms to define and enforce security policies not yet part of DTLS.

- o DTLS usually protects the whole UDP payload. In our case, the protection of the Geneve Header only, for example, would require some further developments to the existing DTLS.
- o IPsec/AH prevents the creation of the Geneve overlay network. IPsec/AH has been defined for end-to-end IP communications. In the case of a Geneve Packet, the two ends are defined by the IP addresses of the Geneve Packet Outer IP Headers. These IP addresses are not necessarily the Geneve NVE, and could instead be those of an Geneve element that belong to the Geneve overlay network and in charge of steering the traffic to another Geneve overlay element. With IPsec/AH, the IP addresses could not be modified, and the Geneve Packet will not be able to be steered across the Geneve overlay network. In this case IPsec/AH could be used for a hop-by-hop security. This would require each node of the Geneve Overlay network to be provisioned appropriately with the IPsec material which would come with significant management issues. In addition, this would not achieve a end-to-end security between the two ends of the Geneve tunnel.
- o IPsec/ESP may also be used without encryption. However, in this case, the port number would be protected, which would prevent Geneve element to redirect the traffic to a different Geneve element using a different port. Such constraint may prevent the overlay network to be operated as an overlay network, that is any on path Geneve element is able to redirect the traffic to another Geneve element that belongs to the overlay network.

4. Scope of the GAO

The Geneve Header Authentication Option (GAO) expects to have the following properties:

- o Provides means to authenticate the Geneve Header, a selected associated set of options as well as part of the Geneve Payload.
- o Provides an anti-replay mechanism. This option does not encrypt any data and as such does not provide any privacy. When privacy

is expected, it can be enforced by the Geneve overlay network using GEO ([I-D.mglt-nvo3-geneve-encryption-option]) as well as by the Tenant's System which may encrypt their communications using IPsec/ESP or TLS. The main purpose of the GAO is to provide means for the infrastructure to ensure that Geneve communications cannot be injected for example by modifying the NVI.

- o Provides authentication - at least in an orchestrated environment - to the two NVEs, but also to any appropriately configured on-path Geneve forwarding element.
- o Provides read access to the Geneve Header for any Geneve on path elements. This option is expected to enable Geneve communications to be secured, while benefiting from all the facilities provided by Geneve.
- o Provide the ability for on path Geneve forwarding elements to add Geneve Options on Geneve authenticated Packets without invalidating the GAO.
- o Provides means with some restrictions for an on-path element to add Geneve Option and authenticate that Geneve Option using a GAO.

5. Terminology

The terminology used in this document has been introduced in [I-D.mglt-nvo3-geneve-security-architecture].

6. GAO Description

For generic format of the Geneve Options is defined in Figure 1. The following values are expected:

- o Option Class: 0x0000
- o Type: C is unset as the GAO can simply be ignored by a NVE or a transit node. The GSP will prevent to accept a GOA that is mandated by the GSP and that has not been validated.
- o R is set to 0.
- o Length: This document only considers the algorithms recommended by [I-D.ietf-ipsecme-rfc7321bis] AUTH_HMAC_SHA2_256_128 and AUTH_HMAC_SHA2_512_256. These algorithms are defined in [RFC4868] with a respective 16 and 32 byte ICV. As a result, the option length is expected to $4 + 28 = 32$ bytes (resp. $4 + 44 = 48$ bytes) which leads to 8 or 12 as the possible values for Length.

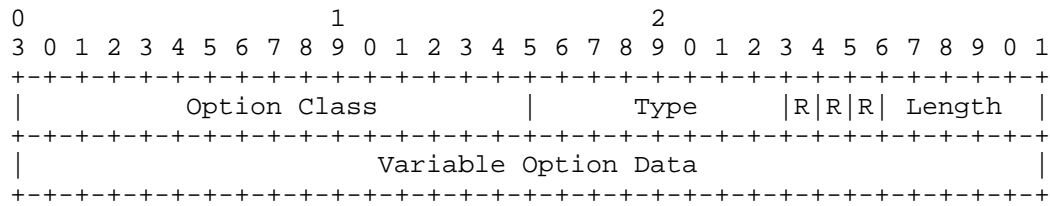


Figure 1: Geneve Option Format

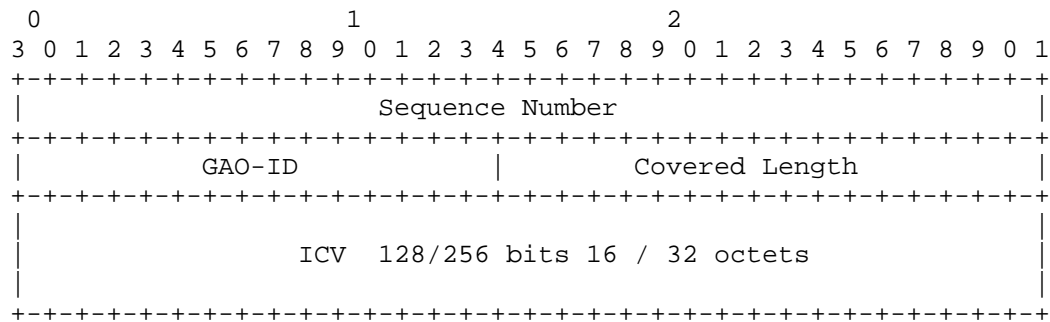


Figure 2: Geneve Authentication Data

- o Sequence Number (32 bits): indicates the Sequence number of the Geneve Header. When the SN is 32 bit long, the whole SN is indicated. When the SN is 64 bit long, only the 32 least significant bits are indicated.
- o GAO-ID (16 bits): indicates the identifier of the GAO. This identifier is useful to retrieve the GSA, with the necessary information to compute the GAO or to validate it.
- o Covered Length (16 bits): indicates in number of bytes following the GAO that are covered by the authentication.
- o ICV contains the HMAC value.

7. GAO Processing

7.1. GAO Placement

A GAO option covers the Geneve Header, the Geneve Options following the GAO as well as the Covered Length appended to the GAO. As a result, any on path (Geneve) element MUST leave the Geneve Fixed Header and the first Covered Length bytes after GAO unchanged.

GAO does not covers the Geneve Options placed between the Geneve Fixed Header and the GAO. In addition, GAO does not cover the bytes located after the Covered Length.

Geneve Options that are expected to be updated by any Geneve forwarding elements MUST be located between the Geneve Fixed Header and the existing GAO.

When a Geneve Packet is received by a Geneve forwarding elements and that element is expected to insert an additional Geneve Option, the Geneve forwarding element MUST NOT insert the Geneve Option in a area covered by a GAO. A safe way to proceed is that Geneve forwarding element that do not understand GAO MUST insert new Geneve Option right after the Geneve Fixed Header. This will result in having the Geneve Option before the existing GAO. When the Geneve forwarding element understand GAO it can consider the covered area by each GAO and place its new option in a non covered area.

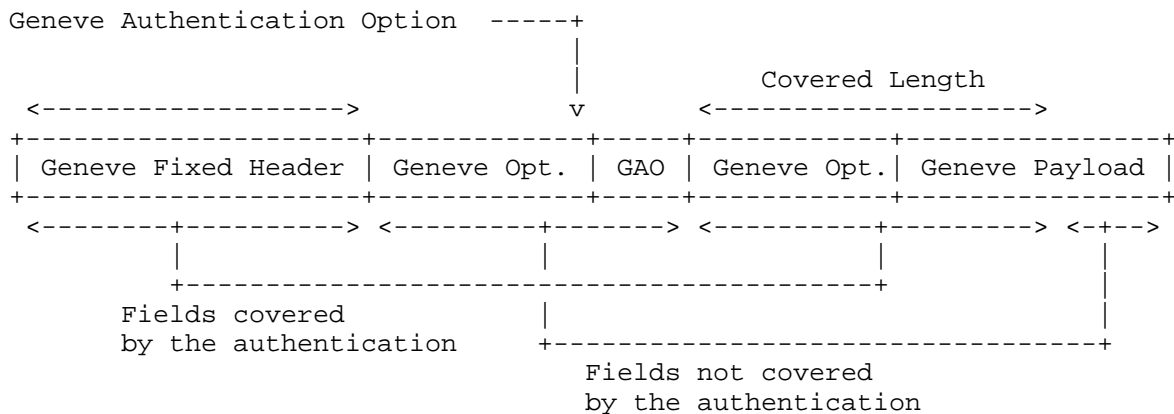


Figure 3: Geneve Authentication Options Placement

7.2. GSA Parameters

This section describes the parameters of the GAS necessary to compute or validate the GAO. These parameters are then latter used to described the processing.

- o GSO ID: The identifier of that GAO. This identifier is used to bind the GAO with the appropriated GSA. It is expected that GSA are uniquely identified on the receiver side. In case collision are supported, the implementation MUST be able to deterministically associate the GAO to the appropriated GSA for example by using IP addresses and UDP ports.

- o GSO Protocol: The security protocol associated with the Geneve Security Option. In our case the protocol is GAO.
- o GSO Authentication Algorithm: This document follows recommendations provided by [I-D.ietf-ipsecme-rfc7321bis] which recommends AUTH_HMAC_SHA2_256_128 and AUTH_HMAC_SHA2_512_256 defined in [RFC4868].
- o GSO Payload Covered Length: the length of the Geneve Payload covered by GAO. The expression of the length can be a number of bytes, but it may also be defined with an abstract designation. For example, a sending node may be willing to authenticate the Geneve Payload up to the ESP layer. In that case, the sending node will have to compute the corresponding Payload Covered Length. This value is only used by the sending node. The receiving node read that value from the GAO.
- o GSO Covered Geneve Options: Indicates the Geneve Options covered by the GAO. This indication is primarily necessary for the sending node and is derived from the Geneve Packet by the receiving node. It MUST be checked by the receiving node to validate the GSA. It might typically be expressed as a list of Geneve Options that needs to be covered by the authentication.
- o GSO Authentication Key: the shared secret necessary compute and validate the HMAC value generated by the Authentication Algorithm specified above.

In order to implement the anti replay mechanisms the following parameters are provided:

- o GSO Sequence Number Size: indicates the size of the SN. This document considers a 32 bit or a 64 bit length.
- o GSO Sequence Number: that designates the Sequence Number last sent or received packet.
- o GSO Anti Replay Window: that indicates the windows that defines out-of order packet or late packets versus a replayed Geneve Packet. Any Geneve Packet with a lower SN than GSO Sequence Number - Anti Replay Windows MUST be rejected.

In order to check the conformity with the GSP:

- o Selectors: The selectors are provided so the receiver can check the Geneve Packet protected by the GSO is conform to the GSP. In other words a valid GSO is not sufficient for the Geneve Packet to be forwarded to the upper layers. Note that the Selectors MUST

match the Geneve Packet associated to the GSA before the GSO is built for outbound Packets. For inbound Geneve Packet the Selectors are those that correspond to the Geneve Packet after the GSO has been validated/decrypted. Selectors are mostly expected to be used by the GSA for incoming Geneve Packet, in order to check the GSA is conform with its GSP.

- o GSA Life time:

7.3. GAO Outbound Processing

Upon receiving a Geneve Packet, the Geneve Security Module performs a GSP DB look up to determine if any security action is required. If the security action is DISCARD, the Geneve Packet is discarded. If the security action is BYPASS, the Geneve Packet is sent to the lower layers for the outer encapsulation without any additional security consideration. If the action is SECURE, the GSP returns the list of GSAs that need to be applied. The list is an ordered list, and the Geneve Security Module performs these GSAs in the received order. (See [I-D.mglt-nvo3-geneve-security-architecture] for more information.)

When a list of GSAs is provided, it is crucial that the implementation updates the Selectors of the further GSAs according to the actions undertaken by the previous GSAs. In most cases, a GSA results in the addition of GSO. The Selectors of the next GSA MUST consider this new GSO, in the Selectors.

The outbound processing consists in the following actions:

1. Generating the Sequence Number
2. Generating a Covered Length
3. Generating the ICV
4. Building the GAO
5. Building the output Geneve Packet

7.3.1. Generating the Sequence Number

The Sequence Number is used to prevent anti replay. The Sequence Number is any number strictly greater than the current value of the GSO Sequence Number mentioned in the GSA.

The size of the GSO Sequence Number is designated by the GSO Sequence Number Size. The GSO Sequence Number can be a 32 bit or 64 bit

number. When the limit or the GSO Sequence number has been reached, the GSA MUST be renewed. In other words, no re-initialization nor rolling mechanisms are expected for the GSO Sequence Number. The Geneve Elements need to take the necessary actions in order to generate GSAs before the limit of the GSO Sequence Number is reached.

The new value of the GSO Sequence Number replaces the former GSO Sequence Number in the GSA.

7.3.2. Generating a Covered Length

The Covered Length describes the number of bytes of the Geneve Packet that are located after the GAO and authenticated by GAO.

The Covered Length includes Geneve Options that are covered by the authentication designated by the GSO Covered Geneve Options as well as a portion of the Geneve Payload designated by the GSO Payload Covered Length.

The covered Geneve Options MUST be immutable, and any on-path Geneve element MUST NOT change any of the Geneve Options covered by GAO. The covered Options MAY be agreed between the two Geneve element, however, by default, it is expected that the sending node will include any immutable Geneve Option. The agreement of the covered Geneve Options is not necessary to validate the GAO. In fact the position of the GAO in the Geneve Packet indicates deterministically the covered Geneve Options. However, Geneve Options that are immutable while not being covered by the GAO will be considered suspicious and as such SHOULD be rejected by the Geneve Security Module of the receiving node. This Geneve Option could have been inserted as well as modified. Of course some Geneve Security Module MAY also specify a list of immutable Geneve Option that are not expected to be covered. In that case such options MUST NOT be removed by the Geneve Security Module.

Overall, the covered Geneve Options is determined by the sending node. In addition that Geneve Options may have varying size, the contribution of the Covered Length is likely to vary for each Geneve Packet.

Similarly, the contribution of the Covered Length by the Geneve Payload is also likely to vary for each Geneve Packet. More specifically, it is more likely that a GSA defines the layers of the Geneve Payload that needs to be authenticated instead of a number of bytes. For example, a GSA may indicate that the Geneve Payload may be covered up to the ESP or (D)TLS layer. In addition, the GSA may also indicate an upper bound value for the Covered Length which could be imposed by hardware or computing restrictions. As a result, the

contribution of the Geneve Payload is determined by the sending node and evaluated for each Geneve Packet.

7.3.2.1. Generating the ICV

The ICV results from applying the GSO Authentication Algorithm with the GSO Authentication Key to the appropriated data.

The appropriated data is build by concatenating the initial string "geneve authentication option" with the Geneve Fixed Header, the GSO Sequence Number, the GSO-ID, the GSO Covered Length, the covered Geneve options as well as the covered part of the Geneve Payload.

All fields of the Geneve Fixed Header are considered, including the Rsv and Reserved fields. It is important to understand that these fields are expected to remain immutable fields.

7.3.2.2. Building the GAO

The GAO is built by concatenating the 32 least significant bits of the GSO Sequence Number, the GAO-ID, the Covered Length and the generated ICV.

7.3.2.3. Building the output Geneve Packet

The GAO is placed before all covered Geneve Options, followed by the Geneve Payload. A Geneve Option that is not covered by the GAO MUST NOT be placed after the GAO. The Geneve Options covered by the GAO MUST remain in the same order as the order considered for generating the ICV. A Geneve Option covered by the GAO MUST NOT be located before the GAO. In addition, a Geneve Element MUST NOT change any bit located after the GAO that is covered by the GAO.

The generated Geneve Packet is then forwarded to the Outer Tunnel encapsulation.

7.3.3. GAO Inbound Processing

Upon receiving a Geneve Packet, the receiving Geneve element determines the Geneve Packet is neither associated with a DISCARD nor with a BYPASS policy, and as such is expected to be SECURED - see [I-D.mglt-nvo3-geneve-security-architecture].

When the Geneve Security Module finds a GAO, the inbound processing consists in the following actions:

1. Computing the Sequence Number

2. Validate the ICV
3. Apply the anti-replay protection
4. Remove the GAO from the Geneve Packet
5. GSP Validation

7.3.3.1. Computing the Sequence Number

When the GSO Sequence Number Size indicates the GSO Sequence Number is coded over 32 bits, the Sequence Number is as indicated in the GAO.

When the GSO Sequence Number Size indicates the GSO Sequence Number is coded over 64 bits, the receiving node needs to evaluate the value of the 32 most significant bits. If the Sequence Number is lower than the 32 least significant bits of the GSO Sequence Number, the receiving node will assume the 32 most significant bits of the Sequence Number are the most significant bits of the GSO Sequence incremented by one. The Sequence Number is evaluated as the combination of its 32 most significant bits and the 32 least significant bits indicated in the GAO.

In case it is not possible to increment these 32 most significant bits, the Sequence Number is considered out of the limit and the Geneve Packet is rejected.

It is worth noting that if the Sequence number MUST NOT be incremented by several order of the most significant bits.

7.3.3.2. ICV Validation

To validate the ICV, the receiving node computes the ICV and compares the computed value with the value carried by the GAO. If the two values match the ICV is validated. In case of mismatch, the Geneve Packet is rejected.

The ICV results from applying the GSO Authentication Algorithm with the GSO Authentication Key to the appropriated data.

The appropriated data is build by concatenating the initial string "geneve authentication option" with the Geneve Fixed Header, the GSO Sequence Number, the GSO-ID, the Covered Length, the covered Geneve data.

All elements are read from the Geneve Fixed Header or the GAO and the covered data is read as the number of bytes indicated by the Covered Length value of the GAO that follow the GAO.

7.3.3.3. Anti Replay Protection

The receiving node reads the Sequence Number and Compare it with the GSO Sequence Number stored in the GSA. The difference Delta is evaluated by computing $\text{GSO Sequence Number} - \text{Sequence Number}$.

If Delta is greater than GSO Anti Replay Window, the Geneve Packet is rejected.

If Delta is strictly negative, the GSO Sequence Number is updated with the value of the Sequence Number.

7.3.3.4. GAO Removal

Once the ICV protection has been verified as well as the anti replay protection, the GAO is removed from the Geneve Packet. The removal of the Option occurs after the UDP decapsulation, thus there is no impact on the Geneve Packet, and, for example, no length needs to be adjusted.

7.3.3.5. GSP Validation

GSP Validation validates a given GAO is conform to the expected GSP. This means that when the GAO has been removed, the resulting Geneve Packet is matched against the GSP DB in order to validate the resulting Geneve Packet is associated to the GSA. Such verification is performed by checking the GSO Selectors.

The Geneve Security Module also checks that the expected part of the Geneve Packet have been covered as expected. This includes the Geneve Options as well as the Geneve Payload Length. In case a mismatch is detected the Geneve Packet MUST be rejected.

Some implementations MAY perform additional checks or transformations. For example, some implementation, unless specified or agreed otherwise, SHOULD remove the immutable Geneve Options that are not covered by the validation.

Once validation is completed, the Geneve Packet is forwarded to the Geneve Layer.

8. IANA Considerations

There are no IANA consideration for this document.

9. Security Considerations

10. Acknowledgment

11. References

11.1. Normative References

[I-D.ietf-ipsecme-rfc7321bis]

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)", draft-ietf-ipsecme-rfc7321bis-06 (work in progress), June 2017.

[I-D.ietf-nvo3-geneve]

Gross, J., Ganga, I., and T. Sridhar, "Geneve: Generic Network Virtualization Encapsulation", draft-ietf-nvo3-geneve-04 (work in progress), March 2017.

[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-dtls13-00 (work in progress), April 2017.

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

[RFC4302] Kent, S., "IP Authentication Header", RFC 4302, DOI 10.17487/RFC4302, December 2005, <<http://www.rfc-editor.org/info/rfc4302>>.

[RFC4868] Kelly, S. and S. Frankel, "Using HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512 with IPsec", RFC 4868, DOI 10.17487/RFC4868, May 2007, <<http://www.rfc-editor.org/info/rfc4868>>.

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

11.2. Informative References

- [I-D.ietf-nvo3-encap]
Boutros, S., Ganga, I., Garg, P., Manur, R., Mizrahi, T., Mozes, D., and E. Nordmark, "NVO3 Encapsulation Considerations", draft-ietf-nvo3-encap-00 (work in progress), June 2017.
- [I-D.mglt-nvo3-geneve-encryption-option]
Migault, D., "Geneve Encryption Option", July 2017, <<https://tools.ietf.org/html/I-D.ietf-nvo3-geneve-encryption-option-00>>.
- [I-D.mglt-nvo3-geneve-security-architecture]
Migault, D., "Geneve Security Architecture", July 2017, <<https://tools.ietf.org/html/I-D.ietf-nvo3-geneve-security-architecture-00>>.
- [I-D.mglt-nvo3-security-requirements]
Migault, D., "Geneve Security Requirements", July 2017, <<https://tools.ietf.org/html/I-D.mglt-nvo3-security-requirements-00>>.
- [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, <<http://www.rfc-editor.org/info/rfc7364>>.

Author's Address

Daniel Migault

Email: daniel.migault@ericsson.com

NVO3
Internet-Draft
Intended status: Standards Track
Expires: December 29, 2017

D. Migault
June 27, 2017

Geneve Header Encryption Option (GEO)
draft-mglt-nvo3-geneve-encryption-option-00

Abstract

This document describes the Geneve Encryption Option (GEO). This option enables a Geneve forwarding element to encrypt the Geneve Header with selected associated Geneve Options as well as a portion of the Geneve Payload.

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 <http://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 29, 2017.

Copyright Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Requirements notation	2
2. Introduction	2
3. GEO Description	2
4. GEO Processing	3
4.1. GEO Placement	3
5. IANA Considerations	4
6. Security Considerations	4
7. Acknowledgment	4
8. References	4
8.1. Normative References	4
8.2. Informative References	5
Author's Address	6

1. Requirements notation

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

2. Introduction

3. GEO Description

For generic format of the Geneve Options is defined in Figure 1. The following values are expected:

- o Option Class: 0x0000
- o Type: C is unset as the GEO can simply be ignored by a NVE or a transit node. The GSP will prevent to accept a GOA that is mandated by the GSP and that has not been validated.
- o R is set to 0.
- o Length: This document only considers the algorithms recommended by [I-D.ietf-ipsecme-rfc7321bis] ENCR_AES_GCM_16 or ENCR_CHACHA20_POLY1305. These algorithms are defined in [RFC4106] and [RFC7539].



4. GEO Processing

4.1. GEO Placement

Geneve Encryption Option



GEO is a termination Geneve Option. The encrypted Geneve Options and portion of the encrypted Geneve Payload are appended to the Geneve Header. They are not encoded as an Geneve Option.

5. IANA Considerations

There are no IANA consideration for this document.

6. Security Considerations

7. Acknowledgment

8. References

8.1. Normative References

[I-D.ietf-ipsecme-rfc4307bis]

Nir, Y., Kivinen, T., Wouters, P., and D. Migault,
"Algorithm Implementation Requirements and Usage Guidance
for IKEv2", draft-ietf-ipsecme-rfc4307bis-18 (work in
progress), March 2017.

[I-D.ietf-ipsecme-rfc7321bis]

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)", draft-ietf-
ipsecme-rfc7321bis-06 (work in progress), June 2017.

[I-D.ietf-nvo3-geneve]

Gross, J., Ganga, I., and T. Sridhar, "Geneve: Generic
Network Virtualization Encapsulation", draft-ietf-
nvo3-geneve-04 (work in progress), March 2017.

[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-dtls13-00 (work in progress), April
2017.

[RFC2119]

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

- [RFC4106] Viega, J. and D. McGrew, "The Use of Galois/Counter Mode (GCM) in IPsec Encapsulating Security Payload (ESP)", RFC 4106, DOI 10.17487/RFC4106, June 2005, <<http://www.rfc-editor.org/info/rfc4106>>.
- [RFC4302] Kent, S., "IP Authentication Header", RFC 4302, DOI 10.17487/RFC4302, December 2005, <<http://www.rfc-editor.org/info/rfc4302>>.
- [RFC4868] Kelly, S. and S. Frankel, "Using HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512 with IPsec", RFC 4868, DOI 10.17487/RFC4868, May 2007, <<http://www.rfc-editor.org/info/rfc4868>>.
- [RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012, <<http://www.rfc-editor.org/info/rfc6347>>.

8.2. Informative References

- [I-D.ietf-nvo3-encap] Boutros, S., Ganga, I., Garg, P., Manur, R., Mizrahi, T., Mozes, D., and E. Nordmark, "NVO3 Encapsulation Considerations", draft-ietf-nvo3-encap-00 (work in progress), June 2017.
- [I-D.mglt-nvo3-geneve-security-architecture] Migault, D., "Geneve Security Architecture", July 2017, <<https://tools.ietf.org/html/I-D.ietf-nvo3-geneve-security-architecture-00>>.
- [I-D.mglt-nvo3-security-requirements] Migault, D., "Geneve Security Requirements", July 2017, <<https://tools.ietf.org/html/I-D.mglt-nvo3-security-requirements-00>>.
- [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, <<http://www.rfc-editor.org/info/rfc7364>>.
- [RFC7539] Nir, Y. and A. Langley, "ChaCha20 and Poly1305 for IETF Protocols", RFC 7539, DOI 10.17487/RFC7539, May 2015, <<http://www.rfc-editor.org/info/rfc7539>>.

Author's Address

Daniel Migault

Email: daniel.migault@ericsson.com

NVO3
Internet-Draft
Intended status: Standards Track
Expires: December 29, 2017

D. Migault
June 27, 2017

Geneve Security Architecture
draft-mglt-nvo3-geneve-security-architecture-00

Abstract

This document describes the Geneve Security Architecture.

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 <http://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 29, 2017.

Copyright Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Requirements notation	2
2. Introduction	2
3. Terminology	3
4. Architecture Overview	4
5. Geneve Security Policies Database	5
5.1. Selectors	6
5.2. Geneve Security Policies	8
5.3. Geneve Security Policies Example	8
6. Geneve Security Association Database	11
6.1. Geneve Security Associations	11
7. Geneve Security Module Packet Processing	13
7.1. Outbound Geneve Processing	13
7.2. Inbound Geneve Packet Processing	13
8. IANA Considerations	14
9. Security Considerations	14
10. Acknowledgment	14
11. References	15
11.1. Normative References	15
11.2. Informational References	15
Author's Address	16

1. Requirements notation

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

2. Introduction

[I-D.ietf-nvo3-encap] and [I-D.mglt-nvo3-security-requirements] clearly state the need to secure the Geneve overlay network and provide means to authenticate the Geneve Header as well as being able to encrypt the Geneve Payload.

Both of these requirements are fulfilled with specific Geneve Security Options. More explicitly, [I-D.mglt-nvo3-geneve-authentication-option] defines an option that authenticate the Geneve fixed Header and optionally a set of Geneve Option as well as a portion of the Geneve Payload. [I-D.mglt-nvo3-geneve-encryption-option] defines a Geneve Option that enables to encrypt a subset of Geneve Options as well as a portion of the Geneve Payload. Further descriptions on how an Geneve Security Option is treated is out of the scope of this document.

This document defines how the Geneve overlay can be secured properly. A Geneve Element may handle different Geneve overlay networks

associated with different level of security. This document defines how to associate a level of security to an Geneve overlay network. In addition, a security level for a given overlay network may result in a combination of multiple Geneve Security Options. As the order these Geneve Security Options are processed matters, it is necessary the sending and receiving Geneve Element have similar behaviours in order to guarantee interoperability while securing a Geneve overlay Network.

This document explains how Geneve Security Policies and Geneve Security Associations are organized to associate a given level of security to an Geneve overlay network. In addition, this document also exposes how the Geneve Security Module implementing the security interacts with the Geneve architecture.

3. Terminology

- o Geneve Elements: designates all elements that handled Geneve Packets. These elements may be terminal elements such as NVEs for example, but can also be on path elements that are expected to manage the flow inside the Geneve overlay network.
- o Geneve Packet: designates the packet that Geneve Elements are expected to handled. It is composed of a Geneve Header and a Geneve Payload. In this document the Outer Ethernet Header and Outer IPv4 Header as well as the Outer UDP Header defined in [I-D.ietf-nvo3-geneve] section 3.1 are not part of the Geneve Packet. Similarly, the Outer Ethernet Header, the Outer IPv6 Header as well as the Outer UDP Header defined in [I-D.ietf-nvo3-geneve] section 3.2 are not part of the Geneve Packet.
- o Geneve Header: is described in [I-D.ietf-nvo3-geneve] section 3.4. The Geneve Header may contain zero or more Geneve options.
- o Geneve Payload: designates the data carried by a Geneve Packet. [I-D.ietf-nvo3-geneve]. In [I-D.ietf-nvo3-geneve] section 3.1 and section 3.2, the Geneve Payload would be the Inner Ethernet Header, the Payload and the Frame Check Sequence.
- o Geneve Fix Header: The Geneve Header without any Geneve Options.
- o Geneve Security Policies (GSP):
- o Geneve Security Policies Data Base (GSP DB):
- o Geneve Security Association (GSA):

- o Geneve Security Association Data Base (GSA DB):
- o Geneve Security Options (GSO): A security option defined for Geneve. Currently the security options that have been defined are GAO or GEO.
- o Geneve Authentication Option (GAO): Geneve Option that describes how to authenticate the Geneve Header as well as part of the Geneve Payload. This option is described in [I-D.mglt-nvo3-geneve-authentication-option].
- o Geneve Encryption Option (GEO): Geneve Option that describe how to encrypt Geneve Options as well as a part of the Geneve Payload. GEO is defined in [I-D.mglt-nvo3-geneve-encryption-option].
- o Geneve Security Module: an implementation responsible to enforce the security of Geneve Packets.

4. Architecture Overview

The Geneve Security Architecture is represented in figure Figure 1. Geneve security is enforced by the Geneve Security Module. The Geneve Security Policies (GSP) define which flows inside a virtual network needs to be secured by associating a action SECURE, BYPASS or DISCARD to each Geneve Packet. When a Geneve Packet is tagged as SECURE, the GSP provides specific structures known as Geneve Security Associations (GSA) that describe how the Geneve Packet MUST be secured. Typically, the GSA defines the type of option (Geneve Authentication Option (GAO) or the Geneve Encryption Option (GEO) to be computed or validated, as well as the necessary material such as the appropriated counters, the necessary keys to compute and validate the GSO. GSP and GSA are respectively stored in GSP Database (GSP DB) and GSA Database (GSA DB).

For outbound traffic, the Geneve Security Module receives a non secured Geneve Packet and is responsible to secure that Geneve Packet with the appropriated GSOs - as defined by the GSP/GSAs. Once the GSO have been added, Outer Encapsulation is performed as described in [I-D.ietf-nvo3-geneve] i.e. the Geneve Packet is being encapsulated with Outer Ethernet / Outer IPv4 or IPv6 / and Outer UDP.

For inbound traffic, the Geneve Security Module defines whether a incoming Geneve Packet must be secured or not as defined by the GSP. If a Geneve Packet does not have to be secured, any GSO found is ignored. Otherwise, the Geneve Security Module validates each GSO, and check the validated GSOs are conformed to the defined GSP. The last step is necessary to make sure that in addition to valid security options, the expected GSO were encountered.

This document assumes that all nodes GSP DB and GSA DB are appropriately provisioned by the control plane.

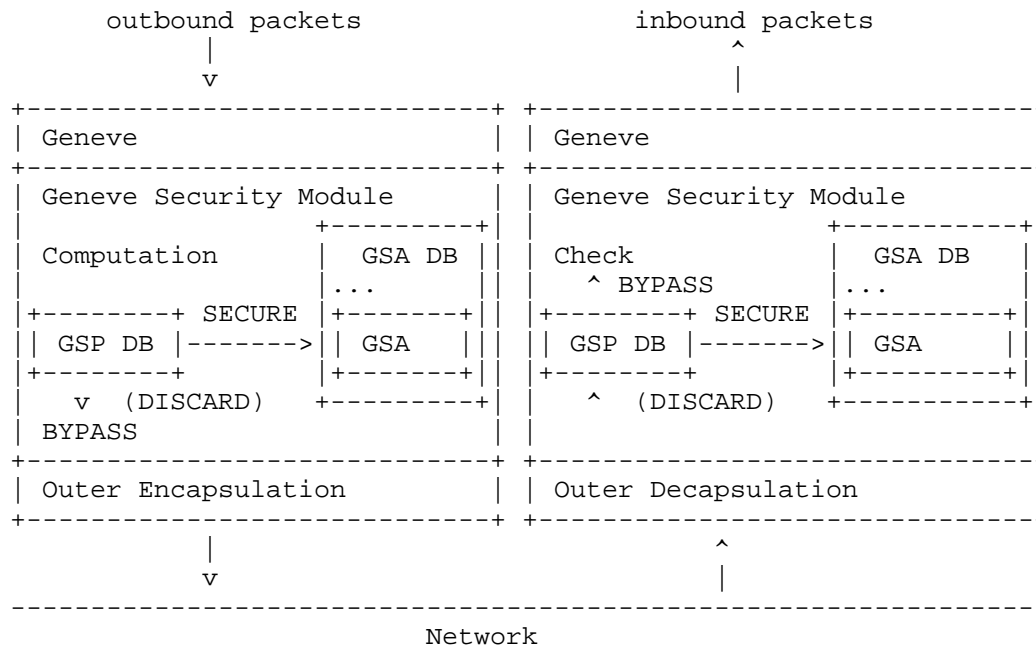


Figure 1: Geneve Security Architecture

5. Geneve Security Policies Database

The GSP DB contains a list of GSP that associates a Geneve Packet with a specific action `BYPASS`, `DISCARD`, `SECURE`.

The matching between a Geneve Packet and an action is performed through Selectors. These Selectors associated to specific values defined whether a Geneve Packet match a given GSP. As GSO may result in encrypting a Selector, a GSP lookup is always performed with a "clear text" Geneve Packet. More specifically, the GSP lookup for a Geneve Packet associated with the SECURE action is performed before the GSO is being added or after the GSO has been validated. For outbound Geneve Packet, a GSP DB look up is performed using the Selectors' value before the GSO is computed. In that case, the GSP will even provide the required structure to generate the GSO. On the other hand, for incoming traffic, the GSO is identified by an identifier carried by the Geneve Packet, a GSP look up is performed once the GSO has been validated / decrypted. As a consequence, the same GSP DB is shared by the sending and the receiving Geneve Element.

When BYPASS is selected, then the Geneve Security Module forwards the matching Geneve to the next layer. As represented in Figure 1, the next layer of an outbound Geneve Packet is the Outer Encapsulation while the next layer of an incoming Geneve Packet is the Geneve layer. When DISCARD is selected, the Geneve Security Module is expected to drop the matching Geneve Packet. When a SECURE action is selected the associated GSAs MUST be provided. For outbound Geneve Packet, the GSAs provided will be used in order to appropriately generate the GSO. On the other hand, for incoming Geneve Packet, the GSAs are returned so the Geneve Security Module can validate the GSO present in the Geneve Packet are conform to the GSP.

It is worth noting that for incoming Geneve Packet, those not tagged as BYPASS or DISCARD are by default considered as tagged as SECURE. This means that the GSP DB may be split into sub databases that contains GSP associated to a specific action. GSP DB (DISCARD/BYPASS) may contain all GSP associated to the DISCARD and BYPASS rules while GSP DB (SECURE) contains all GSP associated to the action SECURE. By doing so, a incoming Geneve Packet may be associated to the SECURE action without performing a lookup on GSP DB (SECURE). This does not prevents the Geneve Security Module from validating the GSO found in the incoming Geneve Packet, as these GSO are carrying a specific Identifier. On the other hand, the GSP DB (SECURE) MUST be lookup in order to validate that all GSO defined by the security policies have been appropriately validated.

5.1. Selectors

The Selectors are the elements read from the packet in order to match a GSP. When a Selector is expected to be found in the Geneve Packet, the Selectors values that match the condition are indicated with a range or a list of matching values. For clarity, this document uses ANY to indicate the full range. When the Selector may not exist or may not be accessible and must be ignored to evaluate the matching condition, it is qualified of being OPAQUE.

Geneve Header Selectors:

- o Geneve Version (2 bits): The version of the Geneve Version. This field is always present and MUST be specified. When all Geneve Version are associated to the same GSP, then all values must be specified with ANY = [0, ..., 4].
- o OAM bit (1 bit): The indication of an OAM indication. When OAM and non OAM traffic is associated to the same GPS, then all values must be indicated with ANY = [0, 1].

- o Critical bit (1 bit): indicates the presence of a critical option. When the presence of a critical option or its absence are associated to the same GSP, then all values must be indicated with ANY = [0, 1]
- o Rsv (6 bits): Currently [I-D.ietf-nvo3-geneve] specifies the field is set to zero by the sender and ignored by the receiver. According to these rules, the sender is expected to DISCARD any non zero values and the receiver is expect to indicate all these values in its GSP.
- o Protocol Type (16 bits): indicates the type of the Geneve Payload. It is likely that only a few types will be specified for matching.
- o VNI: indicates the virtual network identifier. It is also likely that only a small set of VNI values will be provisioned per switches.
- o Reserved (8 bits): (see Rsv)
- o Geneve Options Class - Type List: This fields specifies the Geneve Options that MUST be present. The absence of one of these option result in discarding the Geneve Header. When the presence or absence of a specific Geneve Option has no impact for the GSP selection, the value is set to OPAQUE as they may not be any options.

Additional Selectors are considered within the Geneve Payload. The Selectors provided below are expected to enable different GSP according to the protection of the traffic. Typically, the Geneve overlay network may protect differently traffic that is already protected by the tenants with IPsec, DTLS/TLS, or SSH.

- o Next Header (IPv6) / Protocol (IPv4) (8 bits): For IPv6 this field indicates the presence of an IPv6 Option or the transport layer used after the IPv6 Header. For IPv4 packets, the protocol indicates the layer after the IPv4 Header. This field is typically used to indicate whether IPsec/AH (51), IPsec/ESP (50), TCP (6) or UDP (17) is used. Next Header is a mandatory field and is expected in any IP header. When the matching condition does not consider the Next Header or Protocol number than ANY = [0, ..., 65535] is expected. When non IP packet are expected, OPAQUE is expected.
- o Port Source is typically used to determine how the tenants traffic is being protected by TLS or DTLS. Ports associated to TLS are expected to be 443 https, 636 ldaps, 989 ftps-data, 990 ftps, 992 telnets, 993 imaps, 994 ircs, 995 pop3s, 5061 sips, 22 ssh/scp.

Note that not all transport are associated with a port number. When only transport layers with port numbers are expected to be used (such as TCP or UDP) and the matching condition does not consider the port numbers, ANY = [0, ..., 65535] is expected. When traffic may not have port numbers - such as ICMP traffic, OPAQUE is expected.

- o Port Destination: (see Port Source)

5.2. Geneve Security Policies

Geneve Security Policies are unidirectional. A GSP is composed of:

- o Selectors that express a matching condition
- o Action that defines if the Geneve Packet MUST be DISCARDED, BYPASSED or SECURED. When the associated action is SECURE, then the GSP associates an ordered list of GSA. The GSA contains the description and the necessary material to perform the SECURE action.

The GSP DB is an ordered list of GSP.

5.3. Geneve Security Policies Example

According to [I-D.ietf-nvo3-geneve], the associated Geneve Version is 0, a sender MUST set Rsv and Reserved to zero. When the sender only supports [I-D.ietf-nvo3-geneve], it may performed a sanity check for its outbound packets. The rules can be places at the beginning of the GSP DB.

Selector	Value	Action
Geneve Version	[1 ... 4] (non-zero)	DISCARD
OAM	[0,1] (ANY)	
Critical	[0,1] (ANY)	
Rsv	[0, ... ,63] (ANY)	
Protocol	[0, ..., 65535] (ANY)	
VNI	[0, ..., 16777215] (ANY)	
Reserved	[0, ..., 255] (ANY)	
Geneve Options	OPAQUE	
Next Header	[0, ..., 255] (ANY)	
Port Source	OPAQUE	
Port Destination	OPAQUE	
Geneve Version	[0 ... 4] (ANY)	DISCARD
OAM	[0,1] (ANY)	
Critical	[0,1] (ANY)	
Rsv	[1, ... ,63] (non-zero)	
VNI	[0, ..., 65535] (ANY)	
Reserved	[0, ..., 15] (ANY)	
Geneve Options	OPAQUE	
Next Header	[0, ..., 255] (ANY)	
Port Source	OPAQUE	
Port Destination	OPAQUE	
Geneve Version	[0 ... 4] (ANY)	DISCARD
OAM	[0,1] (ANY)	
Critical	[0,1] (ANY)	
Rsv	[1, ... ,63] (ANY)	
VNI	[0, ..., 65535] (ANY)	
Reserved	[1, ..., 15] (non-zero)	
Geneve Options	OPAQUE	
Next Header	[0, ..., 255] (ANY)	
Port Source	OPAQUE	
Port Destination	OPAQUE	

Figure 2: Example 1: Geneve Security Policy for [I-D.ietf-nvo3-geneve] compliance (sender)

By default a Geneve Security Module may DISCARD any Geneve packet that have no matching This is indicated by the following GSP at the end of the GSP DB.

Selector	Value	Action
Geneve Version	[0 ... 4] (ANY)	DISCARD
OAM	[0,1] (ANY)	
Critical	[0,1] (ANY)	
Rsv	[0, ... ,63] (ANY)	
Protocol	[0, ..., 65535] (ANY)	
VNI	[0, ..., 16777215] (ANY)	
Reserved	[0, ..., 255] (ANY)	
Geneve Options	OPAQUE	
Next Header	[0, ..., 255] (ANY)	
Port Source	OPAQUE	
Port Destination	OPAQUE	

Figure 3: Example 2: Geneve Security Policy for [I-D.ietf-nvo3-geneve] compliance (sender)

The example below details a GSP that proceeds to a specific treatment to the traffic between tenant using ESP. The specific treatment could typically only authenticate the Geneve Packet or partially encrypt the Geneve Payload, in order to only hide the Inner headers - including the ESP header - up to the ESP payload.

In the example, the GSP apply the same GSAs whatever the Geneve Header informations are. More specifically, all virtualized network share the same GSAs.

Selector	Value	Action
Geneve Version	[0 ... 4] (ANY)	SECURE [GSA1, GSA2]
OAM	[0,1] (ANY)	
Critical	[0,1] (ANY)	
Rsv	[0, ... ,63] (ANY)	
Protocol	[0, ..., 65535] (ANY)	
VNI	[0, ..., 16777215] (ANY)	
Reserved	[0, ..., 255] (ANY)	
Geneve Options	OPAQUE	
Next Header	[50] (ESP)	
Port Source	OPAQUE	
Port Destination	OPAQUE	

Figure 4: Example 3: Geneve Security Policy for ESP protect traffic

6. Geneve Security Association Database

GSA DB contains all GSAs. GSA are expected to contain all the necessary information for the Geneve Security Module to compute the GSO by both the sender and the receiver. This includes for example the cryptographic keys to encrypt (resp. authenticate) as well as to decrypt (resp. validate) the Geneve Packet. In addition, the GSA also contains parameters associated to the protection of the security option such as the anti-replay mechanisms as well as the management of that options such as its lifetime.

For outbound traffic, the concerned GSA are provided by the GSP. In this case, it is the purpose of the implementation of Geneve Security Module to provide that appropriated reference. Most likely, the appropriated GSAs will be designated using a memory address.

For inbound traffic, the concerned GSA is designated by the associated GSO with a GSO-ID. In that case the appropriated GSA is retrieved using this index.

6.1. Geneve Security Associations

A GSA contains the following information:

- o GSO ID: The identifier of that GSA. This identifier is used by receiver to bind the appropriated GSO to the appropriated GSA. Note that when the packet is encrypted by the GSO, it may not be possible to associate the GSA using GSP.
- o GSO Protocol: The security protocol associated with the Geneve Security Option. Currently the two GSO are GAO and GEO.

When the security option includes some encryption operation, the following parameters are provided. Note that as recommended by [I-D.ietf-ipsecme-rfc7321bis], encryption is authenticated encryption.

- o GSO Encryption Algorithm: In most cases, the encryption is combined with an authentication performed with the same key.
- o GSO Encryption Key:

When the security option includes a dedicated authentication operation (that is not part of the encryption), the following parameters are provided:

- o GSO Authentication Algorithm:

- o GSO Authentication Key:

The following parameters indicate the coverage of the security

- o GSO Payload Covered Length: the length of the Geneve Payload covered by the GSO. The expression of the length can be a number of bytes, but it may also be defined with an abstract designation. For example, a sending node may be willing to authenticate the Geneve Payload up to the ESP layer. In that case, the sending node will have to compute the corresponding Payload Covered Length. This value is only used by the sending node. The receiving node read that value from the GAO.
- o GSO Covered Geneve Options: Indicates the Geneve Options covered by the GSO. This indication is primarily necessary for the sending node and is derived from the Geneve Packet by the receiving node. It might be checked by the receiving node to validate the GSA. It might typically be expressed as a list of Geneve Options that needs to be covered by the authentication.

In order to implement the anti replay mechanisms the following parameters are provided:

- o GSO Sequence Number Size: indicates the size of the SN. This document considers a 32 bit or a 64 bit length.
- o GSO Last Received Packet: that designates the Sequence Number last sent or received packet.
- o GSO Anti Replay Window: that indicates the minimum acceptable value of the Sequence Number. Any Geneve Packet with a lower SN MUST be rejected. Such SN value is usually derived from the Last Received Packet - Anti Replay Windows.

In order to check the conformity with the GSP:

- o GSO Selectors: The selectors are provided so the receiver can check the Geneve Packet protected by the GSO is conform to the GSP. In other words a valid GSO is not sufficient for the Geneve Packet to be forwarded to the upper layers. Note that the Selectors MUST match the Geneve Packet associated to the GSA before the GSO is built for outbound Packets. For inbound Geneve Packet the Selectors are those that corresponds to the Geneve Packet after the GSO has been validated/decrypted. Selectors are mostly expected to be used by the GSA for incoming Geneve Packet, in order to check the GSA is conform with its GSP.
- o GSA Life time:

7. Geneve Security Module Packet Processing

This section assumes that the GSA is valid. Unvalid GSA MUST be deleted or considered as non existing by either the sender or the receiver.

7.1. Outbound Geneve Processing

- o The Geneve Security Module consults the GSP DB to determine the GSP associated to the Geneve Packet.
 - * When a Geneve Packet is DISCARD, the Geneve Packet is dropped.
 - * When a Geneve Packet is BYPASS, the Geneve Packet is directly forwarded to the lower layers for the outer encapsulation.
 - * When a Geneve Packet is SECURE, the GSP returns one or multiple Geneve Security Association (GAS) of the Geneve Security Association Database (GSA DB). GAS contains the necessary material to compute the GSO for outbound Geneve Packet. When multiple GAS are returned, GAS are applied in the order they are provided. Each computed GSO carries a unique GSA-ID, so the receiver can check the corresponding GSO without performing a GSP DB lookup.
 - * When no matching is found, the Geneve Packet is DISCARDED
- o Geneve Packet is forwarded to the lower layers for the Outer Encapsulation.

7.2. Inbound Geneve Packet Processing

For inbound Geneve Packets:

- o The Geneve Security Module checks the Geneve Packet is associated to a DISCARD or a BYPASS GSP.
 - * If a match occurs the Geneve Packet is either DISCARDED or BYPASSED to the Geneve layer.
 - * Otherwise the Geneve Packet is expected to be SECURED and processed as such by the Geneve Security Module.

When the Geneve Packet is believed to be SECURED.

- o The Geneve Security Module opens a security context which lists the encountered and validated GSO as well as their respective order.

- o The Geneve Security Module inspects the Geneve Header for GSO in a network order and proceeds as follows:
 - * The Geneve Security Module extracts the GSA-ID of the GSO.
 - * The Geneve Security Module performs a GSA DB lookup based on the GSA-ID to retrieve the GSA associated to the Geneve Packet.
 - + If the GSA DB the GSO, the SO is skipped and the Geneve Security Module continue wity the next GSO. In this case, the GSO is treated as a unexpected geneve option.
 - * The Geneve Security Module validates the Geneve Security Option against the GSA. If the validation does not succeeds, the Geneve Packet is discarded.
 - * The Geneve security Module validates the Geneve Packet - once the GSO process has been performed - is conformed with the GSP by checking the resulting Geneve Packet matches the Selectors provided in the GSA.
 - + If the validation is successful, the Geneve Security Module associates the GSA-ID with a validated status in the security context. For this reason it is important to match the GSA Selector with the appropriated Selectors value. In case multiple GSO are combined, the Selectors of the GSA MAY differ from those used for the GSP DB matching.
 - + If a mismatch occurs the Geneve Packet is dropped.

When all Geneve Security Options have been validated, the Geneve Packet is matched against the GSP DB to validate the GSA-ID listed in the security context match those returned by the GSP DB. Note that the receiver and the sender MUST have the same GSA-IDs, however, computation and validation are processed in a different order.

8. IANA Considerations

There are no IANA consideration for this document.

9. Security Considerations

10. Acknowledgment

11. References

11.1. Normative References

- [I-D.ietf-ipsecme-rfc7321bis]
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)", draft-ietf-ipsecme-rfc7321bis-06 (work in progress), June 2017.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<http://www.rfc-editor.org/info/rfc2119>>.

11.2. Informational References

- [I-D.ietf-nvo3-encap]
Boutros, S., Ganga, I., Garg, P., Manur, R., Mizrahi, T., Mozes, D., and E. Nordmark, "NVO3 Encapsulation Considerations", draft-ietf-nvo3-encap-00 (work in progress), June 2017.
- [I-D.ietf-nvo3-geneve]
Gross, J., Ganga, I., and T. Sridhar, "Geneve: Generic Network Virtualization Encapsulation", draft-ietf-nvo3-geneve-04 (work in progress), March 2017.
- [I-D.mglt-nvo3-geneve-authentication-option]
Migault, D., "Geneve Authentication Option", July 2017, <<https://tools.ietf.org/html/I-D.ietf-nvo3-geneve-authentication-option-00>>.
- [I-D.mglt-nvo3-geneve-encryption-option]
Migault, D., "Geneve Encryption Option", July 2017, <<https://tools.ietf.org/html/I-D.ietf-nvo3-geneve-encryption-option-00>>.
- [I-D.mglt-nvo3-security-requirements]
Migault, D., "Geneve Security Requirements", July 2017, <<https://tools.ietf.org/html/I-D.mglt-nvo3-security-requirements-00>>.

Author's Address

Daniel Migault

Email: daniel.migault@ericsson.com

NVO3
Internet-Draft
Intended status: Standards Track
Expires: December 29, 2017

D. Migault
Ericsson
June 27, 2017

Geneve Protocol Security Requirements
draft-mglt-nvo3-geneve-security-requirements-00

Abstract

This draft lists the security requirements associated to the Generic Network Virtualization Encapsulation (Geneve) [I-D.ietf-nvo3-geneve].

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 <http://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 29, 2017.

Copyright Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Requirements notation	2
2. Introduction	2
3. Tenants Isolation	3
3.1. Traffic Injection	4
3.2. Traffic Redirection	6
4. Overlay Network Robustness	7
5. Infrastructure Isolation	8
5.1. Tenants Communication	8
5.2. Overlay Network Architecture	9
6. IANA Considerations	9
7. Security Considerations	9
8. Acknowledgment	10
9. References	10
9.1. Normative References	10
9.2. Informational References	10
Author's Address	10

1. Requirements notation

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

2. Introduction

A cloud provider may administrate Tenant Systems belonging to one or multiple tenants using an Geneve overlay network. The Geneve overlay enables multiple Virtual Networks to coexist over a shared infrastructure, and a Virtual Network may be distributed within a single data center or between different data centers. The Geneve overlay network is constituted by Geneve forwarding elements as well as Network Virtualization Edges (NVE) [RFC8014]. Traffic with a Virtual Network is thus steered between NVEs using Generic Network Virtualization Encapsulation (Geneve) [I-D.ietf-nvo3-geneve].

This document analyses and lists the security requirements to securely deploy Geneve overlay networks. It is expected that these requirements will help design the appropriated security mechanisms for Geneve as well as provides some basis security notion for further Geneve deployments.

In addition, when a tenant subscribes to a cloud provider for hosting its Tenant Systems, the cloud provider manages the Geneve overlay network on behalf of the tenant [RFC7365]. It may also, but not necessarily manage the infrastructure supporting the overlay network. The Geneve security requirements listed in this document aims at

providing the cloud provider the necessary options to ensure the tenant:

1. Tenants Isolation, that is Tenant System inside a Virtual Network are isolated from other Tenants Virtual Networks. This Tenants isolation mostly prevents traffic from one tenant to be redirected to another tenant as well as traffic from one tenant being injected into another tenant.
2. Geneve robustness, that is a rogue elements of the Geneve overlay network will have limited impacts over the Geneve overlay network itself as well as over Tenants Systems.
3. Geneve isolation of the infrastructure, that is information in transit is not subject to passive monitoring. Information in transit concerns both information associated to the Geneve overlay network as well as information exchanged by the Tenant' Systems. Hiding information of the overlay network to the infrastructure is typically required when the overlay network provider and the infrastructure provider are different entities.

3. Tenants Isolation

Tenant Systems isolation prevents communications from one tenant to leak into another Tenant Systems' virtual network. This section is focused on an Geneve overlay network perspective which means:

1. Only communications between NVEs are considered. In other words, the transmission from the NVE to the Tenant System itself is out of the scope of this section. Similarly the security used by the infrastructure to steer Geneve Packets from a NVE to Geneve forwarding element is out of scope either.
2. Isolation is only broken by rogue NVE or rogue Geneve forwarding elements. In other words, breaking isolation using other elements or other protocol layers are out of scope of this section
3. A Geneve NVE SHOULD be able to set different security policies to different flows. These flows MUST be characterized from the Geneve Header and Geneve Options as well as some inner traffic selectors. Typically an NVE SHOULD be able to selectively authenticate only the sections that are not authenticate by the Tenant System. If the Tenant Systems authenticates its communications with TLS, only the IP, transport (TCP / UDP) and TLS/DTLS section should be encrypted while only the IP header and ESP header is expected to be encrypted when tenants' communications are encrypted with ESP.

Suppose Tenant A and Tenant B are two distinct tenants and are expected to remain isolated by the Geneve overlay network. The attacks breaking the isolation considered in this section are the injection of traffic into one virtual network as well as the redirection of one tenant's traffic to a third party.

3.1. Traffic Injection

Traffic injection can target a specific element on the overlay network such as, for example, an NVE, a Geneve forwarding element or eventually specific Tenant System. On a overlay network perspective, the difference of targeting a Tenant's System requires valid MAC and IP addresses of the Tenant's System.

In order to provide integrity protection, 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 is independent from the Geneve overlay network and as such provides protection against any node outside the virtual network including the nodes of the Geneve overlay network to inject packets to a Tenant System. On the other such protection does not protect the virtual network from receiving illegitimate packets that may disrupt the Tenant's System performances.

When Tenant Systems are protected against spoofed packets, the Geneve overlay network may still prevent such spoofed Geneve Packet to be steered into the virtual network. In addition, when the Tenant's System have not enabled such protections, the overlay network should be able to provide a secure infrastructure for hosting these virtual networks and prevent a third party to inject traffic into the overlay. In this section the third party is a node on the infrastructure hosting the Geneve overlay network. In addition, this node could be any Geneve element except the legitimate NVEs (source or destination).

A Geneve overlay network is composed of multiple Geneve forwarding elements steering a Geneve Packet between the two NVEs. The Geneve Packet is forwarded according to the information carried in the Geneve Packet as well as routing tables associated to this information. For that reason, the information carried in the Geneve Header, including Geneve Option MUST be accessible by the intermediary nodes.

In order to prevent traffic injection in one virtual network, the destination Geneve NVE MUST be able to authenticate the incoming traffic sent by the source NVE. Note that this threat model assumes

that the third party injecting traffic does not inject traffic through the NVEs.

Authentication of the whole Geneve Packet may raise the cost of security unnecessarily. In fact it is expected that the Tenant Systems will also protect their end-to-end traffic, as a result, corruption of the Geneve Payload can be detected by the System Tenant. In addition, for the ease of processing, an authenticated Geneve Packet should not impact the processing of the intermediary nodes, unless they are able to check the authentication themselves. A key advantage of validating the authentication by intermediary nodes is that detection can occur earlier, however such requirement may require the use of asymmetric cryptography, which may be balanced by its low performance over symmetric cryptography. As a result the following requirements are associated with the authentication:

- REQ1: A Geneve NVE MUST be able to authenticate the Geneve Header including the immutable Geneve Options.
- REQ2: A Geneve NVE MUST be able to agree that authentication includes or not the Geneve Payload, and if so it SHOULD also be able to indicate that only a portion of it is authenticated.
- REQ3: A Geneve intermediary forwarding element MAY be able to validate the authentication before the packet reaches the Geneve destination tunnel end point.
- REQ4: A Geneve intermediary forwarding element MUST be able to insert an authenticated Geneve Option into a authenticated Geneve Packet - protected by the source Geneve tunnel termination point.
- REQ5: A Geneve intermediary forwarding element not supporting authentication MUST NOT be impacted by the authentication of the Geneve Packet and should be able to handle the Geneve Packet as a non-authenticated Geneve Packet.
- REQ6: A Geneve NVE SHOULD be able to set different security policies to different flows. These flows MUST be characterized from the Geneve Header and Geneve Options as well as some inner traffic selectors. Typically an NVE SHOULD be able to selectively encrypt only the sections that are not encrypted by the Tenant System. If the Tenant Systems encrypts its communications with TLS, only the IP, transport (TCP / UDP) and TLS/DTLS section should be encrypted while only the IP header and ESP header is expected to be encrypted when tenants' communications are encrypted with ESP.

3.2. Traffic Redirection

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, or for actively re-injecting a modified Geneve Packet into the overlay.

Avoiding leaking information is hard to enforced at a Geneve level. However, the Geneve overlay network and the Tenants Systems can lower the consequences of such leakage in case these occurs. The Tenant System may protect partly the data carries over the overlay network using end-to-end encryption such as IPsec/TLS. Doing so provides integrity protection as well as confidentiality for the Tenant's information. Such protection applies even if the source or destination NVE are corrupted.

The purpose of the Geneve overlay network is to limit the information it is aware of to leak. When Tenant Systems are enforcing confidentiality of the information in transit with IPsec or TLS for example, they are still some information revealed the MAC and IP headers of the inner packet may remain unprotected. IN this case, the Geneve overlay network should be able to maintain this information confidential. When Tenant's have not enforced such security the Geneve overlay network should be able to provide a secure infrastructure and prevent leakage of information outside the virtual network. In addition, the information carried by the Geneve Header may also reveal some information on the overlay network itself, its deployment as well as states from the Tenant System. In this the Geneve overlay network should also be able to protect such Geneve Options.

Note that when the overlay network is hosted on an architecture that belongs to another administrative domain, the administrator of the infrastructure is typically able to perform passive monitoring attacks.

In order to protect the Geneve communications between the Geneve tunnel terminating points here are the following requirements:

- REQ7: A Geneve NVE MUST be able to agree that the Geneve Payload or portion of it is encrypted as well as as immutable Geneve Options not intended for the intermediary Geneve nodes.
- REQ8: A Geneve intermediary forwarding element MUST be able to insert an encrypted Geneve Option into a authenticated Geneve Packet - protected by the source Geneve tunnel termination point.

- REQ9: A Geneve intermediary forwarding element MUST be able to insert an encrypted Geneve Option into an encrypted Geneve Packet - protected by the source Geneve NVE.
- REQ10: A Geneve intermediary forwarding element not supporting encryption MUST NOT be impacted by the authentication of the Geneve Packet and should be able to handle the Geneve Packet as an non-protected Geneve Packet.

Re-injection through a Geneve intermediary node is prevented by the authentication. On the other hand, if the re-injection is performed through one of the Geneve NVE, the protection provided by encryption as well as authentication does not apply. The authentication is intended to check integrity toward the data provided by the source Geneve NVE. If that point is corrupted, it is likely to inject corrupted traffic with integrity protection. On the other hand, if the destination Geneve NVE is expected to validate the data, as a result if traffic is injected through that node it is likely to bypass the integrity validation.

4. Overlay Network Robustness

While Tenant isolation prevents one Tenant to inject packets into another Tenants, it does not prevent a rogue or misconfigured node to replay a packet, to load a specific Tenant System with a modified Geneve payload or to abuse the Geneve overlay network.

1. A rogue Geneve overlay forwarding element on path of one Tenants traffic may replay a valid packet to load the network. This can typically be seen as a volumetric attack in order to disrupt the tenants domain, a specific Tenant System or the multi Tenant infrastructure itself. In some cases, especially when the tenants costs are evaluate on the necessary computing resources, such attacks may target an increase of the tenants costs.
2. When traffic between tenants is not protected, a rogue Geneve overlay element may forward a modified packet over a valid Geneve Header. The crafted packet may for example, include a specifically crafted application payload intended for a specific Tenant Systems application. Other examples includes a larger randomly craft payload intended to load one specific application.
3. The Geneve forwarding policies are engineered according to the various types of flows with their associated volumetry and requirements. For example, some OAM flows are expected to be associated with a higher priority then standard data plane flows. Similarly, the use of various Geneve Header parameters or options may introduce different treatments. Updating the Geneve header

may result in counter all optimizations used to setup a performant infrastructure and thus affect the tenants.

Note nodes that may address such attacks MUST be provided means to authenticate the Geneve Packet. More specifically,

In order to avoid the above mentioned attacks, the following requirements should be considered:

REQ11: Geneve Header SHOULD be bound to the forwarded payload. By reading the Geneve Header and the Payload, the Geneve forwarding element SHOULD be able to validate the Geneve Header corresponds to the Geneve payload. In case of mismatch the Geneve forwarding element is expected to discard the packet.

REQ12: Geneve forwarding element SHOULD be provided anti replay mechanisms. By reading the Geneve Header, the Geneve forwarding element is expected to detect a packet has been replayed or at least limit the replay windows. When a packet is detected as being replayed, the Geneve forwarding element is expected to discard this packet.

5. Infrastructure Isolation

The cloud provider managing the Geneve overlay network may be willing to isolate the communications between Tenant Systems as well as the organization of the Geneve overlay from the infrastructure. Such isolation may be performed by encrypting the data in transit within the Geneve overly network.

5.1. Tenants Communication

The main purpose for encrypting tenants communication inside the Geneve overlay network is to prevent that external parties such a infrastructure providers may access to the information exchanged between Tenant System exchanged via the Geneve overlay network. A typical example comes would be the infrastructure provider used by the Geneve overlay network.

In addition, encryption of the data in transit in the Geneve overlay network may also be one way to prevent the leakage of information when tenant isolation is broken. Encryption is not expected to enforce tenant isolation, but if information can hardly be used by another tenant it may limit the interest in breaking such isolation to still information as well as it might reduce the risks of leaking some confidential information.

The requirements correspond to the those protecting against the redirection or passive monitoring attacks in Section 3.2.

IPsec or TLS provides end-to-end encryption for NVE communications. However, as the Geneve Header would be encrypted, these mechanisms cannot be used as general mechanisms for the overlay network.

Encrypting Geneve payload by the NVE prevents disclosing the Geneve payload to third party in case of leakage. However, such service is provided by the cloud provider and the tenant has little control over it. In most cases, if the tenant is willing to enforce data confidentiality, it is recommended that it encrypts communications between Tenants systems using IPsec or TLS. By doing so, the cloud provider would not even have access to such information. While encryption is being performed by the tenant, a cloud provider may be willing to avoid re-encrypting that same content. Instead, the cloud provider may prefer to only encrypt the tenants informations that have not been encrypted by TLS or IPsec. Doing so is expected to reduce the necessary resource for encrypting.

The requirements correspond to the those protecting against the redirection or passive monitoring attacks in Section 3.2.

5.2. Overlay Network Architecture

In addition, to the information exchanged between Tenant Systems, the cloud provider may also avoid revealing the distribution of the Tenant Systems through the data center. In fact a passive attacker may observe the NVI in the Geneve header in order to derive the communication pattern between the Tenant Systems. Other parameters or options may reveal other kind of informations. One possibility is to encrypt the information, but other transformations may also apply.

The requirements correspond to the those protecting against the redirection or passive monitoring attacks in Section 3.2.

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.

8. Acknowledgment

9. References

9.1. Normative References

[I-D.ietf-nvo3-geneve]
Gross, J., Ganga, I., and T. Sridhar, "Geneve: Generic Network Virtualization Encapsulation", draft-ietf-nvo3-geneve-04 (work in progress), March 2017.

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

9.2. Informational References

[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, <<http://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, <<http://www.rfc-editor.org/info/rfc8014>>.

Author's Address

Daniel Migault
Ericsson
8400 boulevard Decarie
Montreal, QC H4P 2N2
Canada

Email: daniel.migault@ericsson.com