Workgroup: NVO3 Working Group
Internet-Draft: draft-ietf-nvo3-encap-12
Published: 19 February 2024
Intended Status: Informational
Expires: 22 August 2024
Authors: S. Boutros, Ed. D. Eastlake, Ed.
Ciena Corporation Futurewei Technologies

Network Virtualization Overlays (NVO3) Encapsulation Considerations

Abstract

The IETF Network Virtualization Overlays (NVO3) Working Group developed considerations for a common encapsulation that addresses various network virtualization overlay technical concerns. This document provides a record, for the benefit of the IETF community, of the considerations arrived at starting from the output of an NVO3 encapsulation design team. These considerations may be helpful with future deliberations by working groups over the choice of encapsulation formats.

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

Based on these considerations, the Working Group determined that Geneve with a few modifications as the common encapsulation. This document provides more details, particularly in Section 7.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

The NVO3 Working Group is chartered to gather requirements and develop solutions for network virtualization data planes based on encapsulation of virtual network traffic over an IP-based underlay data plane. Requirements include due consideration for OAM and security. Based on these requirements the WG was to select, extend, and/or develop one or more data plane encapsulation format(s).

This led to WG drafts and an RFC describing three encapsulations as follows:

*[RFC8926] Geneve: Generic Network Virtualization Encapsulation

*[ietf_intarea_gue] Generic UDP Encapsulation

*[nvo3_vxlan_gpe] Generic Protocol Extension for VXLAN (VXLAN-GPE)

Discussion on the list and in face-to-face meetings identified a number of technical problems with each of these encapsulations. Furthermore, there was clear consensus at the 96th IETF meeting in Berlin that, to maximize interoperability, the working group should progress only one data plane encapsulation. In order to overcome a deadlock on the encapsulation decision, the WG consensus was to form a Design Team [RFC2418] to resolve this issue and provide initial considerations.

2. Design Team and Working Group Process

The Design Team was to select 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 were goals. The Design Team was to specifically avoid selecting a design that is burdensome on hardware implementations but should allow future extensibility. The selected design also needed to operate well with ICMP and in Equal Cost Multi-Path (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.

The output of the Design Team was then prcoessed through the working group resulting in working group consensus for this document.

3. Terminology

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

4. Abbreviations and Acronyms

The following abbreviations and acronyms are used in this document:

ACL	- Access Control List
DT	- NVO3 encapsulation Design Team
ECMP	- Equal Cost Multi-Path
EVPN	- Ethernet VPN [RFC8365]
Geneve	- Generic Network Virtualization Encapsulation [RFC8926]
GPE	- Generic Protocol Extension
GUE	- Generic UDP Encapsulation [ietf_intarea_gue]
HMAC	- Hash based keyed Message Authentication Code [RFC2104]
IEEE	- Institute for Electrical and Electronic Engineers (www.ieee.org)
NIC	- Network Interface Card (refers to network interface hardware which is not necessarily a discrete "card")
NSH	- Network Service Header [RFC8300]
NVA	- Network Virtualization Authority
NVE	- Network Virtual Edge (device)
NVO3	- Network Virtualization Overlays over Layer 3
OAM	- Operations, Administration, and Maintenance [RFC6291]
PWE3	- Pseudowire Emulation Edge to Edge
TCAM	- Ternary Content-Addressable Memory

TLV
- Type, Length, and Value

Transit device - Underlay network devices between NVE(s).

UUID - Universally Unique Identifier

VNI - Virtual Network Identifier

VXLAN - Virtual eXtensible LAN [RFC7348]

5. Encapsulation Issues and Background

The following subsections describe issues with current encapsulations as discussed by the NVO3 WG. Numerous extensions and options have been designed for GUE and Geneve which may help resolve some of these issues but have not yet been validated by the WG.

Also included are diagrams and information on the candidate encapsulations. These are mostly copied from other documents. Since each protocol is assumed to be sent over UDP, an initial UDP Header is shown which would be preceded by an IPv4 or IPv6 Header.

5.1. Geneve

The Geneve packet format, taken from [RFC8926], is shown in Figure 1 below.

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

 Outer UDP Header:
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |           Source Port         |       Dest Port = 6081 Geneve |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |           UDP Length          |           UDP Checksum        |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

 Geneve Header:
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |Ver|  Opt Len  |O|C|    Rsvd.  |          Protocol Type        |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |        Virtual Network Identifier (VNI)       |    Reserved   |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    ~                    Variable-Length Options                    ~
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1: Geneve Header

The type of payload being carried is indicated by an Ethertype [RFC7042] in the Protocol Type field in the Geneve Header; Ethernet itself is represented by Ethertype 0x6558. See [RFC8926] for details concerning UDP header fields. The O bit indicates an OAM packet. The C bit is the "Critical" bit which means that the options must be processed or the packet discarded.

Issues with Geneve [RFC8926] are as follows:

* Can't be implemented cost-effectively in all use cases because variable length header and order of the TLVs makes it 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.

Selection of Geneve despite these issues may be the result of the Geneve design effort assuming that the Geneve header would typically be delivered to a server and parsed in software.

5.2. Generic UDP Encapsulation (GUE)

```
    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

UDP Header:
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |        Source port            |     Dest port = 6080 GUE      |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |        UDP Length             |          Checksum             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

GUE Header:
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | 0 |C|   Hlen  |  Proto/ctype  |             Flags             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   ~                  Extensions Fields (optional)                 ~
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2: GUE Header

The type of payload being carried is indicated by an IANA Internet protocol number in the Proto/ctype field. The C bit indicates a Control packet.

Issues with GUE [ietf_intarea_gue] are as follows:

* There were a significant number of objections to GUE related to the complexity of implementation in hardware, similar to those noted for Geneve above, such as the variable length and possible high maximum length of the header.

5.3. Generic Protocol Extension (GPE) for VXLAN

```
    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 UDP Header:
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |           Source Port         |     Dest Port = 4790 GPE      |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |           UDP Length          |        UDP Checksum           |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

VXLAN-GPE Header
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |R|R|Ver|I|P|B|O|       Reserved                | Next Protocol |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                VXLAN Network Identifier (VNI) |   Reserved    |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: GPE Header

The type of payload being carried is indicated by the Next Protocol field using a VXLAN-GPE-specific registry. The I bit indicates that the VNI is valid. The P bit indicates that the Next Protocol field is valid. The B bit indicates the packet is an ingress replicated Broadcast, Unknown Unicast, or Multicast packet. The O bit indicates an OAM packet.

Issues with VXLAN-GPE [nvo3_vxlan_gpe] are as follows:

*GPE is not day-1 backwards compatible with VXLAN [RFC7348]. 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 were the same.

*GPE is insufficiently extensible. It adds a Next Protocol field and some flag bits to the VXLAN header but is not otherwise extensible.

*Security, e.g., of the VNI, as discussed in Section 6.2.2, has not been addressed by GPE. Although a shim header could be added for security and to support other extensions, this has not been defined yet. More study would be needed to understand the implication of such a shim on offloading in NICs.

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 [RFC8300]).
- *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 extensions are 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. Thus it may be harder to skip the encapsulation header with VXLAN-GPE

6.2. Useful Extensions Use Cases

Non-vendor specific extensions, such as TLVs, MUST follow the standardization process. The following use cases for extensions show 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, and network planning and 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 between 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 because access to the end-system is not feasible or the diversity of parameters and associated probe packets to be created is simply too large. An extension providing an in-band telemetry mechanism [RFC9197] 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. Extension headers are needed to use any sophisticated security.

The possibility of VNI spoofing with an NVO3 protocol is exacerbated by using UDP. Systems typically have no restrictions on applications being able to send to any UDP port so an unprivileged application can trivially spoof VXLAN [RFC7348] packets for instance, including using arbitrary VNIs.

One can envision support of an HMAC-like Message Authentication Code (MAC) [RFC2104] in an NVO3 extension to authenticate the header and the outer IP addresses, thereby preventing attackers from injecting packets with spoofed VNIs.

Another aspect of security is payload security. Essentially this makes packets that look like the following:

```
IP|UDP|NVO3 Encap|DTLS/IPsec-ESP Extension|payload.
```

This is desirable 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 port for DTLS/IPsec [RFC9147]/[RFC6071]).

6.2.3. Group Based 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 [VXLANgroup]. 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.

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 a transit device 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 options 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 an order for some hardware implementation.

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 both for NICs implementing various TCP offload mechanisms and for transit devices and NVEs applying policy or ACLs to the inner payload.

6.4. Extension Size

Extension header length has a significant impact on hardware and software implementations. A maximum total header length that is too small will unnecessarily constrain software flexibility. A maximum total header length that is too large will place a nontrivial cost on hardware implementations. Thus, the DT recommends that there be a minimum and maximum total available extension header length specified. The maximum total header length is determined by the size of the bit field 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 support for the minimum total header length which would then become the de facto maximum total header length.

The recommended minimum total available header length is 64 bytes.

The size of an extension header should always be 4 byte 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. Ordering of Extension Headers

To support hardware nodes at the target NVE or at a transit device that can process one or a few extension headers in TCAM, a control plane in such a deployment can signal a capability to ensure a specific extension header will always appear in a specific order, for example the first one in the packet.

The order of the extension headers should be hardware friendly for both the sender and the receiver and possibly some transit devices also. This may requre that the extension headers and their order be dynamically determined based on the hardware of those devices.

Transit devices don't participate in control plane communication between the end points and are not required to process the extension headers; however, if they do, they may need to process only a small subset of the extension headers that will be consumed by target NVEs.

6.6. TLV versus 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 fields approach to indicate the presence of extensions 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 of different options. However, general TLVs of arbitrary order, size, and repetition are difficult to implement in hardware. A middle ground is to use TLVs with restrictions on their size and alignment, observing that individual TLVs can have a fixed length, and to support via the control plane a method such that an NVE will only receive options that it 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, some other method must be used. Transit devices would have issues with future GUE bit fields being defined for future options as well.

A benefit of TLVs from a hardware perspective is that they are self describing, i.e., all the information is in the TLV. In a bit field 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 an NVE. These can be modules such as a diagnostic module by one vendor that does packet sampling and another module from a different vendor that implements 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 modules, like MAC learning and security, to process different options.

6.7. Control Plane Considerations

Given that we want to allow considerable flexibility and extensibility, e.g., for software NVEs, yet be able to support important extensions in less flexible contexts such as hardware NVEs, it is useful to consider the control plane. By control plane in this section we mean both protocols, such as EVPN [RFC8365] and others, and deployment specific configuration.

If each NVE can express in the control plane that it only supports certain extensions (which could be a single extension, or a few), and the source NVEs only include supported 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 the deployment can provide useful functionality with simplified hardware requirements for the target NVE.

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 extension headers are included in the packet. 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 with this approach different NVEs could desire different extensions or 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 extension headers.

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 target NVEs that do not support them.

The control planes defined today need to add the ability to describe the different encapsulations. Thus, perhaps EVPN [RFC8365] and any other control plane protocol that the IETF defines should have a way to indicate the supported NVO3 extensions and their order, for each of the encapsulations supported.

Developing a separate draft on guidance for option processing and control plane participation should be considered. 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. Having the control plane negotiate the constraints is the most appropriate and flexible way to address these requirements.

6.8. Split NVE

If there is a need for hosts to send and receive options in a split NVE case [RFC8394], 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 this is needed with other encapsulations it can be done by defining an Ethertype so that it can be carried over Ethernet and [IEEE802.1Q].

If there is a 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

Whether we should make the VNI 32-bits or larger was one of the topics considered. The benefit of a 24-bit VNI would be to avoid unnecessary changes with existing proposals and implementations that are almost all, if not all, using 24-bit VNI. If we need a larger VNI, perhaps for a telemetry case, an extension can be used to support that.

7. Recommendations

The Design Team (DT) reported that Geneve was most suitable as a starting point for a proposed standard for network virtualization, for the following reasons given below. This conclusion was supported by the NVO3 Working Group.

1. On whether VNI should be in the base header or in an extension header and whether it should be a 24-bit or 32-bit field (see Section 6.9), it was agreed that VNI is critical information for network virtualization and MUST be present in all packets. It was also agreed that a 24-bit VNI, which is supported by Geneve, matches the existing widely used encapsulation formats, i.e., VXLAN [RFC7348] and NVGRE [RFC7637], and hence is more suitable to use going forward.

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

3. The option of using NSH [RFC8300] with VXLAN-GPE was considered 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 a header length in VXLAN-GPE, which makes skipping over the headers to process inner payload more difficult. A 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). Whether it would make sense to either use NSH for this, or define a new NVO3 options header was explored. However, 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 guide, 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. The TLV style and bit field style of extension were compared. It was deemed that parsing either TLVs or 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. Also 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 TLVs are more flexible, a control plane can restrict the number of option TLVs as well as the order and size of the TLVs to limit this flexibility and make the TLVs simpler for a dataplane implementation to handle.

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

6. It was agreed that the C (Critical) bit in Geneve is helpful. This bit indicates that the header includes options which must be parsed or the packet discarded. It allows a 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 a receiver NVE and thus make it easy for NVE to skip over the options. Thus, the C bit should remain as defined in Geneve.

7. There are already some extensions that are being discussed (see section 6.2) of varying sizes. By using Geneve options it is possible to get in-band parameters like switch id, ingress port, egress port, internal delay, and queue size using TLV extensions for telemetry purpose from switches. It is also possible to add security extension TLVs like HMAC [RFC2104] and DTLS/IPsec [RFC9147]/[RFC6071] to authenticate the Geneve packet header and secure the Geneve packet payload by software or hardware tunnel endpoints. A Group Based Policy extension TLV can be carried as well.

8. There are already implementations of Geneve options deployed in production networks. There is as well new hardware supporting Geneve TLV parsing. In addition, an In-band Telemetry [INT] specification is being developed by P4.org that illustrates the option of INT meta data carried over Geneve. OVN/OVS [OVN] have also defined some option TLV(s) for Geneve.

9. Usage requirements (see Section 6) have been addressed while considering the requirements and implementations in general including software and hardware.

There seems to be interest in standardizing some well-known secure option TLVs to secure the header and payload to guarantee encapsulation header integrity and tenant data privacy. The working group should consider standardizing such option(s).

The following enhancements to Geneve are recommended to make it more suitable to hardware and yet provide flexibility for software:

* The following sort of text is recommended: while TLVs 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 a secure hash that must be processed in a certain order at lowest latency.

* A control plane can negotiate a subset of option TLVs and certain TLV ordering, as well as limiting the total number of option TLVs present in the packet, for example, to allow for hardware capable of processing fewer options. Hence, the control plane needs to have the ability to describe the supported TLVs subset and their order.

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

* Geneve should follow fragmentation recommendations in overlay services like PWE3 and the L2/L3 VPN recommendations to guarantee larger MTU for the tunnel overhead ([RFC3985] Section 5.3).

* Geneve should provide a recommendation for critical bit processing - text could specify 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, it is recommended that Geneve increase the Single TLV option length to 256.

* Geneve address requirements for OAM considerations for alternate marking and for performance measurements that need a 2 bit field in the header should be considered and the need for the current OAM bit in the Geneve Header clarified.

* The WG should 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, T. Sridhar for his valuable comments and feedback, Anoop Ghanwani for his extensive comments, and Ignas Bagdonas.

9. Security Considerations

This document does not introduce any additional security constraints; however, Section 6.2.2 discusses security/integrity extensions and this document suggests, in Section 7, that the the nvo3 WG work on security options for Geneve.

10. IANA Considerations

This document requires no IANA actions.

11. Normative References

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

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

12. Informative References

[ietf_gue_extensions] Herbert, T., Yong, L., and F. Templin, "Extensions for Generic UDP Encapsulation", work in progress, 8 March 2019, <https://datatracker.ietf.org/doc/draft-ietf-intarea-gue-extensions/>.

[ietf_intarea_gue] Herbert, T., Yong, L., and O. Zia, "Generic UDP Encapsulation", work in progress, 26 October 2019.

[IEEE802.1Q] 802.1 WG, IEEE., "Bridges and Bridged Networks", IEEE Std 802.1Q-2014, 3 November 2014.

[INT] P4.org, "In-band Network Telemetry (INT) Dataplane Specification", November 2020, <https://p4.org/p4-spec/docs/INT_v2_1.pdf>.

[nvo3_vxlan_gpe] Maino, F., Kreeger, L., and U. Elzur, "Generic Protocol Extension for VXLAN (VXLAN-GPE)", work in progress, 4 November 2023, <https://datatracker.ietf.org/doc/draft-ietf-nvo3-vxlan-gpe/>.

[OVN] Network, O. V., "", <https://www.openvswitch.org/>.

[RFC2104] Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, DOI 10.17487/RFC2104, February 1997, <https://www.rfc-editor.org/info/rfc2104>.

[RFC2418]
Bradner, S., "IETF Working Group Guidelines and Procedures", BCP 25, RFC 2418, DOI 10.17487/RFC2418, September 1998, <https://www.rfc-editor.org/info/rfc2418>.

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

[RFC6071] Frankel, S. and S. Krishnan, "IP Security (IPsec) and Internet Key Exchange (IKE) Document Roadmap", RFC 6071, DOI 10.17487/RFC6071, February 2011, <https://www.rfc-editor.org/info/rfc6071>.

[RFC6291] Andersson, L., van Helvoort, H., Bonica, R., Romascanu, D., and S. Mansfield, "Guidelines for the Use of the "OAM" Acronym in the IETF", BCP 161, RFC 6291, DOI 10.17487/RFC6291, June 2011, <https://www.rfc-editor.org/info/rfc6291>.

[RFC7042] Eastlake 3rd, D. and J. Abley, "IANA Considerations and IETF Protocol and Documentation Usage for IEEE 802 Parameters", BCP 141, RFC 7042, DOI 10.17487/RFC7042, October 2013, <https://www.rfc-editor.org/info/rfc7042>.

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

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

[RFC8300] Quinn, P., Ed., Elzur, U., Ed., and C. Pignataro, Ed., "Network Service Header (NSH)", RFC 8300, DOI 10.17487/RFC8300, January 2018, <https://www.rfc-editor.org/info/rfc8300>.

[RFC8365] Sajassi, A., Ed., Drake, J., Ed., Bitar, N., Shekhar, R., Uttaro, J., and W. Henderickx, "A Network Virtualization Overlay Solution Using Ethernet VPN (EVPN)", RFC 8365, DOI 10.17487/RFC8365, March 2018, <https://www.rfc-editor.org/info/rfc8365>.

[RFC8394]
Li, Y., Eastlake 3rd, D., Kreeger, L., Narten, T., and D. Black, "Split Network Virtualization Edge (Split-NVE) Control-Plane Requirements", RFC 8394, DOI 10.17487/RFC8394, May 2018, <https://www.rfc-editor.org/info/rfc8394>.

[RFC8926] Gross, J., Ed., Ganga, I., Ed., and T. Sridhar, Ed., "Geneve: Generic Network Virtualization Encapsulation", RFC 8926, DOI 10.17487/RFC8926, November 2020, <https://www.rfc-editor.org/info/rfc8926>.

[RFC9147] Rescorla, E., Tschofenig, H., and N. Modadugu, "The Datagram Transport Layer Security (DTLS) Protocol Version 1.3", RFC 9147, DOI 10.17487/RFC9147, April 2022, <https://www.rfc-editor.org/info/rfc9147>.

[RFC9197] Brockners, F., Ed., Bhandari, S., Ed., and T. Mizrahi, Ed., "Data Fields for In Situ Operations, Administration, and Maintenance (IOAM)", RFC 9197, DOI 10.17487/RFC9197, May 2022, <https://www.rfc-editor.org/info/rfc9197>.

[VXLANgroup] Smith, M. and L. Kreeger, "VXLAN Group Policy Option", work in progress, 22 October 2018, <https://datatracker.ietf.org/doc/html/draft-smith-vxlan-group-policy-05>.

Appendix A. Encapsulation Comparison

A.1. Overview

This section presents a comparison of the three NVO3 encapsulation proposals, Geneve [RFC8926], GUE [ietf_intarea_gue], and VXLAN-GPE [nvo3_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 A.2 below.

A.2. Extensibility

A.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 [nvo3_vxlan_gpe], extensibility can be attained to some extent if the Network Service Header (NSH) [RFC8300] 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). Note that NSH-over-VXLAN-GPE implies an additional overhead of the 8-octets NSH header, in addition to the VXLAN-GPE header.

A.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 [ietf_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. This length field simplifies the parsing by transit devices that skip the encapsulation header without parsing its extensions.

A.2.3. Critical Extensions

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

A.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.

A.3. Encapsulation Header

A.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 [ietf_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 [ietf_gue_extensions].

A.3.2. Next Protocol

All 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 [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.

A.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 [ietf_intarea_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.

A.4. Comparison Summary

The following table summarizes the comparison between the three NVO3 encapsulations. In some cases a plus sign ("+") or minus sign ("-") is used to indicate that the header is stronger or weaker in an area respectively.

	Geneve	GUE	VXLAN-GPE
Outer transport UDP Port Number	UDP/IP 6081	UDP/IP 6080	UDP/IP 4790
Base header length	8 octets	4 octets	8 octets (16 octets using NSH)
Extensibility	Variable length options	Extension fields	No native ext- ensibility. Might use 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 exte- nsion bit	+	-	-
VNI field size	24 bits	32 bits (extension)	24 bits
Next protocol field	16 bits Ethertype registry	8 bits Internet prot- ocol 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	none	27 bits

Figure 4: NVO3 Encapsulations Comparison

Contributors

The following co-authors have contributed to this document:.

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
Huawei

Email: tal.mizrahi.phd@gmail.com

David Mozes

Email: mosesster@gmail.com

Erik Nordmark
ZEDEDA

Email: nordmark@sonic.net

Michael Smith
Cisco

Email: michsmit@cisco.com

Sam Aldrin
Google

Email: aldrin.ietf@gmail.com

Authors' Addresses

Sami Boutros (editor)
Ciena Corporation
United States of America

Email: sboutros@ciena.com

Donald E. Eastlake 3rd (editor)
Futurewei Technologies
2386 Panoramic Circle
Apopka, Florida 32703
United States of America

Phone: +1-508-333-2270
Email: d3e3e3@gmail.com