Service Function Chaining
Internet-Draft
Intended status: Standards Track
Expires: August 14, 2016

S. Kumar, Ed.
L. Kreeger, Ed.
Cisco Systems, Inc.
S. Majee
F5 Networks
W. Haeffner
Vodafone
R. Manur
Broadcom
D. Melman
Marvell

February 11, 2016

UDP Transport for Network Service Header
draft-kumar-sfc-nsh-udp-transport-02

Abstract

This draft describes the transport encapsulation to carry Network Service Header (NSH) over UDP transport protocol. This enables applications and services using NSH to communicate over a simple layer-3 network without topological constraints. It brings down the barrier to deploy NSH by not requiring additional overhead as is typical of overlay encapsulation mechanisms designed on top of UDP.

As a first benefit, this method eases the deployment of Service Function Chaining (SFC) by allowing SFC components to utilize the basic UDP/IP stack available in virtually all network elements and end systems to setup the virtual network and realize Service Function Chains (SFCs).

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 August 14, 2016.

Table of Contents

1. Introduction 2
1.1. Applicability and Operating Environments 3
1.2. Requirements Language 4
2. Definition Of Terms 4
3. NSH UDP Overlay Transport Encapsulation 5
3.1. Stacking And Layering 5
3.2. NSH UDP Overlay Packet Format 5
4. UDP Encapsulation Considerations 7
4.1. UDP Transport End-points 7
4.2. UDP Source Port Considerations 7
4.3. Checksum Considerations 8
4.3.1. IPv4 Checksum Processing 8
4.3.2. IPv6 Checksum Processing 8
4.3.3. UDP-Lite Considerations 10
4.4. Congestion Considerations 10
4.5. MTU and Fragmentation Considerations 11
4.6. Middlebox Considerations 12
4.7. Differentiated Services and ECN Considerations 12
5. Acknowledgements 13
6. IANA Considerations 13
7. Security Considerations 13
8. References 13
8.1. Normative References 13
8.2. Informative References 15
Authors' Addresses 15

1. Introduction

NSH is an encapsulation designed to carry SFC specific information and metadata. It is very flexible in providing fixed and variable length encapsulation options while allowing for a high degree of

extensibility. NSH in addition allows for carrying a variety of packets as payload, there by acting as a shim header between the inner payload and the outer transport.

NSH focuses on the application aspect of the encapsulation while leaving the transport mechanisms out of scope. This design choice is meant to allow NSH to be carried on any transport as required by the application and the use cases.

The transport independence aspect of NSH makes it necessary for existing transport protocols or new ones to carry NSH encapsulated packet as a payload. Given that IP networks are ubiquitous with virtually every device, element, node connected to the IP network possessing the ability to support UDP datagram transport over IP layer, it is one of the most basic of the transports to carry NSH.

UDP as a transport provides many benefits which has made it the de-facto choice for creating virtual networks such as VxLAN [RFC7348]. By nature it is a datagram service and trades reliability for simplicity and reduced overhead. It allows for sufficient entropy, for the network to exploit, in load balancing packets across paths in the network. Likewise, end hosts exploit it to distribute packets between the NICs and processor cores, for optimum performance. To this end, network elements and end hosts, both hardware and software, implement specific mechanisms to optimize UDP packet processing.

UDP datagram service and efficient implementations of it in existing networks is thus a forgone conclusion. These benefits among others, coupled with extensibility aspect of NSH - to implement security, header verification, etc., makes UDP a very simple, widely available and foundational choice for transporting NSH encapsulated packets.

This draft describes the considerations for the creation of on-demand point-to-point lightweight UDP tunnels to transport NSH encapsulated packets, hereafter abbreviated as NSH-OVER-UDP.

1.1. Applicability and Operating Environments

NSH is an encapsulation carrying control information and metadata in addition to the packet or frame for service delivery. NSH base header contains the next protocol field to specify the payload type encapsulated. Supported payload types include IPv4, IPv6 and Ethernet, not including the experimental types. UDP as a transport for NSH hence is tunneling IPv4, IPv6 and Ethernet packets or frames encapsulated in NSH.

This draft follows the usage guidelines outlined in [I-D.ietf-tsvwg-rfc5405bis] in specifying the usage of UDP as a

transport for NSH. [I-D.ietf-tsvwg-rfc5405bis] offers guidelines for application and protocol designers to consider and adopt when using UDP as a transport. It primarily focusses on congestion control to prevent congestion collapse and to provide fairness for all users of capacity along the path while also providing other recommendations. In particular, it identifies two types of applicability for the specification of applications, such as this draft: 1) General Internet and 2) Controlled Environment. Former is broadly the specification targeting the use of UDP for applications over the Internet, which seems to have become inevitable for successful applications even when those applications start out in limited networks. The latter on the other hand is the specification targeting the use of UDP for applications in a controlled environment. Controlled environments are assumed to be well coordinated and well managed. Further, such environments have additional means, in the form carrier grade or other tools and hardware to manage congestion rather than rely on application built-in mechanisms.

NSH and more broadly SFC, in its initial specification, targets only a single administrative domain and falls into the applicability type 2: controlled environment. It is assumed that SFC is deployed over a single or even multiple connected IP networks that are all under the same administrative domain or cooperating domains. It is thus assumed that these controlled networks are traffic engineered and manage congestion through external means. Deploying SFC over the Internet and hence the use of UDP to carry NSH over the Internet is out of scope for this draft.

1.2. 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 RFC 2119 [RFC2119].

2. Definition Of Terms

This document uses some terms defined in SFC architecture [I-D.ietf-sfc-architecture] and NSH [I-D.ietf-sfc-nsh] drafts as mere examples for ease of understanding.

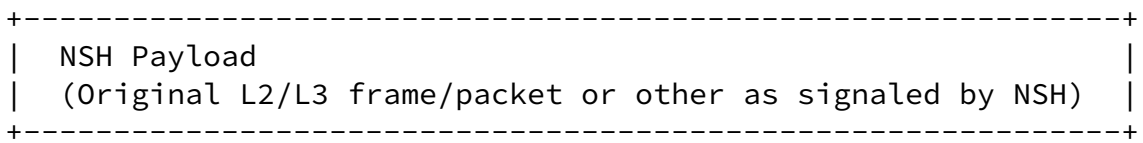
3. NSH UDP Overlay Transport Encapsulation

3.1. Stacking And Layering

A NSH encapsulated packet when carried over an UDP transport looks as depicted in Figure 1.

The original payload, L2 frame, L3 packet, NSH OAM message, etc., is first encapsulated in NSH shim header. The NSH encapsulated packet then becomes the payload for the UDP packet carried over an IPv4 or IPv6 network. The UDP header serves as the L4 transport for NSH and its payload.

Although depicted as a layer3 IP over an L2 network, nothing is assumed about how the L3 network is designed and deployed. It is entirely possible for IPinIP or MPLS or other underpinnings.



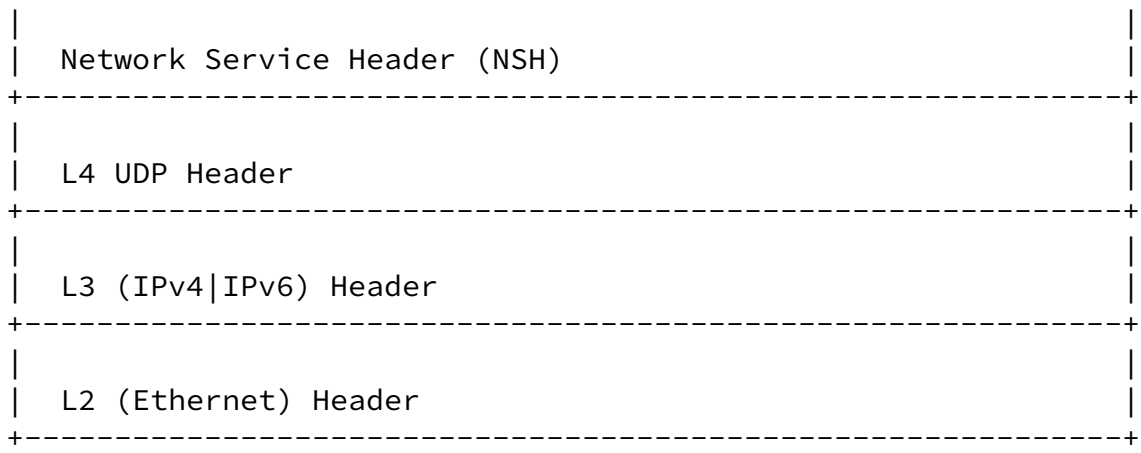
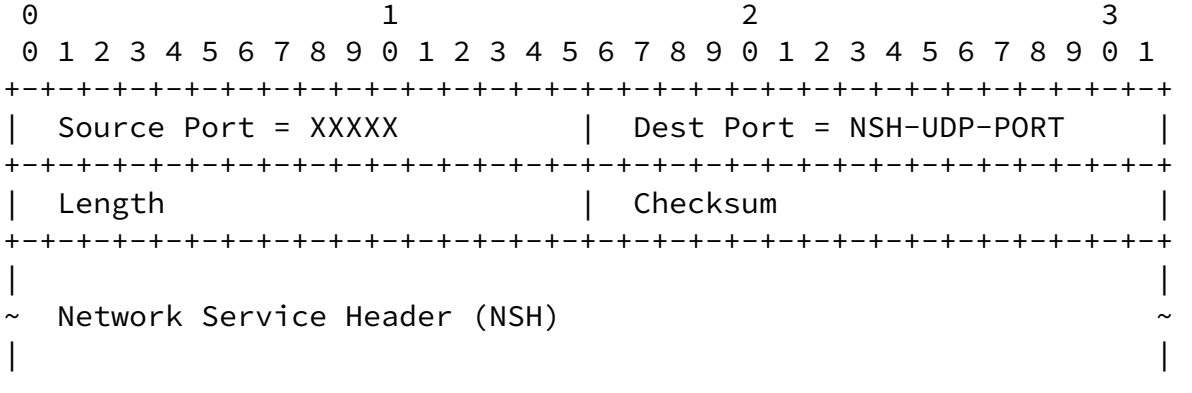


Figure 1: NSH UDP Stack

3.2. NSH UDP Overlay Packet Format

Figure 2 shows the format of the NSH encapsulation transported over UDP.

Rest of the document assumes UDP destination port to be set to NSH-UDP-PORT unless stated otherwise explicitly, when carrying a NSH encapsulated payload packet in UDP transport.



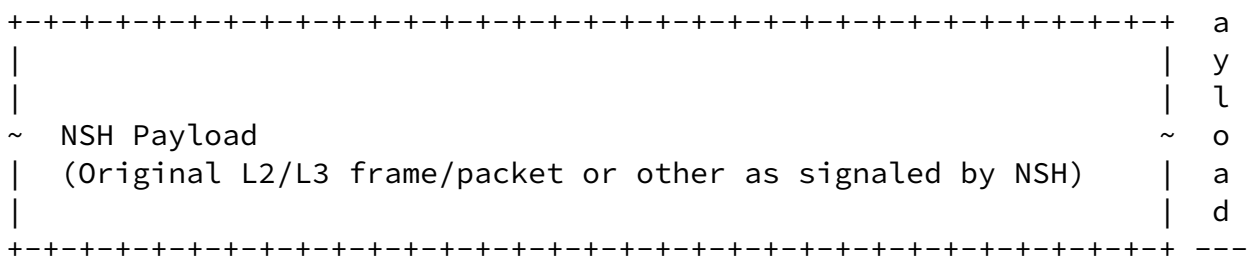


Figure 2: NSH UDP Overlay Encapsulation Format

Source Port :
: The UDP port number computed to provide entropy. See Section 4.2 for details.

Dest Port :
: UDP port number assigned to NSH: NSH-UDP-PORT.

Length :
: Length of the UDP payload. This includes both the UDP header and payload as specified in [RFC0768].

Checksum :
: UDP checksum computed over the pseudo header, NSH and NSH payload or zero. See Section 4.3

NSH :
: The NSH encapsulation header.

NSH Payload :
: The original frame or packet being carried or OAM message, etc.

4. UDP Encapsulation Considerations

4.1. UDP Transport End-points

The UDP transport extends between the two end-points involved in carrying the NSH traffic. The control plane provisioning the NSH encapsulation MUST specify the location of the transport destination when using UDP as the transport, such as the IPv4 or IPv6 address of

the end-point.

In the case of SFC, this UDP transport extends between two SFC components: Classifier and SFF or any two SFFs or SFF and SF or SFF and SFC-proxy. The destination of the UDP transport is thus the IP address used by these components to receive the NSH encapsulated traffic. When UDP transport is required to carry NSH encapsulated traffic, SFC control plane MUST provision the UDP transport destination and the use of UDP as transport.

4.2. UDP Source Port Considerations

The source port used in the UDP transport SHOULD be computed to provide entropy for load balancing along the transmission path, including network elements such as routers and switches as well as end points such as servers. This behavior may in turn be controlled by local-policy at the encapsulating entity.

The source port number SHOULD stay constant and not change for the flow represented within the NSH payload. This is typically done by computing the source UDP port number as a hash over the invariant part of the NSH payload. This could be IP and UDP or IP and TCP part of the NSH payload when the next-protocol field in NSH base header is set to IPv4, for instance. This avoids inducing packet reordering due to the use of NSH UDP transport.

The recommended selection of source port as per [RFC6335], is the dynamic range: 49152-65535. A number in this range SHOULD be selected to reflect the flow contained in NSH payload.

The source port number SHOULT NOT be set to zero by the UDP transport encapsulating entity to mitigate data injection attacks by off-path devices.

In case of carrying UDP over IPv6, network flow label [RFC6437] SHOULD be used with the same value as set in the UDP source port. This allows devices processing NSH encapsulated UDP packets to ECMP [RFC6438] load balance and route at the IP level as opposed to looking inside the UDP header.

The end point receiving and processing the UDP transported NSH

packets SHOULD perform checks to filter packets with invalid source port numbers.

4.3. Checksum Considerations

UDP header checksum is essential to ensure UDP payload is not corrupted. In case of IPv4, IP header checksum enables detection of delivering to the wrong destination. UDP checksum over IPv6 on the other hand enables the detection of UDP payload corruption in addition to the detection of wrong destination. UDP checksum MUST be handled as per [RFC0768] [RFC2460] by both the decapsulator and the encapsulator.

NSH is capable of carrying both IP and non-IP packets. In case of IP packets, NSH payload may already have checksum protection. In such cases the vulnerable portion of the UDP transport carrying NSH is just the NSH header part of the UDP payload. However, given the controlled network environment, this may be low risk and hence checksum protection is optional as per discussion below.

4.3.1. IPv4 Checksum Processing

IPv4 allows for zero checksum and hence the decapsulator MUST accept UDP datagrams received with zero checksum. Checksum in the UDP header MAY be set to zero for performance or other implementation specific reasons by the encapsulator of NSH packet (classifier, SFF, SF-proxy or SF). When a non-zero checksum is set by the encapsulator, it MUST be computed over the IP, UDP headers and the data as defined in the UDP specification [RFC0768]. The receiving entity thus MUST accept a UDP encapsulated NSH packet with non-zero UDP checksum. Receiving entities, of UDP encapsulated NSH packets with non-zero checksum, MUST verify the checksum before accepting the packet.

4.3.2. IPv6 Checksum Processing

IPv6 header does not itself have a checksum but relies on the upper layers such as UDP. UDP over IPv6 hence protects both the source and destination addresses in addition to the payload.

[RFC2460] does not allow the use of zero checksum with UDP. [RFC6935] and [RFC6936] define the guidelines and requirements to be met for using zero checksum with IPv6. Since SFC and NSH are constrained within a single administrative domain, zero checksum method MUST be configurable to override the default option.

The following are the requirements and recommendations for using NSH-OVER-UDP, over an IPv6 transport not performing UDP checksum to verify the integrity of the transport end points.

1. By default, NSH encapsulator node SHOULD use non zero UDP checksum for NSH-OVER-UDP packets.

2. By default, NSH encapsulator node SHOULD NOT send packets with zero checksum for NSH-OVER-UDP packets.

3. By default NSH decapsulator node SHOULD discard NSH-OVER-UDP packets received with zero checksum.

4. NSH encapsulator and decapsulator nodes MUST be configurable to use the zero checksum method for NSH-OVER-UDP packets.

5. When zero checksum method is enabled for use with NSH-OVER-UDP packets, it is RECOMMENDED that it be done for specific IP addresses. The decapsulator MUST check for the validity of the source and destination IP addresses by comparing them against the set of IP address enabled for zero checksum method.

6. NSH encapsulator MAY send both zero and non-zero checksum NSH-OVER-UDP packets to the same destination and the decapsulator nodes MUST receive both zero and non-zero checksum NSH-OVER-UDP packets from the same source

7. A middlebox, if present in the path of NSH-OVER-UDP packets, MUST allow forwarding of NSH-OVER-UDP packets with both zero and non-zero checksum.

8. While using zero checksum, the network administrator MUST ensure that the corruption of packets in the environment is low through means outside the scope of this draft, such as monitoring and analysis of network traffic.

9. Encapsulator and decapsulator components MUST verify the non-zero checksum when in use and provide an integrity mechanism to isolate the cause of corruption when the corruption rate increases.

The above requirements do not change the requirements specified in [RFC2460], further updated in [RFC6935] or, the requirements in [RFC6936] but are adopted for transporting NSH over UDP.

Since NSH is specified for controlled environments, which provides visibility and control over packet corruption in the environment, the

network operator is expected to keep the packet corruption to an acceptably low level. The above requirements further contribute to reducing the corruption rates and hence they are not expected to increase in any way as compared to using UDP win non NSH-OVER-UDP packets in the same environment. Requirements 2, 3 and 5 in section 5 of [RFC6936] are hence satisfied.

Since NSH is an encapsulation header with no requirement on sate maintenance at either the encapsulator or the decapsulator and NSH-OVER-UDP adds no additional state requirements, requirement 4 in section 5 of [RFC6936] is not applicable.

NSH-OVER-UDP has no control feedback mechanism as it does not specify a protocol or additional semantics for its use, requirements 6 and 7 in section 5 of [RFC6936] do not apply.

Since NSH-OVER-UDP is unidirectional, requirement 10 in section 5 of [RFC6936] does not apply.

In summary, NSH-OVER-UDP may use zero checksum method while carried over IPv6 in accordance with the drafts sighted in the above discussion.

4.3.3. UDP-Lite Considerations

NSH when transported over UDP with zero checksum method, either in IPv4 or IPv6 packets, looses the integrity verification provided by the checksum. However, as discussed in Section 4.3.2, deployment in a controlled environment is expected to minimize packet corruption.

NSH payload may consist of packets that may or may not have their own integrity verification mechanisms as in IPv4 or TCP or UDP packets inside NSH. In light of this, if implementations require integrity verification of the payload but want to avoid the redundant integrity checks or, require integrity checks only for the NSH header, should seriously consider UDP-Lite [RFC3828]. UDP-lite shares the UDP name space but uses IP protocol identifier to distinguish itself from UDP.

4.4. Congestion Considerations

Congestion control with a connection less protocol like UDP is a very important consideration to prevent congestion collapse as discussed in depth in [RFC5405] updated by [I-D.ietf-tsvwg-rfc5405bis]. In particular, senders of UDP traffic are expected to control the rate at which packets are sent to a destination in order to minimize the congestions effects along the path to the destination which is shared with other flows between different source and destination tuples.

NSH-OVER-UDP is expected to transport both IP and non IP traffic although former is expected to be the dominant use case. Where IP traffic is carried, it is assumed to be congestion controlled. In such scenarios, additional congestion control in NSH-OVER-UDP is assumed to be unnecessary. However, when non-IP traffic is carried, such as link layer traffic, the rate at which packets are sent to the destination by an NSH-OVER-UDP encapsulator may not be controlled and hence may run into congestion issues in the network impacting throughput of not only other senders and receivers but the throughput of this sender as well. The network operator(s) can minimize or avoid these situations by carful planning to control the rate of transmission of such packets through means outside the scope of NSH-OVER-UDP.

Since NSH-OVER-UDP is expected to be deployed in controlled environments, it is suitable for deployment in such network environments. It MUST NOT be deployed over general Internet unless explicit guarantees are in place to control the sender of such packets to prevent congestion.

The operator of the networks where NSH-OVER-UDP is deployed is expected to impose checks at the egress points of those networks to ensure any traffic that is not congestion controlled does not escape to the Internet.

4.5. MTU and Fragmentation Considerations

Fragmentation severely impacts the performance and efficiency of the elements that process the packet fragments, which includes the routers and middleboxes among others, and the network in general. Fragmentation creates more packets in the network, requires resources in the network elements to buffer and reassemble, which only gets

worse if the fragments are re-ordered (for instance they take different paths in the network), adds processing overhead, to name a few disadvantages. Further, it leads to loss of an entire packet and even flow, if a single fragment is lost. A firewall enforcing policies on the packet content requires entire packet to be reassembled and a loss of a fragment results in dropping the all fragments and blocking the corresponding flow.

An application, as a result, SHOULD NOT send packets that exceed the MTU along the path of the packet. This requires Operators of networks deploying NSH-OVER-UDP are RECOMMENDED to configure the MTU of the network to accommodate NSH and UDP transport encapsulation overhead. This is only feasible when the networks are under single administrative domain or co-operating administrative domains or managed networks. Where it is not possible to set the network-wide MTU to accommodate NSH-OVER-UDP packet overhead, NSH-OVER-UDP

encapsulators SHOULD use path MTU (PMTU) discovery to determine the MTU along the path. Ideally, such PMTU discovery SHOULD be performed by the end application and lower the packet MTU. Such a method fails when the packet traverses multiple administrative domains or the Internet as ICMP messages required for successful operation of PMTU are increasingly being filtered.

Within the same administrative domain, PMTU discovery SHOULD be used by the NSH-OVER-UDP encapsulators to determine the MTU along the path. The determined PMTU MUST over-ride the configured MTU. NSH-OVER-UDP encapsulators MUST fragment the packet being encapsulated prior to encapsulating in UDP. This may result in NSH itself spread across multiple fragments in extreme cases and hence reassembly becomes a requirement to process NSH.

When fragmentation is indeed performed by the NSH-OVER-UDP encapsulators, same source port number MUST be used on all the fragments of the same packet.

4.6. Middlebox Considerations

Middle boxes typically build state and have a notion of flow. Policies are often applied by the operator of such networks and middleboxes to the flow in one direction while expecting the same policy to be applied automatically in the opposite direction of the

flow by the middlebox. State-full behavior of the middleboxes enables such functionality.

NSH-OVER-UDP creates a point-to-point unidirectional tunnel. NSH-UDP-PORT is the destination port while a random port is chosen as the source port as explained in Section 4.2. Since NSH is deployed in a controlled environment, the operator MUST update the middleboxes to allow packets destined to NSH-UDP-PORT. It may further be constrained to specific source and destination IP addresses.

In case of SFC, NSH is used to steer traffic to middleboxes and the middleboxes are expected to parse NSH-OVER-UDP packets to service the NSH payload packets and hence presence of middle boxes are expected.

4.7. Differentiated Services and ECN Considerations

IP Packets carried as payload of NSH inside NSH-OVER-UDP may have differentiated services (DS) [RFC2475] or ECN [RFC3168] or both markings on them. It becomes important to determine the markings for the encapsulating IP packet such as NSH-OVER-UDP when carrying such a marked packet as payload. [RFC2983] and [RFC6040] discuss DS and ECN topics in depth, respectively, as it applies to tunnels.

5. Acknowledgements

The authors would like to thank David Black, Alia Atlas and others for their feedback, review comments and guidance.

6. IANA Considerations

IANA is requested to assign a well-known UDP port number for the purpose defined in this draft, referred to as NSH-UDP-PORT.

7. Security Considerations

Encapsulating NSH in UDP does not alter the security risk of NSH encapsulation and payload and hence security of the payload is as per [I-D.ietf-sfc-nsh]

NSH-OVER-UDP is expected to predominantly carry IP traffic which is checksumed. In increasing number of cases, as in mobile service

provider networks, the traffic is also encrypted. Although it is allowed to use zero UDP checksum with NSH-OVER-UDP, non-zero checksum SHOULD be used to protect against corruption to mitigate privacy concerns.

Use of computed port number for NSH-OVER-UDP source port, as discussed in Section 4.2, provides minimal protection against off-path attacks although it is not a substitute for encryption techniques. However, where source port computation for entropy is disabled a random port SHOULD be selected to mitigate exposure to off-path attacks as described in [RFC6056].

8. References

8.1. Normative References

[I-D.ietf-sfc-nsh]
Quinn, P. and U. Elzur, "Network Service Header", draft-ietf-sfc-nsh-02 (work in progress), January 2016.

[I-D.ietf-tsvwg-rfc5405bis]
Eggert, L., Fairhurst, G., and G. Shepherd, "UDP Usage Guidelines", draft-ietf-tsvwg-rfc5405bis-07 (work in progress), November 2015.

[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>.

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, DOI 10.17487/RFC2460, December 1998, <http://www.rfc-editor.org/info/rfc2460>.

[RFC2475] Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., and W. Weiss, "An Architecture for Differentiated

Services", RFC 2475, DOI 10.17487/RFC2475, December 1998, <http://www.rfc-editor.org/info/rfc2475>.

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

[RFC3168] Ramakrishnan, K., Floyd, S., and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", RFC 3168, DOI 10.17487/RFC3168, September 2001, <http://www.rfc-editor.org/info/rfc3168>.

[RFC3828] Larzon, L-A., Degermark, M., Pink, S., Jonsson, L-E., Ed., and G. Fairhurst, Ed., "The Lightweight User Datagram Protocol (UDP-Lite)", RFC 3828, DOI 10.17487/RFC3828, July 2004, <http://www.rfc-editor.org/info/rfc3828>.

[RFC5405] Eggert, L. and G. Fairhurst, "Unicast UDP Usage Guidelines for Application Designers", BCP 145, RFC 5405, DOI 10.17487/RFC5405, November 2008, <http://www.rfc-editor.org/info/rfc5405>.

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

[RFC6056] Larsen, M. and F. Gont, "Recommendations for Transport-Protocol Port Randomization", BCP 156, RFC 6056, DOI 10.17487/RFC6056, January 2011, <http://www.rfc-editor.org/info/rfc6056>.

[RFC6437] Amante, S., Carpenter, B., Jiang, S., and J. Rajahalme, "IPv6 Flow Label Specification", RFC 6437, DOI 10.17487/RFC6437, November 2011, <http://www.rfc-editor.org/info/rfc6437>.

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

[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>.

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

8.2. Informative References

[I-D.ietf-sfc-architecture] Halpern, J. and C. Pignataro, "Service Function Chaining (SFC) Architecture", draft-ietf-sfc-architecture-11 (work in progress), July 2015.

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

[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>.

Authors' Addresses

Surendra Kumar (editor)
Cisco Systems, Inc.
170 W. Tasman Dr.
San Jose, CA 95134
US

Email: smkumar@cisco.com

Larry Kreeger (editor)
Cisco Systems, Inc.
170 W. Tasman Dr.
San Jose, CA 95134
US

Email: kreeger@cisco.com

Sumandra Majee
F5 Networks
90 Rio Robles
San Jose, CA 95134
US

Email: S.Majee@F5.com

Walter Haeffner
Vodafone
Ferdinand-Braun-Platz 1
Duesseldorf 40549
DE

Email: walter.haeffner@vodafone.com

Rajeev Manur
Broadcom

Email: rmanur@broadcom.com

David Melman
Marvell

Email: davidme@marvell.com