Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 4, 2015

X. Xu
Huawei Technologies
N. Sheth
Juniper Networks
L. Yong
Huawei USA
R. Callon
Juniper Networks
D. Black
EMC Corporation

January 31, 2015

Encapsulating MPLS in UDP
draft-ietf-mpls-in-udp-11

Abstract

This document specifies an IP-based encapsulation for MPLS, called MPLS-in-UDP (User Datagram Protocol) for situations where UDP encapsulation is preferred to direct use of MPLS, e.g., to enable UDP-based ECMP (Equal Cost Multi-Pathing) or link aggregation. The MPLS-in-UDP encapsulation technology must only be deployed within a single network (with a single network operator) or networks of an adjacent set of co-operating network operators where traffic is managed to avoid congestion, rather than over the Internet where congestion control is required. Usage restrictions apply to MPLS-in-UDP usage for traffic that is not congestion controlled and to UDP zero checksum usage with IPv6.

Table of Contents

1. Introduction

This document specifies an IP-based encapsulation for MPLS, i.e. MPLS-in-UDP (User Datagram Protocol), which is applicable in some circumstances where IP-based encapsulation for MPLS is required and further fine-grained load balancing of MPLS packets over IP networks over Equal Cost Multi-Path (ECMP) and/or Link Aggregation Groups (LAG) is required as well. There are already IP-based encapsulations for MPLS that allow for fine-grained load balancing by using some special field in the encapsulation header as an entropy field. However, MPLS-in-UDP can be advantageous since networks have used the

UDP port number fields as a basis for load-balancing solutions for some time.

Like other IP-based encapsulation methods for MPLS, this encapsulation allows for two Label Switching Routers (LSR) to be adjacent on a Label Switched Path (LSP), while separated by an IP network. In order to support this encapsulation, each LSR needs to know the capability to decapsulate MPLS-in-UDP by the remote LSRs. This specification defines only the data plane encapsulation and does not concern itself with how the knowledge to use this encapsulation is conveyed. Specifically, this capability can be either manually configured on each LSR or be dynamically advertised in manners that are outside the scope of this document.

Similarly, the MPLS-in-UDP encapsulation format defined in this document by itself cannot ensure the integrity and privacy of data packets being transported through the MPLS-in-UDP tunnels and cannot enable the tunnel decapsulators to authenticate the tunnel encapsulator. Therefore, in the case where any of the above security issues is concerned, the MPLS-in-UDP SHOULD be secured with IPsec [RFC4301] or DTLS [RFC6347]. For more details, please see Section 6 of Security Considerations.

1.1. Existing Encapsulations

Currently, there are several IP-based encapsulations for MPLS such as MPLS-in-IP, MPLS-in-GRE (Generic Routing Encapsulation) [RFC4023], and MPLS-in-L2TPv3 (Layer Two Tunneling Protocol - Version 3) [RFC4817]. In all these methods, the IP addresses can be varied to achieve load-balancing.

All these IP-based encapsulations for MPLS are specified for both IPv4 and IPv6. In the case of IPv6-based encapsulations, the IPv6 Flow Label can be used for ECMP and LAGs [RFC6438]. However, there is no such entropy field in the IPv4 header.

For MPLS-in-GRE as well as MPLS-in-L2TPv3, protocol fields (the GRE Key and the L2TPv3 Session ID respectively) can be used as the load-balancing key as described in [RFC5640]. For this, intermediate routers need to understand these fields in the context of being used as load-balancing keys.

1.2. Motivations for MPLS-in-UDP Encapsulation

Most existing routers in IP networks are already capable of distributing IP traffic "microflows" [RFC2474] over ECMPs and/or LAG based on the hash of the five-tuple of User Datagram Protocol (UDP) [RFC0768] and Transmission Control Protocol (TCP) packets (i.e.,

source IP address, destination IP address, source port, destination port, and protocol). By encapsulating the MPLS packets into an UDP tunnel and using the source port of the UDP header as an entropy field, the existing load-balancing capability as mentioned above can be leveraged to provide fine-grained load-balancing of MPLS traffic over IP networks.

1.3. Applicability Statements

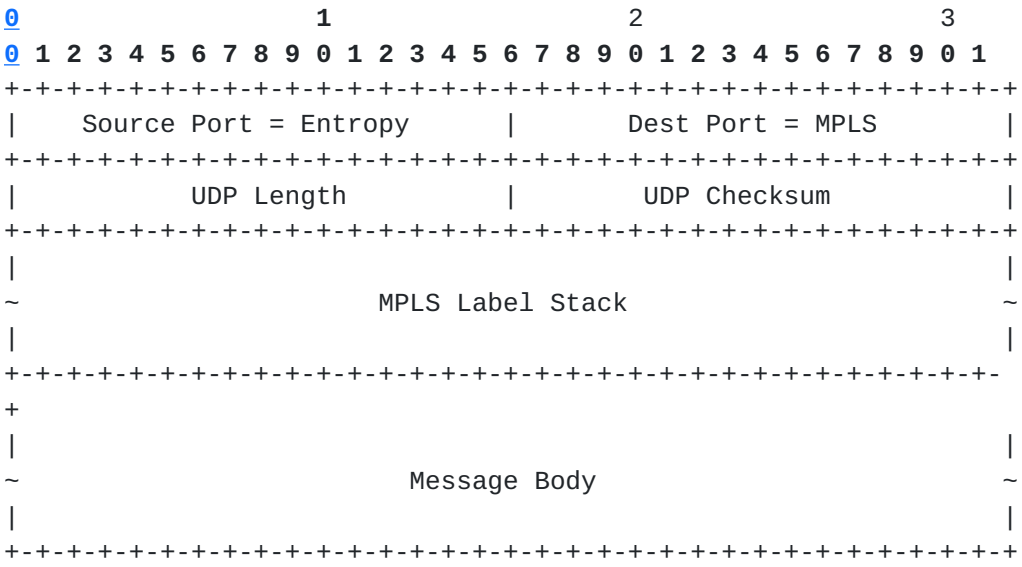
The MPLS-in-UDP encapsulation technology MUST only be deployed within a single network (with a single network operator) or networks of an adjacent set of co-operating network operators where traffic is managed to avoid congestion, rather than over the Internet where congestion control is required. Furthermore, packet filters SHOULD be added to prevent MPLS-in-UDP packets from escaping from the network due to misconfiguation or packet errors.

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. Encapsulation in UDP

MPLS-in-UDP encapsulation format is shown as follows:



Source Port of UDP

This field contains a 16-bit entropy value that is generated by the encapsulator to uniquely identify a flow. What constitutes a flow is locally determined by the encapsulator and therefore

is outside the scope of this document. What algorithm is

actually used by the encapsulator to generate an entropy value is outside the scope of this document.

In case the tunnel does not need entropy, this field of all packets belonging to a given flow SHOULD be set to a randomly selected constant value so as to avoid packet reordering.

To ensure that the source port number is always in the range 49152 to 65535 (Note that those ports less than 49152 are reserved by IANA to identify specific applications/protocols) which may be required in some cases, instead of calculating a 16-bit hash, the encapsulator SHOULD calculate a 14-bit hash and use those 14 bits as the least significant bits of the source port field while the most significant two bits SHOULD be set to binary 11. That still conveys 14 bits of entropy information which would be enough as well in practice.

Destination Port of UDP

This field is set to a value (TBD1) allocated by IANA to indicate that the UDP tunnel payload is an MPLS packet.

UDP Length

The usage of this field is in accordance with the current UDP specification [RFC0768].

UDP Checksum

For IPv4 UDP encapsulation, this field is RECOMMENDED to be set to zero for performance or implementation reasons because the IPv4 header includes a checksum and use of the UDP checksum is optional with IPv4, unless checksum protection of VPN labels is important (See Section 6). For IPv6 UDP encapsulation, the IPv6 header does not include a checksum, so this field MUST contain a UDP checksum that MUST be used as specified in [RFC0768] and [RFC2460] unless one of the exceptions that allows use of UDP zero-checksum mode (as specified in [RFC6935]) applies. See Section 3.1 for specification of these exceptions and additional requirements that apply when UDP zero-checksum mode is used for MPLS-in-UDP traffic over IPv6.

MPLS Label Stack

This field contains an MPLS Label Stack as defined in [RFC3032].

Message Body

This field contains one MPLS message body.

3.1. UDP Checksum Usage with IPv6

When UDP is used over IPv6, the UDP checksum is relied upon to protect both the IPv6 and UDP headers from corruption, and MUST be used unless the requirements in [RFC6935] and [RFC6936] for use of UDP zero-checksum mode with a tunnel protocol are satisfied. MPLS-in-UDP is a tunnel protocol, and there is significant successful deployment of MPLS-in-IPv6 without UDP (i.e., without a checksum that covers the IPv6 header or the MPLS label stack), as described in Section 3.1 of [RFC6936]:

> "There is extensive experience with deployments using tunnel protocols in well-managed networks (e.g., corporate networks and service provider core networks). This has shown the robustness of methods such as Pseudowire Emulation Edge-to-Edge (PWE3) and MPLS that do not employ a transport protocol checksum and that have not specified mechanisms to protect from corruption of the unprotected headers(such as the VPN Identifier in MPLS)".

The UDP checksum MUST be implemented and MUST be used in accordance with [RFC0768] and [RFC2460] for MPLS-in-UDP traffic over IPv6 unless one or more of the following exceptions applies and the additional requirements stated below are also complied with. There are three exceptions that allow use of UDP zero-checksum mode for IPv6 with MPLS-in-UDP, subject to the additional requirements stated below in this section. The three exceptions are:

a. Use of MPLS-in-UDP in networks under single administrative control (such as within a single operator's network) where it is known (perhaps through knowledge of equipment types and lower layer checks) that packet corruption is exceptionally unlikely and where the operator is willing to take the risk of undetected packet corruption.

b. Use of MPLS-in-UDP in networks under single administrative control (such as within a single operator's network) where it is judged through observational measurements (perhaps of historic or current traffic flows that use a non-zero checksum) that the level of packet corruption is tolerably low and where the operator is willing to take the risk of undetected packet corruption.

c. Use of MPLS-in-UDP for traffic delivery for applications that are tolerant of misdelivered or corrupted packets (perhaps through higher layer checksum, validation, and retransmission or

transmission redundancy) where the operator is willing to rely on the applications using the tunnel to survive any corrupt packets.

These exceptions may also be extended to the use of MPLS-in-UDP within a set of closely cooperating network administrations (such as network operators who have agreed to work together in order to jointly provide specific services). Even when one of the above exceptions applies, use of UDP checksums may be appropriate when VPN services are provided over MPLS-in-UDP, see Section 6.

As such, for IPv6, the UDP checksum for MPLS-in-UDP MUST be used as specified in [RFC0768] and [RFC2460] for tunnels that span multiple networks whose network administrations do not cooperate closely, even if each non-cooperating network administration independently satisfies one or more of the exceptions for UDP zero-checksum mode usage with MPLS-in-UDP over IPv6.

The following additional requirements apply to implementation and use of UDP zero-checksum mode for MPLS-in-UDP over IPv6:

a. Use of the UDP checksum with IPv6 MUST be the default configuration of all MPLS-in-UDP implementations.

b. The MPLS-in-UDP implementation MUST comply with all requirements specified in Section 4 of [RFC6936] and with requirement 1 specified in Section 5 of [RFC6936].

c. The tunnel decapsulator SHOULD only allow the use of UDP zero-checksum mode for IPv6 on a single received UDP Destination Port regardless of the encapsulator. The motivation for this requirement is possible corruption of the UDP destination port, which may cause packet delivery to the wrong UDP port. If that other UDP port requires the UDP checksum, the misdelivered packet will be discarded

d. The tunnel decapsulator MUST check that the source and destination IPv6 addresses are valid for the MPLS-in-UDP tunnel on which the packet was received if that tunnel uses UDP zero-checksum mode and discard any packet for which this check fails.

e. The tunnel encapsulator SHOULD use different IPv6 addresses for each MPLS-in-UDP tunnel that uses UDP zero-checksum mode regardless of decapsulator in order to strengthen the decapsulator's check of the IPv6 source address (i.e., the same IPv6 source address SHOULD NOT be used with more than one IPv6 destination address, independent of whether that destination address is a unicast or multicast address). When this is not possible, it is RECOMMENDED to use each source IPv6 address for

as few UDP zero-checksum mode MPLS-in-UDP tunnels (i.e., with as few destination IPv6 addresses) as is feasible.

f. Any middlebox support for MPLS-in-UDP with UDP zero-checksum mode for IPv6 MUST comply with requirements 1 and 8-10 in Section 5 of [RFC6936].

g. Measures SHOULD be taken to prevent IPv6 traffic with zero UDP checksums from "escaping" to the general Internet; see Section 5 for examples of such measures.

h. IPv6 traffic with zero UDP checksums MUST be actively monitored for errors by the network operator.

The above requirements do not change either the requirements specified in [RFC2460] as modified by [RFC6935] or the requirements specified in [RFC6936].

The requirement to check the source IPv6 address in addition to the destination IPv6 address, plus the strong recommendation against reuse of source IPv6 addresses among MPLS-in-UDP tunnels collectively provide some mitigation for the absence of UDP checksum coverage of the IPv6 header. In addition, the MPLS data plane only forwards packets with valid labels (i.e., labels that have been distributed by the tunnel egress LSR), providing some additional opportunity to detect MPLS-in-UDP packet misdelivery when the misdelivered packet contains a label that is not valid for forwarding at the receiving LSR. The expected result for IPv6 UDP zero-checksum mode for MPLS-in-UDP is that corruption of the destination IPv6 address will usually cause packet discard, as offsetting corruptions to the source IPv6 and/or MPLS top label are unlikely. Additional assurance is provided by the restrictions in the above exceptions that limit usage of IPv6 UDP zero-checksum mode to well-managed networks for which MPLS packet corruption has not been a problem in practice.

Hence MPLS-in-UDP is suitable for transmission over lower layers in the well-managed networks that are allowed by the exceptions stated above and the rate of corruption of the inner IP packet on such networks is not expected to increase by comparison to MPLS traffic that is not encapsulated in UDP. For these reasons, MPLS-in-UDP does not provide an additional integrity check when UDP zero-checksum mode is used with IPv6, and this design is in accordance with requirements 2, 3 and 5 specified in Section 5 of [RFC6936].

MPLS does not accumulate incorrect state as a consequence of label stack corruption. A corrupt MPLS label results in either packet discard or forwarding (and forgetting) of the packet without accumulation of MPLS protocol state. Active monitoring of MPLS-in-

UDP traffic for errors is REQUIRED as occurrence of errors will result in some accumulation of error information outside the MPLS protocol for operational and management purposes. This design is in accordance with requirement 4 specified in Section 5 of [RFC6936].

The remaining requirements specified in Section 5 of [RFC6936] are inapplicable to MPLS-in-UDP. Requirements 6 and 7 do not apply because MPLS does not have an MPLS-generic control feedback mechanism. Requirements 8-10 are middlebox requirements that do not apply to MPLS-in-UDP tunnel endpoints, but see Section 3.2 for further middlebox discussion.

In summary, UDP zero-checksum mode for IPv6 is allowed to be used with MPLS-in-UDP when one of the three exceptions specified above applies, provided that the additional requirements specified in this section are complied with. Otherwise the UDP checksum MUST be used for IPv6 as specified in [RFC0768] and [RFC2460].

This entire section and its requirements apply only to use of UDP zero-checksum mode for IPv6; they can be avoided by using the UDP checksum as specified in [RFC0768] and [RFC2460].

3.2. Middlebox Considerations for IPv6 UDP Zero Checksums

IPv6 datagrams with a zero UDP checksum will not be passed by any middlebox that validates the checksum based on [RFC2460] or that updates the UDP checksum field, such as NATs or firewalls. Changing this behavior would require such middleboxes to be updated to correctly handle datagrams with zero UDP checksums. The MPLS-in-UDP encapsulation does not provide a mechanism to safely fall back to using a checksum when a path change occurs redirecting a tunnel over a path that includes a middlebox that discards IPv6 datagrams with a zero UDP checksum. In this case the MPLS-in-UDP tunnel will be black-holed by that middlebox. Recommended changes to allow firewalls, NATs and other middleboxes to support use of an IPv6 zero UDP checksum are described in Section 5 of [RFC6936].

4. Processing Procedures

This MPLS-in-UDP encapsulation causes MPLS packets to be forwarded through "UDP tunnels". When performing MPLS-in-UDP encapsulation by the encapsulator, the entropy value would be generated by the encapsulator and then be filled in the Source Port field of the UDP header. The Destination Port field is set to a value (TBD1) allocated by IANA to indicate that the UDP tunnel payload is an MPLS packet. As for whether the top label of the encapsulated MPLS packet is downstream-assigned or upstream-assigned, it is determined

according to the following criteria which are compatible with [RFC5332]:

a. If the tunnel destination IP address is a unicast address, the top label MUST be downstream-assigned;

b. If the tunnel destination IP address is an IP multicast address, either all encapsulated MPLS packets in the particular tunnel have a downstream-assigned label at the top of the stack, or all encapsulated MPLS packets in that tunnel have an upstream-assigned label at the top of the stack. The means by which this is determined for a particular tunnel is outside the scope of this specification. In the absence of any knowledge about a specific tunnel, the label SHOULD be presumed to be upstream-assigned.

Intermediate routers, upon receiving these UDP encapsulated packets, could balance these packets based on the hash of the five-tuple of UDP packets. Upon receiving these UDP encapsulated packets, the decapsulator would decapsulate them by removing the UDP headers and then process them accordingly. For other common processing procedures associated with tunneling encapsulation technologies including but not limited to Maximum Transmission Unit (MTU) and preventing fragmentation and reassembly, Time to Live (TTL) and differentiated services, the corresponding "Common Procedures" defined in [RFC4023] which are applicable for MPLS-in-IP and MPLS-in-GRE encapsulation formats SHOULD be followed.

Note: Each UDP tunnel is unidirectional, as MPLS-in-UDP traffic is sent to the IANA-allocated UDP Destination Port, and in particular, is never sent back to any port used as a UDP Source Port (which serves solely as a source of entropy). This is at odds with a common middlebox (e.g., firewall) assumption that bidirectional traffic uses a common pair of UDP ports. As a result, arranging to pass bidirectional MPLS-in-UDP traffic through middleboxes may require separate configuration for each direction of traffic.

5. Congestion Considerations

Section 3.1.3 of [RFC5405] discussed the congestion implications of UDP tunnels. As discussed in [RFC5405], because other flows can share the path with one or more UDP tunnels, congestion control [RFC2914] needs to be considered.

A major motivation for encapsulating MPLS in UDP is to improve the use of multipath (such as ECMP) in cases where traffic is to traverse routers which are able to hash on UDP Port and IP address. As such, in many cases this may reduce the occurrence of congestion and

improve usage of available network capacity. However, it is also necessary to ensure that the network, including applications that use the network, responds appropriately in more difficult cases, such as when link or equipment failures have reduced the available capacity.

The impact of congestion must be considered both in terms of the effect on the rest of the network of a UDP tunnel that is consuming excessive capacity, and in terms of the effect on the flows using the UDP tunnels. The potential impact of congestion from a UDP tunnel depends upon what sort of traffic is carried over the tunnel, as well as the path of the tunnel.

MPLS is widely used to carry a wide range of traffic. In many cases MPLS is used to carry IP traffic. IP traffic is generally assumed to be congestion controlled, and thus a tunnel carrying general IP traffic (as might be expected to be carried across the Internet) generally does not need additional congestion control mechanisms. As specified in [RFC5405]:

> "IP-based traffic is generally assumed to be congestion-controlled, i.e., it is assumed that the transport protocols generating IP-based traffic at the sender already employ mechanisms that are sufficient to address congestion on the path. Consequently, a tunnel carrying IP-based traffic should already interact appropriately with other traffic sharing the path, and specific congestion control mechanisms for the tunnel are not necessary".

For this reason, where MPLS-in-UDP tunneling is used to carry IP traffic that is known to be congestion controlled, the UDP tunnels MAY be used within a single network or across multiple networks, with cooperating network operators. Internet IP traffic is generally assumed to be congestion-controlled. Similarly, in general Layer 3 VPNs are carrying IP traffic that is similarly assumed to be congestion controlled.

Whether or not Layer 2 VPN traffic is congestion controlled may depend upon the specific services being offered and what use is being made of the layer 2 services.

However, MPLS is also used in many cases to carry traffic that is not necessarily congestion controlled. For example, MPLS may be used to carry pseudowire or VPN traffic where specific bandwidth guarantees are provided to each pseudowire, or to each VPN.

In such cases network operators may avoid congestion by careful provisioning of their networks, by rate limiting of user data traffic, and/or by using MPLS Traffic Engineering (MPLS-TE) to assign

specific bandwidth guarantees to each LSP. Where MPLS is carried over UDP over IP, the identity of each individual MPLS flow is in general lost and MPLS-TE cannot be used to guarantee bandwidth to specific flows (since many LSPs may be multiplexed over a single UDP tunnel, and many UDP tunnels may be mixed in an IP network).

For this reason, where the MPLS traffic is not congestion controlled, MPLS-in-UDP tunnels MUST only be used within a single operator's network that utilizes careful provisioning (e.g., rate limiting at the entries of the network while over-provisioning network capacity) to ensure against congestion, or within a limited number of networks whose operators closely cooperate in order to jointly provide this same careful provisioning.

As such, MPLS-in-UDP MUST NOT be used over the general Internet, or over non-cooperating network operators, to carry traffic that is not congestion-controlled.

Measures SHOULD be taken to prevent non-congestion-controlled MPLS-in-UDP traffic from "escaping" to the general Internet, e.g.:

a. Physical or logical isolation of the links carrying MPLS-over-UDP from the general Internet.

b. Deployment of packet filters that block the UDP Destination Ports used for MPLS-over-UDP.

c. Imposition of restrictions on MPLS-in-UDP traffic by software tools used to set up MPLS-in-UDP tunnels between specific end systems (as might be used within a single data center).

d. Use of a "Managed Circuit Breaker" for the MPLS traffic as described in [I-D.ietf-tsvwg-circuit-breaker].

6. Security Considerations

The security problems faced with the MPLS-in-UDP tunnel are exactly the same as those faced with MPLS-in-IP and MPLS-in-GRE tunnels [RFC4023]. In other words, the MPLS-in-UDP tunnel as defined in this document by itself cannot ensure the integrity and privacy of data packets being transported through the MPLS-in-UDP tunnel and cannot enable the tunnel decapsulator to authenticate the tunnel encapsulator. In the case where any of the above security issues is concerned, the MPLS-in-UDP tunnel SHOULD be secured with IPsec or DTLS. IPsec was designed as a network security mechanism and therefore it resides at the network layer. As such, if the tunnel is secured with IPsec, the UDP header would not be visible to intermediate routers anymore in either IPsec tunnel or transport

mode. As a result, the meaning of adopting the MPLS-in-UDP tunnel as an alternative to the MPLS-in-GRE or MPLS-in-IP tunnel is lost. By comparison, DTLS is better suited for application security and can better preserve network and transport layer protocol information. Specifically, if DTLS is used, the destination port of the UDP header MUST be set to an IANA-assigned value (TBD2) indicating MPLS-in-UDP with DTLS, and that UDP port MUST NOT be used for other traffic. The UDP source port field can still be used to add entropy, e.g., for load-sharing purposes. DTLS usage is limited to a single DTLS session for any specific tunnel encapsulator/ decapsulator pair (identified by source and destination IP addresses). Both IP addresses MUST be unicast addresses - multicast traffic is not supported when DTLS is used. An MPLS-in-UDP tunnel decapsulator implementation that supports DTLS is expected to be able to establish DTLS sessions with multiple tunnel encapsulators, and likewise an MPLS-in-UDP tunnel encapsulator implementation is expected to be able to establish DTLS sessions with multiple decapsulators (although different source and/or destination IP addresses may be involved - see Section 3.1 for discussion of one situation where use of different source IP addresses is important).

If the tunnel is not secured with IPsec or DTLS, some other method should be used to ensure that packets are decapsulated and forwarded by the tunnel tail only if those packets were encapsulated by the tunnel head. If the tunnel lies entirely within a single administrative domain, address filtering at the boundaries can be used to ensure that no packet with the IP source address of a tunnel endpoint or with the IP destination address of a tunnel endpoint can enter the domain from outside. However, when the tunnel head and the tunnel tail are not in the same administrative domain, this may become difficult, and filtering based on the destination address can even become impossible if the packets must traverse the public Internet. Sometimes only source address filtering (but not destination address filtering) is done at the boundaries of an administrative domain. If this is the case, the filtering does not provide effective protection at all unless the decapsulator of an MPLS-in-UDP validates the IP source address of the packet.

This document does not require that the decapsulator validate the IP source address of the tunneled packets (with the exception that the IPv6 source address MUST be validated when UDP zero-checksum mode is used with IPv6), but it should be understood that failure to do so presupposes that there is effective destination-based (or a combination of source-based and destination-based) filtering at the boundaries. MPLS-based VPN services rely on a VPN label in the MPLS label stack to identify the VPN. Corruption of that label could leak traffic across VPN boundaries. Such leakage is highly undesirable when inter-VPN isolation is used for privacy or security reasons.

When that is the case, UDP checksums SHOULD be used for MPLS-in-UDP with both IPv4 and IPv6, and in particular, UDP zero-checksum mode SHOULD NOT be used with IPv6. Each UDP checksum covers the VPN label, thereby providing increased assurance of isolation among VPNs.

7. IANA Considerations

One UDP destination port number indicating MPLS needs to be allocated by IANA:

Service Name: MPLS-UDP

Transport Protocol(s): UDP

Assignee: IESG <iesg@ietf.org>

Contact: IETF Chair <chair@ietf.org>.

Description: Encapsulate MPLS packets in UDP tunnels.

Reference: This document -- draft-ietf-mpls-in-udp (MPLS WG document).

Port Number: TBD1 -- To be assigned by IANA.

One UDP destination port number indicating MPLS with DTLS needs to be allocated by IANA:

Service Name: MPLS-UDP-DTLS

Transport Protocol(s): UDP

Assignee: IESG <iesg@ietf.org>

Contact: IETF Chair <chair@ietf.org>.

Description: Encapsulate MPLS packets in UDP tunnels with DTLS.

Reference: This document -- draft-ietf-mpls-in-udp (MPLS WG document).

Port Number: TBD2 -- To be assigned by IANA.

8. Contributors

Note that contributors are listed in alphabetical order according to their last names.

Yongbing Fan

China Telecom

Email: fanyb@gsta.com

Adrian Farrel

Juniper Networks

Email: adrian@olddog.co.uk

Zhenbin Li

Huawei Technologies

Email: lizhenbin@huawei.com

Carlos Pignataro

Cisco Systems

Email: cpignata@cisco.com

Curtis Villamizar

Outer Cape Cod Network Consulting, LLC

Email: curtis@occnc.com

9. Acknowledgements

Thanks to Shane Amante, Dino Farinacci, Keshava A K, Ivan Pepelnjak, Eric Rosen, Andrew G. Malis, Kireeti Kompella, Marshall Eubanks, George Swallow, Loa Andersson, Vivek Kumar, Stewart Bryant, Wen Zhang, Joel M. Halpern, Noel Chiappa, Scott Brim, Alia Atlas, Alexander Vainshtein, Joel Jaeggli, Edward Crabbe, Mark Tinka, Lars Eggert, Joe Touch, Lloyd Wood, Gorry Fairhurst, Weiguo Hao, Mark Szczesniak, Stephen Farrell, Zhenxiao Liu and Xing Tong for their valuable comments and suggestions on this document.

Special thanks to Alia Atlas for her insightful suggestion of adding an applicability statement.

Thanks to Daniel King, Gregory Mirsky and Eric Osborne for their valuable MPLS-RT reviews on this document. Thanks to Charlie Kaufman for his SecDir review of this document. Thanks to Nevil Brownlee for the OPSDir review of this document. Thanks to Roni Even for the Gen-ART review of this document. Thanks to Pearl Liang for the IANA review of this documents.

10. References

10.1. Normative References

[RFC0768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, August 1980.

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

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.

[RFC3032] Rosen, E., Tappan, D., Fedorkow, G., Rekhter, Y., Farinacci, D., Li, T., and A. Conta, "MPLS Label Stack Encoding", RFC 3032, January 2001.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

[RFC5332] Eckert, T., Rosen, E., Aggarwal, R., and Y. Rekhter, "MPLS Multicast Encapsulations", RFC 5332, August 2008.

[RFC5405] Eggert, L. and G. Fairhurst, "Unicast UDP Usage Guidelines for Application Designers", BCP 145, RFC 5405, November 2008.

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, January 2012.

[RFC6935] Eubanks, M., Chimento, P., and M. Westerlund, "IPv6 and UDP Checksums for Tunneled Packets", RFC 6935, April 2013.

[RFC6936] Fairhurst, G. and M. Westerlund, "Applicability Statement for the Use of IPv6 UDP Datagrams with Zero Checksums", RFC 6936, April 2013.

10.2. Informative References

[I-D.ietf-tsvwg-circuit-breaker]
Fairhurst, G., "Network Transport Circuit Breakers", draft-ietf-tsvwg-circuit-breaker-00 (work in progress), September 2014.

[RFC2474] Nichols, K., Blake, S., Baker, F., and D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", RFC 2474, December 1998.

[RFC2914] Floyd, S., "Congestion Control Principles", BCP 41, RFC 2914, September 2000.

[RFC4023] Worster, T., Rekhter, Y., and E. Rosen, "Encapsulating MPLS in IP or Generic Routing Encapsulation (GRE)", RFC 4023, March 2005.

[RFC4817] Townsley, M., Pignataro, C., Wainner, S., Seely, T., and J. Young, "Encapsulation of MPLS over Layer 2 Tunneling Protocol Version 3", RFC 4817, March 2007.

[RFC5640] Filsfils, C., Mohapatra, P., and C. Pignataro, "Load-Balancing for Mesh Softwires", RFC 5640, August 2009.

[RFC6438] Carpenter, B. and S. Amante, "Using the IPv6 Flow Label for Equal Cost Multipath Routing and Link Aggregation in Tunnels", RFC 6438, November 2011.

Authors' Addresses

Xiaohu Xu
Huawei Technologies
No.156 Beiqing Rd
Beijing 100095
CHINA

Phone: +86-10-60610041
Email: xuxiaohu@huawei.com

Nischal Sheth
Juniper Networks
1194 N. Mathilda Ave
Sunnyvale, CA 94089
USA

Email: nsheth@juniper.net

Lucy Yong
Huawei USA
5340 Legacy Dr
Plano, TX 75025
USA

Email: Lucy.yong@huawei.com

Ross Callon
Juniper Networks
10 Technology Park Drive
Westford, MA 01886
USA

Email: rcallon@juniper.net

David Black
EMC Corporation
176 South Street
Hopkinton, MA 01748
USA

Email: david.black@emc.com