

V6OPS Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: December 7, 2018

G. Fioccola  
Telecom Italia  
G. Van de Velde  
Nokia  
M. Cociglio  
Telecom Italia  
P. Muley  
Nokia  
June 5, 2018

IPv6 Performance Measurement with Alternate Marking Method  
draft-fioccola-v6ops-ipv6-alt-mark-01

Abstract

This document describes how the alternate marking method in [RFC8321] can be used as the passive performance measurement method in an IPv6 domain, and will discuss the strengths and the weaknesses of the implementation options available to network operations. It proposes how to extend [RFC7837] to apply alternate marking technique.

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].

Status of This Memo

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

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

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

This Internet-Draft will expire on December 7, 2018.

## Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

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

## Table of Contents

1. Introduction . . . . .	2
2. IPv6 application of Alternate Marking . . . . .	3
2.1. IPv6 Extension Headers as Marking Field . . . . .	3
2.2. Other Possibilities . . . . .	5
2.2.1. IPv6 Addresses as Marking Field . . . . .	5
2.2.2. IPv6 Flow Label as Marking Field . . . . .	5
3. Alternate Marking Method Operation . . . . .	6
3.1. Single Mark Measurement . . . . .	6
3.2. Double Mark Measurement . . . . .	7
4. Security Considerations . . . . .	7
5. IANA Considerations . . . . .	7
6. Acknowledgements . . . . .	7
7. References . . . . .	7
7.1. Normative References . . . . .	7
7.2. Informative References . . . . .	7
Authors' Addresses . . . . .	9

## 1. Introduction

This document reports a summary on the possible implementation options for the application of the alternate marking method in an IPv6 domain.

[RFC8321] describes passive performance measurement method, which can be used to measure packet loss, latency and jitter on live traffic. Because this method is based on marking consecutive batches of packets the method often referred as Alternate Marking Method.

This document defines how the alternate marking method can be used to measure packet loss and delay metrics of IPv6 tunneled packets or SRv6 policies.

The IPv6 Header Format defined in [RFC8200] introduces the format of IPv6 addresses, the Extension Headers in the base IPv6 Header and the availability of a 20-bit flow label, that can be considered for the application of the Alternate Marking methodology.

## 2. IPv6 application of Alternate Marking

The application of the alternate marking requires a marking field. The alternatives that can be taken into consideration for the choice of the marking field are the following:

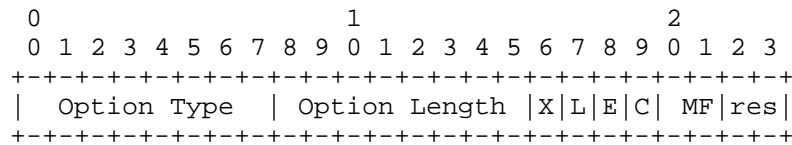
- o Extension Header
- o IPv6 Address
- o Flow Label

### 2.1. IPv6 Extension Headers as Marking Field

A new type of EH may be a solution space proposal (e.g. [RFC8250] and [RFC7837] give a chance).

A possibility can be to use a Hop-By-Hop(HBH) Extension Header(EH). The assumption is that a HBH EH with an alternate marking measurement option can be defined. The router processing can be optimized to handle this use case.

Using a new EH assumes that ALL routers in the domain support this type of headers, which complicates backward compatibility of the technology. The extension of an existing EH (e.g. [RFC7837]) can overcome this issue.



Mark Field (MF) is:

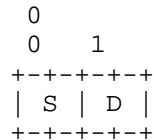


Figure 1: ConEx HBH Option Layout with Mark Field

where:

- o S - Single mark method;
- o D - Double mark method.

The Figure 1 defines a new ConEx HBH (Hop-By-Hop) Option Layout.

This proposal starts from ConEx Destination Option Layout defined in [RFC7837], where the Reserved (res) field is made by four bits that are not used in that specification, in fact they are set to zero by the sender and are ignored by the receiver.

This document aims to introduce the Mark Field (2 bits from 4 bits res field). So the Mark Field (MF) reduces the number of Reserved bits and the Reserved (res) field is now made by 2 bits.

It is important to highlight that the Destination Option Layout is used as Hop-By-Hop Option Layout, since the alternate marking methodology in [RFC8321] allows, by definition, Hop-By-Hop performance measurements.

[I-D.krishnan-conex-ipv6] also tried to introduce a ConEx HBH Options and inspired this proposal.

[I-D.fear-ippm-mpdm] introduces Marking Performance and Diagnostic Metrics (M-PDM) and aims to combine [RFC8250] with [RFC8321], while the extension of [RFC7837], proposed in this document, is optimized to include only marking method without any considerations on how to report and manage, this can be done in-band or out-of-band depending on the case.

## 2.2. Other Possibilities

This section reports the other possibilities that have been discussed.

### 2.2.1. IPv6 Addresses as Marking Field

There is an advantage of using destination addresses (DA) to encode the alternate marking method. In addition to identifying a host, a destination address is also and more fundamentally identifying an exit point from the forwarding domain. It indicates where processing for forwarding to the DA stops, and where other processing of the packet is to occur. Using the DA to encode this alternate marking processing means that it is easy to retrofit into existing devices and models. There is no need to replace existing IPv6 forwarding devices, because they already support DA based forwarding.

However using DA for marking seems a lot expensive.

### 2.2.2. IPv6 Flow Label as Marking Field

Considering the Flow Label, [RFC6294] makes a survey of Proposed Use Cases for the IPv6 Flow Label. The flow label is an immutable field recommended to contain a pseudo-random value, however, often it has the default value of zero. [RFC6436] and [RFC6437] open the door for IPv6 Flow Label to be used in a controlled environment and [RFC6438] describes the use of the IPv6 Flow Label field for load distribution purpose, especially across Equal Cost Multi-Path (ECMP) and/or Link Aggregation Group (LAG) paths. In addition it is possible to mention [I-D.krishnan-6man-header-reserved-bits] that tried to set aside 4 bits from the flow label field for future expansion.

There are few drawbacks to use Flow Label instead of an EH solution or IPv6 Addresses for IPv6 alternate marking, in particular an easier backward compatibility and less bits on the wire. In this way nothing breaks if a transit router does not have the capability of understanding the Flow Label context.

Since the flow-label based load balancing has been defined, the application of the Alternate Marking method to the flow label could be realised with two fundamental assumptions:

- o The original flow-label reconstructed when leaving the controlled domain.
- o The usage of IPv6 tunnels (IPv6inIPv6, IPSec, IPv6 UDP, etc..) or SRv6 policies.

In this case, the controlled domain reflects to the fact that it is a network operator choice that grabs control of packet handling within its own network. In fact, regarding the flow label, four options can be supposed:

- 1) Just do not do anything with Flow Label (leave it default).
- 2) Entropy only and NO alternate marking for performance measurements.
- 3) Alternate marking only and NO usage of entropy.
- 4) Alternate marking and entropy (in this case the entropy SHOULD be based upon a subset of bits because otherwise paths may be changed when the marking changes).

### 3. Alternate Marking Method Operation

[RFC8321] describes in detail the methodology, that we briefly illustrate also here.

#### 3.1. Single Mark Measurement

As explained in the [RFC8321], marking can be applied to delineate blocks of packets based either on equal number of packets in a block or based on equal time interval. The latter method offers better control as it allows better account for capabilities of downstream nodes to report statistics related to batches of packets and, at the same time, time resolution that affects defect detection interval.

If the Single Mark measurement used, then the D flag MUST be set to zero on transmit and ignored by monitoring point.

The S flag is used to create alternate flows to measure the packet loss by switching value of the S flag. Delay metrics MAY be calculated with the alternate flow using any of the following methods:

- o First/Last Batch Packet Delay calculation: timestamps are collected based on order of arrival so this method is sensitive to packet loss and re-ordering.
- o Average Packet Delay calculation: an average delay is calculated by considering the average arrival time of the packets within a single block. This method only provides single metric for the duration of the block and it doesn't give information about the delay distribution.

### 3.2. Double Mark Measurement

Double Mark method allows more detailed measurement of delays for the monitored flow but it requires more nodal and network resources. If the Double Mark method used, then the S flag MUST be used to create the alternate flow. The D flag MUST be used to mark single packets to measure delay jitter.

The first marking (S flag alternation) is needed for packet loss and also for average delay measurement. The second marking (D flag is put to one) creates a new set of marked packets that are fully identified and dedicated for delay. This method is useful to have not only the average delay but also to know more about the statistic distribution of delay values.

### 4. Security Considerations

tbc

### 5. IANA Considerations

tbc

### 6. Acknowledgements

The authors would like to thank Fred Baker, Ole Troan, Robert Hinden, Suresh Krishnan, Brian Carpenter, Roberta Maglione, Tom Herbert, Mark Smith, Joel Halpern, Fernando Gont, Xiaohu Xu and Joel Jaeggli for their comments and feedbacks.

### 7. References

#### 7.1. Normative References

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

#### 7.2. Informative References

[I-D.fear-ippm-mpdm]  
Elkins, N., Fioccola, G., and m. mackermann@bcbsm.com, "IPv6 Marking and Performance and Diagnostic Metrics (MPDM)", draft-fear-ippm-mpdm-00 (work in progress), June 2018.

- [I-D.krishnan-6man-header-reserved-bits]  
Krishnan, S. and J. Halpern, "Reserving bits in the IPv6 header for future use", draft-krishnan-6man-header-reserved-bits-00 (work in progress), October 2010.
- [I-D.krishnan-conex-ipv6]  
Krishnan, S., Kuehlewind, M., and C. Ucendo, "Options for Conex marking in IPv6 packets", draft-krishnan-conex-ipv6-02 (work in progress), March 2011.
- [RFC6294] Hu, Q. and B. Carpenter, "Survey of Proposed Use Cases for the IPv6 Flow Label", RFC 6294, DOI 10.17487/RFC6294, June 2011, <<https://www.rfc-editor.org/info/rfc6294>>.
- [RFC6436] Amante, S., Carpenter, B., and S. Jiang, "Rationale for Update to the IPv6 Flow Label Specification", RFC 6436, DOI 10.17487/RFC6436, November 2011, <<https://www.rfc-editor.org/info/rfc6436>>.
- [RFC6437] Amante, S., Carpenter, B., Jiang, S., and J. Rajahalme, "IPv6 Flow Label Specification", RFC 6437, DOI 10.17487/RFC6437, November 2011, <<https://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, <<https://www.rfc-editor.org/info/rfc6438>>.
- [RFC7837] Krishnan, S., Kuehlewind, M., Briscoe, B., and C. Ralli, "IPv6 Destination Option for Congestion Exposure (ConEx)", RFC 7837, DOI 10.17487/RFC7837, May 2016, <<https://www.rfc-editor.org/info/rfc7837>>.
- [RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <<https://www.rfc-editor.org/info/rfc8200>>.
- [RFC8250] Elkins, N., Hamilton, R., and M. Ackermann, "IPv6 Performance and Diagnostic Metrics (PDM) Destination Option", RFC 8250, DOI 10.17487/RFC8250, September 2017, <<https://www.rfc-editor.org/info/rfc8250>>.



[RFC8321] Fioccola, G., Ed., Capello, A., Cociglio, M., Castaldelli, L., Chen, M., Zheng, L., Mirsky, G., and T. Mizrahi, "Alternate-Marking Method for Passive and Hybrid Performance Monitoring", RFC 8321, DOI 10.17487/RFC8321, January 2018, <<https://www.rfc-editor.org/info/rfc8321>>.

Authors' Addresses

Giuseppe Fioccola  
Telecom Italia  
Torino  
Italy

Email: [giuseppe.fioccola@telecomitalia.it](mailto:giuseppe.fioccola@telecomitalia.it)

Gunter Van de Velde  
Nokia  
Antwerp  
BE

Email: [gunter.van\\_de\\_velde@nokia.com](mailto:gunter.van_de_velde@nokia.com)

Mauro Cociglio  
Telecom Italia  
Torino  
Italy

Email: [mauro.cociglio@telecomitalia.it](mailto:mauro.cociglio@telecomitalia.it)

Praveen Muley  
Nokia  
Mountain View  
USA

Email: [praveen.muley@nokia.com](mailto:praveen.muley@nokia.com)

IPv6 maintenance Working Group (6man)  
Internet-Draft  
Intended status: Standards Track  
Expires: September 25, 2018

F. Gont  
SI6 Networks / UTN-FRH  
C. Huitema  
Private Octopus Inc.  
S. Krishnan  
Ericsson Research  
G. Gont  
SI6 Networks  
M. Garcia Corbo  
SITRANS  
March 24, 2018

Recommendation on Temporary IPv6 Interface Identifiers  
draft-gont-6man-non-stable-iids-04

Abstract

This document specifies a set of requirements for generating temporary addresses, and clarifies the stability requirements for IPv6 addresses, allowing for the use of only temporary addresses.

Status of This Memo

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

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

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

This Internet-Draft will expire on September 25, 2018.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents

carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	2
2. Terminology . . . . .	3
3. Problem statement . . . . .	3
4. Stability Requirements for IPv6 Addresses . . . . .	4
5. Requirements for Temporary IPv6 Addresses . . . . .	4
6. Future Work . . . . .	6
7. IANA Considerations . . . . .	6
8. Security Considerations . . . . .	6
9. Acknowledgements . . . . .	7
10. References . . . . .	7
Authors' Addresses . . . . .	10

## 1. Introduction

IPv6 Stateless Address AutoConfiguration (SLAAC) [RFC4862] has traditionally resulted in stable addresses, since the Interface Identifier (IID) has been generated by embedding a stable layer-2 numeric identifier (e.g., a MAC address). [RFC4941] originally implied, throughout the specification, that temporary addresses are generated and employed along with stable addresses.

While the use of stable addresses (only) or mixed stable and temporary addresses can be desirable in a number of scenarios, there are other scenarios in which, for security and privacy reasons, a node may want to use only temporary address (e.g., a temporary address).

On the other hand, the lack of a formal set of requirements for temporary addresses led to a number of flaws in popular implementations and in the protocol specification itself, such as allowing for the correlation of network activity carried out with different addresses, reusing randomized identifiers across different networks, etc.

This document clarifies the requirements for stability of IPv6 addresses, such that nodes are not required to configure stable addresses, and may instead employ only temporary addresses. It also specifies a set of requirements for the generation of temporary addresses.

## 2. Terminology

Statistically different:

When two values are required to be "statistically different", it means that the equality of those values cannot be caused by anything else other than random chance.

This document employs the definitions of "stable address" and "temporary address" from [RFC7721].

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. Problem statement

When [RFC4941] was written, its authors wanted to prevent privacy and security attacks enabled by addresses that contain "an embedded interface identifier, which remains constant over time". They observed that "Anytime a fixed identifier is used in multiple contexts, it becomes possible to correlate seemingly unrelated activity using this identifier." They were concerned with both on-path attackers who would observe the IP addresses of packets observed in transit, and attackers that would have access to the logs of servers.

Since the publication of [RFC4941] in September 2007, our understanding of threats and mitigations has evolved. The IETF is now officially concerned with Pervasive Monitoring [RFC7258], as well as the wide spread collection of information for advertising and other purposes, for example through the Real Time Bidding protocol used for advertising auctions [RTB25].

### 3.1. Privacy requirements

The widespread deployment of encryption advocated in [RFC7624] is a response to Pervasive Monitoring. Encryption of communication reduces the amount of information that can be collected by monitoring data links, but does not prevent monitoring of IPv6 addresses embedded in clear text packet headers. Stable IPv6 addresses enable the correlation of such data over time.

MAC Address Randomization [IETFMACRandom] is another response to pervasive monitoring. In conjunction with DHCP Anonymity [RFC7844], it ensures that devices cannot be tracked by their MAC Address or their DHCP identifiers when they connect to "hot spots". However, the privacy effects of MAC Address Randomization would be nullified

if a device kept using the same IPv6 address before and after a MAC-address randomization event.

Many Web Browsers have options enabling browsing "in private". However, if the web connections during the private mode use the same IPv6 address as those in the public mode, web tracking systems similar to [RTB25] will quickly find the correlation between the public persona of the user and the supposedly private connection. Similarly, many web browsers have options to "delete history", including deleting "cookies" and other persistent data. Again, if the same IPv6 address is used before and after the deletion of cookies, web tracking systems will easily correlate the new activity with the prior data collection.

Using temporary address alone may not be sufficient to prevent all forms of tracking. It is however quite clear that some usage of temporary addresses is necessary to provide user privacy. It is also clear that the usage of temporary addresses needs to be synchronized with other privacy defining event such as moving to a new network, performing MAC Address Randomization, or changing the privacy posture of a node.

#### 4. Stability Requirements for IPv6 Addresses

Nodes are not required to generate addresses with any specific stability properties. That is, the generation of stable addresses is OPTIONAL. This means that a node may end up configuring only stable addresses, only temporary, or both stable and temporary addresses.

#### 5. Requirements for Temporary IPv6 Addresses

The requirements for temporary IPv6 addresses are as follows:

1. Temporary addresses MUST have a limited lifetime (limited "valid lifetime" and "preferred lifetime" from [RFC4862]), that should be statistically different for different addresses. The lifetime of an address essentially limits the extent to which network activity correlation can be performed for such address.
2. The lifetime of an address MUST be further reduced when privacy-meaningful events (such as a node attaching to a new network) takes place.
3. The resulting Interface Identifiers MUST be statistically different when addresses are configured for different prefixes. That is, when temporary addresses are generated for different autoconfiguration prefixes for the same network interface, the resulting Interface Identifiers must be statistically different.

This means that, given two addresses that employ different prefixes, it must be difficult for an outside entity to tell whether the addresses correspond to the same network interface or even whether they have been generated by the same host.

4. It must be difficult for an outside entity to predict the Interface Identifiers that will be employed for temporary addresses, even with knowledge of the algorithm/method employed to generate them and/or knowledge of the Interface Identifiers previously employed for other temporary addresses.
5. The resulting Interface Identifiers MUST be semantically opaque [RFC7136] and MUST NOT follow any specific patterns.

By definition, temporary addresses have a limited lifetime. This is in contrast with e.g. stable addresses [RFC7217], that are not expected to become invalid under normal circumstances. Employing statistically different lifetimes for different addresses prevents an observer from synchronizing with the temporary address regeneration; that is, from being able to predict when a temporary address will become invalid and a new one regenerated, and thus being able to infer that one newly observed address is actually the result of regenerating a previously observed one.

The lifetime of an address should be further reduced by privacy-meaningful events. For example, a host must not employ the same address across network attachment events. That is, a host that de-attaches from a network and subsequently re-attaches to a (possibly different) network should regenerate all of its temporary addresses. Similarly, a host that implements MAC address randomization should regenerate all of its temporary addresses. Failure to regenerate temporary addresses upon such events would allow the correlation of network activity across such events (e.g., correlation of network activity as a host moves from one network to another). Other events, such as those discussed in Section 3.1 should also trigger the regeneration of all temporary addresses.

Temporary addresses configured for different prefixes should employ statistically different interface identifiers. In general, the reuse of identifiers across different contexts or scopes can be detrimental for security and privacy [I-D.gont-predictable-numeric-ids] [RFC6973] [RFC4941]. For example, a node that deterministically employs the same interface identifier for generating temporary addresses for different prefixes will allow the correlation of network activity.

For security and privacy reasons, the IIDs generated for temporary addresses must be unpredictable by an outside entity. Otherwise, the node may be subject to many (if not all) of the security and privacy

issues that temporary addresses are expected to mitigate (please see [RFC7721]).

Any semantics or patterns in an IID might be leveraged by an attacker to e.g. reduce the search space when performing address-scanning attacks (see [RFC7707], infer the identity of the node, etc.

NOTE:

In the above text, where the "lifetime" of different addresses is required to be statistically different, or where the interface identifiers for different temporary addresses is required to be statistically different, the goal is that an implementation must not deterministically employ the same such values for different addresses. For example, where interface identifiers for different temporary addresses are required to be statistically different, the goal is to e.g. prevent an implementation from computing a single random interface identifier and employing such identifier for the generation of temporary addresses for other prefixes for the same network interface (as was the case with the algorithm specified in [RFC4941]). Therefore, a node is neither required nor expected to e.g. enforce that a newly-generated random interface identifier is not currently employed by any other temporary address configured by the node, or that such interface identifier has not been previously employed for any other temporary address configured by the node.

## 6. Future Work

This document clarifies the requirements for stability requirements for IPv6 addresses, and specifies requirements for temporary addresses. A separate document ([I-D.gont-taps-address-usage-problem-statement]) discusses the trade-offs involved when considering different stability properties of IPv6 addresses.

## 7. IANA Considerations

There are no IANA registries within this document. The RFC-Editor can remove this section before publication of this document as an RFC.

## 8. Security Considerations

This document clarifies the stability requirements for IPv6 addresses, and specifies requirements for the generation of temporary addresses.

The security and privacy properties of IPv6 addresses have been discussed in detail in [RFC7721] and [RFC7707].

## 9. Acknowledgements

The authors would like to thank (in alphabetical order) Brian Carpenter, Lorenzo Colitti, and David Plonka, for providing valuable feedback on earlier versions of this document.

## 10. References

### 10.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC4086] Eastlake 3rd, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", BCP 106, RFC 4086, DOI 10.17487/RFC4086, June 2005, <<https://www.rfc-editor.org/info/rfc4086>>.
- [RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, DOI 10.17487/RFC4291, February 2006, <<https://www.rfc-editor.org/info/rfc4291>>.
- [RFC4862] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, DOI 10.17487/RFC4862, September 2007, <<https://www.rfc-editor.org/info/rfc4862>>.
- [RFC4941] Narten, T., Draves, R., and S. Krishnan, "Privacy Extensions for Stateless Address Autoconfiguration in IPv6", RFC 4941, DOI 10.17487/RFC4941, September 2007, <<https://www.rfc-editor.org/info/rfc4941>>.
- [RFC5453] Krishnan, S., "Reserved IPv6 Interface Identifiers", RFC 5453, DOI 10.17487/RFC5453, February 2009, <<https://www.rfc-editor.org/info/rfc5453>>.
- [RFC7136] Carpenter, B. and S. Jiang, "Significance of IPv6 Interface Identifiers", RFC 7136, DOI 10.17487/RFC7136, February 2014, <<https://www.rfc-editor.org/info/rfc7136>>.



- [RFC7217] Gont, F., "A Method for Generating Semantically Opaque Interface Identifiers with IPv6 Stateless Address Autoconfiguration (SLAAC)", RFC 7217, DOI 10.17487/RFC7217, April 2014, <<https://www.rfc-editor.org/info/rfc7217>>.
- [RFC8064] Gont, F., Cooper, A., Thaler, D., and W. Liu, "Recommendation on Stable IPv6 Interface Identifiers", RFC 8064, DOI 10.17487/RFC8064, February 2017, <<https://www.rfc-editor.org/info/rfc8064>>.

## 10.2. Informative References

- [FIPS-SHS] NIST, "Secure Hash Standard (SHS)", FIPS Publication 180-4, March 2012, <<http://csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf>>.
- [I-D.gont-predictable-numeric-ids] Gont, F. and I. Arce, "Security and Privacy Implications of Numeric Identifiers Employed in Network Protocols", draft-gont-predictable-numeric-ids-02 (work in progress), February 2018.
- [I-D.gont-taps-address-usage-problem-statement] Gont, F., Gont, G., Corbo, M., and C. Huitema, "Problem Statement Regarding IPv6 Address Usage", draft-gont-taps-address-usage-problem-statement-00 (work in progress), February 2018.
- [IANA-RESERVED-IID] IANA, "Reserved IPv6 Interface Identifiers", <<http://www.iana.org/assignments/ipv6-interface-ids>>.
- [IETFMACRandom] Zuniga, JC., "MAC Privacy", November 2014, <<http://www.ietf.org/blog/2014/11/mac-privacy/>>.
- [OPEN-GROUP] The Open Group, "The Open Group Base Specifications Issue 7 / IEEE Std 1003.1-2008, 2016 Edition", Section 4.16 Seconds Since the Epoch, 2016, <<http://pubs.opengroup.org/onlinepubs/9699919799/basedefs/contents.html>>.

- [RAID2015] Ullrich, J. and E. Weippl, "Privacy is Not an Option: Attacking the IPv6 Privacy Extension", International Symposium on Recent Advances in Intrusion Detection (RAID), 2015, <<https://www.sba-research.org/wp-content/uploads/publications/Ullrich2015Privacy.pdf>>.
- [RFC1321] Rivest, R., "The MD5 Message-Digest Algorithm", RFC 1321, DOI 10.17487/RFC1321, April 1992, <<https://www.rfc-editor.org/info/rfc1321>>.
- [RFC3041] Narten, T. and R. Draves, "Privacy Extensions for Stateless Address Autoconfiguration in IPv6", RFC 3041, DOI 10.17487/RFC3041, January 2001, <<https://www.rfc-editor.org/info/rfc3041>>.
- [RFC6059] Krishnan, S. and G. Daley, "Simple Procedures for Detecting Network Attachment in IPv6", RFC 6059, DOI 10.17487/RFC6059, November 2010, <<https://www.rfc-editor.org/info/rfc6059>>.
- [RFC6151] Turner, S. and L. Chen, "Updated Security Considerations for the MD5 Message-Digest and the HMAC-MD5 Algorithms", RFC 6151, DOI 10.17487/RFC6151, March 2011, <<https://www.rfc-editor.org/info/rfc6151>>.
- [RFC6973] Cooper, A., Tschofenig, H., Aboba, B., Peterson, J., Morris, J., Hansen, M., and R. Smith, "Privacy Considerations for Internet Protocols", RFC 6973, DOI 10.17487/RFC6973, July 2013, <<https://www.rfc-editor.org/info/rfc6973>>.
- [RFC7258] Farrell, S. and H. Tschofenig, "Pervasive Monitoring Is an Attack", BCP 188, RFC 7258, DOI 10.17487/RFC7258, May 2014, <<https://www.rfc-editor.org/info/rfc7258>>.
- [RFC7624] Barnes, R., Schneier, B., Jennings, C., Hardie, T., Trammell, B., Huitema, C., and D. Borkmann, "Confidentiality in the Face of Pervasive Surveillance: A Threat Model and Problem Statement", RFC 7624, DOI 10.17487/RFC7624, August 2015, <<https://www.rfc-editor.org/info/rfc7624>>.
- [RFC7707] Gont, F. and T. Chown, "Network Reconnaissance in IPv6 Networks", RFC 7707, DOI 10.17487/RFC7707, March 2016, <<https://www.rfc-editor.org/info/rfc7707>>.

- [RFC7721] Cooper, A., Gont, F., and D. Thaler, "Security and Privacy Considerations for IPv6 Address Generation Mechanisms", RFC 7721, DOI 10.17487/RFC7721, March 2016, <<https://www.rfc-editor.org/info/rfc7721>>.
- [RFC7844] Huitema, C., Mrugalski, T., and S. Krishnan, "Anonymity Profiles for DHCP Clients", RFC 7844, DOI 10.17487/RFC7844, May 2016, <<https://www.rfc-editor.org/info/rfc7844>>.
- [RTB25] Interactive Advertising Bureau (IAB), "Real Time Bidding (RTB) project, OpenRTB API Specification Version 2.5", December 2016, <<http://www.iab.com/wp-content/uploads/2016/03/OpenRTB-API-Specification-Version-2-5-FINAL.pdf>>.

## Authors' Addresses

Fernando Gont  
SI6 Networks / UTN-FRH  
Evaristo Carriego 2644  
Haedo, Provincia de Buenos Aires 1706  
Argentina

Phone: +54 11 4650 8472  
Email: [fgont@si6networks.com](mailto:fgont@si6networks.com)  
URI: <http://www.si6networks.com>

Christian Huitema  
Private Octopus Inc.  
Friday Harbor, WA 98250  
U.S.A.

Email: [huitema@huitema.net](mailto:huitema@huitema.net)  
URI: <http://privateoctopus.com/>

Suresh Krishnan  
Ericsson Research  
8400 Decarie Blvd.  
Town of Mount Royal, QC  
Canada

Email: [suresh.krishnan@ericsson.com](mailto:suresh.krishnan@ericsson.com)

Guillermo Gont  
SI6 Networks  
Evaristo Carriego 2644  
Haedo, Provincia de Buenos Aires 1706  
Argentina

Phone: +54 11 4650 8472  
Email: [ggont@si6networks.com](mailto:ggont@si6networks.com)  
URI: <https://www.si6networks.com>

Madeleine Garcia Corbo  
Servicios de Informacion del Transporte  
Neptuno 358  
Havana City 10400  
Cuba

Email: [madelen.garcial6@gmail.com](mailto:madelen.garcial6@gmail.com)

IPv6 Maintenance (6man) Working Group  
Internet-Draft  
Obsoletes: rfc4941 (if approved)  
Intended status: Standards Track  
Expires: September 6, 2018

F. Gont  
SI6 Networks / UTN-FRH  
S. Krishnan  
Ericsson Research  
March 5, 2018

Privacy Extensions for Stateless Address Autoconfiguration in IPv6  
draft-gont-6man-rfc4941bis-00

Abstract

Nodes use IPv6 stateless address autoconfiguration to generate addresses using a combination of locally available information and information advertised by routers. Addresses are formed by combining network prefixes with an interface identifier. This document describes an extension that causes nodes to generate global scope addresses from interface identifiers that change over time. Changing the interface identifier (and the global scope addresses generated from it) over time makes it more difficult for eavesdroppers and other information collectors to identify when different addresses used in different transactions actually correspond to the same node.

Status of this Memo

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

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

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

This Internet-Draft will expire on September 6, 2018.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of

publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	3
1.1. Terminology . . . . .	3
1.2. Problem Statement . . . . .	3
2. Background . . . . .	5
2.1. Extended Use of the Same Identifier . . . . .	5
2.2. Possible Approaches . . . . .	6
3. Protocol Description . . . . .	8
3.1. Assumptions . . . . .	8
3.2. Generation Of Randomized Interface Identifiers . . . . .	9
3.2.1. Randomized IIDs . . . . .	9
3.2.2. Hash-based generation of randomized interface identifiers . . . . .	9
3.3. Generating Temporary Addresses . . . . .	11
3.4. Expiration of Temporary Addresses . . . . .	12
3.5. Regeneration of Randomized Interface Identifiers . . . . .	13
3.6. Deployment Considerations . . . . .	14
4. Implications of Changing Interface Identifiers . . . . .	16
5. Defined Constants . . . . .	17
6. Future Work . . . . .	18
7. Security Considerations . . . . .	19
8. Significant Changes from RFC RFC4941 . . . . .	20
9. Acknowledgements . . . . .	21
10. References . . . . .	22
10.1. Normative References . . . . .	22
10.2. Informative References . . . . .	23
Authors' Addresses . . . . .	25

## 1. Introduction

Stateless address autoconfiguration [RFC4862] defines how an IPv6 node generates addresses without the need for a DHCPv6 server.

The security and privacy implications of such addresses have been discussed in great detail in [RFC7721],[RFC7217], and RFC7707.

Section 2 provides background information on the issue. Section 3 describes a procedure for generating alternate interface identifiers and global scope addresses. Section 4 discusses implications of changing interface identifiers. The term "global scope addresses" is used in this document to collectively refer to "Global unicast addresses" as defined in [RFC4291] and "Unique local addresses" as defined in [RFC4193].

### 1.1. Terminology

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

The terms "Public address", "stable address", "temporary address", "constant IID", "Stable IID", and "Temporary IID" are to be interpreted as specified in [RFC7721].

### 1.2. Problem Statement

Addresses generated using Stateless address autoconfiguration [RFC4862] contain an embedded interface identifier, which remains stable over time. Anytime a fixed identifier is used in multiple contexts, it becomes possible to correlate seemingly unrelated activity using this identifier.

The correlation can be performed by

- o An attacker who is in the path between the node in question and the peer(s) it is communicating to, and can view the IPv6 addresses present in the datagrams.
- o An attacker who can access the communication logs of the peers with which the node has communicated.

Since the identifier is embedded within the IPv6 address, which is a fundamental requirement of communication, it cannot be easily hidden. This document proposes a solution to this issue by generating interface identifiers which vary over time.

Note that an attacker, who is on path, may be able to perform significant correlation based on

- o The payload contents of the packets on the wire
- o The characteristics of the packets such as packet size and timing

Use of temporary addresses will not prevent such payload based correlation.



## 2. Background

This section discusses the problem in more detail, provides context for evaluating the significance of the concerns in specific environments and makes comparisons with existing practices.

### 2.1. Extended Use of the Same Identifier

The use of a non-changing interface identifier to form addresses is a specific instance of the more general case where a constant identifier is reused over an extended period of time and in multiple independent activities. Anytime the same identifier is used in multiple contexts, it becomes possible for that identifier to be used to correlate seemingly unrelated activity. For example, a network sniffer placed strategically on a link across which all traffic to/from a particular host crosses could keep track of which destinations a node communicated with and at what times. Such information can in some cases be used to infer things, such as what hours an employee was active, when someone is at home, etc. Although it might appear that changing an address regularly in such environments would be desirable to lessen privacy concerns, it should be noted that the network prefix portion of an address also serves as a constant identifier. All nodes at (say) a home, would have the same network prefix, which identifies the topological location of those nodes. This has implications for privacy, though not at the same granularity as the concern that this document addresses. Specifically, all nodes within a home could be grouped together for the purposes of collecting information. If the network contains a very small number of nodes, say just one, changing just the interface identifier will not enhance privacy at all, since the prefix serves as a constant identifier.

One of the requirements for correlating seemingly unrelated activities is the use (and reuse) of an identifier that is recognizable over time within different contexts. IP addresses provide one obvious example, but there are more. Many nodes also have DNS names associated with their addresses, in which case the DNS name serves as a similar identifier. Although the DNS name associated with an address is more work to obtain (it may require a DNS query) the information is often readily available. In such cases, changing the address on a machine over time would do little to address the concerns raised in this document, unless the DNS name is changed as well (see Section 4).

Web browsers and servers typically exchange "cookies" with each other [COOKIES]. Cookies allow web servers to correlate a current activity with a previous activity. One common usage is to send back targeted advertising to a user by using the cookie supplied by the browser to

identify what earlier queries had been made (e.g., for what type of information). Based on the earlier queries, advertisements can be targeted to match the (assumed) interests of the end-user.

The use of a constant identifier within an address is of special concern because addresses are a fundamental requirement of communication and cannot easily be hidden from eavesdroppers and other parties. Even when higher layers encrypt their payloads, addresses in packet headers appear in the clear. Consequently, if a mobile host (e.g., laptop) accessed the network from several different locations, an eavesdropper might be able to track the movement of that mobile host from place to place, even if the upper layer payloads were encrypted.

The security and privacy implications of IPv6 addresses are discussed in detail in [RFC7721], [RFC7707], and [RFC7217].

## 2.2. Possible Approaches

One way to avoid having a stable non-changing address is to use DHCPv6 [DHCPV6] for obtaining addresses. Section 12 of [DHCPV6] discusses the use of DHCPv6 for the assignment and management of "temporary addresses", which are never renewed and provide the same property of temporary addresses described in this document with regards to the privacy concern.

Another approach, compatible with the stateless address autoconfiguration architecture, would be to change the interface identifier portion of an address over time. Changing the interface identifier can make it more difficult to look at the IP addresses in independent transactions and identify which ones actually correspond to the same node, both in the case where the routing prefix portion of an address changes and when it does not.

Many machines function as both clients and servers. In such cases, the machine would need a DNS name for its use as a server. Whether the address stays fixed or changes has little privacy implication since the DNS name remains constant and serves as a constant identifier. When acting as a client (e.g., initiating communication), however, such a machine may want to vary the addresses it uses. In such environments, one may need multiple addresses: a "stable" address and public address registered in the DNS, that is used to accept incoming connection requests from other machines, and a "temporary" address used to shield the identity of the client when it initiates communication. These two cases are roughly analogous to telephone numbers and caller ID, where a user may list their telephone number in the public phone book, but disable the display of its number via caller ID when initiating calls.

On the other hand, a machine that functions only as a client may want to employ only temporary addresses for public communication.

To make it difficult to make educated guesses as to whether two different interface identifiers belong to the same node, the algorithm for generating alternate identifiers must include input that has an unpredictable component from the perspective of the outside entities that are collecting information.

[I-D.gont-6man-non-stable-iids] specifies requirements for temporary addresses. This document specifies a number of algorithms for generating temporary addresses that comply with the aforementioned requirements.

### 3. Protocol Description

The goal of this section is to define procedures that:

1. Do not result in any changes to the basic behavior of addresses generated via stateless address autoconfiguration [RFC4862].
2. Create temporary addresses based on an unpredictable interface identifier for the purpose of initiating outgoing sessions. These temporary addresses would be used for a short period of time (hours to days) and would then be deprecated. Deprecated address can continue to be used for already established connections, but are not used to initiate new connections. New temporary addresses are generated periodically to replace temporary addresses that expire, with the exact time between address generation a matter of local policy.
3. Produce a sequence of temporary global scope addresses from a sequence of interface identifiers that appear to be random in the sense that it is difficult for an outside observer to predict a future address (or identifier) based on a current one and it is difficult to determine previous addresses (or identifiers) knowing only the present one.
4. By default, generate one address for each prefix to be employed for stateless address autoconfiguration.

#### 3.1. Assumptions

The following algorithm assumes that for a given temporary address, an implementation can determine the prefix from which it was generated. When a temporary address is deprecated, a new temporary address is generated. The specific valid and preferred lifetimes for the new address are dependent on the corresponding lifetime values set for the prefix from which it was generated.

Finally, this document assumes that when a node initiates outgoing communication, temporary addresses can be given preference over stable addresses (if available), when the device is configured to do so. [RFC6724] mandates implementations to provide a mechanism, which allows an application to configure its preference for temporary addresses over stable addresses. It also allows for an implementation to prefer temporary addresses by default, so that the connections initiated by the node can use temporary addresses without requiring application-specific enablement. This document also assumes that an API will exist that allows individual applications to indicate whether they prefer to use temporary or stable addresses and override the system defaults.

### 3.2. Generation Of Randomized Interface Identifiers

The following subsections specify some possible algorithms for generating temporary interface identifiers that comply with the requirements in [I-D.gont-6man-non-stable-iids].

#### 3.2.1. Randomized IIDs

One possible approach would be to select a pseudorandom number of the appropriate length. A node employing this algorithm should generate IIDs as follows:

1. Obtain a random number (see [RFC4086] for randomness requirements for security)
2. The Interface Identifier is obtained by taking as many bits from the aforementioned random number (obtained in the previous step) as necessary.

We note that [RFC4291] requires that the Interface IDs of all unicast addresses (except those that start with the binary value 000) be 64 bits long. However, the method discussed in this document could be employed for generating Interface IDs of any arbitrary length, albeit at the expense of reduced entropy (when employing Interface IDs smaller than 64 bits).

3. The resulting Interface Identifier SHOULD be compared against the reserved IPv6 Interface Identifiers [RFC5453] [IANA-RESERVED-IID] and against those Interface Identifiers already employed in an address of the same network interface and the same network prefix. In the event that an unacceptable identifier has been generated, a new interface identifier should be generated, by repeating the algorithm from the first step.

#### 3.2.2. Hash-based generation of randomized interface identifiers

The algorithm in [RFC7217] can be augmented for the generation of temporary addresses. The benefit of this would be that a node could employ a single algorithm for generating stable and temporary addresses, by employing appropriate parameters.

Nodes would employ the following algorithm for generating the temporary IID:

1. Compute a random identifier with the expression:

RID = F(Prefix, MAC\_Address, Network\_ID, Time, DAD\_Counter, secret\_key)

Where:

RID:

Random Identifier

F():

A pseudorandom function (PRF) that MUST NOT be computable from the outside (without knowledge of the secret key). F() MUST also be difficult to reverse, such that it resists attempts to obtain the secret\_key, even when given samples of the output of F() and knowledge or control of the other input parameters. F() SHOULD produce an output of at least 64 bits. F() could be implemented as a cryptographic hash of the concatenation of each of the function parameters. SHA-1 [FIPS-SHS] and SHA-256 are two possible options for F(). Note: MD5 [RFC1321] is considered unacceptable for F() [RFC6151].

Prefix:

The prefix to be used for SLAAC, as learned from an ICMPv6 Router Advertisement message.

MAC\_Address:

The MAC address corresponding to the underlying network interface card. Employing the MAC address in this expression (in replacement of the Net\_Iface parameter of the expression in RFC7217) means that the re-generation of a randomized MAC address will result in a different temporary address.

Network\_ID:

Some network-specific data that identifies the subnet to which this interface is attached -- for example, the IEEE 802.11 Service Set Identifier (SSID) corresponding to the network to which this interface is associated. Additionally, Simple DNA [RFC6059] describes ideas that could be leveraged to generate a Network\_ID parameter. This parameter is SHOULD be employed if some form of "Network\_ID" is available.

Time:

An implementation-dependent representation of time. One possible example is the representation in UNIX-like systems [OPEN-GROUP], that measure time in terms of the number of seconds elapsed since the Epoch (00:00:00 Coordinated Universal Time (UTC), 1 January 1970).

DAD\_Counter:

A counter that is employed to resolve Duplicate Address Detection (DAD) conflicts.

**secret\_key:**

A secret key that is not known by the attacker. The secret key SHOULD be of at least 128 bits. It MUST be initialized to a pseudo-random number (see [RFC4086] for randomness requirements for security) when the operating system is installed or when the IPv6 protocol stack is "bootstrapped" for the first time.

2. The Interface Identifier is finally obtained by taking as many bits from the RID value (computed in the previous step) as necessary, starting from the least significant bit. The resulting Interface Identifier SHOULD be compared against the reserved IPv6 Interface Identifiers [RFC5453] [IANA-RESERVED-IID] and against those Interface Identifiers already employed in an address of the same network interface and the same network prefix. In the event that an unacceptable identifier has been generated, the value DAD\_Counter should be incremented by 1, and the algorithm should be restarted from the first step.

### 3.3. Generating Temporary Addresses

[RFC4862] describes the steps for generating a link-local address when an interface becomes enabled as well as the steps for generating addresses for other scopes. This document extends [RFC4862] as follows. When processing a Router Advertisement with a Prefix Information option carrying a global scope prefix for the purposes of address autoconfiguration (i.e., the A bit is set), the node implementing this specification MUST perform the following steps:

1. Process the Prefix Information Option as defined in [RFC4862], either creating a new stable address or adjusting the lifetimes of existing addresses, both stable and temporary. If a received option will extend the lifetime of a stable address, the lifetimes of temporary addresses should be extended, subject to the overall constraint that no temporary addresses should ever remain "valid" or "preferred" for a time longer than  $(TEMP\_VALID\_LIFETIME - DESYNC\_FACTOR)$  or  $(TEMP\_PREFERRED\_LIFETIME - DESYNC\_FACTOR)$  respectively. The configuration variables `TEMP_VALID_LIFETIME` and `TEMP_PREFERRED_LIFETIME` correspond to approximate target lifetimes for temporary addresses.
2. One way an implementation can satisfy the above constraints is to associate with each temporary address a creation time (called `CREATION_TIME`) that indicates the time at which the address was created. When updating the preferred lifetime of an existing temporary address, it would be set to expire at whichever time is earlier: the time indicated by the received lifetime or  $(CREATION\_TIME + TEMP\_PREFERRED\_LIFETIME - DESYNC\_FACTOR)$ . A

similar approach can be used with the valid lifetime.

3. When a new stable address is created as described in [RFC4862], or if the node has not configured any temporary address for the corresponding prefix, the node SHOULD create a new temporary address for such prefix.
4. When creating a temporary address, the lifetime values MUST be derived from the corresponding prefix as follows:
  - \* Its Valid Lifetime is the lower of the Valid Lifetime of the stable address (if available) or TEMP\_VALID\_LIFETIME
  - \* Its Preferred Lifetime is the lower of the Preferred Lifetime of the stable address (if available) or TEMP\_PREFERRED\_LIFETIME - DESYNC\_FACTOR.
5. A temporary address is created only if this calculated Preferred Lifetime is greater than REGEN\_ADVANCE time units. In particular, an implementation MUST NOT create a temporary address with a zero Preferred Lifetime.
6. New temporary addresses MUST be created by appending the interface's current randomized interface identifier to the prefix that was received.
7. The node MUST Perform duplicate address detection (DAD) on the generated temporary address. If DAD indicates the address is already in use, the node MUST generate a new randomized interface identifier, and repeat the previous steps as appropriate up to TEMP\_IDGEN\_RETRIES times. If after TEMP\_IDGEN\_RETRIES consecutive attempts no non-unique address was generated, the node MUST log a system error and MUST NOT attempt to generate temporary addresses for that interface. Note that DAD MUST be performed on every unicast address generated from this randomized interface identifier.

### 3.4. Expiration of Temporary Addresses

When a temporary address becomes deprecated, a new one MUST be generated. This is done by repeating the actions described in Section 3.3, starting at step 3). Note that, except for the transient period when a temporary address is being regenerated, in normal operation at most one temporary address per prefix should be in a non-deprecated state at any given time on a given interface. Note that if a temporary address becomes deprecated as result of processing a Prefix Information Option with a zero Preferred Lifetime, then a new temporary address MUST NOT be generated. To



ensure that a preferred temporary address is always available, a new temporary address SHOULD be regenerated slightly before its predecessor is deprecated. This is to allow sufficient time to avoid race conditions in the case where generating a new temporary address is not instantaneous, such as when duplicate address detection must be run. The node SHOULD start the address regeneration process `REGEN_ADVANCE` time units before a temporary address would actually be deprecated.

As an optional optimization, an implementation MAY remove a deprecated temporary address that is not in use by applications or upper-layers as detailed in Section 6.

### 3.5. Regeneration of Randomized Interface Identifiers

The frequency at which temporary addresses changes depends on how a device is being used (e.g., how frequently it initiates new communication) and the concerns of the end user. The most egregious privacy concerns appear to involve addresses used for long periods of time (weeks to months to years). The more frequently an address changes, the less feasible collecting or coordinating information keyed on interface identifiers becomes. Moreover, the cost of collecting information and attempting to correlate it based on interface identifiers will only be justified if enough addresses contain non-changing identifiers to make it worthwhile. Thus, having large numbers of clients change their address on a daily or weekly basis is likely to be sufficient to alleviate most privacy concerns.

There are also client costs associated with having a large number of addresses associated with a node (e.g., in doing address lookups, the need to join many multicast groups, etc.). Thus, changing addresses frequently (e.g., every few minutes) may have performance implications.

Nodes following this specification SHOULD generate new temporary addresses on a periodic basis. This can be achieved automatically by generating a new randomized interface identifier at least once every  $(\text{TEMP\_PREFERRED\_LIFETIME} - \text{REGEN\_ADVANCE} - \text{DESYNC\_FACTOR})$  time units. As described above, generating a new temporary address `REGEN_ADVANCE` time units before a temporary address becomes deprecated produces addresses with a preferred lifetime no larger than `TEMP_PREFERRED_LIFETIME`. The value `DESYNC_FACTOR` is a random value (different for each client) that ensures that clients don't synchronize with each other and generate new addresses at exactly the same time. When the preferred lifetime expires, a new temporary address MUST be generated using the new randomized interface identifier.

Because the precise frequency at which it is appropriate to generate new addresses varies from one environment to another, implementations SHOULD provide end users with the ability to change the frequency at which addresses are regenerated. The default value is given in TEMP\_PREFERRED\_LIFETIME and is one day. In addition, the exact time at which to invalidate a temporary address depends on how applications are used by end users. Thus, the suggested default value of one week (TEMP\_VALID\_LIFETIME) may not be appropriate in all environments. Implementations SHOULD provide end users with the ability to override both of these default values.

Finally, when an interface connects to a new link, a new set of temporary addresses MUST be generated immediately. If a device moves from one ethernet to another, generating a new set of temporary addresses ensures that the device uses different randomized interface identifiers for the temporary addresses associated with the two links, making it more difficult to correlate addresses from the two different links as being from the same node. The node MAY follow any process available to it, to determine that the link change has occurred. One such process is described by Detecting Network Attachment [DNA].

### 3.6. Deployment Considerations

Devices implementing this specification MUST provide a way for the end user to explicitly enable or disable the use of temporary addresses. In addition, a site might wish to disable the use of temporary addresses in order to simplify network debugging and operations. Consequently, implementations SHOULD provide a way for trusted system administrators to enable or disable the use of temporary addresses.

Additionally, sites might wish to selectively enable or disable the use of temporary addresses for some prefixes. For example, a site might wish to disable temporary address generation for "Unique local" [RFC4193] prefixes while still generating temporary addresses for all other global prefixes. Another site might wish to enable temporary address generation only for the prefixes 2001::/16 and 2002::/16 while disabling it for all other prefixes. To support this behavior, implementations SHOULD provide a way to enable and disable generation of temporary addresses for specific prefix subranges. This per-prefix setting SHOULD override the global settings on the node with respect to the specified prefix subranges. Note that the pre-prefix setting can be applied at any granularity, and not necessarily on a per subnet basis.

The use of temporary addresses may cause unexpected difficulties with some applications. As described below, some servers refuse to accept

communications from clients for which they cannot map the IP address into a DNS name. In addition, some applications may not behave robustly if temporary addresses are used and an address expires before the application has terminated, or if it opens multiple sessions, but expects them to all use the same addresses.

If a very small number of nodes (say only one) use a given prefix for extended periods of time, just changing the interface identifier part of the address may not be sufficient to ensure privacy, since the prefix acts as a constant identifier. The procedures described in this document are most effective when the prefix is reasonably non static or is used by a fairly large number of nodes.

#### 4. Implications of Changing Interface Identifiers

The desires of protecting individual privacy versus the desire to effectively maintain and debug a network can conflict with each other. Having clients use addresses that change over time will make it more difficult to track down and isolate operational problems. For example, when looking at packet traces, it could become more difficult to determine whether one is seeing behavior caused by a single errant machine, or by a number of them.

Some servers refuse to grant access to clients for which no DNS name exists. That is, they perform a DNS PTR query to determine the DNS name, and may then also perform an AAAA query on the returned name to verify that the returned DNS name maps back into the address being used. Consequently, clients not properly registered in the DNS may be unable to access some services. As noted earlier, however, a node's DNS name (if non-changing) serves as a constant identifier. The wide deployment of the extension described in this document could challenge the practice of inverse-DNS-based "authentication," which has little validity, though it is widely implemented. In order to meet server challenges, nodes could register temporary addresses in the DNS using random names (for example a string version of the random address itself).

Use of the extensions defined in this document may complicate debugging and other operational troubleshooting activities. Consequently, it may be site policy that temporary addresses should not be used. Consequently, implementations **MUST** provide a method for the end user or trusted administrator to override the use of temporary addresses.

## 5. Defined Constants

Constants defined in this document include:

TEMP\_VALID\_LIFETIME -- Default value: 1 week. Users should be able to override the default value.

TEMP\_PREFERRED\_LIFETIME -- Default value: 1 day. Users should be able to override the default value.

REGEN\_ADVANCE -- 5 seconds

MAX\_DESYNC\_FACTOR -- 10 minutes. Upper bound on DESYNC\_FACTOR.

DESYNC\_FACTOR -- A random value within the range 0 - MAX\_DESYNC\_FACTOR. It is computed once at system start (rather than each time it is used) and must never be greater than (TEMP\_VALID\_LIFETIME - REGEN\_ADVANCE).

TEMP\_IDGEN\_RETRIES -- Default value: 3

## 6. Future Work

An implementation might want to keep track of which addresses are being used by upper layers so as to be able to remove a deprecated temporary address from internal data structures once no upper layer protocols are using it (but not before). This is in contrast to current approaches where addresses are removed from an interface when they become invalid [RFC4862], independent of whether or not upper layer protocols are still using them. For TCP connections, such information is available in control blocks. For UDP-based applications, it may be the case that only the applications have knowledge about what addresses are actually in use. Consequently, an implementation generally will need to use heuristics in deciding when an address is no longer in use.

The determination as to whether to use stable versus temporary addresses can in some cases only be made by an application. For example, some applications may always want to use temporary addresses, while others may want to use them only in some circumstances or not at all. Suitable API extensions will likely need to be developed to enable individual applications to indicate with sufficient granularity their needs with regards to the use of temporary addresses. See [I-D.gont-taps-address-usage-problem-statement] for further details. Recommendations on DNS practices to avoid the problem described in Section 4 when reverse DNS lookups fail may be needed. [DNSOP] contains a more detailed discussion of the DNS related issues.

While this document discusses ways of obscuring a user's IP address, the method described is believed to be ineffective against sophisticated forms of traffic analysis. To increase effectiveness, one may need to consider use of more advanced techniques, such as Onion Routing [ONION].

## 7. Security Considerations

Ingress filtering has been and is being deployed as a means of preventing the use of spoofed source addresses in Distributed Denial of Service (DDoS) attacks. In a network with a large number of nodes, new temporary addresses are created at a fairly high rate. This might make it difficult for ingress filtering mechanisms to distinguish between legitimately changing temporary addresses and spoofed source addresses, which are "in-prefix" (They use a topologically correct prefix and non-existent interface ID). This can be addressed by using access control mechanisms on a per address basis on the network egress point.

The security and privacy implications of IPv6 addresses are discussed in great detail in [RFC7721] and [RFC7217].

## 8. Significant Changes from RFC RFC4941

This section summarizes the changes in this document relative to RFC 4941 that an implementer of RFC 4941 should be aware of.

1. The algorithm to generate randomized interface identifiers was replaced by two possible alternative algorithms.
2. Generation of stable addresses is not implied or required by this document.
3. Temporary addresses are *\*not\** disabled by default.



## 9. Acknowledgements

This document is based on [RFC4941] (authored by T. Narten, R. Draves, and S. Krishnan) and [I-D.gont-6man-non-stable-iids] (authored by F. Gont, C. Huitema, G. Gont, and M. Garcia Corbo).

## 10. References

### 10.1. Normative References

- [MD5] Rivest, R., "The MD5 Message-Digest Algorithm", RFC 1321, April 1992.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", RFC 2119, March 1997.
- [RFC4086] Eastlake 3rd, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", BCP 106, RFC 4086, DOI 10.17487/RFC4086, June 2005, <<https://www.rfc-editor.org/info/rfc4086>>.
- [RFC4193] Hinden, R. and B. Haberman, "Unique Local IPv6 Unicast Addresses", RFC 4193, DOI 10.17487/RFC4193, October 2005, <<https://www.rfc-editor.org/info/rfc4193>>.
- [RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, DOI 10.17487/RFC4291, February 2006, <<https://www.rfc-editor.org/info/rfc4291>>.
- [RFC4862] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, DOI 10.17487/RFC4862, September 2007, <<https://www.rfc-editor.org/info/rfc4862>>.
- [RFC4941] Narten, T., Draves, R., and S. Krishnan, "Privacy Extensions for Stateless Address Autoconfiguration in IPv6", RFC 4941, DOI 10.17487/RFC4941, September 2007, <<https://www.rfc-editor.org/info/rfc4941>>.
- [RFC5453] Krishnan, S., "Reserved IPv6 Interface Identifiers", RFC 5453, DOI 10.17487/RFC5453, February 2009, <<https://www.rfc-editor.org/info/rfc5453>>.
- [RFC6724] Thaler, D., Ed., Draves, R., Matsumoto, A., and T. Chown, "Default Address Selection for Internet Protocol Version 6 (IPv6)", RFC 6724, DOI 10.17487/RFC6724, September 2012, <<https://www.rfc-editor.org/info/rfc6724>>.
- [RFC7136] Carpenter, B. and S. Jiang, "Significance of IPv6 Interface Identifiers", RFC 7136, DOI 10.17487/RFC7136, February 2014, <<https://www.rfc-editor.org/info/rfc7136>>.
- [RFC7217] Gont, F., "A Method for Generating Semantically Opaque Interface Identifiers with IPv6 Stateless Address

Autoconfiguration (SLAAC)", RFC 7217, DOI 10.17487/  
RFC7217, April 2014,  
<<https://www.rfc-editor.org/info/rfc7217>>.

## 10.2. Informative References

- [COOKIES] Kristol, D. and L. Montulli, "HTTP State Management Mechanism", RFC 2965, October 2000.
- [DDNS] Vixie, P., Thomson, S., Rekhter, Y., and J. Bound, "Dynamic Updates in the Domain Name System (DNS UPDATE)", RFC 2136, April 1997.
- [DHCP] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, March 1997.
- [DHCPV6] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, July 2003.
- [DNA] Choi, J. and G. Daley, "Detecting Network Attachment in IPv6 Goals", draft-ietf-dna-goals-04 (work in progress), December 2004.
- [DNSOP] Durand, A., Ihren, J., and P. Savola, "Operational Considerations and Issues with IPv6 DNS", draft-ietf-dnsop-ipv6-dns-issues-10 (work in progress), October 2004.
- [FIPS-SHS] NIST, "Secure Hash Standard (SHS)", FIPS Publication 180-4, March 2012, <<http://csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf>>.
- [I-D.gont-6man-non-stable-iids] Gont, F., Huitema, C., Gont, G., and M. Corbo, "Recommendation on Temporary IPv6 Interface Identifiers", draft-gont-6man-non-stable-iids-01 (work in progress), March 2017.
- [I-D.gont-taps-address-usage-problem-statement] Gont, F., Gont, G., Corbo, M., and C. Huitema, "Problem Statement Regarding IPv6 Address Usage", draft-gont-taps-address-usage-problem-statement-00 (work in progress), February 2018.
- [IANA-RESERVED-IIID] IANA, "Reserved IPv6 Interface Identifiers",

<<http://www.iana.org/assignments/ipv6-interface-ids>>.

[ONION] Reed, MGR., Syverson, PFS., and DMG. Goldschlag, "Proxies for Anonymous Routing", Proceedings of the 12th Annual Computer Security Applications Conference, San Diego, CA, December 1996.

[OPEN-GROUP]

The Open Group, "The Open Group Base Specifications Issue 7 / IEEE Std 1003.1-2008, 2016 Edition", Section 4.16 Seconds Since the Epoch, 2016, <<http://pubs.opengroup.org/onlinepubs/9699919799/basedefs/contents.html>>.

[RFC1321] Rivest, R., "The MD5 Message-Digest Algorithm", RFC 1321, DOI 10.17487/RFC1321, April 1992, <<https://www.rfc-editor.org/info/rfc1321>>.

[RFC6059] Krishnan, S. and G. Daley, "Simple Procedures for Detecting Network Attachment in IPv6", RFC 6059, DOI 10.17487/RFC6059, November 2010, <<https://www.rfc-editor.org/info/rfc6059>>.

[RFC6151] Turner, S. and L. Chen, "Updated Security Considerations for the MD5 Message-Digest and the HMAC-MD5 Algorithms", RFC 6151, DOI 10.17487/RFC6151, March 2011, <<https://www.rfc-editor.org/info/rfc6151>>.

[RFC7707] Gont, F. and T. Chown, "Network Reconnaissance in IPv6 Networks", RFC 7707, DOI 10.17487/RFC7707, March 2016, <<https://www.rfc-editor.org/info/rfc7707>>.

[RFC7721] Cooper, A., Gont, F., and D. Thaler, "Security and Privacy Considerations for IPv6 Address Generation Mechanisms", RFC 7721, DOI 10.17487/RFC7721, March 2016, <<https://www.rfc-editor.org/info/rfc7721>>.

Authors' Addresses

Fernando Gont  
SI6 Networks / UTN-FRH  
Evaristo Carriego 2644  
Haedo, Provincia de Buenos Aires 1706  
Argentina

Phone: +54 11 4650 8472  
Email: fgont@si6networks.com  
URI: <http://www.si6networks.com>

Suresh Krishnan  
Ericsson Research  
8400 Decarie Blvd.  
Town of Mount Royal, QC  
Canada

Email: [suresh.krishnan@ericsson.com](mailto:suresh.krishnan@ericsson.com)



INTERNET-DRAFT  
Intended Status: Standard  
Expires: July 2018

T. Herbert  
Quantonium

January 15, 2018

ICMPv6 errors for discarding packets due to processing limits  
draft-herbert-6man-icmp-limits-03

Abstract

Network nodes may discard packets if they are unable to process protocol headers of packets due to processing constraints or limits. When such packets are dropped, the sender receives no indication so it cannot take action to address the cause of discarded packets. This document defines ICMPv6 errors that can be sent by a node that discards packets because it is unable to process the protocol headers. A node that receives such an ICMPv6 error may be able to modify what it sends in future packets to avoid subsequent packet discards.

Status of this Memo

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

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

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

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

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

Copyright and License Notice

Copyright (c) 2018 IETF Trust and the persons identified as the

document authors. All rights reserved.

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

## Table of Contents

1	Introduction . . . . .	3
1.1	Extension header limits . . . . .	3
1.2	Aggregate header limits . . . . .	4
2	ICMPv6 errors for extension header limits . . . . .	4
2.1	Format . . . . .	4
2.2	Unrecognized Next Header type encountered (code 1) . . . . .	5
2.3	Extension header too big (code 4) . . . . .	5
2.4	Extension header chain too long (code 5) . . . . .	6
2.5	Too many options in extension header (code 6) . . . . .	6
3	ICMPv6 error for aggregate header limits . . . . .	6
3.1	Format . . . . .	6
3.2	Usage . . . . .	7
4	Operation . . . . .	8
4.1	Priority of reporting . . . . .	8
4.2	Host response . . . . .	8
5	Security Considerations . . . . .	9
6	IANA Considerations . . . . .	10
6.1	Parameter Problem codes . . . . .	10
6.2	Destination Unreachable codes . . . . .	10
8	Acknowledgments . . . . .	10
7	References . . . . .	10
7.1	Normative References . . . . .	10
7.2	Informative References . . . . .	11
	Author's Address . . . . .	11



## 1 Introduction

This document specifies ICMPv6 errors that can be sent when a node discards a packet due to it being unable to process the necessary protocol headers because of processing constraints or limits. New ICMPv6 code points are defined as an update to [RFC4443].

Four of the errors are specific to processing limits of extension headers; another error is used when the aggregate protocol headers in a packet exceed the processing limits of a node.

### 1.1 Extension header limits

With IPv6, optional internet-layer information is carried in one or more IPv6 Extension Headers [RFC8200]. Extension Headers are placed between the IPv6 header and the Upper-Layer Header in a packet. The term "Header Chain" refers collectively to the IPv6 header, Extension Headers, and Upper-Layer Headers occurring in a packet. Individual extension headers may have a length of 2048 and must fit into one MTU. Destination Options and Hop by Hop Options contain a list of options in Type-length-value (TLV) format. Each option includes a length of the data field in octets, and the minimum size of an option (non-pad type) is two bytes and the maximum length is 257 bytes. The number of options in an extension header is only limited by the length of the extension header and MTU. Options may also be skipped over by a receiver if they are unknown and the Option Type indicates to skip (first two bits are 00).

Per [RFC8200], except for Hop by Hop options, extension headers are not examined or processed by intermediate nodes. Many intermediate nodes, however, do examine extension headers for various purposes. For instance, a node may examine all extension headers to locate the transport header of packet in order to implement transport layer filtering or to track connections to implement a stateful firewall.

Destination hosts are expected to process all extension headers and options in Hop-by-Hop and Destination Options.

Due to the variable lengths, high limits of lengths of extension headers, or potential for Denial of Service attack; many devices impose operational limits of extension headers in packets they can process. [RFC7045] discusses the requirements of intermediate nodes that discard packets because of unrecognized extension headers. When a limit is exceeded, the typical behavior is to silently discard a packet. The limits are non-standard and may be configurable per implementation. Both intermediate nodes and end hosts may institute such limits on extension header processing.

This document defines three Parameter Problem codes and extends the applicability of an existing code that are sent by a node that discards a packet due to processing limits of extension headers being exceeded. A source host that receives an ICMPv6 error can modify the use of extension headers in subsequent packets to the destination in order to avoid further occurrences of packets being discarded.

## 1.2 Aggregate header limits

Many hardware devices implement a parsing buffer of a fixed sized to process packets. The parsing buffer is expected to contain all the headers (often up to a transport layer header for filtering) that a device needs to examine. Parsing buffers have been implemented with various sizes (512 bytes is common, some devices have smaller sizes).

When the aggregate length of headers in a packet exceeds the size of the parsing buffer, a device will typically either discard the packet or defer processing to a software slow path. In either case, no indication of a problem is sent back to the sender.

This document defines one code for ICMPv6 Destination Unreachable that is sent by a node that is unable to process the headers of a packet due to the aggregate size of the packet headers exceeding a processing limit (e.g. exceeding the size of a parsing buffer). A source host that receives an ICMPv6 error can modify the headers used in subsequent packets to try to avoid further occurrences of packets being discarded or relegated to a slow path.

## 2 ICMPv6 errors for extension header limits

Three new codes are defined for Parameter Problem type and applicability of one existing code is extended for ICMPv6 errors for extension header limits.

### 2.1 Format

The format of the ICMPv6 message for an extension header limit exceeded error is:

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								
Type										Code										Checksum																			
Pointer																																							
As much of invoking packet																																							
as possible without the ICMPv6 packet																																							

```
| exceeding the minimum IPv6 MTU [IPv6]
```

### IPv6 Fields:

Destination Address

Copied from the Source Address field of the invoking packet.

## ICMPv6 Fields:

Type

#### 4 (Parameter Problem type)

Code (pertinent to this specification)

- ```

1 - Unrecognized Next Header type encountered
4 - Extension header too big
5 - Extension header chain too long
6 - Too many options in extension header

```

## Pointer

Identifies the octet offset within the invoking packet where the problem occurred.

The pointer will point beyond the end of the ICMPv6 packet if the field having a problem is beyond what can fit in the maximum size of an ICMPv6 error message.

## 2.2 Unrecognized Next Header type encountered (code 1)

[RFC8200] specifies that a destination host should send an "unrecognized next header type" when a Next Header value is unrecognized in a packet. This document extends this to allow intermediate nodes to send this same error for a packet that is discarded because a node does not recognize a Next Header type.

This code SHOULD be sent by an intermediate node that discards a packet because it encounters a Next Header type that is unknown in its examination. The ICMPv6 Pointer field is set to the offset of the unrecognized value within the original packet.

Note that when the original sender receives the ICMPv6 error it can differentiate between the message being sent by a destination host, per [RFC4443], and an error sent by an intermediate host based on matching the source address of the ICMPv6 packet and the destination address of the packet in the ICMPv6 data.

### 2.3 Extension header too big (code 4)

An ICMPv6 Parameter Problem with code for "extension header too big"

SHOULD be sent when a node discards a packet because the size of an extension header exceeds its processing limit. The ICMPv6 Pointer field is set to the offset of the first octet in the extension header that exceeds the limit.

#### 2.4 Extension header chain too long (code 5)

An ICMPv6 Parameter Problem with code for "extension header chain too long" SHOULD be sent when a node discards a packet with an extension header chain because an extension header chains exceeds its processing limit.

There are two different limits that might be applied: a limit on the total size in octets of the header chain, and a limit on the number of extension headers in the chain. This error code is used in both cases. In the case that the a size limit is exceeded, the ICMPv6 Pointer is set to first octet beyond the limit. In the case that the number of extension headers is exceeded, the ICMPv6 Pointer is set to the offset of first octet of the first extension header that is beyond the limit.

#### 2.5 Too many options in extension header (code 6)

An ICMPv6 Parameter Problem with code for "too many options in extension header" SHOULD be sent when a node discards a packet with an extension header that has a number of options that exceed the processing limits of the node. This code is applicable for Destination options or Hop-by-Hop options. The ICMPv6 Pointer field is set to the first octet of the first option that exceeds the limit.

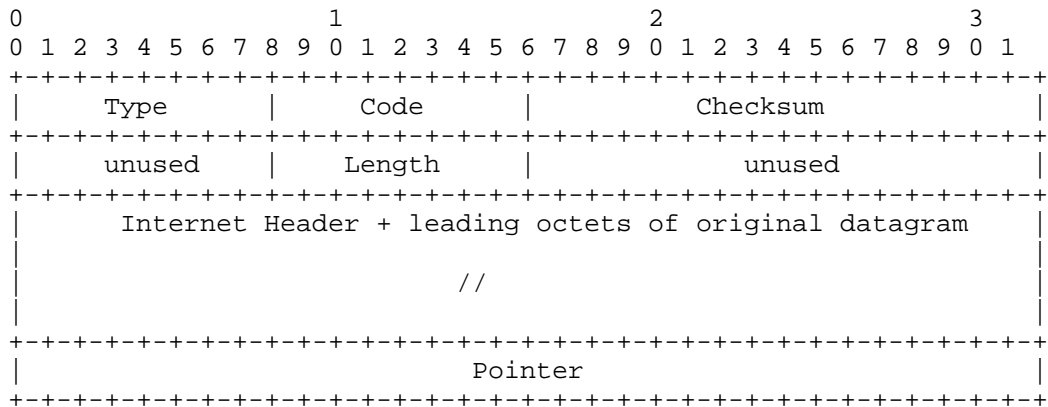
### 3 ICMPv6 error for aggregate header limits

One code is defined for Destination Unreach type for aggregate header limits.

#### 3.1 Format

The error for aggregate header limits employs a multi-part ICMPv6 message format as defined in [RFC4884]. The extended structure contains a pointer to the octet beyond the limit.

The format of the ICMPv6 message for an aggregate header limit exceeded is:



## IPv6 Fields:

## Destination Address

Copied from the Source Address field of the invoking packet.

## ICMPv6 Fields:

## Type

1 (Destination Unreachable type)

## Code (pertinent to this specification)

8 - Headers too long

## Length

Length of the "original datagram" measured in 64 bit words

## Pointer

Identifies the octet offset within the invoking packet where a limit was exceeded.

The pointer will point beyond the end of the original datagram if the field exceeding the limit is beyond what can fit in the maximum size of an ICMPv6 error message.

## 3.2 Usage

An ICMPv6 Destination Unreachable error with code for "headers too long" SHOULD be sent when a node discards a packet because the aggregate length of headers in the packet exceeds the processing limits of the node. The Pointer in the extended ICMPv6 structure is set to the offset of the first octet that exceeds the limit.

## 4 Operation

Nodes that send or receive ICMPv6 errors due to header processing limits MUST generally comply with ICMPv6 processing as specified in [RFC4443].

### 4.1 Priority of reporting

More than one ICMPv6 error may be applicable to report for a packet. For instance, the number of extension headers in a packet might exceed a limit, and the aggregate length of protocol headers might also exceed a limit. Only one ICMPv6 error should be sent for a packet, so a priority is defined to determine which error to report.

The reporting priority of ICMPv6 errors for processing limits is from highest to lowest priority:

- 1) Real error (existing codes)
- 2) Unrecognized Next Header type encountered by an intermediate node
- 3) Too many options in an extension header
- 4) Extension header too big
- 5) Extension header chain too long for number of extension headers limit exceeded
- 6) Extension header chain too long for size of the extension header chain exceeding a limit
- 7) Headers too long

### 4.2 Host response

When a source host receives an ICMPv6 error for a processing limit being exceeded, it SHOULD verify the ICMPv6 error is valid and take an appropriate action.

The ICMPv6 error SHOULD be logged with sufficient detail for debugging packet loss. The details of the error, including the addresses and the offending extension header or data, should be retained. This would be useful for instance to debug when a node is mis-configured and unexpectedly discarding packets, or when a new extension header is being deployed.

A host MAY modify its usage of protocol headers in subsequent packets

to avoid repeated occurrences of the same error.

For ICMPv6 errors cause by extension header limits being exceeded:

- \* An error SHOULD be reported to an application if the application enabled extension headers for its traffic. The application MAY either terminate a connection if extension headers are required, stop using extension headers in packets to the destination indicated in packet of the ICMPv6 error, or attempt modify its use of extension headers or headers to avoid the packet drop.
- \* A host system SHOULD take action if it is automatically inserting extension headers into packets unbeknownst to the application. The host system SHOULD either stop using extension headers or modify its used of extension headers for subsequent packets sent to the destination indicated in the packet of the ICMPv6 error.

## 5 Security Considerations

This document does not introduce any new security concerns for use of ICMPv6 errors. The security considerations for ICMPv6 described in [RFC4443] are applicable.

## 6 IANA Considerations

### 6.1 Parameter Problem codes

IANA is requested to assign the following codes for ICMPv6 type 4 "Parameter Problem":

- 4 - Extension header too big
- 5 - Extension header chain too long
- 6 - Too many options in extension header

### 6.2 Destination Unreachable codes

IANA is requested to assign the following codes for ICMPv6 type 1 "Destination Unreachable":

- 8 - Headers too long

## 8 Acknowledgments

The authors would like to thank Ron Bonica, Bob Hinden for their comments and suggestions that improved this document.

## 7 References

### 7.1 Normative References

- [RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <<https://www.rfc-editor.org/info/rfc8200>>.
- [RFC4443] Conta, A., Deering, S., and M. Gupta, Ed., "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", RFC 4443, DOI 10.17487/RFC4443, March 2006, <<http://www.rfc-editor.org/info/rfc4443>>.
- [RFC7045] Carpenter, B. and S. Jiang, "Transmission and Processing of IPv6 Extension Headers", RFC 7045, DOI 10.17487/RFC7045, December 2013, <<http://www.rfc-editor.org/info/rfc7045>>.
- [RFC4884] Bonica, R., Gan, D., Tappan, D., and C. Pignataro, "Extended ICMP to Support Multi-Part Messages", RFC 4884, DOI 10.17487/RFC4884, April 2007, <<https://www.rfc-editor.org/info/rfc4884>>.



## 7.2 Informative References

### Author's Address

Tom Herbert  
Quantonium  
Santa Clara, CA  
USA

Email: [tom@herbertland.com](mailto:tom@herbertland.com)

Network Working Group  
Internet-Draft  
Updates: 5175 (if approved)  
Intended status: Standards Track  
Expires: October 18, 2018

R. Hinden  
Check Point Software  
B. Carpenter  
Univ. of Auckland  
April 16, 2018

IPv6 Router Advertisement IPv6-Only Flag  
draft-hinden-ipv4flag-04

Abstract

This document specifies a Router Advertisement Flag to indicate to hosts that the administrator has configured the router to advertise that the link is IPv6-Only. This document updates RFC5175.

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 October 18, 2018.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

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

## Table of Contents

|                                                      |   |
|------------------------------------------------------|---|
| 1. Introduction . . . . .                            | 2 |
| 2. Requirements Language . . . . .                   | 3 |
| 3. Applicability Statements . . . . .                | 3 |
| 4. IPv6-Only Flag . . . . .                          | 4 |
| 5. Router and Operational Considerations . . . . .   | 5 |
| 6. Host Behavior Considerations . . . . .            | 5 |
| 7. IANA Considerations . . . . .                     | 6 |
| 8. Security Considerations . . . . .                 | 6 |
| 9. Acknowledgments . . . . .                         | 7 |
| 10. Change log [RFC Editor: Please remove] . . . . . | 7 |
| 11. References . . . . .                             | 8 |
| 11.1. Normative References . . . . .                 | 8 |
| 11.2. Informative References . . . . .               | 9 |
| Authors' Addresses . . . . .                         | 9 |

## 1. Introduction

This document specifies a Router Advertisement Flag to indicate to hosts that the administrator has configured the router to advertise that the link is IPv6-Only. The flag does not apply to non-default IPv6 routers.

Hosts that support IPv4 and IPv6, usually called dual stack hosts, need to also work efficiently on IPv6 only links. That is, a link where there are no IPv4 routers and/or IPv4 services. Dual stack is the default configuration for most current host operating systems such as Windows 10, IOS, Android, Linux, and BSD, as well as devices such as printers. Monitoring of IPv6-only link, for example at the IETF 100 meeting in Singapore, shows that current dual stack hosts will create local auto-configured IPv4 addresses and attempt to reach IPv4 services. This may be a problem for several reasons:

- o It may result in an undesirable level of Layer 2 broadcast traffic, especially on large wireless networks.
- o In particular, this may overload switches in multi-segment wireless networks because it will create IPv4 state for every dual stack host.
- o Such traffic may drain battery power on wireless hosts that have no interest in link-local IPv4 traffic. [RFC7772] indicates how this risk might be quantified.
- o Similarly, hosts may waste battery power on futile attempts to access IPv4 services.

- o On an IPv6-only link, IPv4 might be used for malicious purposes and pass unnoticed by IPv6-only monitoring mechanisms.

Some of these problems could be mitigated by configuring the Layer 2 infrastructure to drop IPv4 and DHCPv4 traffic by filtering Ethertypes 0x0800 and 0x806. However although this would limit the traffic to a single segment, it would not eliminate it.

This document defines a mechanism that a router administrator can use to inform hosts that this is an IPv6-Only link on their default routers such that they can disable IPv4 on this link, mitigating all of the above problems.

Because there is no IPv4 support on IPv6-only routers, the only way to notify the dual stack hosts that this link is IPv6-Only is to use an IPv6 mechanism. An active notification will be much more precise than attempting to deduce this fact by the lack of IPv4 responses or traffic.

IPv4-only hosts, and dual-stack hosts that do not recognize the new flag, will continue to attempt IPv4 operations, in particular IPv4 discovery protocols typically sent as link-layer broadcasts. This legacy traffic cannot be prevented by any IPv6 mechanism. The value of the new flag is limited to hosts that recognize it.

This document specifies a new flag for Router Advertisement Flag [RFC5175]. It updates [RFC5175] to add this flag.

## 2. Requirements Language

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

## 3. Applicability Statements

The mechanism is designed to allow administrators to notify hosts that the link is IPv6-Only. It SHOULD be only used in IPv6-Only links.

Dual stack hosts that have a good reason to use IPv4, for example IPv4 link-local services, can continue to do so. This is consistent with the SHOULD language in this document.

Administrators SHOULD only use this mechanism if they are certain that the link is IPv6-Only. For example, in cases where there is a

need to continue to use IPv4 or there are IPv4 only routers, setting this flag to 1 is a configuration error.

#### 4. IPv6-Only Flag

RFC5175 currently defines the flags in the NDP Router Advertisement message and these flags are registered in the IANA IPv6 ND Router Advertisement flags Registry [IANA-RF]. This currently contains the following one-bit flags defined in published RFCs:

```
  0 1 2 3 4 5 6 7
+-----+
|M|O|H|Prf|P|R|R|
+-----+
```

|     |                                              |
|-----|----------------------------------------------|
| M   | Managed Address Configuration Flag [RFC4861] |
| O   | Other Configuration Flag [RFC4861]           |
| H   | Mobile IPv6 Home Agent Flag [RFC3775]        |
| Prf | Router Selection Preferences [RFC4191]       |
| P   | Neighbor Discovery Proxy Flag [RFC4389]      |
| R   | Reserved                                     |

This document defines bit 6 to be the IPv6-Only Flag:

#### 6 IPv6-Only Flag

This flag has two values. These are:

|   |                               |
|---|-------------------------------|
| 0 | This is not an IPv6-Only link |
| 1 | This is an IPv6-Only link     |

RFC 5175 requires that unused flag bits be set to zero. Therefore, a router that does not support the new flag will not appear to assert that this is an IPv6-Only link.

Hosts receiving the Router Advertisement SHOULD only process this flag if the advertising router is a Default Router. Specifically, if the Lifetime field in the Router Advertisement is not zero, otherwise it SHOULD be ignored. This is done to allow some IPv6 routers to advertise information without being a Default Router and providing IPv6 connectivity.

## 5. Router and Operational Considerations

Default IPv6 routers that are on an IPv6-Only link SHOULD be configured to set the IPv6-Only flag to 1. In all other cases the flag SHOULD not be set to 1.

The intent is that the administrator of the router configures the router to set the IPv6-Only flag if she/he wants to tell the hosts on the link that the link is IPv6-Only. This is a configuration flag, not something that the router decides on it's own.

Operators of large IPv6-only wireless links are advised to also use Layer 2 techniques to drop IPv4 and DHCPv4 packets (Ethertypes 0x0800 and 0x806) at all switches, and to ensure that IPv4 and DHCPv4 Layer 3 features are disabled in all switches.

## 6. Host Behavior Considerations

If there are multiple IPv6 default routers on a link, they might send different values of the flag. If at least one IPv6 default router sends the flag with value 0, a dual stack host SHOULD not assume that the link is IPv6-Only. If all IPv6 default routers send the flag with value 1, a dual stack host SHOULD assume that this is an IPv6-Only link.

A host that receives only RAs with the flag set to 1 SHOULD not attempt any IPv4 operations, unless it subsequently receives at least one RA with the flag set to zero. As soon as such an RA is received, IPv4 operations SHOULD be started.

A host MAY choose to delay all IPv4 operations at start-up until a reasonable time has elapsed for RA messages to arrive. If all RAs received have the flag set, a host SHOULD also choose to not attempt IPv4 operations until an application asks it to, specifically delay performing DHCPv4 until it gets a request from an application to use IPv4. This would avoid attempting to obtain IPv4 addresses if there are no applications trying to use IPv4.

In all of the above, the flag's value is considered valid for the lifetime of the default router concerned, unless a subsequent RA delivers a different flag value. If a default router expires (i.e., no RA is received that refreshes its lifetime), the host must remove this router's flag value from consideration. If the result is that all surviving default routers have the flag set to 1, the host SHOULD assume that the link is IPv6-Only. In other words, at any given time, the state of the flag as seen by the host is the logical AND of the flags sent by all unexpired default IPv6 routers.

## 7. IANA Considerations

IANA is requested to assign the new Router Advertisement flag defined in Section 4 of this document. Bit 6 is the next available bit in this registry, IANA is requested to use this bit unless there is a reason to use another bit in this registry.

IANA is also requested to register this new flag bit in the IANA IPv6 ND Router Advertisement flags Registry [IANA-RF].

## 8. Security Considerations

This document shares the security issues with other parts of IPv6 Neighbor Discovery. General techniques to protect Router Advertisement traffic such as Router Guard [RFC6105] are useful in protecting these vulnerabilities.

A bad actor could use this mechanism to attempt turn off IPv4 service on a link that is using IPv4, by sending Router Advertisements with the IPv6-Only Flag set to 1. In that case, as long as there are routers sending Router Advertisements with this Flag set to 0, they would override this attack given the mechanism in Section 4. Specifically a host would only turn off IPv4 service if it wasn't hearing any Router Advertisement with the Flag set to 0. If the advice in Section 5 is followed, this attack will fail.

Conversely, a bad actor could use this mechanism to turn on, or pretend to turn on, IPv4 service on an IPv6-only link, by sending Router Advertisements with the Flag set to 0. However, this is really no different than what such a bad actor can do anyway, if they have the ability to configure a bogus router in the first place. The advice in Section 5 will minimize such an attack by limiting it to a single link.

Note that manipulating the Router Preference [RFC4191] will not affect either of these attacks: any IPv6-Only Flag of 0 will always override all Flags set to 1.

The new flag is neutral from an IPv6 privacy viewpoint, since it does not affect IPv6 operations in any way. From an IPv4 privacy viewpoint, it has the potential benefit of suppressing unnecessary traffic that might reveal the existence of a host and the correlation between its hardware and IPv4 addresses.

## 9. Acknowledgments

A closely related proposal was published earlier as [I-D.ietf-sunset4-noipv4].

Helpful comments were received from Lorenzo Colitti, David Farmer, Fernando Gont, Erik Kline, Jen Linkova, Veronika McKillop, Michael Richardson, Mark Smith, Barbara Stark, Ole Troan, James Woodyatt, and other members of the 6MAN working group.

Bjoern Zeeb has also produced a variant of this proposal and proposed an IPv6 transition plan in [I-D.bz-v4goawayflag].

## 10. Change log [RFC Editor: Please remove]

draft-hinden-ipv4flag-04, 2018-April-16:

Changed the name of the document and flag to be the IPv6-Only flag.

Rewrote text to make it affirmative that this is used by an administrator to tell the hosts that the link is IPv6-Only.

Added an Applicability Statements section to scope the intend use.

Changed requirement language to upper case, added Requirements Language section with references to [RFC2119] and [RFC8174].

Editorial changes.

draft-hinden-ipv4flag-03, 2018-Feb-15:

Changed terminology to use "link" instead of "network".

Improved text in Section 4. "Host Behavior Considerations" and added suggestion to only perform IPv4 if an application requests it.

Added clarification that the bit is set because an administrator configured the router to send it.

Editorial changes.

draft-hinden-ipv4flag-02, 2018-Feb-15:

Improved text in introduction.

Added reference to current IANA registry in Section 2.

Editorial changes.



draft-hinden-ipv4flag-01, 2017-Dec-12

Inverted name of flag from "Available" to "Unavailable".

Added problem description and clarified scope.

Added router and operational considerations.

Added host behavior considerations.

Extended security considerations.

Added Acknowledgment section, including reference to prior sunset4 draft.

draft-hinden-ipv4flag-00, 2017-Nov-17:

Original version.

## 11. References

### 11.1. Normative References

- [IANA-RF] "IPv6 ND Router Advertisement flags",  
<<https://www.iana.org/assignments/icmpv6-parameters/icmpv6-parameters.xhtml#icmpv6-parameters-11>>.
- [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>>.
- [RFC4191] Draves, R. and D. Thaler, "Default Router Preferences and More-Specific Routes", RFC 4191, DOI 10.17487/RFC4191, November 2005, <<https://www.rfc-editor.org/info/rfc4191>>.
- [RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, DOI 10.17487/RFC4861, September 2007, <<https://www.rfc-editor.org/info/rfc4861>>.
- [RFC5175] Haberman, B., Ed. and R. Hinden, "IPv6 Router Advertisement Flags Option", RFC 5175, DOI 10.17487/RFC5175, March 2008, <<https://www.rfc-editor.org/info/rfc5175>>.

- [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>>.

## 11.2. Informative References

- [I-D.bz-v4goawayflag]  
Zeeb, B., "IPv6 Router Advertisement IPv4 GoAway Flag", draft-bz-v4goawayflag-00 (work in progress), March 2018.
- [I-D.ietf-sunset4-noipv4]  
Perreault, S., George, W., Tsou, T., Yang, T., and J. Tremblay, "Turning off IPv4 Using DHCPv6 or Router Advertisements", draft-ietf-sunset4-noipv4-01 (work in progress), December 2014.
- [RFC6105] Levy-Abegnoli, E., Van de Velde, G., Popoviciu, C., and J. Mohacsi, "IPv6 Router Advertisement Guard", RFC 6105, DOI 10.17487/RFC6105, February 2011, <<https://www.rfc-editor.org/info/rfc6105>>.
- [RFC7772] Yourtchenko, A. and L. Colitti, "Reducing Energy Consumption of Router Advertisements", BCP 202, RFC 7772, DOI 10.17487/RFC7772, February 2016, <<https://www.rfc-editor.org/info/rfc7772>>.

## Authors' Addresses

Robert M. Hinden  
Check Point Software  
959 Skyway Road  
San Carlos, CA 94070  
USA

Email: [bob.hinden@gmail.com](mailto:bob.hinden@gmail.com)

Brian Carpenter  
Department of Computer Science  
University of Auckland  
PB 92019  
Auckland 1142  
New Zealand

Email: [brian.e.carpenter@gmail.com](mailto:brian.e.carpenter@gmail.com)

Internet Engineering Task Force  
Internet-Draft  
Obsoletes: 6434 (if approved)  
Intended status: Best Current Practice  
Expires: January 17, 2019

T. Chown  
Jisc  
J. Loughney  
Intel  
T. Winters  
UNH-IOL  
July 16, 2018

IPv6 Node Requirements  
draft-ietf-6man-rfc6434-bis-09

Abstract

This document defines requirements for IPv6 nodes. It is expected that IPv6 will be deployed in a wide range of devices and situations. Specifying the requirements for IPv6 nodes allows IPv6 to function well and interoperate in a large number of situations and deployments.

This document obsoletes RFC 6434, and in turn RFC 4294.

Status of This Memo

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

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

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

This Internet-Draft will expire on January 17, 2019.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents

carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

|                                                                                        |    |
|----------------------------------------------------------------------------------------|----|
| 1. Introduction . . . . .                                                              | 3  |
| 1.1. Scope of This Document . . . . .                                                  | 4  |
| 1.2. Description of IPv6 Nodes . . . . .                                               | 4  |
| 2. Requirements Language . . . . .                                                     | 5  |
| 3. Abbreviations Used in This Document . . . . .                                       | 5  |
| 4. Sub-IP Layer . . . . .                                                              | 5  |
| 5. IP Layer . . . . .                                                                  | 6  |
| 5.1. Internet Protocol Version 6 - RFC 8200 . . . . .                                  | 6  |
| 5.2. Support for IPv6 Extension Headers . . . . .                                      | 7  |
| 5.3. Protecting a node from excessive EH options . . . . .                             | 8  |
| 5.4. Neighbor Discovery for IPv6 - RFC 4861 . . . . .                                  | 9  |
| 5.5. SEcure Neighbor Discovery (SEND) - RFC 3971 . . . . .                             | 10 |
| 5.6. IPv6 Router Advertisement Flags Option - RFC 5175 . . . . .                       | 11 |
| 5.7. Path MTU Discovery and Packet Size . . . . .                                      | 11 |
| 5.7.1. Path MTU Discovery - RFC 8201 . . . . .                                         | 11 |
| 5.7.2. Minimum MTU considerations . . . . .                                            | 12 |
| 5.8. ICMP for the Internet Protocol Version 6 (IPv6) - RFC 4443 . . . . .              | 12 |
| 5.9. Default Router Preferences and More-Specific Routes - RFC 4191 . . . . .          | 12 |
| 5.10. First-Hop Router Selection - RFC 8028 . . . . .                                  | 12 |
| 5.11. Multicast Listener Discovery (MLD) for IPv6 - RFC 3810 . . . . .                 | 12 |
| 5.12. Explicit Congestion Notification (ECN) - RFC 3168 . . . . .                      | 13 |
| 6. Addressing and Address Configuration . . . . .                                      | 13 |
| 6.1. IP Version 6 Addressing Architecture - RFC 4291 . . . . .                         | 13 |
| 6.2. Host Address Availability Recommendations . . . . .                               | 13 |
| 6.3. IPv6 Stateless Address Autoconfiguration - RFC 4862 . . . . .                     | 14 |
| 6.4. Privacy Extensions for Address Configuration in IPv6 - RFC 4941 . . . . .         | 15 |
| 6.5. Stateful Address Autoconfiguration (DHCPv6) - RFC 3315 . . . . .                  | 15 |
| 6.6. Default Address Selection for IPv6 - RFC 6724 . . . . .                           | 16 |
| 7. DNS . . . . .                                                                       | 16 |
| 8. Configuring Non-Address Information . . . . .                                       | 16 |
| 8.1. DHCP for Other Configuration Information . . . . .                                | 16 |
| 8.2. Router Advertisements and Default Gateway . . . . .                               | 17 |
| 8.3. IPv6 Router Advertisement Options for DNS Configuration - RFC 8106 . . . . .      | 17 |
| 8.4. DHCP Options versus Router Advertisement Options for Host Configuration . . . . . | 17 |
| 9. Service Discovery Protocols . . . . .                                               | 18 |

|                                                                                        |    |
|----------------------------------------------------------------------------------------|----|
| 10. IPv4 Support and Transition . . . . .                                              | 18 |
| 10.1. Transition Mechanisms . . . . .                                                  | 18 |
| 10.1.1. Basic Transition Mechanisms for IPv6 Hosts and<br>Routers - RFC 4213 . . . . . | 18 |
| 11. Application Support . . . . .                                                      | 18 |
| 11.1. Textual Representation of IPv6 Addresses - RFC 5952 . .                          | 18 |
| 11.2. Application Programming Interfaces (APIs) . . . . .                              | 18 |
| 12. Mobility . . . . .                                                                 | 19 |
| 13. Security . . . . .                                                                 | 20 |
| 13.1. Requirements . . . . .                                                           | 21 |
| 13.2. Transforms and Algorithms . . . . .                                              | 21 |
| 14. Router-Specific Functionality . . . . .                                            | 21 |
| 14.1. IPv6 Router Alert Option - RFC 2711 . . . . .                                    | 22 |
| 14.2. Neighbor Discovery for IPv6 - RFC 4861 . . . . .                                 | 22 |
| 14.3. Stateful Address Autoconfiguration (DHCPv6) - RFC 3315 .                         | 22 |
| 14.4. IPv6 Prefix Length Recommendation for Forwarding - BCP<br>198 . . . . .          | 23 |
| 15. Constrained Devices . . . . .                                                      | 23 |
| 16. IPv6 Node Management . . . . .                                                     | 23 |
| 16.1. Management Information Base (MIB) Modules . . . . .                              | 23 |
| 16.1.1. IP Forwarding Table MIB . . . . .                                              | 24 |
| 16.1.2. Management Information Base for the Internet<br>Protocol (IP) . . . . .        | 24 |
| 16.1.3. Interface MIB . . . . .                                                        | 24 |
| 16.2. YANG Data Models . . . . .                                                       | 24 |
| 16.2.1. IP Management YANG Model . . . . .                                             | 24 |
| 16.2.2. Interface Management YANG Model . . . . .                                      | 24 |
| 17. Security Considerations . . . . .                                                  | 24 |
| 18. IANA Considerations . . . . .                                                      | 24 |
| 19. Authors and Acknowledgments . . . . .                                              | 24 |
| 19.1. Authors and Acknowledgments (Current Document) . . . .                           | 25 |
| 19.2. Authors and Acknowledgments from RFC 6434 . . . . .                              | 25 |
| 19.3. Authors and Acknowledgments from RFC 4294 . . . . .                              | 25 |
| 20. Appendix: Changes from RFC 6434 . . . . .                                          | 25 |
| 21. Appendix: Changes from RFC 4294 . . . . .                                          | 27 |
| 22. References . . . . .                                                               | 28 |
| 22.1. Normative References . . . . .                                                   | 28 |
| 22.2. Informative References . . . . .                                                 | 35 |
| Authors' Addresses . . . . .                                                           | 40 |

## 1. Introduction

This document defines common functionality required by both IPv6 hosts and routers. Many IPv6 nodes will implement optional or additional features, but this document collects and summarizes requirements from other published Standards Track documents in one place.

This document tries to avoid discussion of protocol details and references RFCs for this purpose. This document is intended to be an applicability statement and to provide guidance as to which IPv6 specifications should be implemented in the general case and which specifications may be of interest to specific deployment scenarios. This document does not update any individual protocol document RFCs.

Although this document points to different specifications, it should be noted that in many cases, the granularity of a particular requirement will be smaller than a single specification, as many specifications define multiple, independent pieces, some of which may not be mandatory. In addition, most specifications define both client and server behavior in the same specification, while many implementations will be focused on only one of those roles.

This document defines a minimal level of requirement needed for a device to provide useful internet service and considers a broad range of device types and deployment scenarios. Because of the wide range of deployment scenarios, the minimal requirements specified in this document may not be sufficient for all deployment scenarios. It is perfectly reasonable (and indeed expected) for other profiles to define additional or stricter requirements appropriate for specific usage and deployment environments. For example, this document does not mandate that all clients support DHCP, but some deployment scenarios may deem it appropriate to make such a requirement. For example, NIST has defined profiles for specialized requirements for IPv6 in target environments (see [USGv6]).

As it is not always possible for an implementer to know the exact usage of IPv6 in a node, an overriding requirement for IPv6 nodes is that they should adhere to Jon Postel's Robustness Principle: "Be conservative in what you do, be liberal in what you accept from others" [RFC0793].

#### 1.1. Scope of This Document

IPv6 covers many specifications. It is intended that IPv6 will be deployed in many different situations and environments. Therefore, it is important to develop requirements for IPv6 nodes to ensure interoperability.

#### 1.2. Description of IPv6 Nodes

From the Internet Protocol, Version 6 (IPv6) Specification [RFC8200], we have the following definitions:

- IPv6 node - a device that implements IPv6.
- IPv6 router - a node that forwards IPv6 packets not explicitly addressed to itself.
- IPv6 host - any IPv6 node that is not a router.

## 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] 8174 [RFC8174] when, and only when, they appear in all capitals, as show here.

## 3. Abbreviations Used in This Document

|      |                                   |
|------|-----------------------------------|
| AH   | Authentication Header             |
| DAD  | Duplicate Address Detection       |
| ESP  | Encapsulating Security Payload    |
| ICMP | Internet Control Message Protocol |
| IKE  | Internet Key Exchange             |
| MIB  | Management Information Base       |
| MLD  | Multicast Listener Discovery      |
| MTU  | Maximum Transmission Unit         |
| NA   | Neighbor Advertisement            |
| NBMA | Non-Broadcast Multiple Access     |
| ND   | Neighbor Discovery                |
| NS   | Neighbor Solicitation             |
| NUD  | Neighbor Unreachability Detection |
| PPP  | Point-to-Point Protocol           |

## 4. Sub-IP Layer

An IPv6 node MUST include support for one or more IPv6 link-layer specifications. Which link-layer specifications an implementation should include will depend upon what link-layers are supported by the hardware available on the system. It is possible for a conformant IPv6 node to support IPv6 on some of its interfaces and not on others.

As IPv6 is run over new layer 2 technologies, it is expected that new specifications will be issued. In the following, we list some of the layer 2 technologies for which an IPv6 specification has been developed. It is provided for informational purposes only and may not be complete.

- Transmission of IPv6 Packets over Ethernet Networks [RFC2464]

- Transmission of IPv6 Packets over Frame Relay Networks Specification [RFC2590]
- Transmission of IPv6 Packets over IEEE 1394 Networks [RFC3146]
- Transmission of IPv6, IPv4, and Address Resolution Protocol (ARP) Packets over Fibre Channel [RFC4338]
- Transmission of IPv6 Packets over IEEE 802.15.4 Networks [RFC4944]
- Transmission of IPv6 via the IPv6 Convergence Sublayer over IEEE 802.16 Networks [RFC5121]
- IP version 6 over PPP [RFC5072]

In addition to traditional physical link-layers, it is also possible to tunnel IPv6 over other protocols. Examples include:

- Teredo: Tunneling IPv6 over UDP through Network Address Translations (NATs) [RFC4380]
- Section 3 of "Basic Transition Mechanisms for IPv6 Hosts and Routers" [RFC4213]

## 5. IP Layer

### 5.1. Internet Protocol Version 6 - RFC 8200

The Internet Protocol Version 6 is specified in [RFC8200]. This specification MUST be supported.

The node MUST follow the packet transmission rules in RFC 8200.

All conformant IPv6 implementations MUST be capable of sending and receiving IPv6 packets; forwarding functionality MAY be supported. Nodes MUST always be able to send, receive, and process fragment headers.

IPv6 nodes MUST not create overlapping fragments. Also, when reassembling an IPv6 datagram, if one or more of its constituent fragments is determined to be an overlapping fragment, the entire datagram (and any constituent fragments) MUST be silently discarded. See [RFC5722] for more information.

As recommended in [RFC8021], nodes MUST NOT generate atomic fragments, i.e., where the fragment is a whole datagram. As per [RFC6946], if a receiving node reassembling a datagram encounters an atomic fragment, it should be processed as a fully reassembled



packet, and any other fragments that match this packet should be processed independently.

To mitigate a variety of potential attacks, nodes SHOULD avoid using predictable fragment Identification values in Fragment Headers, as discussed in [RFC7739].

All nodes SHOULD support the setting and use of the IPv6 Flow Label field as defined in the IPv6 Flow Label specification [RFC6437]. Forwarding nodes such as routers and load distributors MUST NOT depend only on Flow Label values being uniformly distributed. It is RECOMMENDED that source hosts support the flow label by setting the Flow Label field for all packets of a given flow to the same value chosen from an approximation to a discrete uniform distribution.

## 5.2. Support for IPv6 Extension Headers

RFC 8200 specifies extension headers and the processing for these headers.

Extension headers (except for the Hop-by-Hop Options header) are not processed, inserted, or deleted by any node along a packet's delivery path, until the packet reaches the node (or each of the set of nodes, in the case of multicast) identified in the Destination Address field of the IPv6 header.

Any unrecognized extension headers or options MUST be processed as described in RFC 8200. Note that where Section 4 of RFC 8200 refers to the action to be taken when a Next Header value in the current header is not recognized by a node, that action applies whether the value is an unrecognized Extension Header or an unrecognized upper layer protocol (ULP).

An IPv6 node MUST be able to process these extension headers. An exception is Routing Header type 0 (RH0), which was deprecated by [RFC5095] due to security concerns and which MUST be treated as an unrecognized routing type.

Further, [RFC7045] adds specific requirements for processing of Extension Headers, in particular that any forwarding node along an IPv6 packet's path, which forwards the packet for any reason, SHOULD do so regardless of any extension headers that are present.

As per RFC 8200, when a node fragments an IPv6 datagram, it MUST include the entire IPv6 Header Chain in the first fragment. The Per-Fragment headers MUST consist of the IPv6 header plus any extension headers that MUST be processed by nodes en route to the destination, that is, all headers up to and including the Routing header if

present, else the Hop-by-Hop Options header if present, else no extension headers. On reassembly, if the first fragment does not include all headers through an Upper-Layer header, then that fragment SHOULD be discarded and an ICMP Parameter Problem, Code 3, message SHOULD be sent to the source of the fragment, with the Pointer field set to zero. See [RFC7112] for a discussion of why oversized IPv6 Extension Header chains are avoided.

Defining new IPv6 extension headers is not recommended, unless there are no existing IPv6 extension headers that can be used by specifying a new option for that IPv6 extension header. A proposal to specify a new IPv6 extension header MUST include a detailed technical explanation of why an existing IPv6 extension header can not be used for the desired new function, and in such cases need to follow the format described in Section 8 of RFC 8200. For further background reading on this topic, see [RFC6564].

### 5.3. Protecting a node from excessive EH options

As per RFC 8200, end hosts are expected to process all extension headers, destination options, and hop-by-hop options in a packet. Given that the only limit on the number and size of extension headers is the MTU, the processing of received packets could be considerable. It is also conceivable that a long chain of extension headers might be used as a form of denial-of-service attack. Accordingly, a host may place limits on the number and sizes of extension headers and options it is willing to process.

A host MAY limit the number of consecutive PAD1 options in destination options or hop-by-hop options to seven. In this case, if the more than seven consecutive PAD1 options are present the packet MAY be silently discarded. The rationale is that if padding of eight or more bytes is required then the PADN option SHOULD be used.

A host MAY limit number of bytes in a PADN option to be less than eight. In such a case, if a PADN option is present that has a length greater than seven then the packet SHOULD be silently discarded. The rationale for this guideline is that the purpose of padding is for alignment and eight bytes is the maximum alignment used in IPv6.

A host MAY disallow unknown options in destination options or hop-by-hop options. This SHOULD be configurable where the default is to accept unknown options and process them per [RFC8200]. If a packet with unknown options is received and the host is configured to disallow them, then the packet SHOULD be silently discarded.

A host MAY impose a limit on the maximum number of non-padding options allowed in the destination options and hop-by-hop extension

headers. If this feature is supported the maximum number SHOULD be configurable and the default value SHOULD be set to eight. The limits for destination options and hop-by-hop options may be separately configurable. If a packet is received and the number of destination or hop-by-hop options exceeds the limit, then the packet SHOULD be silently discarded.

A host MAY impose a limit on the maximum length of destination options or hop-by-hop options extension header. This value SHOULD be configurable and the default is to accept options of any length. If a packet is received and the length of destination or hop-by-hop options extension header exceeds the length limit, then the packet SHOULD be silently discarded.

#### 5.4. Neighbor Discovery for IPv6 - RFC 4861

Neighbor Discovery is defined in [RFC4861]; the definition was updated by [RFC5942]. Neighbor Discovery SHOULD be supported. RFC 4861 states:

Unless specified otherwise (in a document that covers operating IP over a particular link type) this document applies to all link types. However, because ND uses link-layer multicast for some of its services, it is possible that on some link types (e.g., Non-Broadcast Multi-Access (NBMA) links), alternative protocols or mechanisms to implement those services will be specified (in the appropriate document covering the operation of IP over a particular link type). The services described in this document that are not directly dependent on multicast, such as Redirects, next-hop determination, Neighbor Unreachability Detection, etc., are expected to be provided as specified in this document. The details of how one uses ND on NBMA links are addressed in [RFC2491].

Some detailed analysis of Neighbor Discovery follows:

Router Discovery is how hosts locate routers that reside on an attached link. Hosts MUST support Router Discovery functionality.

Prefix Discovery is how hosts discover the set of address prefixes that define which destinations are on-link for an attached link. Hosts MUST support Prefix Discovery.

Hosts MUST also implement Neighbor Unreachability Detection (NUD) for all paths between hosts and neighboring nodes. NUD is not required for paths between routers. However, all nodes MUST respond to unicast Neighbor Solicitation (NS) messages.

[RFC7048] discusses NUD, in particular cases where it behaves too impatiently. It states that if a node transmits more than a certain number of packets, then it SHOULD use the exponential backoff of the retransmit timer, up to a certain threshold point.

Hosts MUST support the sending of Router Solicitations and the receiving of Router Advertisements. The ability to understand individual Router Advertisement options is dependent on supporting the functionality making use of the particular option.

[RFC7559] discusses packet loss resiliency for Router Solicitations, and requires that nodes MUST use a specific exponential backoff algorithm for RS retransmissions.

All nodes MUST support the sending and receiving of Neighbor Solicitation (NS) and Neighbor Advertisement (NA) messages. NS and NA messages are required for Duplicate Address Detection (DAD).

Hosts SHOULD support the processing of Redirect functionality. Routers MUST support the sending of Redirects, though not necessarily for every individual packet (e.g., due to rate limiting). Redirects are only useful on networks supporting hosts. In core networks dominated by routers, Redirects are typically disabled. The sending of Redirects SHOULD be disabled by default on routers intended to be deployed on core networks. They MAY be enabled by default on routers intended to support hosts on edge networks.

"IPv6 Host-to-Router Load Sharing" [RFC4311] includes additional recommendations on how to select from a set of available routers. [RFC4311] SHOULD be supported.

#### 5.5. SEcure Neighbor Discovery (SEND) - RFC 3971

SEND [RFC3971] and Cryptographically Generated Addresses (CGAs) [RFC3972] provide a way to secure the message exchanges of Neighbor Discovery. SEND has the potential to address certain classes of spoofing attacks, but it does not provide specific protection for threats from off-link attackers.

There have been relatively few implementations of SEND in common operating systems and platforms since its publication in 2005, and thus deployment experience remains very limited to date.

At this time, support for SEND is considered optional. Due to the complexity in deploying SEND, and its heavyweight provisioning, its deployment is only likely to be considered where nodes are operating in a particularly strict security environment.

## 5.6. IPv6 Router Advertisement Flags Option - RFC 5175

Router Advertisements include an 8-bit field of single-bit Router Advertisement flags. The Router Advertisement Flags Option extends the number of available flag bits by 48 bits. At the time of this writing, 6 of the original 8 single-bit flags have been assigned, while 2 remain available for future assignment. No flags have been defined that make use of the new option, and thus, strictly speaking, there is no requirement to implement the option today. However, implementations that are able to pass unrecognized options to a higher-level entity that may be able to understand them (e.g., a user-level process using a "raw socket" facility) MAY take steps to handle the option in anticipation of a future usage.

## 5.7. Path MTU Discovery and Packet Size

### 5.7.1. Path MTU Discovery - RFC 8201

"Path MTU Discovery for IP version 6" [RFC8201] SHOULD be supported. From [RFC8200]:

It is strongly recommended that IPv6 nodes implement Path MTU Discovery [RFC8201], in order to discover and take advantage of path MTUs greater than 1280 octets. However, a minimal IPv6 implementation (e.g., in a boot ROM) may simply restrict itself to sending packets no larger than 1280 octets, and omit implementation of Path MTU Discovery.

The rules in [RFC8200] and [RFC5722] MUST be followed for packet fragmentation and reassembly.

As described in RFC 8201, nodes implementing Path MTU Discovery and sending packets larger than the IPv6 minimum link MTU are susceptible to problematic connectivity if ICMPv6 messages are blocked or not transmitted. For example, this will result in connections that complete the TCP three-way handshake correctly but then hang when data is transferred. This state is referred to as a black-hole connection [RFC2923]. Path MTU Discovery relies on ICMPv6 Packet Too Big (PTB) to determine the MTU of the path (and thus these MUST not be filtered, as per the recommendation in [RFC4890]).

An alternative to Path MTU Discovery defined in RFC 8201 can be found in [RFC4821], which defines a method for Packetization Layer Path MTU Discovery (PLPMTUD) designed for use over paths where delivery of ICMPv6 messages to a host is not assured.

#### 5.7.2. Minimum MTU considerations

While an IPv6 link MTU can be set to 1280 bytes, it is recommended that for IPv6 UDP in particular, which includes DNS operation, the sender use a large MTU if they can, in order to avoid gratuitous fragmentation-caused packet drops.

#### 5.8. ICMP for the Internet Protocol Version 6 (IPv6) - RFC 4443

ICMPv6 [RFC4443] MUST be supported. "Extended ICMP to Support Multi-Part Messages" [RFC4884] MAY be supported.

#### 5.9. Default Router Preferences and More-Specific Routes - RFC 4191

"Default Router Preferences and More-Specific Routes" [RFC4191] provides support for nodes attached to multiple (different) networks, each providing routers that advertise themselves as default routers via Router Advertisements. In some scenarios, one router may provide connectivity to destinations the other router does not, and choosing the "wrong" default router can result in reachability failures. In order to resolve this scenario IPv6 Nodes MUST implement [RFC4191] and SHOULD implement the Type C host role defined in RFC4191.

#### 5.10. First-Hop Router Selection - RFC 8028

In multihomed scenarios, where a host has more than one prefix, each allocated by an upstream network that is assumed to implement BCP 38 ingress filtering, the host may have multiple routers to choose from.

Hosts that may be deployed in such multihomed environments SHOULD follow the guidance given in [RFC8028].

#### 5.11. Multicast Listener Discovery (MLD) for IPv6 - RFC 3810

Nodes that need to join multicast groups MUST support MLDv2 [RFC3810]. MLD is needed by any node that is expected to receive and process multicast traffic and in particular MLDv2 is required for support for source-specific multicast (SSM) as per [RFC4607].

Previous versions of this document only required MLDv1 ([RFC2710]) to be implemented on all nodes. Since participation of any MLDv1-only nodes on a link require that all other nodes on the link then operate in version 1 compatibility mode, the requirement to support MLDv2 on all nodes was upgraded to a MUST. Further, SSM is now the preferred multicast distribution method, rather than ASM.

Note that Neighbor Discovery (as used on most link types -- see Section 5.4) depends on multicast and requires that nodes join Solicited Node multicast addresses.

#### 5.12. Explicit Congestion Notification (ECN) - RFC 3168

An ECN-aware router sets a mark in the IP header in order to signal impending congestion, rather than dropping a packet. The receiver of the packet echoes the congestion indication to the sender, which can then reduce its transmission rate as if it detected a dropped packet.

Nodes SHOULD support [RFC3168] by implementing an interface for the upper layer to access and set the ECN bits in the IP header. The benefits of using ECN are documented in [RFC8087].

### 6. Addressing and Address Configuration

#### 6.1. IP Version 6 Addressing Architecture - RFC 4291

The IPv6 Addressing Architecture [RFC4291] MUST be supported.

The current IPv6 Address Architecture is based on a 64-bit boundary for subnet prefixes. The reasoning behind this decision is documented in [RFC7421].

Implementations MUST also support the Multicast flag updates documented in [RFC7371]

#### 6.2. Host Address Availability Recommendations

Hosts may be configured with addresses through a variety of methods, including SLAAC, DHCPv6, or manual configuration.

[RFC7934] recommends that networks provide general-purpose end hosts with multiple global IPv6 addresses when they attach, and it describes the benefits of and the options for doing so. Routers SHOULD support [RFC7934] for assigning multiple address to a host. Host SHOULD support assigning multiple addresses as described in [RFC7934].

Nodes SHOULD support the capability to be assigned a prefix per host as documented in [RFC8273]. Such an approach can offer improved host isolation and enhanced subscriber management on shared network segments.

### 6.3. IPv6 Stateless Address Autoconfiguration - RFC 4862

Hosts MUST support IPv6 Stateless Address Autoconfiguration. It is RECOMMENDED, as described in [RFC8064], that unless there is a specific requirement for MAC addresses to be embedded in an IID, nodes follow the procedure in [RFC7217] to generate SLAAC-based addresses, rather than using [RFC4862]. Addresses generated through RFC7217 will be the same whenever a given device (re)appears on the same subnet (with a specific IPv6 prefix), but the IID will vary on each subnet visited.

Nodes that are routers MUST be able to generate link-local addresses as described in [RFC4862].

From RFC 4862:

The autoconfiguration process specified in this document applies only to hosts and not routers. Since host autoconfiguration uses information advertised by routers, routers will need to be configured by some other means. However, it is expected that routers will generate link-local addresses using the mechanism described in this document. In addition, routers are expected to successfully pass the Duplicate Address Detection procedure described in this document on all addresses prior to assigning them to an interface.

All nodes MUST implement Duplicate Address Detection. Quoting from Section 5.4 of RFC 4862:

Duplicate Address Detection MUST be performed on all unicast addresses prior to assigning them to an interface, regardless of whether they are obtained through stateless autoconfiguration, DHCPv6, or manual configuration, with the following [exceptions noted therein].

"Optimistic Duplicate Address Detection (DAD) for IPv6" [RFC4429] specifies a mechanism to reduce delays associated with generating addresses via Stateless Address Autoconfiguration [RFC4862]. RFC 4429 was developed in conjunction with Mobile IPv6 in order to reduce the time needed to acquire and configure addresses as devices quickly move from one network to another, and it is desirable to minimize transition delays. For general purpose devices, RFC 4429 remains optional at this time.

[RFC7527] discusses enhanced DAD, and describes an algorithm to automate the detection of looped back IPv6 ND messages used by DAD. Nodes SHOULD implement this behaviour where such detection is beneficial.



#### 6.4. Privacy Extensions for Address Configuration in IPv6 - RFC 4941

A node using Stateless Address Autoconfiguration [RFC4862] to form a globally unique IPv6 address using its MAC address to generate the IID will see that IID remain the same on any visited network, even though the network prefix part changes. Thus it is possible for 3rd party device to track the activities of the node they communicate with, as that node moves around the network. Privacy Extensions for Stateless Address Autoconfiguration [RFC4941] address this concern by allowing nodes to configure an additional temporary address where the IID is effectively randomly generated. Privacy addresses are then used as source addresses for new communications initiated by the node.

General issues regarding privacy issues for IPv6 addressing are discussed in [RFC7721].

RFC 4941 SHOULD be supported. In some scenarios, such as dedicated servers in a data center, it provides limited or no benefit, or may complicate network management. Thus devices implementing this specification MUST provide a way for the end user to explicitly enable or disable the use of such temporary addresses.

Note that RFC4941 can be used independently of traditional SLAAC, or of RFC7217-based SLAAC.

Implementers of RFC 4941 should be aware that certain addresses are reserved and should not be chosen for use as temporary addresses. Consult "Reserved IPv6 Interface Identifiers" [RFC5453] for more details.

#### 6.5. Stateful Address Autoconfiguration (DHCPv6) - RFC 3315

DHCPv6 [RFC3315] can be used to obtain and configure addresses. In general, a network may provide for the configuration of addresses through SLAAC, DHCPv6, or both. There will be a wide range of IPv6 deployment models and differences in address assignment requirements, some of which may require DHCPv6 for stateful address assignment. Consequently, all hosts SHOULD implement address configuration via DHCPv6.

In the absence of observed Router Advertisement messages, IPv6 nodes MAY initiate DHCP to obtain IPv6 addresses and other configuration information, as described in Section 5.5.2 of [RFC4862].

Where devices are likely to be carried by users and attached to multiple visited networks, DHCPv6 client anonymity profiles SHOULD be supported as described in [RFC7844] to minimise the disclosure of

identifying information. Section 5 of RFC7844 describes operational considerations on the use of such anonymity profiles.

#### 6.6. Default Address Selection for IPv6 - RFC 6724

IPv6 nodes will invariably have multiple addresses configured simultaneously, and thus will need to choose which addresses to use for which communications. The rules specified in the Default Address Selection for IPv6 [RFC6724] document MUST be implemented. [RFC8028] updates rule 5.5 from [RFC6724]; implementations SHOULD implement this rule.

#### 7. DNS

DNS is described in [RFC1034], [RFC1035], [RFC3363], and [RFC3596]. Not all nodes will need to resolve names; those that will never need to resolve DNS names do not need to implement resolver functionality. However, the ability to resolve names is a basic infrastructure capability on which applications rely, and most nodes will need to provide support. All nodes SHOULD implement stub-resolver [RFC1034] functionality, as in [RFC1034], Section 5.3.1, with support for:

- AAAA type Resource Records [RFC3596];
- reverse addressing in ip6.arpa using PTR records [RFC3596];
- Extension Mechanisms for DNS (EDNS0) [RFC6891] to allow for DNS packet sizes larger than 512 octets.

Those nodes are RECOMMENDED to support DNS security extensions [RFC4033] [RFC4034] [RFC4035].

A6 Resource Records, which were only ever defined with Experimental status in [RFC3363], are now classified as Historic, as per [RFC6563].

#### 8. Configuring Non-Address Information

##### 8.1. DHCP for Other Configuration Information

DHCP [RFC3315] Specifies a mechanism for IPv6 nodes to obtain address configuration information (see Section 6.5) and to obtain additional (non-address) configuration. If a host implementation supports applications or other protocols that require configuration that is only available via DHCP, hosts SHOULD implement DHCP. For specialized devices on which no such configuration need is present, DHCP may not be necessary.

An IPv6 node can use the subset of DHCP (described in [RFC3736]) to obtain other configuration information.

If an IPv6 node implements DHCP it MUST implement the DNS options [RFC3646] as most deployments will expect these options are available.

## 8.2. Router Advertisements and Default Gateway

There is no defined DHCPv6 Gateway option.

Nodes using the Dynamic Host Configuration Protocol for IPv6 (DHCPv6) are thus expected to determine their default router information and on-link prefix information from received Router Advertisements.

## 8.3. IPv6 Router Advertisement Options for DNS Configuration - RFC 8106

Router Advertisement Options have historically been limited to those that are critical to basic IPv6 functionality. Originally, DNS configuration was not included as an RA option, and DHCP was the recommended way to obtain DNS configuration information. Over time, the thinking surrounding such an option has evolved. It is now generally recognized that few nodes can function adequately without having access to a working DNS resolver, and thus a Standards Track document has been published to provide this capability [RFC8106].

Implementations MUST include support for the DNS RA option [RFC8106].

## 8.4. DHCP Options versus Router Advertisement Options for Host Configuration

In IPv6, there are two main protocol mechanisms for propagating configuration information to hosts: Router Advertisements (RAs) and DHCP. RA options have been restricted to those deemed essential for basic network functioning and for which all nodes are configured with exactly the same information. Examples include the Prefix Information Options, the MTU option, etc. On the other hand, DHCP has generally been preferred for configuration of more general parameters and for parameters that may be client-specific. Generally speaking, however, there has been a desire to define only one mechanism for configuring a given option, rather than defining multiple (different) ways of configuring the same information.

One issue with having multiple ways of configuring the same information is that interoperability suffers if a host chooses one mechanism but the network operator chooses a different mechanism. For "closed" environments, where the network operator has significant influence over what devices connect to the network and thus what

configuration mechanisms they support, the operator may be able to ensure that a particular mechanism is supported by all connected hosts. In more open environments, however, where arbitrary devices may connect (e.g., a WIFI hotspot), problems can arise. To maximize interoperability in such environments, hosts would need to implement multiple configuration mechanisms to ensure interoperability.

## 9. Service Discovery Protocols

[RFC6762] and [RFC6763] describe multicast DNS (mDNS) and DNS-Based Service Discovery (DNS-SD) respectively. These protocols, collectively commonly referred to as the 'Bonjour' protocols after their naming by Apple, provide the means for devices to discover services within a local link and, in the absence of a unicast DNS service, to exchange naming information.

Where devices are to be deployed in networks where service discovery would be beneficial, e.g., for users seeking to discover printers or display devices, mDNS and DNS-SD SHOULD be supported.

## 10. IPv4 Support and Transition

IPv6 nodes MAY support IPv4.

### 10.1. Transition Mechanisms

#### 10.1.1. Basic Transition Mechanisms for IPv6 Hosts and Routers - RFC 4213

If an IPv6 node implements dual stack and tunneling, then [RFC4213] MUST be supported.

## 11. Application Support

### 11.1. Textual Representation of IPv6 Addresses - RFC 5952

Software that allows users and operators to input IPv6 addresses in text form SHOULD support "A Recommendation for IPv6 Address Text Representation" [RFC5952].

### 11.2. Application Programming Interfaces (APIs)

There are a number of IPv6-related APIs. This document does not mandate the use of any, because the choice of API does not directly relate to on-the-wire behavior of protocols. Implementers, however, would be advised to consider providing a common API or reviewing existing APIs for the type of functionality they provide to applications.

"Basic Socket Interface Extensions for IPv6" [RFC3493] provides IPv6 functionality used by typical applications. Implementers should note that RFC3493 has been picked up and further standardized by the Portable Operating System Interface (POSIX) [POSIX].

"Advanced Sockets Application Program Interface (API) for IPv6" [RFC3542] provides access to advanced IPv6 features needed by diagnostic and other more specialized applications.

"IPv6 Socket API for Source Address Selection" [RFC5014] provides facilities that allow an application to override the default Source Address Selection rules of [RFC6724].

"Socket Interface Extensions for Multicast Source Filters" [RFC3678] provides support for expressing source filters on multicast group memberships.

"Extension to Sockets API for Mobile IPv6" [RFC4584] provides application support for accessing and enabling Mobile IPv6 [RFC6275] features.

## 12. Mobility

Mobile IPv6 [RFC6275] and associated specifications [RFC3776] [RFC4877] allow a node to change its point of attachment within the Internet, while maintaining (and using) a permanent address. All communication using the permanent address continues to proceed as expected even as the node moves around. The definition of Mobile IP includes requirements for the following types of nodes:

- mobile nodes
- correspondent nodes with support for route optimization
- home agents
- all IPv6 routers

At the present time, Mobile IP has seen only limited implementation and no significant deployment, partly because it originally assumed an IPv6-only environment rather than a mixed IPv4/IPv6 Internet. Recently, additional work has been done to support mobility in mixed-mode IPv4 and IPv6 networks [RFC5555].

More usage and deployment experience is needed with mobility before any specific approach can be recommended for broad implementation in all hosts and routers. Consequently, [RFC6275], [RFC5555], and

associated standards such as [RFC4877] are considered a MAY at this time.

IPv6 for 3GPP [RFC7066] lists a snapshot of required IPv6 Functionalities at the time the document was published that would need to be implemented, going above and beyond the recommendations in this document. Additionally a 3GPP IPv6 Host MAY implement [RFC7278] for delivering IPv6 prefixes on the LAN link.

### 13. Security

This section describes the specification for security for IPv6 nodes.

Achieving security in practice is a complex undertaking. Operational procedures, protocols, key distribution mechanisms, certificate management approaches, etc., are all components that impact the level of security actually achieved in practice. More importantly, deficiencies or a poor fit in any one individual component can significantly reduce the overall effectiveness of a particular security approach.

IPsec either can provide end-to-end security between nodes or can provide channel security (for example, via a site-to-site IPsec VPN), making it possible to provide secure communication for all (or a subset of) communication flows at the IP layer between pairs of internet nodes. IPsec has two standard operating modes, Tunnel-mode and Transport-mode. In Tunnel-mode, IPsec provides network-layer security and protects an entire IP packet by encapsulating the original IP packet and then pre-pending a new IP header. In Transport-mode, IPsec provides security for the transport-layer (and above) by encapsulating only the transport-layer (and above) portion of the IP packet (i.e., without adding a 2nd IP header).

Although IPsec can be used with manual keying in some cases, such usage has limited applicability and is not recommended.

A range of security technologies and approaches proliferate today (e.g., IPsec, Transport Layer Security (TLS), Secure SHell (SSH), TLS VPNS, etc.) No one approach has emerged as an ideal technology for all needs and environments. Moreover, IPsec is not viewed as the ideal security technology in all cases and is unlikely to displace the others.

Previously, IPv6 mandated implementation of IPsec and recommended the key management approach of IKE. This document updates that recommendation by making support of the IPsec Architecture [RFC4301] a SHOULD for all IPv6 nodes. Note that the IPsec Architecture requires (e.g., Section 4.5 of RFC 4301) the implementation of both

manual and automatic key management. Currently, the recommended automated key management protocol to implement is IKEv2 [RFC7296].

This document recognizes that there exists a range of device types and environments where approaches to security other than IPsec can be justified. For example, special-purpose devices may support only a very limited number or type of applications, and an application-specific security approach may be sufficient for limited management or configuration capabilities. Alternatively, some devices may run on extremely constrained hardware (e.g., sensors) where the full IPsec Architecture is not justified.

Because most common platforms now support IPv6 and have it enabled by default, IPv6 security is an issue for networks that are ostensibly IPv4-only; see [RFC7123] for guidance on this area.

#### 13.1. Requirements

"Security Architecture for the Internet Protocol" [RFC4301] SHOULD be supported by all IPv6 nodes. Note that the IPsec Architecture requires (e.g., Section 4.5 of [RFC4301]) the implementation of both manual and automatic key management. Currently, the default automated key management protocol to implement is IKEv2. As required in [RFC4301], IPv6 nodes implementing the IPsec Architecture MUST implement ESP [RFC4303] and MAY implement AH [RFC4302].

#### 13.2. Transforms and Algorithms

The current set of mandatory-to-implement algorithms for the IPsec Architecture are defined in "Cryptographic Algorithm Implementation Requirements For ESP and AH" [RFC8221]. IPv6 nodes implementing the IPsec Architecture MUST conform to the requirements in [RFC8221]. Preferred cryptographic algorithms often change more frequently than security protocols. Therefore, implementations MUST allow for migration to new algorithms, as RFC 8221 is replaced or updated in the future.

The current set of mandatory-to-implement algorithms for IKEv2 are defined in "Cryptographic Algorithms for Use in the Internet Key Exchange Version 2 (IKEv2)" [RFC8247]. IPv6 nodes implementing IKEv2 MUST conform to the requirements in [RFC8247] and/or any future updates or replacements to [RFC8247].

#### 14. Router-Specific Functionality

This section defines general host considerations for IPv6 nodes that act as routers. Currently, this section does not discuss detailed

routing-specific requirements. For the case of typical home routers, [RFC7084] defines basic requirements for customer edge routers.

#### 14.1. IPv6 Router Alert Option - RFC 2711

The IPv6 Router Alert Option [RFC2711] is an optional IPv6 Hop-by-Hop Header that is used in conjunction with some protocols (e.g., RSVP [RFC2205] or Multicast Listener Discovery (MLDv2) [RFC3810]). The Router Alert option will need to be implemented whenever such protocols that mandate its use are implemented. See Section 5.11.

#### 14.2. Neighbor Discovery for IPv6 - RFC 4861

Sending Router Advertisements and processing Router Solicitations MUST be supported.

Section 7 of [RFC6275] includes some mobility-specific extensions to Neighbor Discovery. Routers SHOULD implement Sections 7.3 and 7.5, even if they do not implement Home Agent functionality.

#### 14.3. Stateful Address Autoconfiguration (DHCPv6) - RFC 3315

A single DHCP server ([RFC3315] or [RFC4862]) can provide configuration information to devices directly attached to a shared link, as well as to devices located elsewhere within a site. Communication between a client and a DHCP server located on different links requires the use of DHCP relay agents on routers.

In simple deployments, consisting of a single router and either a single LAN or multiple LANs attached to the single router, together with a WAN connection, a DHCP server embedded within the router is one common deployment scenario (e.g., [RFC7084]). There is no need for relay agents in such scenarios.

In more complex deployment scenarios, such as within enterprise or service provider networks, the use of DHCP requires some level of configuration, in order to configure relay agents, DHCP servers, etc. In such environments, the DHCP server might even be run on a traditional server, rather than as part of a router.

Because of the wide range of deployment scenarios, support for DHCP server functionality on routers is optional. However, routers targeted for deployment within more complex scenarios (as described above) SHOULD support relay agent functionality. Note that "Basic Requirements for IPv6 Customer Edge Routers" [RFC7084] requires implementation of a DHCPv6 server function in IPv6 Customer Edge (CE) routers.



#### 14.4. IPv6 Prefix Length Recommendation for Forwarding - BCP 198

Forwarding nodes MUST conform to BCP 198 [RFC7608] and thus IPv6 implementations of nodes that may forward packets MUST conform to the rules specified in Section 5.1 of [RFC4632].

#### 15. Constrained Devices

The target for this document is general IPv6 nodes. In this Section, we briefly discuss considerations for constrained devices.

In the case of constrained nodes, with limited CPU, memory, bandwidth or power, support for certain IPv6 functionality may need to be considered due to those limitations. While the requirements of this document are RECOMMENDED for all nodes, including constrained nodes, compromises may need to be made in certain cases. Where such compromises are made, the interoperability of devices should be strongly considered, particularly where this may impact other nodes on the same link, e.g., only supporting MLDv1 will affect other nodes.

The IETF 6LowPAN (IPv6 over Low Power LWPAN) WG defined six RFCs, including a general overview and problem statement ([RFC4919], the means by which IPv6 packets are transmitted over IEEE 802.15.4 networks [RFC4944] and ND optimisations for that medium [RFC6775].

IPv6 nodes that are battery-powered SHOULD implement the recommendations in [RFC7772].

#### 16. IPv6 Node Management

Network management MAY be supported by IPv6 nodes. However, for IPv6 nodes that are embedded devices, network management may be the only possible way of controlling these nodes.

Existing network management protocols include SNMP [RFC3411], NETCONF [RFC6241] and RESTCONF [RFC8040].

##### 16.1. Management Information Base (MIB) Modules

[RFC8096] clarifies the obsoleted status of various IPv6-specific MIB modules.

The following two MIB modules SHOULD be supported by nodes that support a Simple Network Management Protocol (SNMP) agent.

#### 16.1.1. IP Forwarding Table MIB

The IP Forwarding Table MIB [RFC4292] SHOULD be supported by nodes that support an SNMP agent.

#### 16.1.2. Management Information Base for the Internet Protocol (IP)

The IP MIB [RFC4293] SHOULD be supported by nodes that support an SNMP agent.

#### 16.1.3. Interface MIB

The Interface MIB [RFC2863] SHOULD be supported by nodes the support an SNMP agent.

#### 16.2. YANG Data Models

The following YANG data models SHOULD be supported by nodes that support a NETCONF or RESTCONF agent.

##### 16.2.1. IP Management YANG Model

The IP Management YANG Model [I-D.ietf-netmod-rfc7277bis] SHOULD be supported by nodes that support NETCONF or RESTCONF.

##### 16.2.2. Interface Management YANG Model

The Interface Management YANG Model [I-D.ietf-netmod-rfc7223bis] SHOULD be supported by nodes that support NETCONF or RESTCONF.

#### 17. Security Considerations

This document does not directly affect the security of the Internet, beyond the security considerations associated with the individual protocols.

Security is also discussed in Section 13 above.

#### 18. IANA Considerations

This document does not require any IANA actions.

#### 19. Authors and Acknowledgments

### 19.1. Authors and Acknowledgments (Current Document)

For this version of the IPv6 Node Requirements document, the authors would like to thank Brian Carpenter, Dave Thaler, Tom Herbert, Erik Kline, Mohamed Boucadair, and Michayla Newcombe for their contributions.

### 19.2. Authors and Acknowledgments from RFC 6434

Ed Jankiewicz and Thomas Narten were named authors of the previous iteration of this document, RFC6434.

For this version of the document, the authors thanked Hitoshi Asaeda, Brian Carpenter, Tim Chown, Ralph Droms, Sheila Frankel, Sam Hartman, Bob Hinden, Paul Hoffman, Pekka Savola, Yaron Sheffer, and Dave Thaler.

### 19.3. Authors and Acknowledgments from RFC 4294

The original version of this document (RFC 4294) was written by the IPv6 Node Requirements design team, which had the following members: Jari Arkko, Marc Blanchet, Samita Chakrabarti, Alain Durand, Gerard Gstaad, Jun-ichiro Itojun Hagino, Atsushi Inoue, Masahiro Ishiyama, John Loughney, Rajiv Raghunarayan, Shoichi Sakane, Dave Thaler, and Juha Wiljakka.

The authors would like to thank Ran Atkinson, Jim Bound, Brian Carpenter, Ralph Droms, Christian Huitema, Adam Machalek, Thomas Narten, Juha Ollila, and Pekka Savola for their comments. Thanks to Mark Andrews for comments and corrections on DNS text. Thanks to Alfred Hoenes for tracking the updates to various RFCs.

## 20. Appendix: Changes from RFC 6434

There have been many editorial clarifications as well as significant additions and updates. While this section highlights some of the changes, readers should not rely on this section for a comprehensive list of all changes.

1. Restructured sections
2. Added 6LoWPAN to link layers as it has some deployment.
3. Removed DOD IPv6 Profile as it hasn't been updated.
4. Updated to MLDv2 support to a MUST since nodes are restricted if MLDv1 is used.

5. Require DNS RA Options so SLAAC-only devices can get DNS, RFC8106 is a MUST.
6. Require RFC3646 DNS Options for DHCPv6 implementations.
7. Added RESTCONF and NETCONF as possible options to Network management.
8. Added section on constrained devices.
9. Added text on RFC7934, address availability to hosts (SHOULD).
10. Added text on RFC7844, anonymity profiles for DHCPv6 clients.
11. mDNS and DNS-SD added as updated service discovery.
12. Added RFC8028 as a SHOULD as a method for solving multi-prefix network
13. Added ECN RFC3168 as a SHOULD
14. Added reference to RFC7123 for Security over IPv4-only networks
15. Removed Jumbograms RFC2675 as they aren't deployed.
16. Updated Obseleted RFCs to the new version of the RFC including 2460, 1981, 7321, 4307
17. Added RFC7772 for power comsumptions considerations
18. Added why /64 boundries for more detail - RFC 7421
19. Added a Unique IPv6 Prefix per Host to support currently deployed IPv6 networks
20. Clarified RFC7066 was snapshot for 3GPP
21. Updated 4191 as a MUST, SHOULD for Type C Host as it helps solve multi-prefix problem
22. Removed IPv6 over ATM since there aren't many deployments
23. Added a note in Section 6.6 for RFC6724 Section 5.5/
24. Added MUST for BCP 198 for forwarding IPv6 packets
25. Added reference to RFC8064 for stable address creation.

26. Added text on protection from excessive EH options
  27. Added text on dangers of 1280 MTU UDP, esp. wrt DNS traffic
  28. Added text to clarify RFC8200 behaviour for unrecognized EHs or unrecognized ULPs
  29. Removed dated email addresses from design team acknowledgements for RFC 4294.
21. Appendix: Changes from RFC 4294

There have been many editorial clarifications as well as significant additions and updates. While this section highlights some of the changes, readers should not rely on this section for a comprehensive list of all changes.

1. Updated the Introduction to indicate that this document is an applicability statement and is aimed at general nodes.
2. Significantly updated the section on Mobility protocols, adding references and downgrading previous SHOULDs to MAYs.
3. Changed Sub-IP Layer section to just list relevant RFCs, and added some more RFCs.
4. Added section on SEND (it is a MAY).
5. Revised section on Privacy Extensions [RFC4941] to add more nuance to recommendation.
6. Completely revised IPsec/IKEv2 section, downgrading overall recommendation to a SHOULD.
7. Upgraded recommendation of DHCPv6 to SHOULD.
8. Added background section on DHCP versus RA options, added SHOULD recommendation for DNS configuration via RAs (RFC6106), and cleaned up DHCP recommendations.
9. Added recommendation that routers implement Sections 7.3 and 7.5 of [RFC6275].
10. Added pointer to subnet clarification document [RFC5942].
11. Added text that "IPv6 Host-to-Router Load Sharing" [RFC4311] SHOULD be implemented.

12. Added reference to [RFC5722] (Overlapping Fragments), and made it a MUST to implement.
13. Made "A Recommendation for IPv6 Address Text Representation" [RFC5952] a SHOULD.
14. Removed mention of "DNAME" from the discussion about [RFC3363].
15. Numerous updates to reflect newer versions of IPv6 documents, including [RFC4443], [RFC4291], [RFC3596], and [RFC4213].
16. Removed discussion of "Managed" and "Other" flags in RAs. There is no consensus at present on how to process these flags, and discussion of their semantics was removed in the most recent update of Stateless Address Autoconfiguration [RFC4862].
17. Added many more references to optional IPv6 documents.
18. Made "A Recommendation for IPv6 Address Text Representation" [RFC5952] a SHOULD.
19. Added reference to [RFC5722] (Overlapping Fragments), and made it a MUST to implement.
20. Updated MLD section to include reference to Lightweight MLD [RFC5790].
21. Added SHOULD recommendation for "Default Router Preferences and More-Specific Routes" [RFC4191].
22. Made "IPv6 Flow Label Specification" [RFC6437] a SHOULD.

## 22. References

### 22.1. Normative References

- [RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, DOI 10.17487/RFC1034, November 1987, <<https://www.rfc-editor.org/info/rfc1034>>.
- [RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <<https://www.rfc-editor.org/info/rfc1035>>.
- [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>>.

- [RFC2710] Deering, S., Fenner, W., and B. Haberman, "Multicast Listener Discovery (MLD) for IPv6", RFC 2710, DOI 10.17487/RFC2710, October 1999, <<https://www.rfc-editor.org/info/rfc2710>>.
- [RFC2711] Partridge, C. and A. Jackson, "IPv6 Router Alert Option", RFC 2711, DOI 10.17487/RFC2711, October 1999, <<https://www.rfc-editor.org/info/rfc2711>>.
- [RFC2863] McCloghrie, K. and F. Kastenholz, "The Interfaces Group MIB", RFC 2863, DOI 10.17487/RFC2863, June 2000, <<https://www.rfc-editor.org/info/rfc2863>>.
- [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, <<https://www.rfc-editor.org/info/rfc3168>>.
- [RFC3315] Droms, R., Ed., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, DOI 10.17487/RFC3315, July 2003, <<https://www.rfc-editor.org/info/rfc3315>>.
- [RFC3411] Harrington, D., Presuhn, R., and B. Wijnen, "An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks", STD 62, RFC 3411, DOI 10.17487/RFC3411, December 2002, <<https://www.rfc-editor.org/info/rfc3411>>.
- [RFC3596] Thomson, S., Huitema, C., Ksinant, V., and M. Souissi, "DNS Extensions to Support IP Version 6", STD 88, RFC 3596, DOI 10.17487/RFC3596, October 2003, <<https://www.rfc-editor.org/info/rfc3596>>.
- [RFC3736] Droms, R., "Stateless Dynamic Host Configuration Protocol (DHCP) Service for IPv6", RFC 3736, DOI 10.17487/RFC3736, April 2004, <<https://www.rfc-editor.org/info/rfc3736>>.
- [RFC3810] Vida, R., Ed. and L. Costa, Ed., "Multicast Listener Discovery Version 2 (MLDv2) for IPv6", RFC 3810, DOI 10.17487/RFC3810, June 2004, <<https://www.rfc-editor.org/info/rfc3810>>.
- [RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, DOI 10.17487/RFC4033, March 2005, <<https://www.rfc-editor.org/info/rfc4033>>.

- [RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, DOI 10.17487/RFC4034, March 2005, <<https://www.rfc-editor.org/info/rfc4034>>.
- [RFC4035] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Protocol Modifications for the DNS Security Extensions", RFC 4035, DOI 10.17487/RFC4035, March 2005, <<https://www.rfc-editor.org/info/rfc4035>>.
- [RFC4213] Nordmark, E. and R. Gilligan, "Basic Transition Mechanisms for IPv6 Hosts and Routers", RFC 4213, DOI 10.17487/RFC4213, October 2005, <<https://www.rfc-editor.org/info/rfc4213>>.
- [RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, DOI 10.17487/RFC4291, February 2006, <<https://www.rfc-editor.org/info/rfc4291>>.
- [RFC4292] Haberman, B., "IP Forwarding Table MIB", RFC 4292, DOI 10.17487/RFC4292, April 2006, <<https://www.rfc-editor.org/info/rfc4292>>.
- [RFC4293] Routhier, S., Ed., "Management Information Base for the Internet Protocol (IP)", RFC 4293, DOI 10.17487/RFC4293, April 2006, <<https://www.rfc-editor.org/info/rfc4293>>.
- [RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, DOI 10.17487/RFC4301, December 2005, <<https://www.rfc-editor.org/info/rfc4301>>.
- [RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, DOI 10.17487/RFC4303, December 2005, <<https://www.rfc-editor.org/info/rfc4303>>.
- [RFC4311] Hinden, R. and D. Thaler, "IPv6 Host-to-Router Load Sharing", RFC 4311, DOI 10.17487/RFC4311, November 2005, <<https://www.rfc-editor.org/info/rfc4311>>.
- [RFC4443] Conta, A., Deering, S., and M. Gupta, Ed., "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", STD 89, RFC 4443, DOI 10.17487/RFC4443, March 2006, <<https://www.rfc-editor.org/info/rfc4443>>.
- [RFC4607] Holbrook, H. and B. Cain, "Source-Specific Multicast for IP", RFC 4607, DOI 10.17487/RFC4607, August 2006, <<https://www.rfc-editor.org/info/rfc4607>>.



- [RFC4632] Fuller, V. and T. Li, "Classless Inter-domain Routing (CIDR): The Internet Address Assignment and Aggregation Plan", BCP 122, RFC 4632, DOI 10.17487/RFC4632, August 2006, <<https://www.rfc-editor.org/info/rfc4632>>.
- [RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, DOI 10.17487/RFC4861, September 2007, <<https://www.rfc-editor.org/info/rfc4861>>.
- [RFC4862] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, DOI 10.17487/RFC4862, September 2007, <<https://www.rfc-editor.org/info/rfc4862>>.
- [RFC4941] Narten, T., Draves, R., and S. Krishnan, "Privacy Extensions for Stateless Address Autoconfiguration in IPv6", RFC 4941, DOI 10.17487/RFC4941, September 2007, <<https://www.rfc-editor.org/info/rfc4941>>.
- [RFC5095] Abley, J., Savola, P., and G. Neville-Neil, "Deprecation of Type 0 Routing Headers in IPv6", RFC 5095, DOI 10.17487/RFC5095, December 2007, <<https://www.rfc-editor.org/info/rfc5095>>.
- [RFC5453] Krishnan, S., "Reserved IPv6 Interface Identifiers", RFC 5453, DOI 10.17487/RFC5453, February 2009, <<https://www.rfc-editor.org/info/rfc5453>>.
- [RFC5722] Krishnan, S., "Handling of Overlapping IPv6 Fragments", RFC 5722, DOI 10.17487/RFC5722, December 2009, <<https://www.rfc-editor.org/info/rfc5722>>.
- [RFC5790] Liu, H., Cao, W., and H. Asaeda, "Lightweight Internet Group Management Protocol Version 3 (IGMPv3) and Multicast Listener Discovery Version 2 (MLDv2) Protocols", RFC 5790, DOI 10.17487/RFC5790, February 2010, <<https://www.rfc-editor.org/info/rfc5790>>.
- [RFC5942] Singh, H., Beebe, W., and E. Nordmark, "IPv6 Subnet Model: The Relationship between Links and Subnet Prefixes", RFC 5942, DOI 10.17487/RFC5942, July 2010, <<https://www.rfc-editor.org/info/rfc5942>>.
- [RFC5952] Kawamura, S. and M. Kawashima, "A Recommendation for IPv6 Address Text Representation", RFC 5952, DOI 10.17487/RFC5952, August 2010, <<https://www.rfc-editor.org/info/rfc5952>>.

- [RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <<https://www.rfc-editor.org/info/rfc6241>>.
- [RFC6437] Amante, S., Carpenter, B., Jiang, S., and J. Rajahalme, "IPv6 Flow Label Specification", RFC 6437, DOI 10.17487/RFC6437, November 2011, <<https://www.rfc-editor.org/info/rfc6437>>.
- [RFC6564] Krishnan, S., Woodyatt, J., Kline, E., Hoagland, J., and M. Bhatia, "A Uniform Format for IPv6 Extension Headers", RFC 6564, DOI 10.17487/RFC6564, April 2012, <<https://www.rfc-editor.org/info/rfc6564>>.
- [RFC6724] Thaler, D., Ed., Draves, R., Matsumoto, A., and T. Chown, "Default Address Selection for Internet Protocol Version 6 (IPv6)", RFC 6724, DOI 10.17487/RFC6724, September 2012, <<https://www.rfc-editor.org/info/rfc6724>>.
- [RFC6762] Cheshire, S. and M. Krochmal, "Multicast DNS", RFC 6762, DOI 10.17487/RFC6762, February 2013, <<https://www.rfc-editor.org/info/rfc6762>>.
- [RFC6763] Cheshire, S. and M. Krochmal, "DNS-Based Service Discovery", RFC 6763, DOI 10.17487/RFC6763, February 2013, <<https://www.rfc-editor.org/info/rfc6763>>.
- [RFC6775] Shelby, Z., Ed., Chakrabarti, S., Nordmark, E., and C. Bormann, "Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)", RFC 6775, DOI 10.17487/RFC6775, November 2012, <<https://www.rfc-editor.org/info/rfc6775>>.
- [RFC6891] Damas, J., Graff, M., and P. Vixie, "Extension Mechanisms for DNS (EDNS(0))", STD 75, RFC 6891, DOI 10.17487/RFC6891, April 2013, <<https://www.rfc-editor.org/info/rfc6891>>.
- [RFC6946] Gont, F., "Processing of IPv6 "Atomic" Fragments", RFC 6946, DOI 10.17487/RFC6946, May 2013, <<https://www.rfc-editor.org/info/rfc6946>>.
- [RFC7045] Carpenter, B. and S. Jiang, "Transmission and Processing of IPv6 Extension Headers", RFC 7045, DOI 10.17487/RFC7045, December 2013, <<https://www.rfc-editor.org/info/rfc7045>>.

- [RFC7048] Nordmark, E. and I. Gashinsky, "Neighbor Unreachability Detection Is Too Impatient", RFC 7048, DOI 10.17487/RFC7048, January 2014, <<https://www.rfc-editor.org/info/rfc7048>>.
- [RFC7112] Gont, F., Manral, V., and R. Bonica, "Implications of Oversized IPv6 Header Chains", RFC 7112, DOI 10.17487/RFC7112, January 2014, <<https://www.rfc-editor.org/info/rfc7112>>.
- [RFC7217] Gont, F., "A Method for Generating Semantically Opaque Interface Identifiers with IPv6 Stateless Address Autoconfiguration (SLAAC)", RFC 7217, DOI 10.17487/RFC7217, April 2014, <<https://www.rfc-editor.org/info/rfc7217>>.
- [I-D.ietf-netmod-rfc7223bis] Bjorklund, M., "A YANG Data Model for Interface Management", draft-ietf-netmod-rfc7223bis-03 (work in progress), January 2018.
- [I-D.ietf-netmod-rfc7277bis] Bjorklund, M., "A YANG Data Model for IP Management", draft-ietf-netmod-rfc7277bis-03 (work in progress), January 2018.
- [RFC7296] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", STD 79, RFC 7296, DOI 10.17487/RFC7296, October 2014, <<https://www.rfc-editor.org/info/rfc7296>>.
- [RFC7527] Asati, R., Singh, H., Beebee, W., Pignataro, C., Dart, E., and W. George, "Enhanced Duplicate Address Detection", RFC 7527, DOI 10.17487/RFC7527, April 2015, <<https://www.rfc-editor.org/info/rfc7527>>.
- [RFC7559] Krishnan, S., Anipko, D., and D. Thaler, "Packet-Loss Resiliency for Router Solicitations", RFC 7559, DOI 10.17487/RFC7559, May 2015, <<https://www.rfc-editor.org/info/rfc7559>>.
- [RFC7608] Boucadair, M., Petrescu, A., and F. Baker, "IPv6 Prefix Length Recommendation for Forwarding", BCP 198, RFC 7608, DOI 10.17487/RFC7608, July 2015, <<https://www.rfc-editor.org/info/rfc7608>>.

- [RFC8021] Gont, F., Liu, W., and T. Anderson, "Generation of IPv6 Atomic Fragments Considered Harmful", RFC 8021, DOI 10.17487/RFC8021, January 2017, <<https://www.rfc-editor.org/info/rfc8021>>.
- [RFC8028] Baker, F. and B. Carpenter, "First-Hop Router Selection by Hosts in a Multi-Prefix Network", RFC 8028, DOI 10.17487/RFC8028, November 2016, <<https://www.rfc-editor.org/info/rfc8028>>.
- [RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <<https://www.rfc-editor.org/info/rfc8040>>.
- [RFC8064] Gont, F., Cooper, A., Thaler, D., and W. Liu, "Recommendation on Stable IPv6 Interface Identifiers", RFC 8064, DOI 10.17487/RFC8064, February 2017, <<https://www.rfc-editor.org/info/rfc8064>>.
- [RFC8106] Jeong, J., Park, S., Beloeil, L., and S. Madanapalli, "IPv6 Router Advertisement Options for DNS Configuration", RFC 8106, DOI 10.17487/RFC8106, March 2017, <<https://www.rfc-editor.org/info/rfc8106>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <<https://www.rfc-editor.org/info/rfc8200>>.
- [RFC8201] McCann, J., Deering, S., Mogul, J., and R. Hinden, Ed., "Path MTU Discovery for IP version 6", STD 87, RFC 8201, DOI 10.17487/RFC8201, July 2017, <<https://www.rfc-editor.org/info/rfc8201>>.
- [RFC8221] Wouters, P., Migault, D., Mattsson, J., Nir, Y., and T. Kivinen, "Cryptographic Algorithm Implementation Requirements and Usage Guidance for Encapsulating Security Payload (ESP) and Authentication Header (AH)", RFC 8221, DOI 10.17487/RFC8221, October 2017, <<https://www.rfc-editor.org/info/rfc8221>>.

- [RFC8247] Nir, Y., Kivinen, T., Wouters, P., and D. Migault, "Algorithm Implementation Requirements and Usage Guidance for the Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 8247, DOI 10.17487/RFC8247, September 2017, <<https://www.rfc-editor.org/info/rfc8247>>.

## 22.2. Informative References

- [RFC0793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, DOI 10.17487/RFC0793, September 1981, <<https://www.rfc-editor.org/info/rfc793>>.
- [RFC2205] Braden, R., Ed., Zhang, L., Berson, S., Herzog, S., and S. Jamin, "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification", RFC 2205, DOI 10.17487/RFC2205, September 1997, <<https://www.rfc-editor.org/info/rfc2205>>.
- [RFC2464] Crawford, M., "Transmission of IPv6 Packets over Ethernet Networks", RFC 2464, DOI 10.17487/RFC2464, December 1998, <<https://www.rfc-editor.org/info/rfc2464>>.
- [RFC2491] Armitage, G., Schulter, P., Jork, M., and G. Harter, "IPv6 over Non-Broadcast Multiple Access (NBMA) networks", RFC 2491, DOI 10.17487/RFC2491, January 1999, <<https://www.rfc-editor.org/info/rfc2491>>.
- [RFC2590] Conta, A., Malis, A., and M. Mueller, "Transmission of IPv6 Packets over Frame Relay Networks Specification", RFC 2590, DOI 10.17487/RFC2590, May 1999, <<https://www.rfc-editor.org/info/rfc2590>>.
- [RFC2923] Lahey, K., "TCP Problems with Path MTU Discovery", RFC 2923, DOI 10.17487/RFC2923, September 2000, <<https://www.rfc-editor.org/info/rfc2923>>.
- [RFC3146] Fujisawa, K. and A. Onoe, "Transmission of IPv6 Packets over IEEE 1394 Networks", RFC 3146, DOI 10.17487/RFC3146, October 2001, <<https://www.rfc-editor.org/info/rfc3146>>.
- [RFC3363] Bush, R., Durand, A., Fink, B., Gudmundsson, O., and T. Hain, "Representing Internet Protocol version 6 (IPv6) Addresses in the Domain Name System (DNS)", RFC 3363, DOI 10.17487/RFC3363, August 2002, <<https://www.rfc-editor.org/info/rfc3363>>.

- [RFC3493] Gilligan, R., Thomson, S., Bound, J., McCann, J., and W. Stevens, "Basic Socket Interface Extensions for IPv6", RFC 3493, DOI 10.17487/RFC3493, February 2003, <<https://www.rfc-editor.org/info/rfc3493>>.
- [RFC3542] Stevens, W., Thomas, M., Nordmark, E., and T. Jinmei, "Advanced Sockets Application Program Interface (API) for IPv6", RFC 3542, DOI 10.17487/RFC3542, May 2003, <<https://www.rfc-editor.org/info/rfc3542>>.
- [RFC3646] Droms, R., Ed., "DNS Configuration options for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3646, DOI 10.17487/RFC3646, December 2003, <<https://www.rfc-editor.org/info/rfc3646>>.
- [RFC3678] Thaler, D., Fenner, B., and B. Quinn, "Socket Interface Extensions for Multicast Source Filters", RFC 3678, DOI 10.17487/RFC3678, January 2004, <<https://www.rfc-editor.org/info/rfc3678>>.
- [RFC6275] Perkins, C., Ed., Johnson, D., and J. Arkko, "Mobility Support in IPv6", RFC 6275, DOI 10.17487/RFC6275, July 2011, <<https://www.rfc-editor.org/info/rfc6275>>.
- [RFC3776] Arkko, J., Devarapalli, V., and F. Dupont, "Using IPsec to Protect Mobile IPv6 Signaling Between Mobile Nodes and Home Agents", RFC 3776, DOI 10.17487/RFC3776, June 2004, <<https://www.rfc-editor.org/info/rfc3776>>.
- [RFC3971] Arkko, J., Ed., Kempf, J., Zill, B., and P. Nikander, "SEcure Neighbor Discovery (SEND)", RFC 3971, DOI 10.17487/RFC3971, March 2005, <<https://www.rfc-editor.org/info/rfc3971>>.
- [RFC3972] Aura, T., "Cryptographically Generated Addresses (CGA)", RFC 3972, DOI 10.17487/RFC3972, March 2005, <<https://www.rfc-editor.org/info/rfc3972>>.
- [RFC4191] Draves, R. and D. Thaler, "Default Router Preferences and More-Specific Routes", RFC 4191, DOI 10.17487/RFC4191, November 2005, <<https://www.rfc-editor.org/info/rfc4191>>.
- [RFC4302] Kent, S., "IP Authentication Header", RFC 4302, DOI 10.17487/RFC4302, December 2005, <<https://www.rfc-editor.org/info/rfc4302>>.

- [RFC4338] DeSanti, C., Carlson, C., and R. Nixon, "Transmission of IPv6, IPv4, and Address Resolution Protocol (ARP) Packets over Fibre Channel", RFC 4338, DOI 10.17487/RFC4338, January 2006, <<https://www.rfc-editor.org/info/rfc4338>>.
- [RFC4380] Huitema, C., "Teredo: Tunneling IPv6 over UDP through Network Address Translations (NATs)", RFC 4380, DOI 10.17487/RFC4380, February 2006, <<https://www.rfc-editor.org/info/rfc4380>>.
- [RFC4429] Moore, N., "Optimistic Duplicate Address Detection (DAD) for IPv6", RFC 4429, DOI 10.17487/RFC4429, April 2006, <<https://www.rfc-editor.org/info/rfc4429>>.
- [RFC4584] Chakrabarti, S. and E. Nordmark, "Extension to Sockets API for Mobile IPv6", RFC 4584, DOI 10.17487/RFC4584, July 2006, <<https://www.rfc-editor.org/info/rfc4584>>.
- [RFC4821] Mathis, M. and J. Heffner, "Packetization Layer Path MTU Discovery", RFC 4821, DOI 10.17487/RFC4821, March 2007, <<https://www.rfc-editor.org/info/rfc4821>>.
- [RFC4877] Devarapalli, V. and F. Dupont, "Mobile IPv6 Operation with IKEv2 and the Revised IPsec Architecture", RFC 4877, DOI 10.17487/RFC4877, April 2007, <<https://www.rfc-editor.org/info/rfc4877>>.
- [RFC4884] Bonica, R., Gan, D., Tappan, D., and C. Pignataro, "Extended ICMP to Support Multi-Part Messages", RFC 4884, DOI 10.17487/RFC4884, April 2007, <<https://www.rfc-editor.org/info/rfc4884>>.
- [RFC4890] Davies, E. and J. Mohacsi, "Recommendations for Filtering ICMPv6 Messages in Firewalls", RFC 4890, DOI 10.17487/RFC4890, May 2007, <<https://www.rfc-editor.org/info/rfc4890>>.
- [RFC4919] Kushalnagar, N., Montenegro, G., and C. Schumacher, "IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals", RFC 4919, DOI 10.17487/RFC4919, August 2007, <<https://www.rfc-editor.org/info/rfc4919>>.
- [RFC4944] Montenegro, G., Kushalnagar, N., Hui, J., and D. Culler, "Transmission of IPv6 Packets over IEEE 802.15.4 Networks", RFC 4944, DOI 10.17487/RFC4944, September 2007, <<https://www.rfc-editor.org/info/rfc4944>>.

- [RFC5014] Nordmark, E., Chakrabarti, S., and J. Laganier, "IPv6 Socket API for Source Address Selection", RFC 5014, DOI 10.17487/RFC5014, September 2007, <<https://www.rfc-editor.org/info/rfc5014>>.
- [RFC5072] Varada, S., Ed., Haskins, D., and E. Allen, "IP Version 6 over PPP", RFC 5072, DOI 10.17487/RFC5072, September 2007, <<https://www.rfc-editor.org/info/rfc5072>>.
- [RFC5121] Patil, B., Xia, F., Sarikaya, B., Choi, JH., and S. Madanapalli, "Transmission of IPv6 via the IPv6 Convergence Sublayer over IEEE 802.16 Networks", RFC 5121, DOI 10.17487/RFC5121, February 2008, <<https://www.rfc-editor.org/info/rfc5121>>.
- [RFC5555] Soliman, H., Ed., "Mobile IPv6 Support for Dual Stack Hosts and Routers", RFC 5555, DOI 10.17487/RFC5555, June 2009, <<https://www.rfc-editor.org/info/rfc5555>>.
- [RFC6563] Jiang, S., Conrad, D., and B. Carpenter, "Moving A6 to Historic Status", RFC 6563, DOI 10.17487/RFC6563, March 2012, <<https://www.rfc-editor.org/info/rfc6563>>.
- [RFC7066] Korhonen, J., Ed., Arkko, J., Ed., Savolainen, T., and S. Krishnan, "IPv6 for Third Generation Partnership Project (3GPP) Cellular Hosts", RFC 7066, DOI 10.17487/RFC7066, November 2013, <<https://www.rfc-editor.org/info/rfc7066>>.
- [RFC7084] Singh, H., Beebe, W., Donley, C., and B. Stark, "Basic Requirements for IPv6 Customer Edge Routers", RFC 7084, DOI 10.17487/RFC7084, November 2013, <<https://www.rfc-editor.org/info/rfc7084>>.
- [RFC7123] Gont, F. and W. Liu, "Security Implications of IPv6 on IPv4 Networks", RFC 7123, DOI 10.17487/RFC7123, February 2014, <<https://www.rfc-editor.org/info/rfc7123>>.
- [RFC7278] Byrne, C., Drown, D., and A. Vizdal, "Extending an IPv6 /64 Prefix from a Third Generation Partnership Project (3GPP) Mobile Interface to a LAN Link", RFC 7278, DOI 10.17487/RFC7278, June 2014, <<https://www.rfc-editor.org/info/rfc7278>>.
- [RFC7371] Boucadair, M. and S. Venaas, "Updates to the IPv6 Multicast Addressing Architecture", RFC 7371, DOI 10.17487/RFC7371, September 2014, <<https://www.rfc-editor.org/info/rfc7371>>.



- [RFC7421] Carpenter, B., Ed., Chown, T., Gont, F., Jiang, S., Petrescu, A., and A. Yourtchenko, "Analysis of the 64-bit Boundary in IPv6 Addressing", RFC 7421, DOI 10.17487/RFC7421, January 2015, <<https://www.rfc-editor.org/info/rfc7421>>.
- [RFC7721] Cooper, A., Gont, F., and D. Thaler, "Security and Privacy Considerations for IPv6 Address Generation Mechanisms", RFC 7721, DOI 10.17487/RFC7721, March 2016, <<https://www.rfc-editor.org/info/rfc7721>>.
- [RFC7739] Gont, F., "Security Implications of Predictable Fragment Identification Values", RFC 7739, DOI 10.17487/RFC7739, February 2016, <<https://www.rfc-editor.org/info/rfc7739>>.
- [RFC7772] Yourtchenko, A. and L. Colitti, "Reducing Energy Consumption of Router Advertisements", BCP 202, RFC 7772, DOI 10.17487/RFC7772, February 2016, <<https://www.rfc-editor.org/info/rfc7772>>.
- [RFC7844] Huitema, C., Mrugalski, T., and S. Krishnan, "Anonymity Profiles for DHCP Clients", RFC 7844, DOI 10.17487/RFC7844, May 2016, <<https://www.rfc-editor.org/info/rfc7844>>.
- [RFC7934] Colitti, L., Cerf, V., Cheshire, S., and D. Schinazi, "Host Address Availability Recommendations", BCP 204, RFC 7934, DOI 10.17487/RFC7934, July 2016, <<https://www.rfc-editor.org/info/rfc7934>>.
- [RFC8087] Fairhurst, G. and M. Welzl, "The Benefits of Using Explicit Congestion Notification (ECN)", RFC 8087, DOI 10.17487/RFC8087, March 2017, <<https://www.rfc-editor.org/info/rfc8087>>.
- [RFC8096] Fenner, B., "The IPv6-Specific MIB Modules Are Obsolete", RFC 8096, DOI 10.17487/RFC8096, April 2017, <<https://www.rfc-editor.org/info/rfc8096>>.
- [RFC8273] Brzozowski, J. and G. Van de Velde, "Unique IPv6 Prefix per Host", RFC 8273, DOI 10.17487/RFC8273, December 2017, <<https://www.rfc-editor.org/info/rfc8273>>.
- [POSIX] IEEE, "IEEE Std. 1003.1-2008 Standard for Information Technology -- Portable Operating System Interface (POSIX), ISO/IEC 9945:2009", <<http://www.ieee.org>>.

[USGv6] National Institute of Standards and Technology, "A Profile for IPv6 in the U.S. Government - Version 1.0", July 2008, <<https://doi.org/10.6028/NIST.SP.500-267Ar1>>.

#### Authors' Addresses

Tim Chown  
Jisc  
Lumen House, Library Avenue  
Harwell Oxford, Didcot OX11 0SG  
United Kingdom

Email: [tim.chown@jisc.ac.uk](mailto:tim.chown@jisc.ac.uk)

John Loughney  
Intel  
Santa Clara, CA  
USA

Email: [john.loughney@gmail.com](mailto:john.loughney@gmail.com)

Timothy Winters  
University of New Hampshire, Interoperability Lab (UNH-IOL)  
Durham, NH  
United States

Email: [twinters@iol.unh.edu](mailto:twinters@iol.unh.edu)

Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: April 24, 2020

C. Filsfils, Ed.  
D. Dukes, Ed.  
Cisco Systems, Inc.  
S. Previdi  
Huawei  
J. Leddy  
Individual  
S. Matsushima  
Softbank  
D. Voyer  
Bell Canada  
October 22, 2019

IPv6 Segment Routing Header (SRH)  
draft-ietf-6man-segment-routing-header-26

Abstract

Segment Routing can be applied to the IPv6 data plane using a new type of Routing Extension Header called the Segment Routing Header. This document describes the Segment Routing Header and how it is used by Segment Routing capable nodes.

Status of This Memo

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

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

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

This Internet-Draft will expire on April 24, 2020.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

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

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

## Table of Contents

|                                                                                 |    |
|---------------------------------------------------------------------------------|----|
| 1. Introduction . . . . .                                                       | 3  |
| 1.1. Requirements Language . . . . .                                            | 3  |
| 2. Segment Routing Header . . . . .                                             | 4  |
| 2.1. SRH TLVs . . . . .                                                         | 6  |
| 2.1.1. Padding TLVs . . . . .                                                   | 8  |
| 2.1.2. HMAC TLV . . . . .                                                       | 9  |
| 3. SR Nodes . . . . .                                                           | 12 |
| 3.1. Source SR Node . . . . .                                                   | 12 |
| 3.2. Transit Node . . . . .                                                     | 12 |
| 3.3. SR Segment Endpoint Node . . . . .                                         | 12 |
| 4. Packet Processing . . . . .                                                  | 13 |
| 4.1. Source SR Node . . . . .                                                   | 13 |
| 4.1.1. Reduced SRH . . . . .                                                    | 13 |
| 4.2. Transit Node . . . . .                                                     | 14 |
| 4.3. SR Segment Endpoint Node . . . . .                                         | 14 |
| 4.3.1. FIB Entry Is Locally Instantiated SRv6 SID . . . . .                     | 14 |
| 4.3.2. FIB Entry Is A Local Interface . . . . .                                 | 16 |
| 4.3.3. FIB Entry Is A Non-Local Route . . . . .                                 | 17 |
| 4.3.4. FIB Entry Is A No Match . . . . .                                        | 17 |
| 5. Intra SR Domain Deployment Model . . . . .                                   | 17 |
| 5.1. Securing the SR Domain . . . . .                                           | 17 |
| 5.2. SR Domain as A Single System with Delegation Among<br>Components . . . . . | 18 |
| 5.3. MTU Considerations . . . . .                                               | 19 |
| 5.4. ICMP Error Processing . . . . .                                            | 19 |
| 5.5. Load Balancing and ECMP . . . . .                                          | 19 |
| 5.6. Other Deployments . . . . .                                                | 20 |
| 6. Illustrations . . . . .                                                      | 20 |
| 6.1. Abstract Representation of an SRH . . . . .                                | 20 |
| 6.2. Example Topology . . . . .                                                 | 21 |
| 6.3. Source SR Node . . . . .                                                   | 22 |
| 6.3.1. Intra SR Domain Packet . . . . .                                         | 22 |
| 6.3.2. Inter SR Domain Packet - Transit . . . . .                               | 22 |
| 6.3.3. Inter SR Domain Packet - Internal to External . . . . .                  | 23 |
| 6.4. Transit Node . . . . .                                                     | 23 |
| 6.5. SR Segment Endpoint Node . . . . .                                         | 23 |
| 6.6. Delegation of Function with HMAC Verification . . . . .                    | 23 |
| 6.6.1. SID List Verification . . . . .                                          | 24 |

|       |                                                 |    |
|-------|-------------------------------------------------|----|
| 7.    | Security Considerations . . . . .               | 24 |
| 7.1.  | Source Routing Attacks . . . . .                | 25 |
| 7.2.  | Service Theft . . . . .                         | 25 |
| 7.3.  | Topology Disclosure . . . . .                   | 25 |
| 7.4.  | ICMP Generation . . . . .                       | 26 |
| 7.5.  | Applicability of AH . . . . .                   | 26 |
| 8.    | IANA Considerations . . . . .                   | 26 |
| 8.1.  | Segment Routing Header Flags Registry . . . . . | 27 |
| 8.2.  | Segment Routing Header TLVs Registry . . . . .  | 27 |
| 9.    | Implementation Status . . . . .                 | 27 |
| 9.1.  | Linux . . . . .                                 | 28 |
| 9.2.  | Cisco Systems . . . . .                         | 28 |
| 9.3.  | FD.io . . . . .                                 | 28 |
| 9.4.  | Barefoot . . . . .                              | 28 |
| 9.5.  | Juniper . . . . .                               | 28 |
| 9.6.  | Huawei . . . . .                                | 29 |
| 10.   | Contributors . . . . .                          | 29 |
| 11.   | Acknowledgements . . . . .                      | 29 |
| 12.   | References . . . . .                            | 29 |
| 12.1. | Normative References . . . . .                  | 29 |
| 12.2. | Informative References . . . . .                | 31 |
|       | Authors' Addresses . . . . .                    | 32 |

## 1. Introduction

Segment Routing can be applied to the IPv6 data plane using a new type of Routing Header called the Segment Routing Header. This document describes the Segment Routing Header and how it is used by Segment Routing capable nodes.

The Segment Routing Architecture [RFC8402] describes Segment Routing and its instantiation in two data planes; MPLS and IPv6.

The encoding of IPv6 segments in the Segment Routing Header is defined in this document.

This document uses the terms Segment Routing, SR Domain, SRv6, Segment ID (SID), SRv6 SID, Active Segment, and SR Policy as defined in [RFC8402].

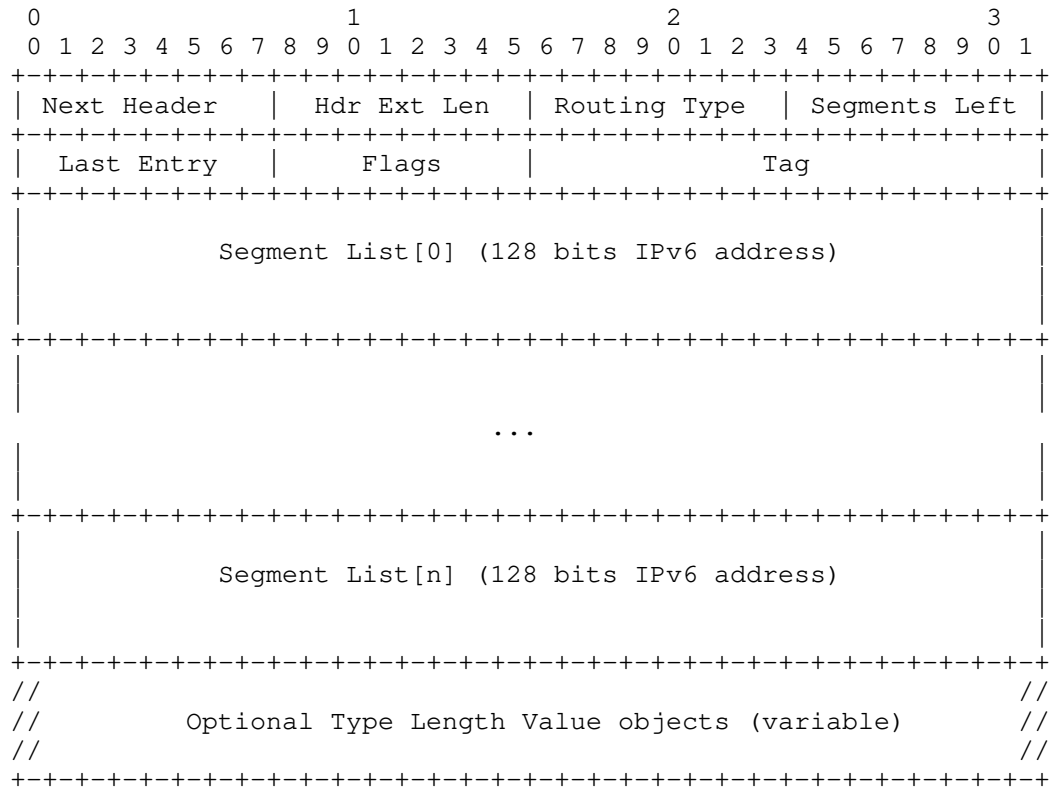
### 1.1. Requirements Language

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

## 2. Segment Routing Header

Routing Headers are defined in [RFC8200]. The Segment Routing Header has a new Routing Type (suggested value 4) to be assigned by IANA.

The Segment Routing Header (SRH) is defined as follows:



where:

- o Next Header: Defined in [RFC8200] Section 4.4
- o Hdr Ext Len: Defined in [RFC8200] Section 4.4
- o Routing Type: TBD, to be assigned by IANA (suggested value: 4).
- o Segments Left: Defined in [RFC8200] Section 4.4
- o Last Entry: contains the index (zero based), in the Segment List, of the last element of the Segment List.

- o Flags: 8 bits of flags. Section 8.1 creates an IANA registry for new flags to be defined. The following flags are defined:

```

  0 1 2 3 4 5 6 7
+-----+
|U U U U U U U U|
+-----+

```

U: Unused and for future use. MUST be 0 on transmission and ignored on receipt.

- o Tag: tag a packet as part of a class or group of packets, e.g., packets sharing the same set of properties. When tag is not used at source it MUST be set to zero on transmission. When tag is not used during SRH Processing it SHOULD be ignored. Tag is not used when processing the SID defined in Section 4.3.1. It may be used when processing other SIDs that are not defined in this document. The allocation and use of tag is outside the scope of this document.
- o Segment List[n]: 128 bit IPv6 addresses representing the nth segment in the Segment List. The Segment List is encoded starting from the last segment of the SR Policy. I.e., the first element of the segment list (Segment List [0]) contains the last segment of the SR Policy, the second element contains the penultimate segment of the SR Policy and so on.
- o Type Length Value (TLV) are described in Section 2.1.

In the SRH, the Next Header, Hdr Ext Len, Routing Type, and Segments Left fields are defined in Section 4.4 of [RFC8200]. Based on the constraints in that section, Next Header, Header Ext Len, and Routing Type are not mutable while Segments Left is mutable.

The mutability of the TLV value is defined by the most significant bit in the type, as specified in Section 2.1.

Section 4.3 defines the mutability of the remaining fields in the SRH (Flags, Tag, Segment List) in the context of the SID defined in this document.

New SIDs defined in the future MUST specify the mutability properties of the Flags, Tag, and Segment List and indicate how the HMAC TLV (Section 2.1.2) verification works. Note, that in effect these fields are mutable.

Consistent with the source routing model, the source of the SRH always knows how to set the segment list, Flags, Tag and TLVs of the SRH for use within the SR Domain. How it achieves this is outside the scope of this document, but may be based on topology, available SIDs and their mutability properties, the SRH mutability requirements of the destination, or any other information.

## 2.1. SRH TLVs

This section defines TLVs of the Segment Routing Header.

A TLV provides meta-data for segment processing. The only TLVs defined in this document are the HMAC (Section 2.1.2) and PAD (Section 2.1.1) TLVs. While processing the SID defined in Section 4.3.1, all TLVs are ignored unless local configuration indicates otherwise (Section 4.3.1.1.1). Thus, TLV and HMAC support is optional for any implementation, however, an implementation adding or parsing TLVs MUST support PAD TLVs. Other documents may define additional TLVs and processing rules for them.

TLVs are present when the Hdr Ext Len is greater than (Last Entry+1)\*2.

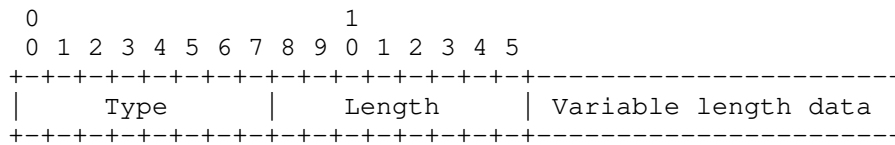
While processing TLVs at a segment endpoint, TLVs MUST be fully contained within the SRH as determined by the Hdr Ext Len. Detection of TLVs exceeding the boundary of the SRH Hdr Ext Len results in an ICMP Parameter Problem, Code 0, message to the Source Address, pointing to the Hdr Ext Len field of the SRH, and the packet being discarded.

An implementation MAY limit the number and/or length of TLVs it processes based on local configuration. It MAY:

- o Limit the number of consecutive Pad1 (Section 2.1.1.1) options to 1. If padding of more than one byte is required, then PadN (Section 2.1.1.2) should be used.
- o Limit the length in PadN to 5.
- o Limit the maximum number of non-Pad TLVs to be processed.
- o Limit the maximum length of all TLVs to be processed.

The implementation MAY stop processing additional TLVs in the SRH when these configured limits are exceeded.





Type: An 8 bit codepoint from Segment Routing Header TLVs Registry TBD IANA Reference. Unrecognized Types MUST be ignored on receipt.

Length: The length of the Variable length data in bytes.

Variable length data: Length bytes of data that is specific to the Type.

Type Length Value (TLV) entries contain OPTIONAL information that may be used by the node identified in the Destination Address (DA) of the packet.

Each TLV has its own length, format and semantic. The codepoint allocated (by IANA) to each TLV Type defines both the format and the semantic of the information carried in the TLV. Multiple TLVs may be encoded in the same SRH.

The highest-order bit of the TLV type (bit 0) specifies whether or not the TLV data of that type can change en route to the packet's final destination:

0: TLV data does not change en route

1: TLV data does change en route

All TLVs specify their alignment requirements using an  $xn+y$  format. The  $xn+y$  format is defined as per [RFC8200]. The SR Source nodes use the  $xn+y$  alignment requirements of TLVs and Padding TLVs when constructing an SRH.

The "Length" field of the TLV is used to skip the TLV while inspecting the SRH in case the node doesn't support or recognize the Type. The "Length" defines the TLV length in octets, not including the "Type" and "Length" fields.

The following TLVs are defined in this document:

Padding TLVs

HMAC TLV

Additional TLVs may be defined in the future.

## 2.1.1.1. Padding TLVs

There are two types of Padding TLVs, pad1 and padN, the following applies to both:

Padding TLVs are used for meeting the alignment requirement of the subsequent TLVs.

Padding TLVs are used to pad the SRH to a multiple of 8 octets.

Padding TLVs are ignored by a node processing the SRH TLV.

Multiple Padding TLVs MAY be used in one SRH

## 2.1.1.1.1. PAD1

Alignment requirement: none

```

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

```

Type: to be assigned by IANA (Suggested value 0)

A single Pad1 TLV MUST be used when a single byte of padding is required. A Pad1 TLV MUST NOT be used if more than one consecutive byte of padding is required.

## 2.1.1.1.2. PADN

Alignment requirement: none

```

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
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|           Type           |           Length           |           Padding (variable)           |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
//                               Padding (variable)                               //
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

Type: to be assigned by IANA (suggested value 4).

Length: 0 to 5

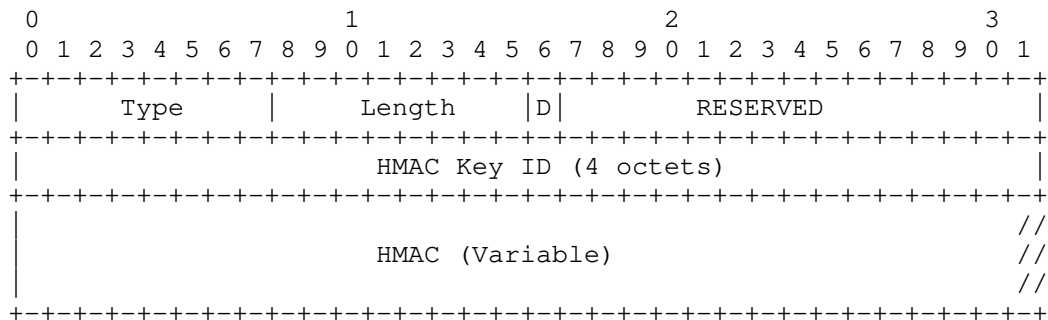
Padding: Length octets of padding. Padding bits have no semantic. They MUST be set to 0 on transmission and ignored on receipt.

The PadN TLV MUST be used when more than one byte of padding is required.

#### 2.1.1.2. HMAC TLV

Alignment requirement: 8n

The keyed Hashed Message Authentication Code (HMAC) TLV is OPTIONAL and has the following format:



where:

- o Type: to be assigned by IANA (suggested value 5).
- o Length: The length of the variable length data in bytes.
- o D: 1 bit. 1 indicates the Destination Address verification is disabled due to use of reduced segment list, Section 4.1.1.
- o RESERVED: 15 bits. MUST be 0 on transmission.
- o HMAC Key ID: A 4 octet opaque number which uniquely identifies the pre-shared key and algorithm used to generate the HMAC.
- o HMAC: Keyed HMAC, in multiples of 8 octets, at most 32 octets.

The HMAC TLV is used to verify that the SRH applied to a packet was selected by an authorized party, and to ensure that the segment list is not modified after generation. This also allows for verification that the current segment (by virtue of being in the authorized segment list) is authorized for use. The SR Domain ensures the source node is permitted to use the source address in the packet via ingress filtering mechanisms as defined in BCP 84 [RFC3704], or other strategies as appropriate.

#### 2.1.2.1. HMAC Generation and Verification

Local configuration determines when to check for an HMAC. This local configuration is outside the scope of this document. It may be based on the active segment at an SR Segment endpoint node, the result of an ACL that considers incoming interface, HMAC Key ID, or other packet fields.

An implementation that supports the generation and verification of the HMAC supports the following default behavior, as defined in the remainder of this section.

The HMAC verification begins by checking the current segment is equal to the destination address of the IPv6 header. The check is successful when either

- o HMAC D bit is 1 and Segments Left is greater than Last Entry.
- o HMAC Segments Left is less than or equal to Last Entry and destination address is equal to Segment List [Segments Left].

The HMAC field is the output of the HMAC computation as defined in [RFC2104], using:

- o key: the pre-shared key identified by HMAC Key ID
- o HMAC algorithm: identified by the HMAC Key ID
- o Text: a concatenation of the following fields from the IPv6 header and the SRH, as it would be received at the node verifying the HMAC:
  - \* IPv6 header: source address (16 octets)
  - \* SRH: Last Entry (1 octet)
  - \* SRH: Flags (1 octet)
  - \* SRH: HMAC 16 bits following Length
  - \* SRH: HMAC Key ID (4 octets)
  - \* SRH: all addresses in the Segment List (variable octets)

The HMAC digest is truncated to 32 octets and placed in the HMAC field of the HMAC TLV.

For HMAC algorithms producing digests less than 32 octets, the digest is placed in the lowest order octets of the HMAC field. Subsequent octets MUST be set to zero such that the HMAC length is a multiple of 8 octets.

If HMAC verification is successful, processing proceeds as normal.

If HMAC verification fails, an ICMP error message (parameter problem, error code 0, pointing to the HMAC TLV) SHOULD be generated (but rate limited) and SHOULD be logged and the packet discarded.

#### 2.1.2.2. HMAC Pre-Shared Key Algorithm

The HMAC Key ID field allows for the simultaneous existence of several hash algorithms (SHA-256, SHA3-256 ... or future ones) as well as pre-shared keys.

The HMAC Key ID field is opaque, i.e., it has neither syntax nor semantic except as an identifier of the right combination of pre-shared key and hash algorithm.

At the HMAC TLV generating and verification nodes, the Key ID uniquely identifies the pre-shared key and HMAC algorithm.

At the HMAC TLV generating node, the Text for the HMAC computation is set to the IPv6 header fields and SRH fields as they would appear at the verification node(s), not necessarily the same as the source node sending a packet with the HMAC TLV.

Pre-shared key roll-over is supported by having two key IDs in use while the HMAC TLV generating node and verifying node converge to a new key.

The HMAC TLV generating node may need to revoke an SRH for which it previously generated an HMAC. Revocation is achieved by allocating a new key and key ID, then rolling over the key ID associated with the SRH to be revoked. The HMAC TLV verifying node drops packets with the revoked SRH.

An implementation supporting HMAC can support multiple hash functions. An implementation supporting HMAC MUST implement SHA-2 [FIPS180-4] in its SHA-256 variant.

The selection of pre-shared key and algorithm, and their distribution is outside the scope of this document. Some options may include:

- o in the configuration of the HMAC generating or verifying nodes, either by static configuration or any SDN oriented approach

- o dynamically using a trusted key distribution protocol such as [RFC6407]

While key management is outside the scope of this document, the recommendations of BCP 107 [RFC4107] should be considered when choosing the key management system.

### 3. SR Nodes

There are different types of nodes that may be involved in segment routing networks: source SR nodes originate packets with a segment in the destination address of the IPv6 header, transit nodes that forward packets destined to a remote segment, and SR segment endpoint nodes that process a local segment in the destination address of an IPv6 header.

#### 3.1. Source SR Node

A Source SR Node is any node that originates an IPv6 packet with a segment (i.e. SRv6 SID) in the destination address of the IPv6 header. The packet leaving the source SR Node may or may not contain an SRH. This includes either:

A host originating an IPv6 packet.

An SR domain ingress router encapsulating a received packet in an outer IPv6 header, followed by an optional SRH.

The mechanism through which a segment in the destination address of the IPv6 header and the Segment List in the SRH, is derived is outside the scope of this document.

#### 3.2. Transit Node

A transit node is any node forwarding an IPv6 packet where the destination address of that packet is not locally configured as a segment nor a local interface. A transit node is not required to be capable of processing a segment nor SRH.

#### 3.3. SR Segment Endpoint Node

A SR segment endpoint node is any node receiving an IPv6 packet where the destination address of that packet is locally configured as a segment or local interface.

#### 4. Packet Processing

This section describes SRv6 packet processing at the SR source, Transit and SR segment endpoint nodes.

##### 4.1. Source SR Node

A Source node steers a packet into an SR Policy. If the SR Policy results in a segment list containing a single segment, and there is no need to add information to the SRH flag or to add TLV, the DA is set to the single segment list entry and the SRH MAY be omitted.

When needed, the SRH is created as follows:

Next Header and Hdr Ext Len fields are set as specified in [RFC8200].

Routing Type field is set as TBD (to be allocated by IANA, suggested value 4).

The DA of the packet is set with the value of the first segment.

The first element of the SRH Segment List is the ultimate segment. The second element is the penultimate segment, and so on.

The Segments Left field is set to  $n-1$  where  $n$  is the number of elements in the SR Policy.

The Last Entry field is set to  $n-1$  where  $n$  is the number of elements in the SR Policy.

TLVs (including HMAC) may be set according to their specification.

The packet is forwarded toward the packet's Destination Address (the first segment).

##### 4.1.1. Reduced SRH

When a source does not require the entire SID list to be preserved in the SRH, a reduced SRH may be used.

A reduced SRH does not contain the first segment of the related SR Policy (the first segment is the one already in the DA of the IPv6 header), and the Last Entry field is set to  $n-2$  where  $n$  is the number of elements in the SR Policy.

#### 4.2. Transit Node

As specified in [RFC8200], the only node allowed to inspect the Routing Extension Header (and therefore the SRH), is the node corresponding to the DA of the packet. Any other transit node MUST NOT inspect the underneath routing header and MUST forward the packet toward the DA according to its IPv6 routing table.

When a SID is in the destination address of an IPv6 header of a packet, it's routed through an IPv6 network as an IPv6 address. SIDs, or the prefix(es) covering SIDs, and their reachability may be distributed by means outside the scope of this document. For example, [RFC5308] or [RFC5340] may be used to advertise a prefix covering the SIDs on a node.

#### 4.3. SR Segment Endpoint Node

Without constraining the details of an implementation, the SR segment endpoint node creates Forwarding Information Base (FIB) entries for its local SIDs.

When an SRv6-capable node receives an IPv6 packet, it performs a longest-prefix-match lookup on the packets destination address. This lookup can return any of the following:

- \* A FIB entry that represents a locally instantiated SRv6 SID
- \* A FIB entry that represents a local interface, not locally instantiated as an SRv6 SID
- \* A FIB entry that represents a non-local route
- \* No Match

##### 4.3.1. FIB Entry Is Locally Instantiated SRv6 SID

This document, and section, defines a single SRv6 SID. Future documents may define additional SRv6 SIDs. In which case, the entire content of this section will be defined in that document.

If the FIB entry represents a locally instantiated SRv6 SID, process the next header chain of the IPv6 header as defined in section 4 of [RFC8200]. Section 4.3.1.1 describes how to process an SRH, Section 4.3.1.2 describes how to process an upper layer header or no next header.

Processing this SID modifies the Segments Left and, if configured to process TLVs, it may modify the "variable length data" of TLV types that change en route. Therefore Segments Left is mutable and TLVs that change en route are mutable. The remainder of the SRH (Flags,



Tag, Segment List, and TLVs that do not change en route) are immutable while processing this SID.

#### 4.3.1.1. SRH Processing

```
S01. When an SRH is processed {
S02.   If Segments Left is equal to zero {
S03.     Proceed to process the next header in the packet,
        whose type is identified by the Next Header field in
        the Routing header.
S04.   }
S05.   Else {
S06.     If local configuration requires TLV processing {
S07.       Perform TLV processing (see TLV Processing)
S08.     }
S09.     max_last_entry = ( Hdr Ext Len / 2 ) - 1
S10.     If ((Last Entry > max_last_entry) or
S11.        (Segments Left is greater than (Last Entry+1))) {
S12.       Send an ICMP Parameter Problem, Code 0, message to
        the Source Address, pointing to the Segments Left
        field, and discard the packet.
S13.     }
S14.     Else {
S15.       Decrement Segments Left by 1.
S16.       Copy Segment List[Segments Left] from the SRH to the
        destination address of the IPv6 header.
S17.       If the IPv6 Hop Limit is less than or equal to 1 {
S18.         Send an ICMP Time Exceeded -- Hop Limit Exceeded in
        Transit message to the Source Address and discard
        the packet.
S19.       }
S20.       Else {
S21.         Decrement the Hop Limit by 1
S22.         Resubmit the packet to the IPv6 module for transmission
        to the new destination.
S23.       }
S24.     }
S25.   }
S26. }
```

#### 4.3.1.1.1. TLV Processing

Local configuration determines how TLVs are to be processed when the Active Segment is a local SID defined in this document. The definition of local configuration is outside the scope of this document.

For illustration purpose only, two example local configurations that may be associated with a SID are provided below.

Example 1:

For any packet received from interface I2  
    Skip TLV processing

Example 2:

For any packet received from interface I1  
    If first TLV is HMAC {  
        Process the HMAC TLV  
    }  
    Else {  
        Discard the packet  
    }

#### 4.3.1.2. Upper-layer Header or No Next Header

When processing the Upper-layer header of a packet matching a FIB entry locally instantiated as an SRv6 SID defined in this document.

```
IF (Upper-layer Header is IPv4 or IPv6) and
    local configuration permits {
    Perform IPv6 decapsulation
    Resubmit the decapsulated packet to the IPv4 or IPv6 module
}
ELSE {
    Send an ICMP parameter problem message to the Source Address and
    discard the packet.  Error code (TBD by IANA) "SR Upper-layer
    Header Error", pointer set to the offset of the upper-layer
    header.
}
```

A unique error code allows an SR Source node to recognize an error in SID processing at an endpoint.

#### 4.3.2. FIB Entry Is A Local Interface

If the FIB entry represents a local interface, not locally instantiated as an SRv6 SID, the SRH is processed as follows:

    If Segments Left is zero, the node must ignore the Routing header and proceed to process the next header in the packet, whose type is identified by the Next Header field in the Routing Header.

    If Segments Left is non-zero, the node must discard the packet and send an ICMP Parameter Problem, Code 0, message to the packet's Source Address, pointing to the unrecognized Routing Type.

#### 4.3.3. FIB Entry Is A Non-Local Route

Processing is not changed by this document.

#### 4.3.4. FIB Entry Is A No Match

Processing is not changed by this document.

### 5. Intra SR Domain Deployment Model

The use of the SIDs exclusively within the SR Domain and solely for packets of the SR Domain is an important deployment model.

This enables the SR Domain to act as a single routing system.

This section covers:

- o securing the SR Domain from external attempt to use its SIDs
- o SR Domain as a single system with delegation between components
- o handling packets of the SR Domain

#### 5.1. Securing the SR Domain

Nodes outside the SR Domain are not trusted: they cannot directly use the SIDs of the domain. This is enforced by two levels of access control lists:

1. Any packet entering the SR Domain and destined to a SID within the SR Domain is dropped. This may be realized with the following logic. Other methods with equivalent outcome are considered compliant:
  - \* allocate all the SID's from a block S/s
  - \* configure each external interface of each edge node of the domain with an inbound infrastructure access list (IACL) which drops any incoming packet with a destination address in S/s
  - \* Failure to implement this method of ingress filtering exposes the SR Domain to source routing attacks as described and referenced in [RFC5095]
2. The distributed protection in #1 is complemented with per node protection, dropping packets to SIDs from source addresses outside the SR Domain. This may be realized with the following

logic. Other methods with equivalent outcome are considered compliant:

- \* assign all interface addresses from prefix A/a
- \* at node k, all SIDs local to k are assigned from prefix Sk/sk
- \* configure each internal interface of each SR node k in the SR Domain with an inbound IACL which drops any incoming packet with a destination address in Sk/sk if the source address is not in A/a.

## 5.2. SR Domain as A Single System with Delegation Among Components

All intra SR Domain packets are of the SR Domain. The IPv6 header is originated by a node of the SR Domain, and is destined to a node of the SR Domain.

All inter domain packets are encapsulated for the part of the packet journey that is within the SR Domain. The outer IPv6 header is originated by a node of the SR Domain, and is destined to a node of the SR Domain.

As a consequence, any packet within the SR Domain is of the SR Domain.

The SR Domain is a system in which the operator may want to distribute or delegate different operations of the outer most header to different nodes within the system.

An operator of an SR domain may choose to delegate SRH addition to a host node within the SR domain, and validation of the contents of any SRH to a more trusted router or switch attached to the host. Consider a top of rack switch (T) connected to host (H) via interface (I). H receives an SRH (SRH1) with a computed HMAC via some SDN method outside the scope of this document. H classifies traffic it sources and adds SRH1 to traffic requiring a specific SLA. T is configured with an IACL on I requiring verification of the SRH for any packet destined to the SID block of the SR Domain (S/s). T checks and verifies that SRH1 is valid, contains an HMAC TLV and verifies the HMAC.

An operator of the SR Domain may choose to have all segments in the SR Domain verify the HMAC. This mechanism would verify that the SRH segment list is not modified while traversing the SR Domain.

### 5.3. MTU Considerations

An SR Domain ingress edge node encapsulates packets traversing the SR Domain, and needs to consider the MTU of the SR Domain. Within the SR Domain, well known mitigation techniques are RECOMMENDED, such as deploying a greater MTU value within the SR Domain than at the ingress edges.

Encapsulation with an outer IPv6 header and SRH share the same MTU and fragmentation considerations as IPv6 tunnels described in [RFC2473]. Further investigation on the limitation of various tunneling methods (including IPv6 tunnels) are discussed in [I-D.ietf-intarea-tunnels] and SHOULD be considered by operators when considering MTU within the SR Domain.

### 5.4. ICMP Error Processing

ICMP error packets generated within the SR Domain are sent to source nodes within the SR Domain. The invoking packet in the ICMP error message may contain an SRH. Since the destination address of a packet with an SRH changes as each segment is processed, it may not be the destination used by the socket or application that generated the invoking packet.

For the source of an invoking packet to process the ICMP error message, the ultimate destination address of the IPv6 header may be required. The following logic is used to determine the destination address for use by protocol error handlers.

- o Walk all extension headers of the invoking IPv6 packet to the routing extension header preceding the upper layer header.
  - \* If routing header is type TBD IANA (SRH)
    - + The SID at Segment List[0] may be used as the destination address of the invoking packet.

ICMP errors are then processed by upper layer transports as defined in [RFC4443].

For IP packets encapsulated in an outer IPv6 header, ICMP error handling is as defined in [RFC2473].

### 5.5. Load Balancing and ECMP

For any inter domain packet, the SR Source node MUST impose a flow label computed based on the inner packet. The computation of the

flow label is as recommended in [RFC6438] for the sending Tunnel End Point.

For any intra domain packet, the SR Source node SHOULD impose a flow label computed as described in [RFC6437] to assist ECMP load balancing at transit nodes incapable of computing a 5-tuple beyond the SRH.

At any transit node within an SR domain, the flow label MUST be used as defined in [RFC6438] to calculate the ECMP hash toward the destination address. If flow label is not used, the transit node would likely hash all packets between a pair of SR Edge nodes to the same link.

At an SR segment endpoint node, the flow label MUST be used as defined in [RFC6438] to calculate any ECMP hash used to forward the processed packet to the next segment.

## 5.6. Other Deployments

Other deployment models and their implications on security, MTU, HMAC, ICMP error processing and interaction with other extension headers are outside the scope of this document.

## 6. Illustrations

This section provides illustrations of SRv6 packet processing at SR source, transit and SR segment endpoint nodes.

### 6.1. Abstract Representation of an SRH

For a node  $k$ , its IPv6 address is represented as  $A_k$ , its SRv6 SID is represented as  $S_k$ .

IPv6 headers are represented as the tuple of (source, destination). For example, a packet with source address  $A_1$  and destination address  $A_2$  is represented as  $(A_1, A_2)$ . The payload of the packet is omitted.

An SR Policy is a list of segments. A list of segments is represented as  $\langle S_1, S_2, S_3 \rangle$  where  $S_1$  is the first SID to visit,  $S_2$  is the second SID to visit and  $S_3$  is the last SID to visit.

$(SA, DA) (S_3, S_2, S_1; SL)$  represents an IPv6 packet with:

- o Source Address is SA, Destination Addresses is DA, and next-header is SRH.
- o SRH with SID list  $\langle S_1, S_2, S_3 \rangle$  with SegmentsLeft = SL.

- o Note the difference between the <> and () symbols. <S1, S2, S3> represents a SID list where the leftmost segment is the first segment. Whereas, (S3, S2, S1; SL) represents the same SID list but encoded in the SRH Segment List format where the leftmost segment is the last segment. When referring to an SR policy in a high-level use-case, it is simpler to use the <S1, S2, S3> notation. When referring to an illustration of detailed behavior, the (S3, S2, S1; SL) notation is more convenient.

At its SR Policy headend, the Segment List <S1,S2,S3> results in SRH (S3,S2,S1; SL=2) represented fully as:

```
Segments Left=2
Last Entry=2
Flags=0
Tag=0
Segment List[0]=S3
Segment List[1]=S2
Segment List[2]=S1
```

## 6.2. Example Topology

The following topology is used in examples below:

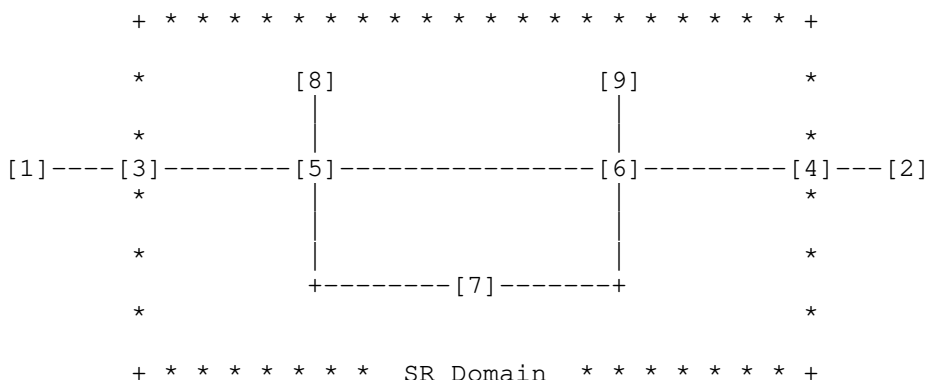


Figure 3

- o 3 and 4 are SR Domain edge routers
- o 5, 6, and 7 are all SR Domain routers
- o 8 and 9 are hosts within the SR Domain
- o 1 and 2 are hosts outside the SR Domain

- o The SR domain implements ingress filtering as per Section 5.1 and no external packet can enter the domain with a destination address equal to a segment of the domain.

### 6.3. Source SR Node

#### 6.3.1. Intra SR Domain Packet

When host 8 sends a packet to host 9 via an SR Policy <S7,A9> the packet is

P1: (A8,S7) (A9,S7; SL=1)

##### 6.3.1.1. Reduced Variant

When host 8 sends a packet to host 9 via an SR Policy <S7,A9> and it wants to use a reduced SRH, the packet is

P2: (A8,S7) (A9; SL=1)

#### 6.3.2. Inter SR Domain Packet - Transit

When host 1 sends a packet to host 2, the packet is

P3: (A1,A2)

The SR Domain ingress router 3 receives P3 and steers it to SR Domain egress router 4 via an SR Policy <S7, S4>. Router 3 encapsulates the received packet P3 in an outer header with an SRH. The packet is

P4: (A3, S7) (S4, S7; SL=1) (A1, A2)

If the SR Policy contains only one segment (the egress router 4), the ingress Router 3 encapsulates P3 into an outer header (A3, S4) without SRH. The packet is

P5: (A3, S4) (A1, A2)

##### 6.3.2.1. Reduced Variant

The SR Domain ingress router 3 receives P3 and steers it to SR Domain egress router 4 via an SR Policy <S7, S4>. If router 3 wants to use a reduced SRH, Router 3 encapsulates the received packet P3 in an outer header with a reduced SRH. The packet is

P6: (A3, S7) (S4; SL=1) (A1, A2)



### 6.3.3. Inter SR Domain Packet - Internal to External

When host 8 sends a packet to host 1, the packet is encapsulated for the portion of its journey within the SR Domain. From 8 to 3 the packet is

P7: (A8,S3) (A8,A1)

In the opposite direction, the packet generated from 1 to 8 is

P8: (A1,A8)

At node 3 P8 is encapsulated for the portion of its journey within the SR domain, with the outer header destined to segment S8. Resulting in

P9: (A3,S8) (A1,A8)

At node 8 the outer IPv6 header is removed by S8 processing, then processed again when received by A8.

### 6.4. Transit Node

Nodes 5 acts as transit nodes for packet P1, and sends packet

P1: (A8,S7) (A9,S7;SL=1)

on the interface toward node 7.

### 6.5. SR Segment Endpoint Node

Node 7 receives packet P1 and, using the logic in Section 4.3.1, sends packet

P7: (A8,A9) (A9,S7; SL=0)

on the interface toward router 6.

### 6.6. Delegation of Function with HMAC Verification

This section describes how a function may be delegated within the SR Domain. In the following sections consider a host 8 connected to a top of rack 5.

#### 6.6.1. SID List Verification

An operator may prefer to apply the SRH at source 8, while 5 verifies the SID list is valid.

For illustration purpose, an SDN controller provides 8 an SRH terminating at node 9, with segment list <S5,S7,S6,A9>, and HMAC TLV computed for the SRH. The HMAC key ID and key associated with the HMAC TLV is shared with 5. Node 8 does not know the key. Node 5 is configured with an IACL applied to the interface connected to 8, requiring HMAC verification for any packet destined to S/s.

Node 8 originates packets with the received SRH including HMAC TLV.

P15: (A8,S5) (A9,S6,S7,S5;SL=3;HMAC)

Node 5 receives and verifies the HMAC for the SRH, then forwards the packet to the next segment

P16: (A8,S7) (A9,S6,S7,S5;SL=2;HMAC)

Node 6 receives

P17: (A8,S6) (A9,S6,S7,S5;SL=1;HMAC)

Node 9 receives

P18: (A8,A9) (A9,S6,S7,S5;SL=0;HMAC)

This use of an HMAC is particularly valuable within an enterprise based SR Domain [SRN].

### 7. Security Considerations

This section reviews security considerations related to the SRH, given the SRH processing and deployment models discussed in this document.

As described in Section 5, it is necessary to filter packets ingress to the SR Domain, destined to SIDs within the SR Domain (i.e., bearing a SID in the destination address). This ingress filtering is via an IACL at SR Domain ingress border nodes. Additional protection is applied via an IACL at each SR Segment Endpoint node, filtering packets not from within the SR Domain, destined to SIDs in the SR Domain. ACLs are easily supported for small numbers of prefixes, making summarization important, and when the prefixes requiring filtering is kept to a seldom changing set.

Additionally, ingress filtering of IPv6 source addresses as recommended in BCP38 [RFC2827] SHOULD be used.

#### 7.1. Source Routing Attacks

An SR domain implements distributed and per node protection as described in section 5.1. Additionally, domains deny traffic with spoofed addresses by implementing the recommendations in BCP 84 [RFC3704].

Full implementation of the recommended protection blocks the attacks documented in [RFC5095] from outside the SR domain, including bypassing filtering devices, reaching otherwise unreachable Internet systems, network topology discovery, bandwidth exhaustion, and defeating anycast.

Failure to implement distributed and per node protection allows attackers to bypass filtering devices and exposes the SR Domain to these attacks.

Compromised nodes within the SR Domain may mount the attacks listed above along with other known attacks on IP networks (e.g. DOS/DDOS, topology discovery, man-in-the-middle, traffic interception/siphoning).

#### 7.2. Service Theft

Service theft is defined as the use of a service offered by the SR Domain by a node not authorized to use the service.

Service theft is not a concern within the SR Domain as all SR Source nodes and SR segment endpoint nodes within the domain are able to utilize the services of the Domain. If a node outside the SR Domain learns of segments or a topological service within the SR domain, IACL filtering denies access to those segments.

#### 7.3. Topology Disclosure

The SRH is unencrypted and may contain SIDs of some intermediate SR-nodes in the path towards the destination within the SR Domain. If packets can be snooped within the SR Domain, the SRH may reveal topology, traffic flows, and service usage.

This is applicable within an SR Domain, but the disclosure is less relevant as an attacker has other means of learning topology, flows, and service usage.

#### 7.4. ICMP Generation

The generation of ICMPv6 error messages may be used to attempt denial-of-service attacks by sending an error-causing destination address or SRH in back-to-back packets. An implementation that correctly follows Section 2.4 of [RFC4443] would be protected by the ICMPv6 rate-limiting mechanism.

#### 7.5. Applicability of AH

The SR Domain is a trusted domain, as defined in [RFC8402] Section 2 and Section 8.2. The SR Source is trusted to add an SRH (optionally verified as having been generated by a trusted source via the HMAC TLV in this document), and segments advertised within the domain are trusted to be accurate and advertised by trusted sources via a secure control plane. As such the SR Domain does not rely on the Authentication Header (AH) as defined in [RFC4302] to secure the SRH.

The use of SRH with AH by an SR source node, and processing at a SR segment endpoint node is not defined in this document. Future documents may define use of SRH with AH and its processing.

#### 8. IANA Considerations

This document makes the following registrations in the Internet Protocol Version 6 (IPv6) Parameters "Routing Type" registry maintained by IANA:

| Suggested Value | Description                  | Reference     |
|-----------------|------------------------------|---------------|
| 4               | Segment Routing Header (SRH) | This document |

This document makes the following registrations in "Type 4 - Parameter Problem" message of the "Internet Control Message Protocol version 6 (ICMPv6) Parameters" registry maintained by IANA:

| CODE     | NAME/DESCRIPTION            |
|----------|-----------------------------|
| TBD IANA | SR Upper-layer Header Error |

This section provides guidance to the Internet Assigned Numbers Authority (IANA) regarding registration of values related to the SRH, in accordance with BCP 26, [RFC8126].

The following terms are used here with the meanings defined in BCP 26: "namespace", "assigned value", "registration".

The following policies are used here with the meanings defined in BCP 26 [RFC8126]: "IETF Review".

#### 8.1. Segment Routing Header Flags Registry

This document requests the creation of a new IANA managed registry to identify SRH Flags Bits. The registration procedure is "IETF Review". Suggested registry name is "Segment Routing Header Flags". Flags is 8 bits.

#### 8.2. Segment Routing Header TLVs Registry

This document requests the creation of a new IANA managed registry to identify SRH TLVs. The registration procedure is "IETF Review". Suggested registry name is "Segment Routing Header TLVs". A TLV is identified through an unsigned 8 bit codepoint value, with assigned values 0-127 for TLVs that do not change en route, and 128-255 for TLVs that may change en route. The following codepoints are defined in this document:

| Assigned Value | Description              | Reference     |
|----------------|--------------------------|---------------|
| 0              | Pad1 TLV                 | This document |
| 1              | Reserved                 | This document |
| 2              | Reserved                 | This document |
| 3              | Reserved                 | This document |
| 4              | PadN TLV                 | This document |
| 5              | HMAC TLV                 | This document |
| 6              | Reserved                 | This document |
| 124-126        | Experimentation and Test | This document |
| 127            | Reserved                 | This document |
| 252-254        | Experimentation and Test | This document |
| 255            | Reserved                 | This document |

Values 1,2,3,6 were defined in draft versions of this specification and are Reserved for backwards compatibility with early implementations and should not be reassigned. Values 127 and 255 are Reserved to allow for expansion of the Type field in future specifications if needed.

#### 9. Implementation Status

This section is to be removed prior to publishing as an RFC.

See [I-D.matsushima-spring-srv6-deployment-status] for updated deployment and interoperability reports.

### 9.1. Linux

Name: Linux Kernel v4.14

Status: Production

Implementation: adds SRH, performs END processing, supports HMAC TLV

Details: <https://irtf.org/anrw/2017/anrw17-final3.pdf> and  
[I-D.filsfils-spring-srv6-interop]

### 9.2. Cisco Systems

Name: IOS XR and IOS XE

Status: Production (IOS XR), Pre-production (IOS XE)

Implementation: adds SRH, performs END processing, no TLV processing

Details: [I-D.filsfils-spring-srv6-interop]

### 9.3. FD.io

Name: VPP/Segment Routing for IPv6

Status: Production

Implementation: adds SRH, performs END processing, no TLV processing

Details: [https://wiki.fd.io/view/VPP/Segment\\_Routing\\_for\\_IPv6](https://wiki.fd.io/view/VPP/Segment_Routing_for_IPv6) and  
[I-D.filsfils-spring-srv6-interop]

### 9.4. Barefoot

Name: Barefoot Networks Tofino NPU

Status: Prototype

Implementation: performs END processing, no TLV processing

Details: [I-D.filsfils-spring-srv6-interop]

### 9.5. Juniper

Name: Juniper Networks Trio and vTrio NPU's

Status: Prototype & Experimental

Implementation: SRH insertion mode, Process SID where SID is an interface address, no TLV processing

#### 9.6. Huawei

Name: Huawei Systems VRP Platform

Status: Production

Implementation: adds SRH, performs END processing, no TLV processing

#### 10. Contributors

Kamran Raza, Zafar Ali, Brian Field, Daniel Bernier, Ida Leung, Jen Linkova, Ebben Aries, Tomoya Kosugi, Eric Vyncke, David Lebrun, Dirk Steinberg, Robert Raszuk, Dave Barach, John Brzozowski, Pierre Francois, Nagendra Kumar, Mark Townsley, Christian Martin, Roberta Maglione, James Connolly, Aloys Augustin contributed to the content of this document.

#### 11. Acknowledgements

The authors would like to thank Ole Troan, Bob Hinden, Ron Bonica, Fred Baker, Brian Carpenter, Alexandru Petrescu, Punit Kumar Jaiswal, David Lebrun, Benjamin Kaduk, Frank Xialiang, Mirja Kuhlewind, Roman Danyliw, Joe Touch, and Magnus Westerlund for their comments to this document.

#### 12. References

##### 12.1. Normative References

[FIPS180-4]

National Institute of Standards and Technology, "FIPS 180-4 Secure Hash Standard (SHS)", March 2012, <<http://csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf>>.

[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>>.

[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>>.

- [RFC2473] Conta, A. and S. Deering, "Generic Packet Tunneling in IPv6 Specification", RFC 2473, DOI 10.17487/RFC2473, December 1998, <<https://www.rfc-editor.org/info/rfc2473>>.
- [RFC2827] Ferguson, P. and D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing", BCP 38, RFC 2827, DOI 10.17487/RFC2827, May 2000, <<https://www.rfc-editor.org/info/rfc2827>>.
- [RFC3704] Baker, F. and P. Savola, "Ingress Filtering for Multihomed Networks", BCP 84, RFC 3704, DOI 10.17487/RFC3704, March 2004, <<https://www.rfc-editor.org/info/rfc3704>>.
- [RFC4107] Bellovin, S. and R. Housley, "Guidelines for Cryptographic Key Management", BCP 107, RFC 4107, DOI 10.17487/RFC4107, June 2005, <<https://www.rfc-editor.org/info/rfc4107>>.
- [RFC4302] Kent, S., "IP Authentication Header", RFC 4302, DOI 10.17487/RFC4302, December 2005, <<https://www.rfc-editor.org/info/rfc4302>>.
- [RFC5095] Abley, J., Savola, P., and G. Neville-Neil, "Deprecation of Type 0 Routing Headers in IPv6", RFC 5095, DOI 10.17487/RFC5095, December 2007, <<https://www.rfc-editor.org/info/rfc5095>>.
- [RFC6407] Weis, B., Rowles, S., and T. Hardjono, "The Group Domain of Interpretation", RFC 6407, DOI 10.17487/RFC6407, October 2011, <<https://www.rfc-editor.org/info/rfc6407>>.
- [RFC6437] Amante, S., Carpenter, B., Jiang, S., and J. Rajahalme, "IPv6 Flow Label Specification", RFC 6437, DOI 10.17487/RFC6437, November 2011, <<https://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, <<https://www.rfc-editor.org/info/rfc6438>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <<https://www.rfc-editor.org/info/rfc8200>>.



- [RFC8402] Filsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402, July 2018, <<https://www.rfc-editor.org/info/rfc8402>>.

## 12.2. Informative References

- [I-D.filsfils-spring-srv6-interop]  
Filsfils, C., Clad, F., Camarillo, P., Abdelsalam, A., Salsano, S., Bonaventure, O., Horn, J., and J. Liste, "SRv6 interoperability report", draft-filsfils-spring-srv6-interop-02 (work in progress), March 2019.
- [I-D.ietf-intarea-tunnels]  
Touch, J. and M. Townsley, "IP Tunnels in the Internet Architecture", draft-ietf-intarea-tunnels-10 (work in progress), September 2019.
- [I-D.matsushima-spring-srv6-deployment-status]  
Matsushima, S., Filsfils, C., Ali, Z., and Z. Li, "SRv6 Implementation and Deployment Status", draft-matsushima-spring-srv6-deployment-status-02 (work in progress), October 2019.
- [RFC4443] Conta, A., Deering, S., and M. Gupta, Ed., "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", STD 89, RFC 4443, DOI 10.17487/RFC4443, March 2006, <<https://www.rfc-editor.org/info/rfc4443>>.
- [RFC5308] Hopps, C., "Routing IPv6 with IS-IS", RFC 5308, DOI 10.17487/RFC5308, October 2008, <<https://www.rfc-editor.org/info/rfc5308>>.
- [RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, DOI 10.17487/RFC5340, July 2008, <<https://www.rfc-editor.org/info/rfc5340>>.
- [RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <<https://www.rfc-editor.org/info/rfc8126>>.
- [SRN] Lebrun, D., Jadin, M., Clad, F., Filsfils, C., and O. Bonaventure, "Software Resolved Networks: Rethinking Enterprise Networks with IPv6 Segment Routing", 2018, <<https://inl.info.ucl.ac.be/system/files/sosr18-final15-embedfonts.pdf>>.

## Authors' Addresses

Clarence Filsfils (editor)  
Cisco Systems, Inc.  
Brussels  
BE

Email: cfilsfil@cisco.com

Darren Dukes (editor)  
Cisco Systems, Inc.  
Ottawa  
CA

Email: ddukes@cisco.com

Stefano Previdi  
Huawei  
Italy

Email: stefano@previdi.net

John Leddy  
Individual  
US

Email: john@leddy.net

Satoru Matsushima  
Softbank

Email: satoru.matsushima@g.softbank.co.jp

Daniel Voyer  
Bell Canada

Email: daniel.voyer@bell.ca

Network  
Internet-Draft  
Intended status: Standards Track  
Expires: 29 March 2022

C. Weiqiang  
China Mobile  
G. Mirsky  
Ericsson  
P. Shaofu  
L. Aihua  
ZTE Corporation  
G. Mishra  
Verizon Inc.  
25 September 2021

Unified Identifier in IPv6 Segment Routing Networks  
draft-mirsky-6man-unified-id-sr-10

Abstract

Segment Routing architecture leverages the paradigm of source routing. It can be realized in a network data plane by prepending the packet with a list of instructions, a.k.a. segments. A segment can be encoded as a Multi-Protocol Label Switching (MPLS) label, IPv4 address, or IPv6 address. Segment Routing can be applied in the MPLS data plane by encoding segments in the MPLS label stack. It also can be applied to the IPv6 data plane by encoding a list of segment identifiers in IPv6 Segment Routing Extension Header (SRH). This document extends the use of the SRH to unified segment identifiers encoded, for example, as MPLS label or IPv4 address, to compress the SRH, and support more detailed network programming and interworking between SR-MPLS and SRv6 domains.

Status of This Memo

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

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

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

This Internet-Draft will expire on 29 March 2022.

## Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

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

## Table of Contents

|                                                                                      |    |
|--------------------------------------------------------------------------------------|----|
| 1. Introduction . . . . .                                                            | 3  |
| 1.1. Conventions used in this document . . . . .                                     | 3  |
| 1.1.1. Acronyms . . . . .                                                            | 3  |
| 1.1.2. Requirements Language . . . . .                                               | 4  |
| 2. Segment Routing Extension Header: Benefits and Challenges . .                     | 4  |
| 3. Unified SIDs in IPv6 Segment Routing Extension Header . . . .                     | 4  |
| 4. Operations with Unified Segment Identifier . . . . .                              | 6  |
| 4.1. Procedures of 32bits MPLS Label within SRH . . . . .                            | 7  |
| 4.1.1. Packet Forwarding Based on UET-MPLS U-SID . . . . .                           | 8  |
| 4.2. Procedures of 32bits IP Address within SRH . . . . .                            | 9  |
| 4.2.1. Packet Forwarding Based on UET-32 U-SID . . . . .                             | 10 |
| 5. The Use Case of Unified Segment Identifier . . . . .                              | 11 |
| 5.1. Nesting Interworking Between SR-MPLS and SRv6 Using Binding<br>U-SID . . . . .  | 12 |
| 5.2. Flat Interworking Between Different UET Domains Using<br>Mixing U-SID . . . . . | 15 |
| 5.2.1. UET Capability Advertisement . . . . .                                        | 15 |
| 5.2.2. SRv6 SID Allocated per UEC . . . . .                                          | 16 |
| 5.2.3. Packets Forwarding Procedures . . . . .                                       | 17 |
| 6. Control Plane in Support of Unified SID . . . . .                                 | 21 |
| 7. SRH with U-SID Pseudo-code . . . . .                                              | 22 |
| 8. U-SID supporting SRv6 programming . . . . .                                       | 23 |
| 9. Benefits . . . . .                                                                | 24 |
| 10. Implementation Considerations . . . . .                                          | 24 |
| 11. IANA Considerations . . . . .                                                    | 24 |
| 12. Security Considerations . . . . .                                                | 24 |
| 13. Contributors . . . . .                                                           | 24 |
| 14. Acknowledgements . . . . .                                                       | 25 |
| 15. Normative References . . . . .                                                   | 25 |
| Authors' Addresses . . . . .                                                         | 27 |

## 1. Introduction

Segment Routing architecture [RFC8402] leverages the paradigm of source routing. It can be realized in a network data plane by prepending the packet with a list of instructions, a.k.a. segment identifiers (SIDs). A segment can be encoded as a Multi-Protocol Label Switching (MPLS) label, IPv4 address, or IPv6 address. Segment Routing can be applied in MPLS data plane by encoding 20-bits SIDs in MPLS label stack [RFC8660]. It also can be applied to IPv6 data plane by encoding a list of 128-bits SIDs in IPv6 Segment Routing Extension Header (SRH) [RFC8754].

This document extends the use of the SRH [RFC8754] to unified identifiers encoded as MPLS label or IPv4 address to support more detailed network programming and interworking between SR-MPLS and SRv6 domains.

### 1.1. Conventions used in this document

#### 1.1.1. Acronyms

SR: Segment Routing

SRH: Segment Routing Extension Header

MPLS: Multiprotocol Label Switching

SR-MPLS: Segment Routing using MPLS data plane

SID: Segment Identifier

IGP: Interior Gateway Protocol

DA: Destination Address

ILM: Incoming Label Map

FEC: Forwarding Equivalence Class

FTN: FEC-to-NHLFE map

OAM: Operation, Administration, and Maintenance

TE: Traffic Engineering

SRv6: Segment Routing in IPv6

U-SID: Unified Segment Identifier

PSP: Penultimate Segment Popping

FIB: Forwarding Information Base

UET: U-SID Encapsulation Type

UEC: U-SID Encapsulation Capability

### 1.1.2. Requirements Language

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

## 2. Segment Routing Extension Header: Benefits and Challenges

Many functions related to Operation, Administration, and Maintenance (OAM) require identification of the SR tunnel ingress and the path, constructed by segments, between the ingress and the egress SR nodes. Combination of IPv6 encapsulation [RFC8200] and SRH [RFC8754], referred to as SRv6, comply with these requirements while it is challenging when applying SR in MPLS networks, also referred to as SR-MPLS.

On the other hand, the size of IPv6 SID presents a scaling challenge to use topological instructions that define strict explicit traffic-engineered (TE) path or support network programming in combination with service-based instructions. At the same time, that is where SR-MPLS approach provides better results due to the smaller SID length. It can be used to compress the SRv6 header size when a smaller namespace of available SIDs is sufficient for addressing the particular network.

SR-MPLS is broadly used in metro networks. With the gradual deployment of SRv6 in the core networks, supporting interworking between SR-MPLS and SRv6 becomes necessary for operators. It is operationally more efficient and straightforward if SRv6 can use the same size SIDs as in SR-MPLS. The SRH can be extended to define the same, as in SR-MPLS, SID length to support the unified segment identifier (U-SID). As a result, end-to-end SR tunnel may use U-SIDs across SR-MPLS and SRv6 domains.

## 3. Unified SIDs in IPv6 Segment Routing Extension Header

SRH format has been defined in Section 3 of [RFC8754] as presented in Figure 1

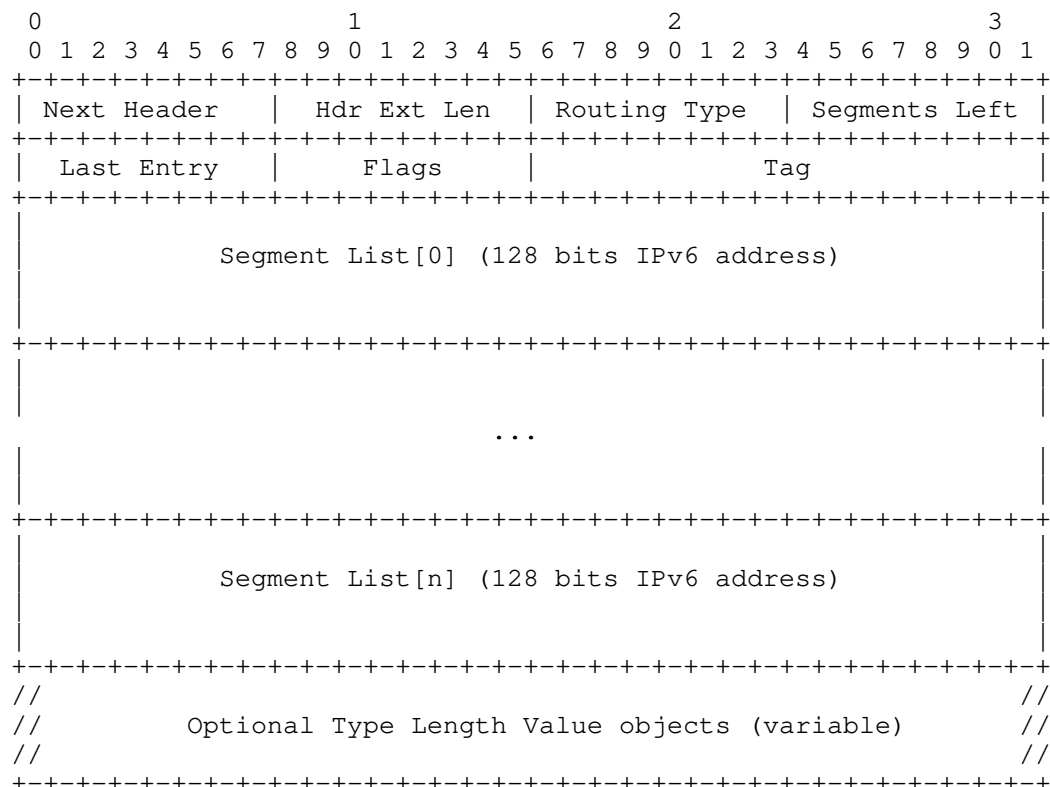


Figure 1: SRH format

This document defines a new field Size in the SRH Flags field as a two-bits field, termed as UET (U-SID Encapsulation Type) flag, to indicate which type of SIDs are encoded in SRH. The UET flag has the following values:

0b00 - indicate a 128-bits SID, an IPv6 address, termed as UET-128 U-SID.

0b01 - indicate a 32-bits SID, termed as UET-32 U-SID. In some environments, the context could be of IPv4 address, while in some other cases, it could represent an index of list or range of IPv4/IPv6 addresses. Another interpretation of 32-bits SID could be as a complementary element of an IPv4/IPv6 prefix. Setting the interpretation might be made through the control plane-based signaling and is outside the scope of this document. If this SID represents a complementary part of an IPv4/IPv6 prefix, the original IP address can be re-constructed by using, for example,

mapping, stitching, shifting, or translating operation. Specification of such a mechanism is outside the scope of this document.

0b10 - indicate a 32-bits SID, termed as UET-MPLS U-SID, which includes an MPLS label in the leftmost 20-bits as displayed in Figure 2. Information in the Context field could be interpreted as a flavor of a particular network programming behavior. Specification of the network programming using this type of U-SID is outside the scope of this document. [Ed.note. Replace with reference to the U-SID network programming document.]

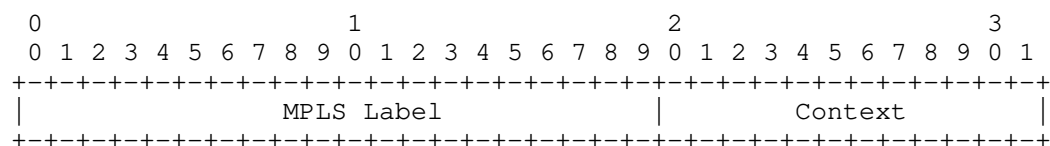


Figure 2: Format of Unified SID with MPLS Label

0b11 - indicate a 16-bits SID, termed as UET-16 U-SID. It is similar to 32bits SID and suitable for scenes with higher compression efficiency

This document also introduces a compatible operation on Segment Left field, also termed as SRH.SL. The relationship between the value of SRH.UET and the interpretation of the SRH.SL is as follows:

- \* if SRH.UET Flag is UET-128, SRH.SL represents the count of 128bits-SID entries in SRH;
- \* if SRH.UET Flag is UET-32 or UET-MPLS, SRH.SL represents the count of 32bits-SID entries in SRH;
- \* if SRH.UET Flag is UET-16, SRH.SL represents the count of 16bits-SID entries in SRH.

#### 4. Operations with Unified Segment Identifier

When SRH is used to include 32-bits long U-SIDs, the ingress and transit nodes of an SR tunnel act as described in Section 5.1 and Section 5.2 of [RFC8754] respectively.



#### 4.1. Procedures of 32bits MPLS Label within SRH

This section describes how the UET-MPLS type of U-SID is used to encode a compressed SRH. In this case, an ILM (Incoming Label Map) entry can be used to map a U-SID to an IPv6 address. As a result, it is not necessary to introduce a new type of index-based mapping table. For the ILM entry of Adjacency-SID, the mapping result copied to DA (Destination Address) is the remote interface IPv6 address. For the ILM entry of Node-SID, the mapping result copied into DA is a remote node loopback IPv6 address.

Operations on an MPLS label of U-SID type are the same as those defined in [RFC8663]. However, SR-MPLS over SRH has the following advantages compared with SR-MPLS over UDP:

- \* SRH is flexible to extend flags or sub-TLVs for service requirements, but not in the case of SR-MPLS over UDP.
- \* Labels in SRH can meet 8 bytes alignment requirements as per [RFC8200], but not in the case of SR-MPLS over UDP.
- \* The source address and the complete path information of the SR policy are not discarded, but not in the case of SR-MPLS over UDP.
- \* The forwarding performance of SR-MPLS over SRH is better than the UDP method because it only updates the destination address rather than frequently removing and adding outer headers.

Procedures of SR-MPLS over IP of [RFC8663] described how to construct an adjusted SR-MPLS FTN (FEC-to-NHLFE map) and ILM entry towards a prefix-SID when next-hops are IP-only routers. The action of FTN and ILM entry will steer the packet along an outer tunnel to the destination node that has originated the FEC (Forwarding Equivalence Class). UDP header is removed and put again at each segment endpoint. However, for SR-MPLS over SRH in this document, the proposed solution is not dependent on adjusted FIB (Forwarding Information Base) entry. That is because no action is needed to get from the FIB entry. A traditional ILM entry (maybe without out-label because of IP-only next-hop) is enough to get the FEC information, i.e., map a U-SID to an IPv6 address and copy to DA. Note that an implementation can get FEC and next-hop/interface forwarding information from the ILM entry, avoiding extra FIB lookup. An SRv6 policy chosen to encapsulate the U-SID list within SRH is determined at the ingress node of this SRv6 policy. The SRH is preserved along the SR to the egress. However, in the case of PSP (Penultimate Segment Popping), the behavior is different from SR-MPLS over IP/UDP method [RFC8663], so the source address (i.e., the ingress of the SRv6 policy) is not discarded.

## 4.1.1. Packet Forwarding Based on UET-MPLS U-SID

The packet forwarding based on UET-MPLS U-SID is similar to the processing described in [RFC8663]. But it differs from that in FIB action and segment list processing. For completeness, we repeat the description of [RFC8663] with modification as follows.

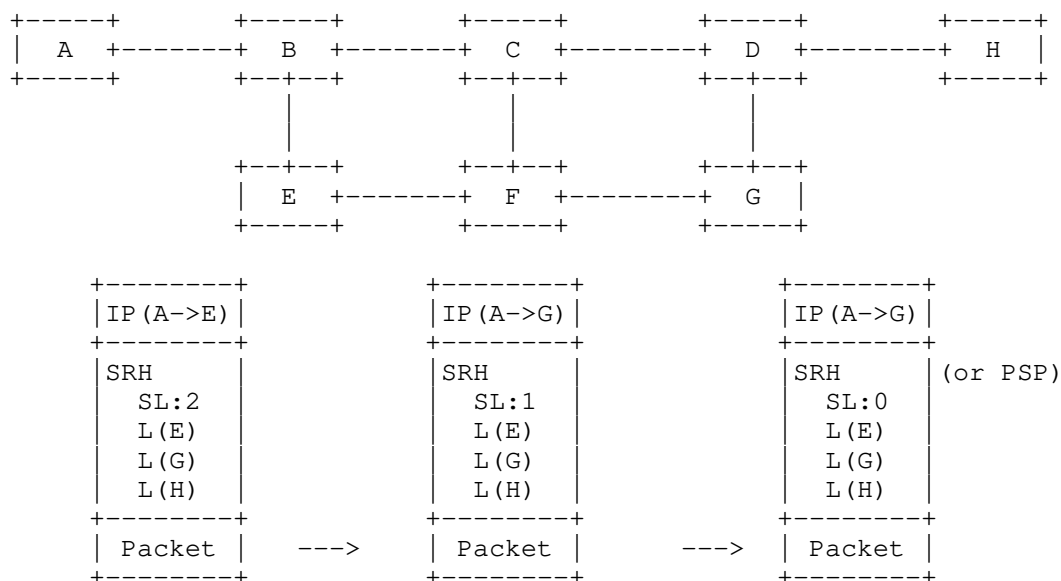


Figure 3: Packet Forwarding Example with UET-MPLS U-SID

In the example shown in Figure 3, assume that routers A, E, G, and H are U-SID capable (i.e., both SR-MPLS and SRv6 capable) while the remaining routers (B, C, D, and F) are only capable of forwarding IP packets. Routers A, E, G, and H advertise their Segment Routing related information via IS-IS or OSPF.

Now assume that router A (the Domain ingress) wants to send a packet to router H (the Domain egress) via an SRv6 policy with the explicit path {E->G->H}. Router A will impose an MPLS label stack within SRH on the packet corresponding to the explicit path. Router A searches ILM entry by the top label (that indicated router E), get the FEC information and next-hop/interface forwarding information, a loopback IPv6 address of E, and then copies to DA and sends the packet. SRH.UET is set to UET-MPLS and the value of SRH.SL is 2.

When the IPv6 packet arrives at router E, router E picks the next segment (label) within SRH based on the SRH.SL value of 2, searches ILM entry by the next label, get the FEC information and next-hop/interface forwarding information, a loopback IPv6 address of G, and then copy to DA and sends the packet. SRH.UET is set to UET-MPLS, and the value of SRH.SL is 1.

When the IPv6 packet arrives at router G, router G gets the next segment (label) within SRH based on the SRH.SL value of 1, then looks up ILM entry by the next label, gets the FEC information and next-hop/interface forwarding information, a loopback IPv6 address of H, and then copies it to IP DA and transmits the packet. Because the value of SRH.SL is 0; the SRH can be removed if the behavior flavor codepoint of the above next segment (label) is set to PSP.

#### 4.2. Procedures of 32bits IP Address within SRH

This section describes how the UET-32 type of U-SID is used to encode a compressed SRH.

[RFC6554] specifies the Source Routing Header (to avoid confusion with Segment Routing Header, we call it SRH3 according to type 3) for use strictly between RPL (Routing Protocol for Low-Power and Lossy Networks) routers in the same RPL routing domain. It introduces mechanisms to compact the source route entries when all entries share the same prefix with the IPv6 Destination Address of a packet carrying an SRH. For each entry in Address[1..n] within the Routing header, the shared prefix octets are not carried, but only a shorter truncated piece of the original 128bits. During packet forwarding, the shorter entry gets one by one and restored to the original IPv6 address. The Segment Left field represents the number of segments remaining, i.e., the number of explicitly listed intermediate nodes still to be visited before reaching the final destination, not the number of 128bits entries.

The described above mechanism, introduced in SRH3, could also be brought to Segment Routing Header (SRH). However, unlike in SRH3, using explicit fields within the Routing header to indicate the number of prefix octets common with the IPv6 Destination Address, this document introduces a new Flavor for Endpoint Behavior, defined in [RFC8986], termed as UET Flavor, for SRv6 SIDs. The UET Flavor of the current active SID indicates the next SID's compressed length within SRH, thus preparing the next SID of the corresponding length.

The UET Flavor information of a SID can be stored in the local SID entry of that SID.

This section defines the following two UET Flavors for Endpoint Behavior:

**UET-32 Flavor:** a SID with UET-32 Flavor means in SRH that the next SID is a 32bits IPv4 address or number.

**UET-16 Flavor:** a SID with UET-16 Flavor means in SRH that the next SID is a 16bits address or number.

For the convenience of expression, we can use UET-128 Flavor for the case when the next SID is a traditional 128bits IPv6 address. Note that UET-128 Flavor is not defined in the document.

An SRv6 SID MUST NOT have multiple UET Flavors at the same time.

#### 4.2.1. Packet Forwarding Based on UET-32 U-SID

This section describes the packet forwarding based on UET-32 U-SID. For UET-16 U-SID, it is similar.

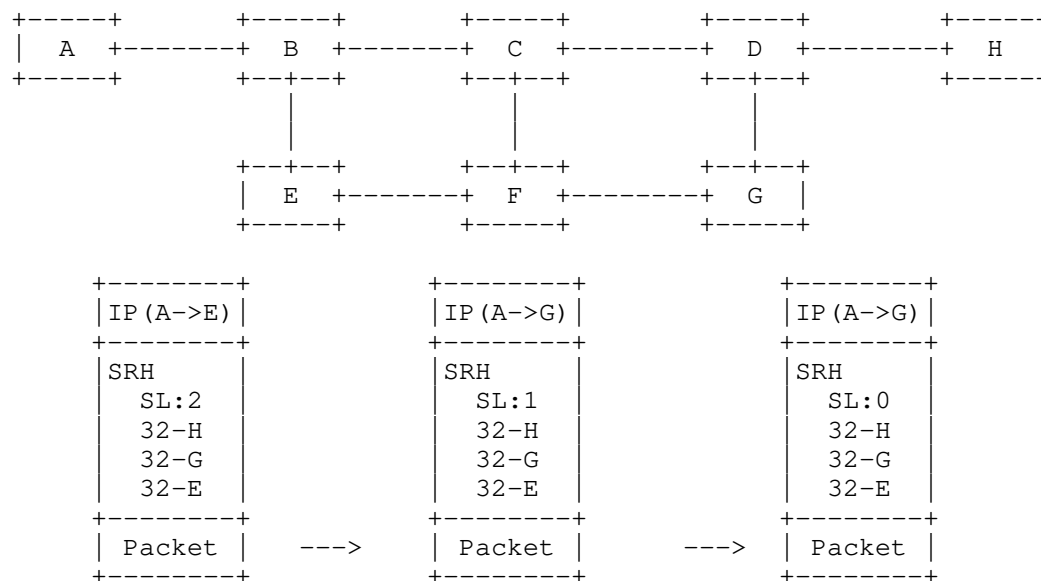


Figure 4: Packet Forwarding Example with UET-32 U-SID

In the example shown in Figure 4, assume that routers A, E, G, and H are U-SID capable while the remaining routers (B, C, D, and F) are only capable of forwarding IP packets. Routers A, E, G, and H

advertise their Segment Routing related information via IS-IS or OSPF, especially SRv6 SIDs with SID structure and UET-32 Flavor information.

Suppose that router A allocates an END SID B:32-A::, router E allocates an END SID B:32-E::, router G allocates an END SID B:32-G::, and router H allocates an END SID B:32-H::. All these SIDs have the same SID structure, i.e., share the same common prefix B (also known as the SRv6 SID Locator Block), and the sum of the Node Length, Function Length, Argument Length of each SID are the same.

Now assume that router A (the Domain ingress) wants to send a packet to router H (the Domain egress) via an SRv6 policy with the explicit path {E->G->H}. Router A will impose a UET-32 U-SID stack within SRH on the packet that corresponds to that explicit path. The U-SID stack consists of three shorter 32bits UET-32 U-SIDs, which are 32-E, 32-G, 32-H. Router A gets the first U-SID 32-E from SRH and restores it to the original IPv6 address B:32-E::, then copy it to DA and sends the packet according to IPv6 FIB lookup. SRH.UET is initially set to UET-32 and the value of SRH.SL is 2.

When the IPv6 packet arrives at router E, match the local SID entry of B:32-E::. Router E get the next U-32 32-G within SRH based on the SRH.SL value of 2, and restore it to the original IPv6 address B:32-G::, then copy it to DA and sends the packet according to IPv6 FIB lookup. SRH.UET remains unchanged, and the value of SRH.SL is 1.

When the IPv6 packet arrives at router G, match the local SID entry of B:32-G::. Router G gets the next U-32 32-H within SRH based on the SRH.SL value of 1, and restore it to the original IPv6 address B:32-H::, then copy it to DA and sends the packet according to IPv6 FIB lookup. SRH.UET remains unchanged, and the value of SRH.SL is 0. The SRH can be removed if the local SID entry of B:32-G:: has PSP Flavor.

When the IPv6 packet arrives at router H, match the local SID entry of B:32-H:: and Proceed to process the next header in the packet.

## 5. The Use Case of Unified Segment Identifier

In addition to being used for compression, U-SID can also be used in interworking between SR-MPLS and SRv6 domains. SR-MPLS is often used in a metro network, for example, in the backhaul metro network of CMCC. If the core network uses SRv6, for example, the core network of the same operator, U-SID can be used in the SRv6 domain to interwork with SR-MPLS in the metro network to form an end-to-end SR policy or tunnel.

### 5.1. Nesting Interworking Between SR-MPLS and SRv6 Using Binding U-SID

SR-MPLS uses SR SIDs as MPLS labels in the MPLS stack, and the SIDs are 32-bits long. SRv6 uses SR SIDs as IPv6 extension header in SRH, and the SIDs are 128-bits long.

The type UET-MPLS of U-SID uses the same 32-bits long SIDs in the MPLS stack and SRH. Thus, four 32-bits long U-SIDs can be placed in the space of a single 128-bits long header. The encapsulation is illustrated in Figure 5.

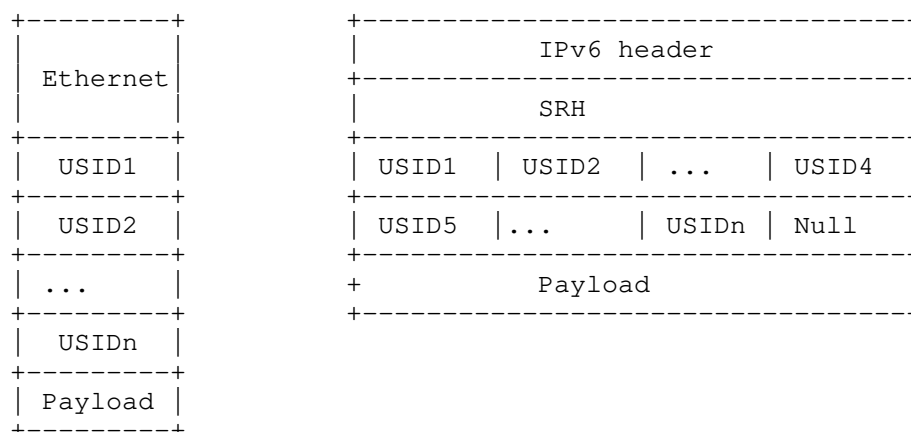


Figure 5: 32-bits long U-SIDs Encapsulation

This document RECOMMENDS using Binding SID for interworking because Binding SID allows hiding the difference between U-SID types of different domains. Additionally, a headend with only classical SRv6 SRH encapsulation capability, i.e., no capability to put multiple short U-SIDs to a single 128bits entry, will not need to upgrade.

Although Binding SID that is allocated for the specific SR policy instance will bring more states on some domain border nodes, the SR policy instance itself maybe pre-exist due to other requirements. The SR policy is created within each UET domain that can be upgraded separately.

To interwork, an MPLS Binding SID could be allocated for an SRv6 policy, used to hide the details of the UET-128 domain (classical SRv6) for a traditional MPLS Label stack. Similarly, an SRv6 Binding SID could be allocated for an SR-MPLS policy, used to hide the UET-MPLS domain's details for a conventional SRv6 SRH. An SRv6 Binding SID allocated for an SRv6 policy that enables the UET-32 compression

style will hide the details of the UET-32 domain for a traditional SRv6 SRH. There may be other combinations that are not discussed in the document.

Note that in some cases, Binding SID will cause multiple SRH to be inserted in the IPv6 header.

The SR-MPLS and SRv6 interworking is illustrated in Figure 6. An end-to-end SR path from A to F crosses the SR-MPLS and SRv6 domains. The SR-MPLS domain could be using the IPv4 or IPv6 address family. The SRv6 border nodes (E/G) receive SR-MPLS packets and forward them into the SRv6 domain using an SR-MPLS Binding SID [RFC8660].

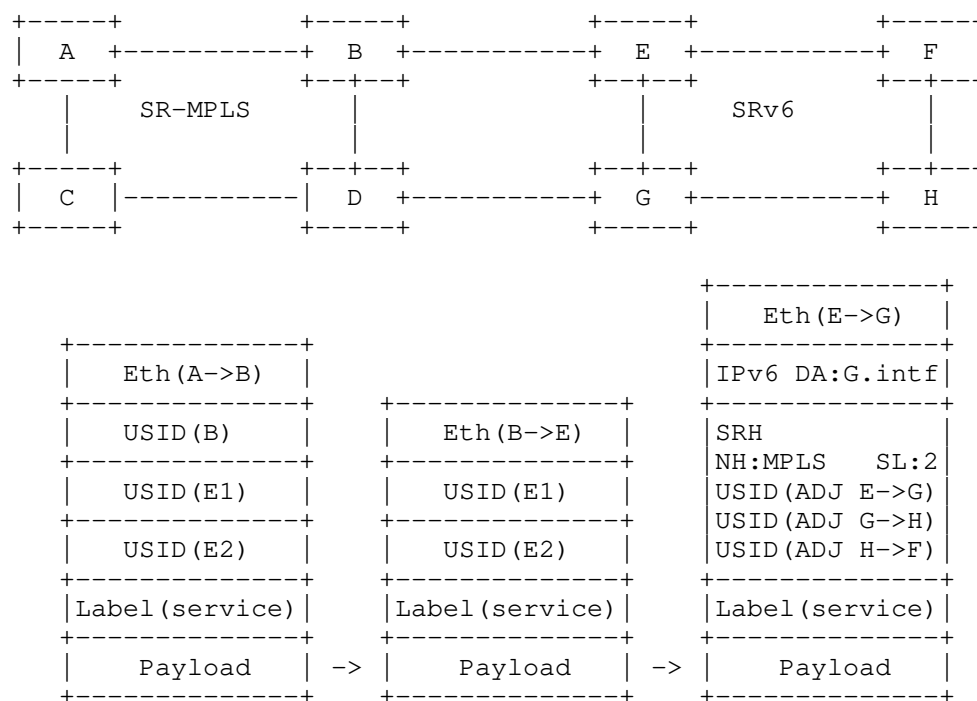


Figure 6: SR-MPLS and SRv6 interworking

The SRv6 edge node E assigns two SIDs, e.g., E1 and E2, E1 is an SR-MPLS Node-SID, E2 is an SR-MPLS Binding-SID, which represents an SRv6 policy (from E to F, via segment list E-G-H-F) with U-SID encapsulation. At the headend A, the end-to-end segment list could be B-E1-E2. Figure 3 demonstrates an example of the packet forwarding, where U-SID is an MPLS label.

The reverse interworking is illustrated in Figure 7. An end-to-end SR path from F to A crosses the SRv6 and SR-MPLS domains. The SRv6 border nodes (E/G) receive SRv6 packets and forward them into the SR-MPLS domain using an SR-MPLS Binding SID or normal Prefix/Adjacency SID.

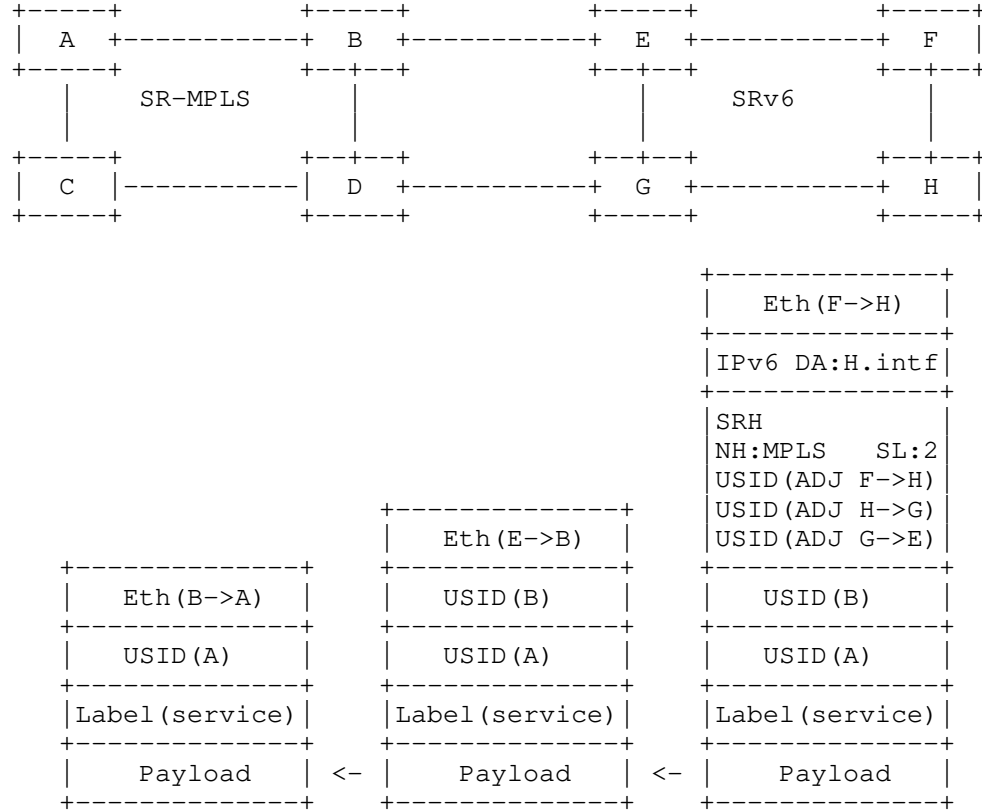


Figure 7: SR-MPLS and SRv6 reverse interworking

The SRv6 edge node F assigns an SR-MPLS Binding-SID F2, which represents an SRv6 policy (from F to E, via segment list F-H-G-E) with U-SID encapsulation. At the headend F, the end-to-end segment list could be F2-B-A.



## 5.2. Flat Interworking Between Different UET Domains Using Mixing U-SID

U-SRH can provide a different interworking scheme to support an end-to-end SR tunnel or policy using a mixing type of U-SIDs if more headend nodes have been upgraded to support encapsulating mixing U-SID in SRH. For example, a SID list could contain some 128bits classical SIDs, some 32bits U-SIDs (IP or Label), and some 16bits U-SIDs at the same time. For this purpose, each U-SID in SRH must meet the alignment requirement. For example, a UET-32 U-SID is stored in a 4-byte alignment, and a UET-16 U-SID is stored in a 2-byte alignment.

The interworking of different UET domains is illustrated in Figure 8. An end-to-end SR tunnel or policy from S to D with segment list <X, ABR1, Y, ABR2, Z, D>, crosses the UET-128 domain, UET-32 domain and UET-MPLS domain. Note that any order of UET domains is also possible and is similar to the case displayed in Figure 8.

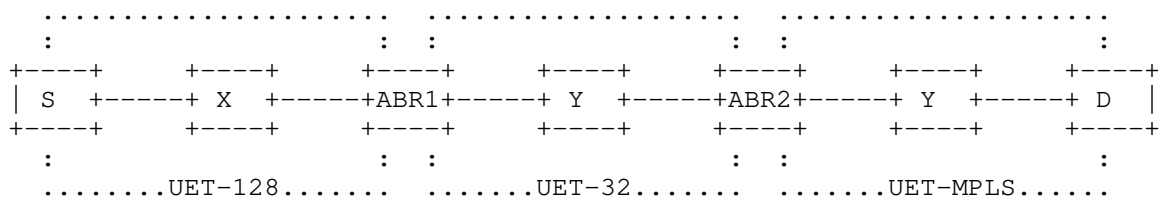


Figure 8: Interworking between different UET SID

### 5.2.1. UET Capability Advertisement

In an SRv6 network, each node can configure its U-SID Encapsulation Capability (UEC), and advertise it to other nodes. A controller can collect UEC information of all nodes. Typical UEC is:

UEC-128: Support classical 128bits SRv6 SID, which is the default capability of an SRv6 node.

UEC-32: Support shorter 32bits IPv4 address or number.

UEC-MPLS: Support shorter 32bits MPLS label.

UEC-16: Support shorter 16bits number.

Each node can support one or more UECs. Refer to Figure 8, node S/X/ABR1 can configure to support UEC-128 capability, node ABR1/Y/ABR2 can configure to support UEC-32 capability, and node ABR2/Y/D can configure to support UEC-MPLS capability.

A UET domain is constructed by several connected SRv6 nodes with the same UEC. For example, a UET-128 domain is constructed by the connected nodes all with UEC-128.

#### 5.2.2. SRv6 SID Allocated per UEC

An SRv6 SID is allocated per UEC. For example, an SRv6 Node can allocate different END SIDs each for UEC-128, UEC-32, UEC-MPLS, etc.

The local SID entry of each SRv6 SID allocated per UEC will explicitly have the specific UET Flavor attribute information.

In addition to the two UET Flavors, i.e., UET-32 and UET-16 Flavors that is defined in Section 4.2, below is described a third UET Flavor for SRv6 SID:

UET-MPLS Flavor: a SID with UET-MPLS Flavor means in SRH the next SID is a 32bits MPLS label.

Each node allocates its SRv6 SID per UEC and advertises it to other nodes with additional UET-Flavor. A controller can collect these SIDs to be used for E2E SID List programming.

To save label resources, an MPLS label is not allocated per UEC. Relevant UET-Flavor information can be directly inserted in the context field of the label item in SRH. However, to meet the SRH processing restrictions defined in [RFC8754], it is possible to allocate MPLS labels for some of the topology-related SRv6 SIDs, which will consume more label resources.

Refer to the scenario presented in Figure 8, where each node may allocate the following SRv6 SID per UEC.

Node S: 128bits-END-SID-S for UEC-128.

Node X: 128bits-END-SID-X for UEC-128.

Node ABR1: 128bits-END-SID-ABR1 for UEC-128, and 128bits-END-SID-ABR1' for UEC-32.

Node Y: 128bits-END-SID-Y for UEC-32.

Node ABR2: 128bits-END-SID-ABR2 for UEC-32, and 128bits-END-SID-ABR2' for UEC-MPLS.

Node Z: 32bits-PREFIX-SID-Z. Note that MPLS Label allocation is independent with UEC.

Node D: 32bits-PREFIX-SID-D. Note that MPLS Label allocation is independent with UEC.

Note that the above SRv6 SID itself is always a 128bits IPv6 address, with no relationship with its UET Flavor attribute. The UET Flavor attribute indicates the next SID type, i.e., 128bits classical SID, 32bits IPv4 address, or 32bits MPLS Label, etc.

### 5.2.3. Packets Forwarding Procedures

Consider that the controller computes an E2E segment list <X, ABR1, Y, ABR2, Z, D>.

For the above E2E segment list, the controller knows which UET domain does each segment node belongs to, especially that ABR1 and ABR2 are the border nodes between different UET domains. Controller will select appropriate SID with specific UET Flavor attribute to indicate the UET domain which the next SID belongs to, i.e., whether the next SID is a classical IPv6 address or a shorter truncated value.

The SID list informed to headend could be:

FSU: First SID UET, which indicates the compression result of the first SID, in this example, is set to UET-128.

No.1 SID: 128bits-END-SID-X (with BL|TL info of itself, and UET-128 Flavor to indicate the compression result of the next SID)

No.2 SID: 128bits-END-SID-ABR1' (with BL|TL info of itself, and UET-32 Flavor to indicate the compression result of the next SID)

No.3 SID: 128bits-END-SID-Y (with BL|TL info of itself, and UET-32 Flavor to indicate the compression result of the next SID)

No.4 SID: 128bits-END-SID-ABR2' (with BL|TL info of itself, and UET-MPLS Flavor to indicate the compression result of the next SID)

No.5 SID: 32bits-PREFIX-SID-Z, (with UET-MPLS Flavor to indicate the compression result of the next SID)

No.6 SID: 32bits-PREFIX-SID-D, (with UET-128 Flavor to indicate the compression result of the next SID)

Note:

FSU indicates the compression result of the first SRv6 SID itself, while the UET Flavor attribute of the first SID just indicates the compression result of the second SRv6 SID.

BL is the Block Length of SRv6 SID. TL is the Truncated Length of SRv6 SID, i.e., the compression result.

The headend analysis of how to get the compressed SID List:

FSU is UET-128, so the first SID 128bits-END-SID-X uses 128 bits.

The No.1 SID, 128bits-END-SID-X, has UET-128 Flavor, which means the next SID, 128bits-END-SID-ABR1', also needs to use 128 bits.

The No.2 SID, 128bits-END-SID-ABR1' has UET-32 Flavor, which means the next SID, 128bits-END-SID-Y, needs to be compressed as 32 bits IPv4 address.

The No.3 SID, 128bits-END-SID-Y, has UET-32 Flavor, which means the next SID, 128bits-END-SID-ABR2', needs to be compressed as 32 bits IPv4 address.

The No.4 SID, 128bits-END-SID-ABR2', has UET-MPLS Flavor, which means the next SID, 32bits-PREFIX-SID-Z, needs to use 32 bits.

The No.5 SID, 32bits-PREFIX-SID-Z, has UET-MPLS Flavor, which means the next SID, 32bits-PREFIX-SID-D, needs to use 32 bits.

The No.6 SID, 32bits-PREFIX-SID-D, has UET-128 Flavor, which means the next SID, maybe a VPN service SRv6 SID, needs to use 128 bits.

Note that in some cases, an overlay VPN service SRv6 SID could be compressed. At that time, the last SID within the underlay segment list may select the UET-32 or UET-16 Flavor attribute.

Thus, the headend can get the following compressed SID List:

128 bits-END-SID-X with UET-128 Flavor

128 bits-END-SID-ABR1' with UET-32 Flavor

32 bits of 128 bits-END-SID-Y with UET-32 Flavor

32 bits of 128 bits-END-SID-ABR2' with UET-MPLS Flavor

32 bits-PREFIX-SID-Z (with UET-MPLS Flavor in context.field)

32 bits-PREFIX-SID-D (with UET-128 Flavor context.field)

At the However, to meet the SRH processing restrictions defined in [RFC8754], it is possible to allocate MPLS labels for some of the topology-related SRv6 SIDs, which will consume more label resources. At the headend, the encapsulated SRH could be:

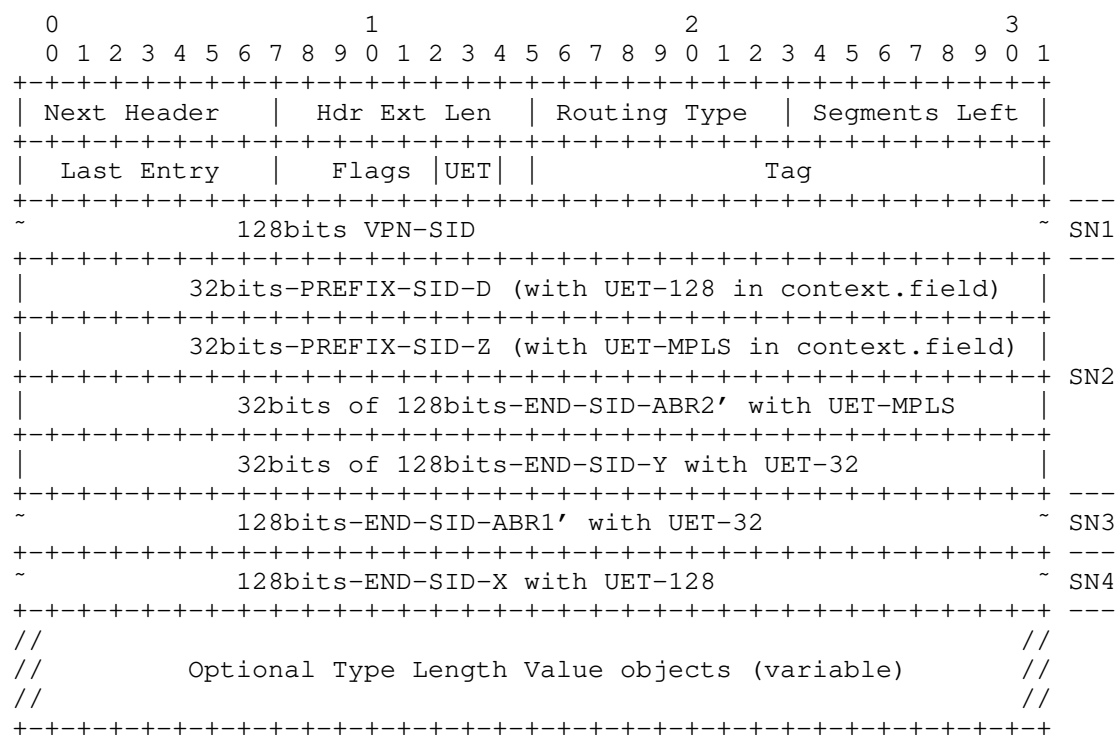


Figure 9: SRH including different UET SID

The initial SRH.SL is set to 4: the number of 128bits based SIDs in SRH, and the initial SRH.UET is set to UET-128, according to FSU, which represents the first UET domain.

During the process of packets passing through multiple UET domains, if SRH.UET change from UET-128 to UET-32 or UET-MPLS, SRH.SL will quadruple, i.e.,  $SRH.SL = SRH.SL * 4$ , which is the number of 32bits based SIDs in SRH. When SRH.UET changed from UET-32 or UET-MPLS to UET-128, SRH.SL will revert to its original size, i.e.,  $SRH.SL = SRH.SL / 4$ , which is the number of 128bits based SIDs in SRH.

Similarly, when SRH.UET changes from UET-128 to UET-16,  $SRH.SL = SRH.SL * 8$ , from UET-32 to UET-16,  $SRH.SL = SRH.SL * 2$ , vice versa.

Refer to Figure 8, next we will describe the process of packets passing through each UET domain.

Before transmitting packets to segment node 1 (SN1) X, the headend S decrements SRH.SL by one and gets the first 128bits SID from SRH.List[], 128bits-END-SID-X with UET-128. Then copies to DA, and then the lookups up FIB to where send the packet. Now, the SRH.SL value is 3 and SRH.UET is UET-128.

At the SN1 X, the local SID matches the DA and has the UET-128 Flavor attribute. Hence, SRH.UET has not changed. It decrements SRH.SL by one, gets the next 128bits SID from SRH.List[], 128bits-END-SID-ABR1' with UET-32, copies the value to DA, and then looks up FIB to where to send the packet. At this time, SRH.SL is 2 and SRH.UET is UET-128.

At the SN2 ABR1, the local SID matches the DA has UET-32 Flavor attribute. Hence, SRH.UET has changed from UET-128 to UET-32. The node will firstly calculate  $\text{SRH.SL} * 4$ , then decrement SRH.SL by 1, get the next 32bits SID from SRH.List[], 32bits of 128bits-END-SID-Y with UET-32, convert it to a complete IPv6 SID, copy to DA, and lookup FIB to send packets. At this time, SRH.SL is 7 and SRH.UET is UET-32.

At the SN3 Y, the local SID matches the DA has UET-32 Flavor attribute. Thus, SRH.UET has not changed. The node will decrement SRH.SL by 1, get the next 32bits SID from SRH.List[], 32bits of 128bits-END-SID-ABR2' with UET-MPLS, convert it to a complete IPv6 SID, copy to DA, and lookup FIB to send packets. At this time, SRH.SL is 6, SRH.UET is UET-32.

At the SN4 ABR2, the local SID matches the DA has UET-MPLS Flavor attribute. Hence, SRH.UET has changed from UET-32 to UET-MPLS. Because the size of SID has not changed, the node will decrement SRH.SL by 1, get the next 32bits SID from SRH.List[], 32bits-PREFIX-SID-Z (with UET-MPLS in context.field), map it to a complete IPv6 prefix FEC by ILM entry, copy to DA, and lookup FIB (or directly get forwarding information from ILM entry) to send packets. Note that the UET information in the context.field needs to be compared with the value in SRH.UET. Since values are equal no change and no additional processing. At this time, SRH.SL is 5, SRH.UET is 0b02.

At the SN5 Z, the address route entry matches the DA and has no UET Flavor attribute. As a result, SRH.UET has not changed. The node decrements SRH.SL by 1, will get the next 32bits SID from SRH.List[], 32bits-PREFIX-SID-D (with UET-128 in context.field), map it to a complete IPv6 prefix FEC by ILM entry, copy to DA, and lookup FIB (or directly get forwarding information from ILM entry) to send packets.

Note that the UET information in context.field needs to be compared to SRH.UET. Because it is changed from UET-MPLS to UET-128, the SRH.SL will be reverted to its original size, i.e., let  $SRH.SL / 4$ . At this time, SRH.SL is 1, SRH.UET is UET-128.

At the SN6 D, the address route entry matched by DA has no associated UET Flavor attribute. Hence, SRH.UET has not changed. The node decrements SRH.SL by 1, will get the next 128bits SID from SRH.List[], 128bits VPN-SID, and follow the rest process described in [RFC8986].

## 6. Control Plane in Support of Unified SID

The introduction of the Unified Identifier may rely on the existing SR extensions to the routing protocols. But some enhancements in the control plane are still required. This section analyzes control plane protocols and identifies necessary extensions.

Each node in the SRv6 domain needs to advertise its U-SID Encapsulation Capability, this information can be carried within SRv6-Capabilities sub-TLV defined in [I-D.ietf-lsr-isis-srv6-extensions] and SRv6 Capabilities TLV defined in [I-D.ietf-lsr-ospfv3-srv6-extensions]. It need also allocate SRv6 SID (Topology type and Service Function type) per UEC and advertise to other nodes, the advertisement of SRv6 END SID, END.X SID, LAN END.X SID defined in [I-D.ietf-lsr-isis-srv6-extensions] and [I-D.ietf-lsr-ospfv3-srv6-extensions] need to be extended to carry UET-Flavor information. This information can be collected and sent to the central controller through BGP-LS. The controller then can send the computed segment list to the headend through BGP or PCEP, and each segment will include explicit UET Flavor information. The detailed procedures are outside the scope of this document.

The SR-MPLS extensions to Interior Gateway Protocols (IGP), IS-IS [RFC8667], OSPF [RFC8665], and OSPFv3 [RFC8666], defined how 20-bits and 32-bits SIDs advertised and bound to SR objects and/or instructions. Extensions to BGP Link-state address family [RFC9085] enabled propagation of segment information of variable length via BGP. Already defined SR-MPLS extensions can be used to get MPLS U-SID mapping FIB entry, and it can coexist with SRv6 extensions to the same IGP/BGP-LS instance. For simplicity, this document suggests using the existing mature SR-MPLS control plane and FIB entry for the MPLS U-SID advertisement and mapping entry. However, it is possible to base it on SRv6 related TLVs/sub-TLVs to advertise the MPLS U-SID. It is outside the scope of this specification and will be discussed in another document.

## 7. SRH with U-SID Pseudo-code

Processing of SRH with U-SID is demonstrated in the following pseudo-code:

Headend sending packet:

```
S01. set initial SRH.UET, respond to the FSU, i.e.,
    the compressed result of the first SID;
S02. set initial SRH.SL, it is the count of 128bits-based SIDs;
S03. if (SRH.UET == UET-128) {
S04.   SRH.SL --;
S05.   Get SRH.List[SRH.SL], 128bits, copy to IPv6 Header DA; Or,
    headend know the first SID before SRH encapsulation,
    just copy it to DA.
S06.   FIB lookup according to DA, and forward packet;
S07. }
S08. else if (SRH.UET == UET-32) {
S09.   SRH.SL = SRH.SL * 4;
S10.   SRH.SL --;
S11.   Get SRH.List[SRH.SL], 32bits, convert to 128bits SRv6 SID, copy
    to IPv6 Header DA; Or, headend know the first SID before SRH
    encapsulation, just copy it to DA;
S12.   FIB lookup according to DA, and forward packet;
S13. }
S14. else if (SRH.UET == UET-MPLS) {
S15.   SRH.SL = SRH.SL * 4;
S16.   SRH.SL --;
S17.   Get SRH.List[SRH.SL], 32bits, lookup ILM entry and map it to 128
    IPv6 address, copy it to IPv6 Header DA; Or, headend know
    the first SID before SRH encapsulation, just copy it to DA;
S18.   FIB lookup according to DA, or directly get forwarding
    information from ILM entry and forward packet;
S19. }
S20. else if (SRH.UET == UET-16) {
S21.   SRH.SL = SRH.SL * 8;
S22.   SRH.SL --;
S23.   Get SRH.List[SRH.SL], 16bits, convert to 128bits SRv6 SID, copy
    to IPv6 Header DA; Or, headend know the first SID before SRH
    encapsulation, just copy it to DA;
S24.   FIB lookup based on DA and forward packet;
S25. }
```

Transit/Egress receive packets:



```
S01. If DA matched local SID entry, copy the UET attr of local SID entry
    to SRH.UET, check when SRH.UET changed from UET-128 to UET-32 or
        UET-MPLS, SRH.SL*4, when from UET-32 or UET-MPLS to UET-128,
        SRH.SL / 4, similar treatment of UET-16 related SRH.SL update;
    Else If DA matched normal address route entry,
        SRH.UET no update;
S02. if (SRH.SL == 0) {
S03.     process the inner payload;
S04. }
S05. else {
S06.     if (SRH.UET == UET-128) {
S07.         SRH.SL -- ;
S08.         Get SRH.List[SRH.SL], 128bits, copy it to IPv6 Header DA;
S09.         FIB lookup based on DA and forward packet;
S10.     }
S11.     else if (SRH.UET == UET-32) {
S12.         SRH.SL -- ;
S13.         Get SRH.List[SRH.SL], 32bits, convert to 128bits SRv6 SID, copy
            to IPv6 Header DA;
S14.         FIB lookup based on DA and forward packet;
S15.     }
S16.     else if (SRH.UET == UET-MPLS) {
S17.         SRH.SL --
S18.         Get SRH.List[SRH.SL], 32bits, lookup ILM entry, map it to
            128bits IPv6 address, copy it to IPv6 Header DA;
S19.         Get UET info from SRH.List[SRH.SL] Context Field, copy it to
            SRH.UET. Check if SRH.UET changed from UET-MPLS to UET-128,
            SRH.SL/4;
S20.         FIB lookup according to DA, or, directly get forwarding
            information from ILM entry, and forward packet;
S21.     }
S22.     else if (SRH.UET == UET-16) {
S23.         SRH.SL -- ;
S24.         Get SRH.List[SRH.SL], 16bits, convert to 128bits SRv6 SID, copy
            to IPv6 Header DA;
S25.         FIB lookup based on DA and forward packet;
S26.     }
S27. }
```

#### 8. U-SID supporting SRv6 programming

U-SID can support SRv6 programming defined by [RFC8986]. The details will be described in another document.

## 9. Benefits

To be discussed in the next version.

## 10. Implementation Considerations

The Unified SID solution has been already implemented and tested by two companies:

- \* Centec has conducted its PoC, and the report is available at <https://cloud.tencent.com/developer/article/1540023>.
- \* Broadcom, in its lab, also conducted PoC testing of the U-SID solution.

## 11. IANA Considerations

IANA is requested to allocate the two-bits long field from the Segment Routing Header Flags registry referred to as Size.

## 12. Security Considerations

This specification inherits all security considerations of [RFC8402] and [RFC8754].

## 13. Contributors

Wan Xiaolan  
New H3C Technologies Co. Ltd  
No.8, Yongjia Road, Haidian District  
Beijing  
China

Email: wxlan@h3c.com

Cheng Wei  
Centec  
Building B, No.5 Xing Han Street, Suzhou Industrial Park  
Suzhou  
China

Email: Chengw@centecnetworks.com

S.Zadok  
Broadcom  
Israel

Email: shay.zadok@broadcom.com

#### 14. Acknowledgements

TBD

#### 15. Normative References

[I-D.ietf-lsr-isis-srv6-extensions]

Psenak, P., Filsfils, C., Bashandy, A., Decraene, B., and Z. Hu, "IS-IS Extensions to Support Segment Routing over IPv6 Dataplane", Work in Progress, Internet-Draft, draft-ietf-lsr-isis-srv6-extensions-17, 18 June 2021, <<https://datatracker.ietf.org/doc/html/draft-ietf-lsr-isis-srv6-extensions-17>>.

[I-D.ietf-lsr-ospfv3-srv6-extensions]

Li, Z., Hu, Z., Cheng, D., Talaulikar, K., and P. Psenak, "OSPFv3 Extensions for SRv6", Work in Progress, Internet-Draft, draft-ietf-lsr-ospfv3-srv6-extensions-02, 15 February 2021, <<https://datatracker.ietf.org/doc/html/draft-ietf-lsr-ospfv3-srv6-extensions-02>>.

- [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>>.
- [RFC6554] Hui, J., Vasseur, JP., Culler, D., and V. Manral, "An IPv6 Routing Header for Source Routes with the Routing Protocol for Low-Power and Lossy Networks (RPL)", RFC 6554, DOI 10.17487/RFC6554, March 2012, <<https://www.rfc-editor.org/info/rfc6554>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <<https://www.rfc-editor.org/info/rfc8200>>.
- [RFC8402] Filsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402, July 2018, <<https://www.rfc-editor.org/info/rfc8402>>.
- [RFC8660] Bashandy, A., Ed., Filsfils, C., Ed., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing with the MPLS Data Plane", RFC 8660, DOI 10.17487/RFC8660, December 2019, <<https://www.rfc-editor.org/info/rfc8660>>.
- [RFC8663] Xu, X., Bryant, S., Farrel, A., Hassan, S., Henderickx, W., and Z. Li, "MPLS Segment Routing over IP", RFC 8663, DOI 10.17487/RFC8663, December 2019, <<https://www.rfc-editor.org/info/rfc8663>>.
- [RFC8665] Psenak, P., Ed., Previdi, S., Ed., Filsfils, C., Gredler, H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPF Extensions for Segment Routing", RFC 8665, DOI 10.17487/RFC8665, December 2019, <<https://www.rfc-editor.org/info/rfc8665>>.
- [RFC8666] Psenak, P., Ed. and S. Previdi, Ed., "OSPFv3 Extensions for Segment Routing", RFC 8666, DOI 10.17487/RFC8666, December 2019, <<https://www.rfc-editor.org/info/rfc8666>>.

- [RFC8667] Previdi, S., Ed., Ginsberg, L., Ed., Filsfils, C., Bashandy, A., Gredler, H., and B. Decraene, "IS-IS Extensions for Segment Routing", RFC 8667, DOI 10.17487/RFC8667, December 2019, <<https://www.rfc-editor.org/info/rfc8667>>.
- [RFC8754] Filsfils, C., Ed., Dukes, D., Ed., Previdi, S., Leddy, J., Matsushima, S., and D. Voyer, "IPv6 Segment Routing Header (SRH)", RFC 8754, DOI 10.17487/RFC8754, March 2020, <<https://www.rfc-editor.org/info/rfc8754>>.
- [RFC8986] Filsfils, C., Ed., Camarillo, P., Ed., Leddy, J., Voyer, D., Matsushima, S., and Z. Li, "Segment Routing over IPv6 (SRv6) Network Programming", RFC 8986, DOI 10.17487/RFC8986, February 2021, <<https://www.rfc-editor.org/info/rfc8986>>.
- [RFC9085] Previdi, S., Talaulikar, K., Ed., Filsfils, C., Gredler, H., and M. Chen, "Border Gateway Protocol - Link State (BGP-LS) Extensions for Segment Routing", RFC 9085, DOI 10.17487/RFC9085, August 2021, <<https://www.rfc-editor.org/info/rfc9085>>.

## Authors' Addresses

Cheng Weiqiang  
China Mobile  
Beijing,  
China

Email: [chengweiqiang@chinamobile.com](mailto:chengweiqiang@chinamobile.com)

Greg Mirsky  
Ericsson

Email: [gregimirsky@gmail.com](mailto:gregimirsky@gmail.com)

Peng Shaofu  
ZTE Corporation  
No.50 Software Avenue, Yuhuatai District  
Nanjing,  
China

Email: [peng.shaofu@zte.com.cn](mailto:peng.shaofu@zte.com.cn)

Liu Aihua  
ZTE Corporation  
Zhongxing Industrial Park, Nanshan District  
Shenzhen,  
China

Email: liu.aihua@zte.com.cn

Gyan S. Mishra  
Verizon Inc.  
13101 Columbia Pike  
Silver Spring, MD 20904  
United States of America

Phone: 301 502-1347  
Email: gyan.s.mishra@verizon.com