

Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: May 01, 2018

G. Fairhurst
T. Jones
University of Aberdeen
M. Tuexen
I. Ruengeler
Muenster University of Applied Sciences
October 30, 2017

Packetization Layer Path MTU Discovery for Datagram Transports
draft-fairhurst-tsvwg-datagram-plpmtud-01.txt

Abstract

This document describes a robust method for Path MTU Discovery (PMTUD) for datagram packetization layers. It allows these layers to probe an Internet path with progressively larger packets to determine a maximum packet size. This method is described as an extension to RFC 1191 and RFC 8201, which specify ICMP-based Path MTU Discovery for IP versions 4 and 6. The document provides functionality for datagram transports that is equivalent to the packetization layer PMTUD specification for TCP, specified in RFC4821.

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 May 01, 2018.

Copyright Notice

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

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text

as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Terminology	4
3. Features required to provide PLPMTUD at the Transport Layer	6
3.1. PMTU Probe Packets	8
3.2. Validation of the current effective PMTU	9
3.3. Reduction of the effective PMTU	9
4. Datagram PLPMTUD	9
4.1. Probing	10
4.2. Timers	10
4.3. Constants	11
4.4. Variables	11
4.5. State Machine	11
5. Specification of Protocol-Specific Methods	13
5.1. UDP and UDP-Lite	13
5.1.1. UDP Options	14
5.1.2. UDP Options required for PLPMTUD	14
5.1.2.1. Echo Request Option	14
5.1.2.2. Echo Response Option	14
5.1.3. Sending UDP-Option Probe Packets	14
5.1.4. Validating the Path with UDP Options	15
5.1.5. Handling of PTB Messages by UDP	15
5.2. SCTP	15
5.2.1. SCTP/IP4 and SCTP/IPv6	15
5.2.1.1. Sending SCTP Probe Packets	15
5.2.1.2. Validating the Path with SCTP	16
5.2.1.3. PTB Message Handling by SCTP	16
5.2.2. SCTP/UDP	16
5.2.2.1. Sending SCTP/UDP Probe Packets	16
5.2.2.2. Validating the Path with SCTP/UDP	16
5.2.2.3. Handling of PTB Messages by SCTP/UDP	16
5.2.3. SCTP/DTLS	16
5.2.3.1. Sending SCTP/DTLS Probe Packets	17
5.2.3.2. Validating the Path with SCTP/DTLS	17
5.2.3.3. Handling of PTB Messages by SCTP/DTLS	17
5.3. Other IETF Transports	17
6. Acknowledgements	17
7. IANA Considerations	17
8. Security Considerations	17
9. References	17
9.1. Normative References	17
9.2. Informative References	19
Appendix A. Event-driven state changes	19
Appendix B. Revision Notes	22
Authors' Addresses	23

1. Introduction

The IETF has specified datagram transport using UDP, SCTP, SCTP/UDP, DCCP, and DCCP/UDP, as well as protocols layered on top of these transports.

Classical Path Maximum Transmission Unit Discovery (PMTUD) can be used with any transport that is able to process ICMP Packet Too Big (PTB) messages (e.g., [RFC1191] and [RFC8201]). It adjusts the effective Path MTU (PMTU), based on reception of ICMP Path too Big (PTB) messages to decrease the PMTU when a packet is sent with a size larger than the value supported along a path, and a method that from time-to-time increases the packet size in attempt to discover an increase in the supported PMTU.

However, Classical PMTUD is subject to protocol failures. One failure arises when traffic using a packet size larger than the actual supported PMTU is blackholed (silently discarded). This may happen when ICMP PTB messages are not delivered back to the sender for some reason [RFC2923]). For example, ICMP messages are increasingly filtered by middleboxes (including Firewalls) [RFC4890], and may not be correctly processed by tunnel endpoints.

Another failure could result if a system not on the path sends a PTB that attempts to force the sender to change the effective PMTU [RFC8201]. A sender could protect itself by using the quoted packet within the PTB message payload to verify that the received PTB message was generated in response to a packet that had actually been sent. However, there are situations where a sender is unable to provide this verification (e.g., when the PTB message does not include sufficient information, often the case for IPv4; or where the information corresponds to an encrypted packet). At the network layer there also could be insufficient context to perform this verification, which depends on information about the active transport flows (e.g., the socket/address pairs being used, and other protocol header information). This verification is more straight forward at a the Packetization Layer (PL) or a higher layer.

The term Packetization Layer has been introduced to describe the layer that is responsible for placing data blocks into the payload of packets and selecting an appropriate maximum packet size. This function is often performed by a transport protocol, but can also be performed by other encapsulation methods working below the application.

In contrast to PMTUD, Packetization Layer Path MTU Discovery (PLPMTUD) [RFC4821] does not rely upon reception and verification of PTB messages. It is therefore more robust than Classical PMTUD. This has become the recommended approach for implementing PMTU discovery with TCP. It uses a general strategy where the PL searches for an appropriate PMTU by sending probe packets along the network path with a progressively larger packet size. If a probe packet is successfully delivered (as determined by the PL), then the effective Path MTU is raised to the probe size.

PLPMTUD introduces flexibility in the implementation of PMTU discovery. At one extreme, it can be configured to only perform PTB black hole recovery to increase the robustness of Classical PMTUD, or at the other extreme, all PTB processing can be disabled and PLPMTUD can completely replace Classical PMTUD. PLPMTUD can also include additional consistency checks without increasing the risk of blackholing.

The UDP-Guidelines [RFC8085] state "an application SHOULD either use the path MTU information provided by the IP layer or implement Path MTU Discovery (PMTUD)", but does not provide a mechanism for discovering the largest size of unfragmented datagram than can be used on a path. PLPMTUD has not currently been specified for UDP, while Section 10.2 of [RFC4821] recommends a PLPMTUD probing method for SCTP that utilises heartbeat messages as packet probes, but does not provide a complete specification. This document provides the details to complete that specification. Similarly, the method defined in this specification could be used with the Datagram Congestion Control Protocol (DCCP) [RFC4340] requires implementations to support Classical PMTUD and states that a DCCP sender "MUST maintain the maximum packet size (MPS) allowed for each active DCCP session". It also defines the current congestion control maximum packet size (CCMPS) supported by a path. This recommends use of PMTUD, and suggests use of control packets (DCCP-Sync) as path probe packets, because they do not risk application data loss. The document also contains information that enables the implementation of PLPMTUD with other datagram transports

Section 4 of this document presents a set of algorithms for datagram protocols to discover a maximum size for the effective PMTU across a path. The methods described rely on features of the PL Section 3 and apply to transport protocols over IPv4 and IPv6. It does not require cooperation from the lower layers (except that they are consistent about which packet sizes are acceptable). It can utilise PTB messages when these are available.

2. Terminology

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

Other terminology is directly copied from [RFC4821], and the definitions in [RFC1122].

Black-Holed: When the sender is unaware that packets are not delivered to the destination endpoint (e.g., when the sender is unaware of a change in the path to one with a smaller PMTU).

Classical Path MTU Discovery: Classical PMTUD is a process described in [RFC1191] and [RFC8201], in which nodes rely on PTB messages to

learn the largest size of unfragmented datagram than can be used across a path.

Datagram: A datagram is a transport-layer protocol data unit, transmitted in the payload of an IP packet.

Effective PMTU: The current estimated value for PMTU used by a Packetization Layer.

EMTU_S: The Effective MTU for sending (EMTU_S) is defined in [RFC1122] as "the maximum IP datagram size that may be sent, for a particular combination of IP source and destination addresses...".

EMTU_R: The Effective MTU for receiving (EMTU_R) is designated in [RFC1122] as "the largest datagram size that can be reassembled by EMTU_R ("Effective MTU to receive")".

Link: A communication facility or medium over which nodes can communicate at the link layer, i.e., a layer below the IP layer. Examples are Ethernet LANs and Internet (or higher) layer and tunnels.

Link MTU: The Maximum Transmission Unit (MTU) is the size in bytes of the largest IP packet, including the IP header and payload, that can be transmitted over a link. Note that this could more properly be called the IP MTU, to be consistent with how other standards organizations use the acronym MTU. This includes the IP header, but excludes link layer headers and other framing that is not part of IP or the IP payload. Other standards organizations generally define link MTU to include the link layer headers.

MPS: The Maximum Packet Size (MPS), the largest size of application data block that may be sent unfragmented across a path. In PLPMTUD this quantity is derived from Effective PMTU by taking into consideration the size of the application and lower protocol layer headers, and may be limited by the application protocol.

Packet: An IP header plus the IP payload.

Packetization Layer (PL): The layer of the network stack that places data into packets and performs transport protocol functions.

Path: The set of link and routers traversed by a packet between a source node and a destination node.

Path MTU (PMTU): The minimum of the link MTU of all the links forming a path between a source node and a destination node.

PLPMTUD: Packetization Layer Path MTU Discovery, the method described in this document for datagram PLs, which is an extension to Classical PMTU Discovery.

3. Features required to provide PLPMTUD at the Transport Layer

TCP PLPMTUD has been defined using standard TCP protocol mechanisms. All of the requirements in [RFC4821] also apply to use of the technique with a datagram PL. Unlike TCP, some datagram PLs require additional mechanisms to implement PLPMTUD.

There are ten requirements for performing the datagram PLPMTUD method described in this specification:

1. PMTU parameters: A PLPMTUD sender is REQUIRED to provide information about the maximum size of packet that can be transmitted by the sender on the local link (the Link MTU and MAY utilize similar information about the receiver when this is supplied (note this may be less than EMTU_R). Some applications also have a maximum transport protocol data unit (PDU) size, in which case there may be no benefit from probing for a size larger than this (unless a transport allows multiplexing multiple applications PDUs into the same datagram.)
2. Effective PMTU: A datagram application MUST be able to choose the size of datagrams sent to the network, up to the effective PMTU, or a smaller value (such as the MPS) derived from this. This value is managed by the PMTUD method. The effective PMTU (specified in Section 1 of [RFC1191]) is equivalent to the EMTU_S (specified in [RFC1122]).
3. Probe packets: On request, a PLPMTUD sender is REQUIRED to be able to transmit a packet larger than the current effective PMTU (but always with a total size less than the link MTU), which the method can use as a probe packet. In IPv4, a probe packet is always sent with the Don't Fragment (DF) bit set and without network layer endpoint fragmentation.
4. Processing PTB messages: A PLPMTUD sender MAY optionally utilize PTB messages received from the network layer to help identify when a path does not support the current size of packet probe. Any received PTB message SHOULD/MUST be verified before it is used to update the PMTU discovery information [RFC8201]. This verification confirms that the PTB message was sent in response to a packet originating by the sender, and needs to be performed before the PMTU discovery method reacts to the PTB message. When the router link MTU is indicated in the PTB message this MAY be

used by datagram PLPMTUD to reduce the size of a probe, but MUST NOT be used increase the effective PMTU.

5. Reception feedback: The destination PL endpoint is REQUIRED to provide a feedback method that indicates when a probe packet has been received by the destination endpoint. The local PL endpoint is REQUIRED to pass this feedback to the sender PLPMTUD method.
6. Probing and congestion control: The isolated loss of a probe packet SHOULD NOT be treated as an indication of congestion and its loss not directly trigger a congestion control reaction.
7. Probe loss recovery: If the data block carried by a probe message needs to be sent reliably, the PL (or layers above) MUST arrange retransmission/repair of any resulting loss. This method MUST be robust in the case where packet probes are lost due to other reasons (including link transmission error, congestion). The PLPMTUD method treats isolated loss of a probe packet (with or without an PTB message) as a potential indication of a PMTU limit on the path. The PL is permitted to retransmit any data included in a lost probe packet without adjusting its congestion window.
8. Cached effective PMTU: The sender MUST cache the effective PMTU value between probes and needs also to consider the disruption that could be incurred by an unsuccessful probe - both upon the flow that incurs a probe loss, and other flows that experience the effect of additional probe traffic.
9. Shared effective PMTU state: The specification of PLPMTUD [RFC4821] states: "If PLPMTUD updates the MTU for a particular path, all Packetization Layer sessions that share the path representation (as described in Section 5.2 of [RFC4821]) SHOULD be notified to make use of the new MTU and make the required congestion control adjustments". Such methods need to be robust to the wide variety of underlying network forwarding behaviours. Considerations about caching have been noted [RFC8201].

In addition the following design principles are stated:

- o Suitable MPS: The PLPMTUD method SHOULD avoid forcing an application to use an arbitrary small MPS (effective PMTU) for transmission while the method is searching for the currently supported PMTU. Datagram PLs do not necessarily support fragmentation of PDUs larger than the PMTU. A reduced MPS can adversely impact the performance of a datagram application.
- o Path validation: The PLPMTUD method MUST be robust to path changes that could have occurred since the path characteristics were last confirmed.
- o Datagram reordering: A method MUST be robust to the possibility that a flow encounters reordering, or has the traffic (including probe packets) is divided over more than one network path.

- o When to probe: The PLPMTUD method SHOULD determine whether the path capacity has increased since it last measured the path. This determines when the path should again be probed.

3.1. PMTU Probe Packets

PMTU discovery relies upon the sender being able to generate probe messages with a specific size. TCP is able to generate probe packets by choosing to appropriately segment data being sent [RFC4821].

In contrast, datagram PLs either have to request an application to send a data block with a specified size, or to utilise padding functions to extend the datagram beyond the size of the application data block. Protocols that permit exchange of control messages (without an application data block) could alternatively prefer to generate a probe packet by extending a control message with padding data.

When the method fails to validate the PMTU for the path, the required size of probe packet can need to be less than the size of the data block generated by an application. In this case, the PL could provide a way to fragment a datagram at the PL, or could instead utilise a control packet with padding.

A receiver needs to be able to distinguish in-band data from any added padding, and ensure that any added padding is not passed to an application at the receiver.

This results in three ways that a sender can create a probe packet:

Probing using application data: A probe packet that contains a data block supplied by an application that matches the size required for the probe. This requires a method to request the application to issue a data block of the desired probe size. If the application/transport needs protection from the loss of this probe packet, the application/transport may perform transport-layer retransmission/repair of the data block (e.g., by retransmission after loss is detected or by duplicating the data block in a datagram without the padding data).

Probing using application data: A probe packet that contains a data block supplied by an application that is combined with padding to inflate the length of the datagram to the size required for the probe. If the application/transport needs protection from the loss of this probe packet, the application/transport may perform transport-layer retransmission/repair of the data block (e.g., by retransmission after loss is detected or by duplicating the data

block in a datagram without the padding data).

Probing using application data: A probe packet that contains only control information and padding to inflate the packet to the size required for the probe. Since these probe packets do not carry any application-supplied data block, they do not typically require retransmission, although they do still consume network capacity.

3.2. Validation of the current effective PMTU

The PL needs a method to determine when packet probes have been successfully received end-to-end across a network path.

Transport protocols can include end-to-end methods that detect and report reception of specific datagrams that they send (e.g., DCCP and SCTP provide keep-alive/heartbeat features). This can also be used by PLPMTUD to acknowledge reception of a probe packet.

A PL that does not acknowledge data reception (e.g., UDP and UDP-Lite) is unable to detect when the packets it sends are discarded because their size is greater than the actual PMTUD. These PLs need to either reply on application protocol to detect this, or use of an additional transport method such as UDP-Options [I-D.ietf-tsvwg-udp-options], and then need to send a reachability probe (e.g., periodically solicit a response) to determine if the current effective PMTU is still supported by the network path.

PMTU discovery can also utilise PTB messages to detect when the actual PMTU supported by a network path is less than the current size of datagrams that are being sent.

3.3. Reduction of the effective PMTU

When the current effective PMTU is no longer supported by the network path, the transport needs to detect this and reduce the effective PMTU.

- o A PL that sends a datagram larger than the actual PMTU that includes no application data block, or one that does not attempt to provide any retransmission, can send a new probe packet with an updated probe size.
- o A PL that wishes to resend the application data block, may need to re-fragment the data block to a smaller datagram size. This could utilise network-layer or PL fragmentation when these are available.

4. Datagram PLPMTUD

This section specifies Datagram PLPMTUD.

The central idea of PLPMTU discovery is probing by a sender. Probe packets of increasing size are sent to find out the maximum size of a user message that is completely transferred across the network path from the sender to the destination. If a PTB message is received from a router or middlebox, this information ought to be verified and SHOULD used. The PTB messages can improve performance compared to one that relies solely on probing.

4.1. Probing

The PLPMTUD method utilises a timer to trigger the generation of probe packets. The `probe_timer` is started each time a probe packet is sent to the destination and is cancelled when receipt of the probe packet is acknowledged. Each time the `probe_timer` expires, the `probe_error_counter` is incremented, and the probe packet is retransmitted. The counter is initialised to zero when a probe packet is first sent with a particular size. The maximum number of retransmissions per probing size is configured (`MAX_PROBES`). If the value of the `PROBE_COUNT` exceeds `MAX_PROBES`, probing will be stopped and the last successfully probed PMTU is set as the effective PMTU.

Once probing is completed, the sender continues to use the effective PMTU until either a PTB message is received or the `PMTU_RAISE_TIMER` expires. If the PL is unable to verify reachability to the destination endpoint after probing has completed, the method uses a `REACHABILITY_TIMER` to periodically repeat a probe packet for the current effective PMTU size, while the `PMTU_RAISE_TIMER` is running. If the resulting probe packet is not acknowledged (i.e. the `PROBE_TIMER` expires), the method re-starts probing for the PMTU.

4.2. Timers

This method utilises three timers:

`PROBE_TIMER`: Configured to expire after a period longer than the maximum time to receive an acknowledgment to a probe packet.

`PMTU_RAISE_TIMER`: Configured to the period a sender ought to continue use the current effective PMTU, after which it re-commences probing for a higher PMTU. This timer has a period of 600 secs, as recommended by [RFC4821].

`REACHABILITY_TIMER`: Configured to the period a sender ought to wait before confirming the current effective PMTU is still supported. This is less than the `PMTU_RAISE_TIMER`.

An implementation could implement the various timers using a single

timer process.

4.3. Constants

The following constants are defined:

MAX_PROBES: The maximum value of the PROBE_ERROR_COUNTER.

MIN_PMTU: The smallest allowed probe packet size. This value is 1280 bytes as specified in [RFC2460].

BASE_PMTU: The BASE_PMTU is a considered a size that should work in most cases. The size equal to or larger than the minimum permitted and smaller than the maximum allowed. In the case of IPv6, this value is 1280 bytes as specified in [RFC2460]. When using IPv4, a size of 1200 is RECOMMENDED.

MAX_PMTU: This is the largest size of PMTU that is probed. It must be less than or equal to the minimum of the local MTU of the outgoing interface and the destination effective MTU for receiving.

4.4. Variables

This method utilises a set of variables:

effective PMTU: The effective PMTU is the maximum size of datagram that the method has currently determined can be supported along the entire path.

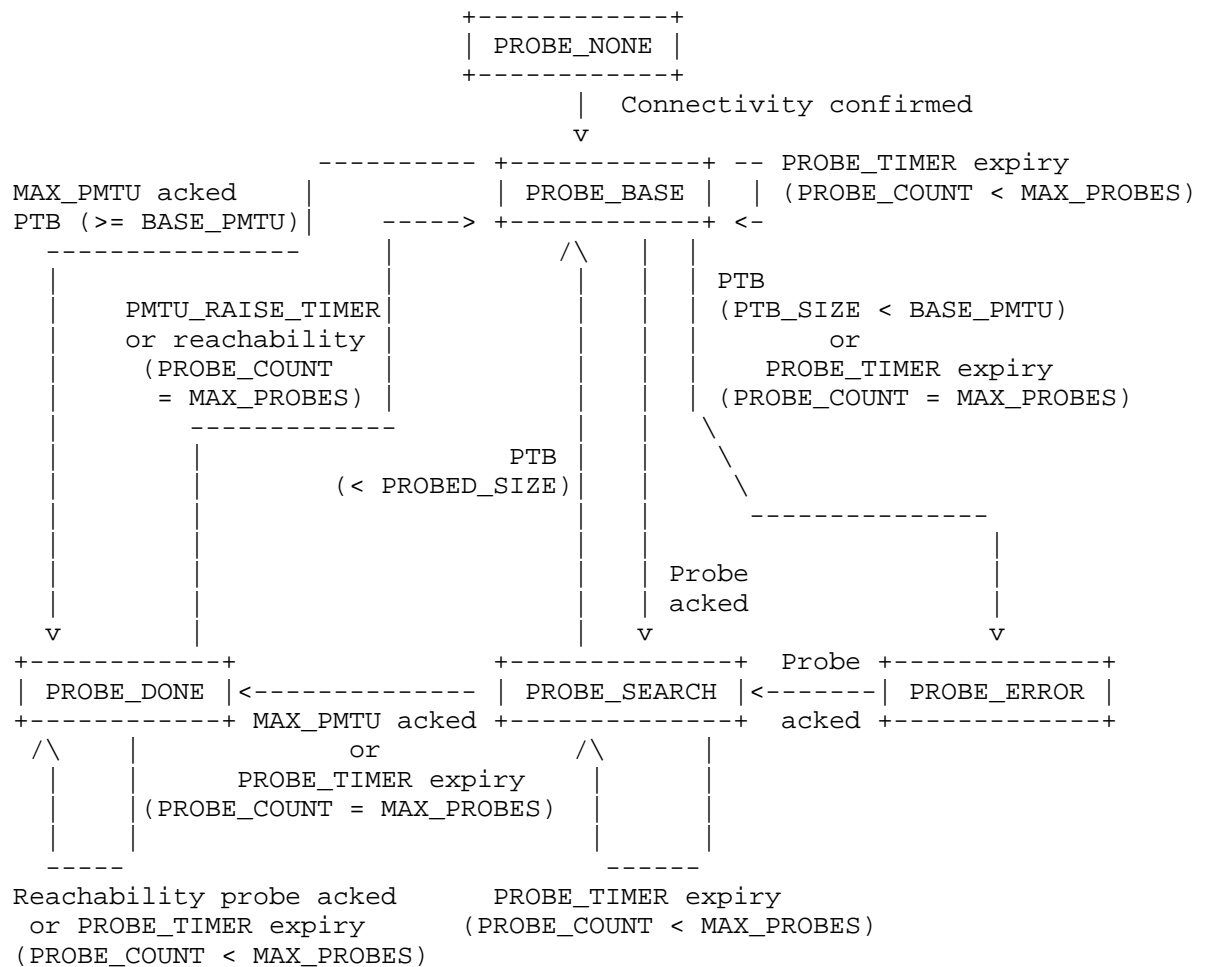
PROBED_SIZE: The PROBED_SIZE is the size of the current probe packet. This is a tentative value for the effective PMTU, which is awaiting confirmation by an acknowledgment.

PROBE_COUNT: This is a count of the number of unsuccessful probe packets that have been sent with size PROBED_SIZE. The value is initialised to zero when a particular size of PROBED_SIZE is first attempted.

PTB_SIZE: The PTB_Sizde is value returned by a verified PTB message indicating the local MTU size of a router along the path.

4.5. State Machine

A state machine for Datagram PLPMTUD is depicted in Figure 1. If multihoming is supported, a state machine is needed for each active path.



The following states are defined to reflect the probing process.

PROBE_NONE: The PROBE_NONE state is the initial state before probing has started. PLPMTUD is not performed in this state. The state transitions to PROBE_BASE, when a path has been confirmed, i.e. when a packet has arrived on this path. The effective PMTU is set to the BASE_PMTU size. Probing ought to start immediately after connection setup to prevent the loss of user data.

PROBE_BASE: The PROBE_BASE state is the starting point for datagram PLPMTUD, and used to confirm whether the BASE_PMTU size is supported by the network path. On entry, the PROBED_SIZE is set

to the `BASE_PMTU` size and the `PROBE_COUNT` is set to zero. A probe packet is sent, and the `PROBE_TIMER` is started. The state is left when the `PROBE_COUNT` reaches `MAX_PROBES`; a PTB message is received, or a probe packet is acknowledged.

PROBE_SEARCH: The `PROBE_SEARCH` state is the main probing state. This state is entered either when probing for the `BASE_PMTU` was successful or when there is a successful reachability test in the `PROBE_ERROR` state. On entry, the effective PMTU is set to the last acknowledged `PROBED_SIZE`.

On the first probe packet for each probed size, the `PROBE_COUNT` is set to zero. Each time a probe packet is acknowledged, the effective PMTU is set to the `PROBED_SIZE`, and then the `PROBED_SIZE` is increased. When a probe packet is not acknowledged within the period of the `PROBE_TIMER`, the `PROBE_COUNT` is incremented and the probe packet is retransmitted. The state is exited when the `PROBE_COUNT` reaches `MAX_PROBES`; a PTB message is verified; or a probe of size `PMTU_MAX` is acknowledged.

PROBE_ERROR: The `PROBE_ERROR` state represents the case where the network path is not known to support an effective PMTU of at least the `BASE_PMTU` size. It is entered when either a probe of size `BASE_PMTU` has not been acknowledged or a verified PTB message indicates a smaller link MTU than the `BASE_PMTU`. On entry, the `PROBE_COUNT` is set to zero and the `PROBED_SIZE` is set to the `MIN_PMTU` size, and the effective PMTU is reset to `MIN_PMTU` size. In this state, a probe packet is sent, and the `PROBE_TIMER` is started. The state transitions to the `PROBE_SEARCH` state when a probe packet is acknowledged.

PROBE_DONE: The `PROBE_DONE` state indicates a successful end to a probing phase. Datagram PLPMTUD remains in this state until either the `PMTU_RAISE_TIMER` expires or a PTB message is verified.

When PLPMTUD uses an unacknowledged PL and is in the `PROBE_DONE` state, a `REACHABILITY_TIMER` periodically resets the `PROBE_COUNT` and schedules a probe packet with the size of the effective PMTU. If the probe packet fails to be acknowledged after `MAX_PROBES` attempts, the method enters the `PROBE_BASE` state. An acknowledged PL SHOULD NOT continue to probe in this state.

Appendix Appendix A contains an informative description of key events:

5. Specification of Protocol-Specific Methods

This section specifies protocol-specific details for datagram PLPMTUD for IETF-specified transport protocols.

5.1. UDP and UDP-Lite

The current specifications of UDP and UDP-Lite [RFC3828] do not define a method in the RFC-series that supports PLPMTUD. In particular, these transport do not provide the transport layer features needed to implement datagram PLPMTUD.

5.1.1. UDP Options

UDP-Options [I-D.ietf-tsvwg-udp-options] supply the additional functionality required to implement datagram PLPMTUD. This enables padding to be added to UDP datagrams and can be used to provide feedback acknowledgement of received probe packets.

5.1.2. UDP Options required for PLPMTUD

This subsection proposes two new UDP-Options that add support for requesting a datagram response be sent and to mark this datagram as a response to a request.

<< We may define a parameter in an Option to indicate the EMTU_R to the peer.>>

5.1.2.1. Echo Request Option

The Echo Request Option allows a sending endpoint to solicit a response from a destination endpoint. The Echo Request carries a four byte token set by the sender.

```

+-----+-----+-----+
| Kind=9 | Len=6 | Token      |
+-----+-----+-----+
 1 byte   1 byte   4 bytes

```

5.1.2.2. Echo Response Option

The Echo Response Option is generated by the PL in response to reception of a previously received Echo Request. The Token field is associates the response with the Token value carried in the most recently-received Echo Request. The rate of generation of UDP packets carrying an Echo Response Option MAY be rate-limited.

```

+-----+-----+-----+
| Kind=10 | Len=6 | Token      |
+-----+-----+-----+
 1 byte   1 byte   4 bytes

```

5.1.3. Sending UDP-Option Probe Packets

This method specifies a probe packet that does not carry an application data block. The probe packet consists of a UDP datagram header followed by a UDP Option containing the ECHOREQ option, which is followed by NOP Options to pad the remainder of the datagram payload. The NOP padding is used to control the length of the probe

packet.

A UDP Option carrying the ECHORES option is used to provide feedback when the probe packet is received at the destination endpoint.

5.1.4. Validating the Path with UDP Options

Since UDP is an unacknowledged PL, a sender that does not have higher-layer information confirming correct delivery of datagrams SHOULD implement the REACHABILITY_TIMER to periodically send probe packets while in the PROBE_DONE state.

5.1.5. Handling of PTB Messages by UDP

Normal ICMP verification MUST be performed as specified in Section 5.2 of [RFC8085]. This requires that the PL verifies each received PTB messages to verify these are received in response to transmitted traffic. A verified PTB message MAY be used as input to the PLPMTUD algorithm.

5.2. SCTP

Section 10.2 of [RFC4821] specifies a recommended PLPMTUD probing method for SCTP. It recommends the use of the PAD chunk, defined in [RFC4820] to be attached to a minimum length HEARTBEAT chunk to build a probe packet. This enables probing without affecting the transfer of user messages and without interfering with congestion control. This is preferred to the use of DATA chunks (with padding as required) to serve as path probes.

<< We might define a parameter contained in the INIT and INIT ACK chunk to indicate the MTU to the peer. However, multihoming makes this a bit complex, so it might not be worth doing.>>

5.2.1. SCTP/IP4 and SCTP/IPv6

The base protocol is specified in [RFC4960].

5.2.1.1. Sending SCTP Probe Packets

Probe packets consist of an SCTP common header followed by a HEARTBEAT chunk and a PAD chunk. The PAD chunk is used to control the length of the probe packet. The HEARTBEAT chunk is used to trigger the sending of a HEARTBEAT ACK chunk. The reception of the HEARTBEAT ACK chunk acknowledges reception of a successful probe.

The HEARTBEAT chunk carries a Heartbeat Information parameter which should include, besides the information suggested in [RFC4960], the probing size, which is the MTU size the complete datagram will add up to. The size of the PAD chunk is therefore computed by reducing the probing size by the IPv4 or IPv6 header size, the SCTP common header, the HEARTBEAT request and the PAD chunk header. The payload of the PAD chunk contains arbitrary data.

To avoid the fragmentation of retransmitted data, probing starts right after the handshake before data is sent. Assuming normal behaviour (i.e., the PMTU is smaller than or equal to the interface MTU), this process will take a few RTTs depending on the number of PMTU sizes probed. The Heartbeat timer can be used to implement the PROBE_TIMER.

5.2.1.2. Validating the Path with SCTP

Since SCTP provides an acknowledged PL, a sender does MUST NOT implement the REACHABILITY_TIMER while in the PROBE_DONE state.

5.2.1.3. PTB Message Handling by SCTP

Normal ICMP verification MUST be performed as specified in Appendix C of [RFC4960]. This requires that the first 8 bytes of the SCTP common header are quoted in the payload of the PTB message, which can be the case for ICMPv4 and is normally the case for ICMPv6. When the verification is completed, the router Link MTU indicated in the PTB message SHOULD be used with the PLPMTUD algorithm.

5.2.2. SCTP/UDP

The UDP encapsulation of SCTP is specified in [RFC6951].

5.2.2.1. Sending SCTP/UDP Probe Packets

Packet probing can be performed as specified in Section 5.2.1.1. The maximum payload is reduced by 8 bytes, which has to be considered when filling the PAD chunk.

5.2.2.2. Validating the Path with SCTP/UDP

Since SCTP provides an acknowledged PL, a sender does MUST NOT implement the REACHABILITY_TIMER while in the PROBE_DONE state.

5.2.2.3. Handling of PTB Messages by SCTP/UDP

Normal ICMP verification MUST be performed for PTB messages as specified in Appendix C of [RFC4960]. This requires that the first 8 bytes of the SCTP common header are contained in the PTB message, which can be the case for ICMPv4 (but note the UDP header also consumes a part of the quoted packet header) and is normally the case for ICMPv6. When the verification is completed, the router Link MTU size indicated in the PTB message SHOULD be used with the PLPMTUD algorithm.

5.2.3. SCTP/DTLS

The DTLS encapsulation of SCTP is specified in [I-D.ietf-tsvwg-sctp-dtls-encaps]. It is used for data channels in WebRTC implementations.

5.2.3.1. Sending SCTP/DTLS Probe Packets

Packet probing can be done as specified in Section 5.2.1.1.

5.2.3.2. Validating the Path with SCTP/DTLS

Since SCTP provides an acknowledged PL, a sender does **MUST NOT** implement the REACHABILITY_TIMER while in the PROBE_DONE state.

5.2.3.3. Handling of PTB Messages by SCTP/DTLS

It is not possible to perform normal ICMP verification as specified in [RFC4960], since even if the ICMP contains enough information, the reflected SCTP common header would be encrypted. Therefore it is not possible to process PTB messages at the PL.

5.3. Other IETF Transports

QUIC is a UDP-based transport that provides reception feedback [I-D .ietf-quic-transport].

<< This section will be completed in a future revision of this ID >>

6. Acknowledgements

This work was partially funded by the European Union's Horizon 2020 research and innovation programme under grant agreement No. 644334 (NEAT). The views expressed are solely those of the author(s).

7. IANA Considerations

This memo includes no request to IANA.

If there are no requirements for IANA, the section will be removed during conversion into an RFC by the RFC Editor.

8. Security Considerations

The security considerations for the use of UDP and SCTP are provided in the references RFCs. Security guidance for applications using UDP is provided in the UDP-Guidelines [RFC8085].

PTB messages could potentially be used to cause a node to inappropriately reduce the effective PMTU. A node supporting PLPMTUD SHOULD appropriately verify the payload of PTB messages to ensure these are received in response to transmitted traffic (i.e., a reported error condition that corresponds to a datagram actually sent by the path layer).

9. References

9.1. Normative References

- [I-D.ietf-quick-transport]
Iyengar, J. and M. Thomson, "QUIC: A UDP-Based Multiplexed and Secure Transport", Internet-Draft draft-ietf-quick-transport-04, June 2017.
- [I-D.ietf-tsvwg-sctp-dtls-encaps]
Tuexen, M., Stewart, R., Jesup, R. and S. Loreto, "DTLS Encapsulation of SCTP Packets", Internet-Draft draft-ietf-tsvwg-sctp-dtls-encaps-09, January 2015.
- [I-D.ietf-tsvwg-udp-options]
Touch, J., "Transport Options for UDP", Internet-Draft draft-ietf-tsvwg-udp-options-01, June 2017.
- [RFC1122] Braden, R., Ed., "Requirements for Internet Hosts - Communication Layers", STD 3, RFC 1122, DOI 10.17487/RFC1122, October 1989, <<http://www.rfc-editor.org/info/rfc1122>>.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<http://www.rfc-editor.org/info/rfc2119>>.
- [RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, DOI 10.17487/RFC2460, December 1998, <<http://www.rfc-editor.org/info/rfc2460>>.
- [RFC3828] Larzon, L-A., Degermark, M., Pink, S., Jonsson, L-E. Ed., and G. Fairhurst, Ed., "The Lightweight User Datagram Protocol (UDP-Lite)", RFC 3828, DOI 10.17487/RFC3828, July 2004, <<http://www.rfc-editor.org/info/rfc3828>>.
- [RFC4820] Tuexen, M., Stewart, R. and P. Lei, "Padding Chunk and Parameter for the Stream Control Transmission Protocol (SCTP)", RFC 4820, DOI 10.17487/RFC4820, March 2007, <<https://www.rfc-editor.org/info/rfc4820>>.
- [RFC4960] Stewart, R., Ed., "Stream Control Transmission Protocol", RFC 4960, DOI 10.17487/RFC4960, September 2007, <<https://www.rfc-editor.org/info/rfc4960>>.
- [RFC6951] Tuexen, M. and R. Stewart, "UDP Encapsulation of Stream Control Transmission Protocol (SCTP) Packets for End-Host to End-Host Communication", RFC 6951, DOI 10.17487/RFC6951, May 2013, <<https://www.rfc-editor.org/info/rfc6951>>.
- [RFC8085] Eggert, L., Fairhurst, G. and G. Shepherd, "UDP Usage Guidelines", BCP 145, RFC 8085, DOI 10.17487/RFC8085, March 2017, <<http://www.rfc-editor.org/info/rfc8085>>.

- [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>>.

9.2. Informative References

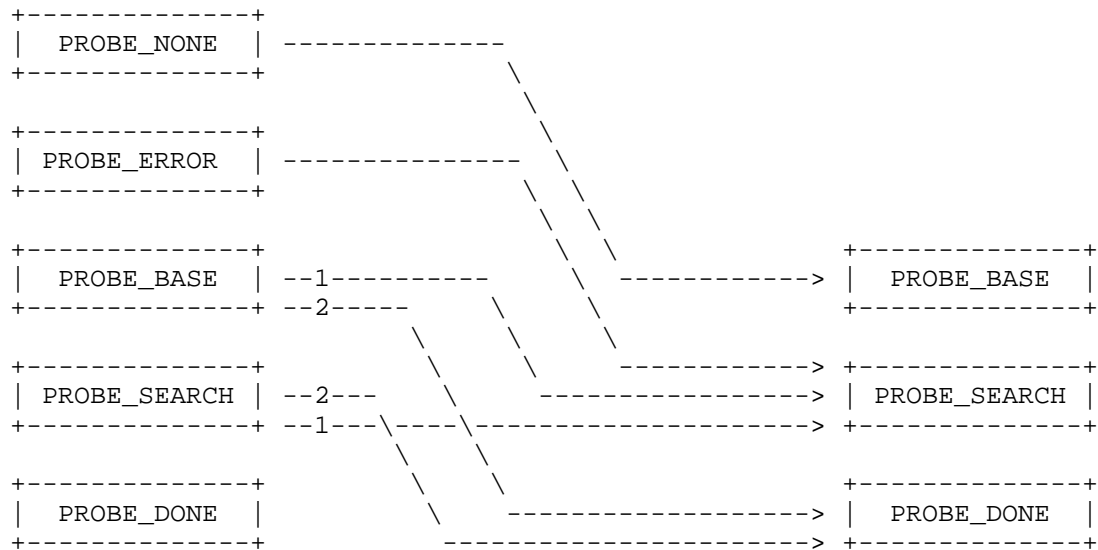
- [RFC1191] Mogul, J.C. and S.E. Deering, "Path MTU discovery", RFC 1191, DOI 10.17487/RFC1191, November 1990, <<http://www.rfc-editor.org/info/rfc1191>>.
- [RFC2923] Lahey, K., "TCP Problems with Path MTU Discovery", RFC 2923, DOI 10.17487/RFC2923, September 2000, <<https://www.rfc-editor.org/info/rfc2923>>.
- [RFC4340] Kohler, E., Handley, M. and S. Floyd, "Datagram Congestion Control Protocol (DCCP)", RFC 4340, DOI 10.17487/RFC4340, March 2006, <<https://www.rfc-editor.org/info/rfc4340>>.
- [RFC4821] Mathis, M. and J. Heffner, "Packetization Layer Path MTU Discovery", RFC 4821, DOI 10.17487/RFC4821, March 2007, <<http://www.rfc-editor.org/info/rfc4821>>.
- [RFC4890] Davies, E. and J. Mohacsi, "Recommendations for Filtering ICMPv6 Messages in Firewalls", RFC 4890, DOI 10.17487/RFC4890, May 2007, <<http://www.rfc-editor.org/info/rfc4890>>.

Appendix A. Event-driven state changes

This appendix contains an informative description of key events:

Path Setup: When a new path is initiated, the state is set to PROBE_NONE. As soon as the path is confirmed, the state changes to PROBE_BASE and the probing mechanism for this path is started. A probe packet with the size of the BASE_PMTU is sent.

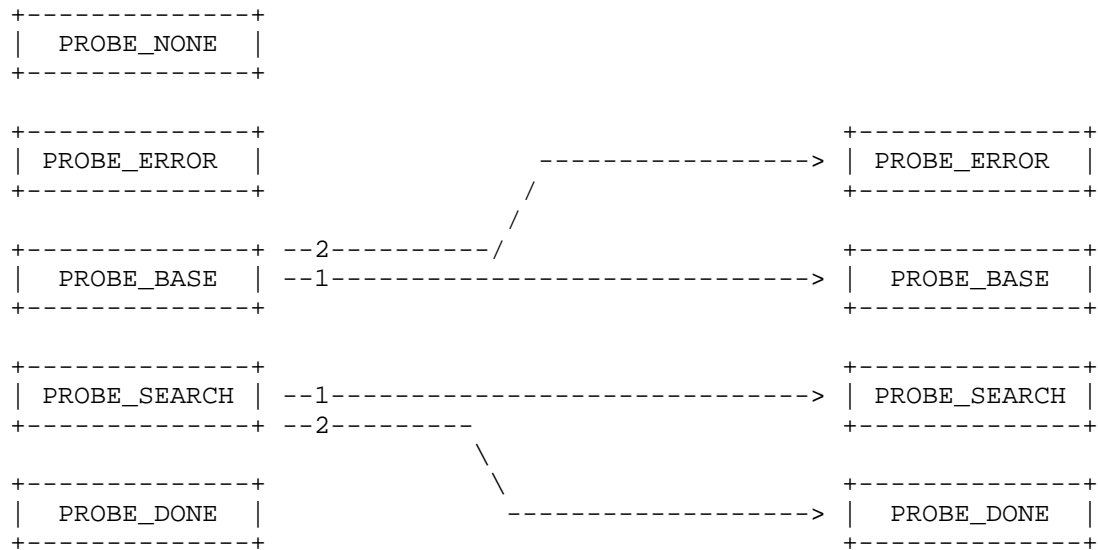
Arrival of an Acknowledgment: Depending on the probing state, the reaction differs according to Figure 4, which is just a simplification of Figure 1 focusing on this event.



Condition 1: The maximum PMTU size has not yet been reached.

Condition 2: The maximum PMTU size has been reached.

Probing timeout: The PROBE_COUNT is initialised to zero each time the value of PROBED_SIZE is changed. The PROBE_TIMER is started each time a probe packet is sent. It is stopped when an acknowledgment arrives that confirms delivery of a probe packet. If the probe packet is not acknowledged before the PROBE_TIMER expires, the PROBE_ERROR_COUNTER is incremented. When the PROBE_COUNT equals the value MAX_PROBES, the state is changed, otherwise a new probe packet of the same size (PROBED_SIZE) is resent. The state transitions are illustrated in Figure 5. This shows a simplification of Figure 1 with a focus only on this event.



Condition 1: The maximum number of probe packets has not been reached. Condition 2: The maximum number of probe packets has been reached.

PMTU raise timer timeout: The path through the network can change over time. It is impossible to discover whether a path change has increased in the actual PMTU by exchanging packets less than or equal to the effective PMTU. This requires PLPMTUD to periodically send a probe packet to detect whether a larger PMTU is possible. This probe packet is generated by the PMTU_RAISE_TIMER. When the timer expires, probing is restarted with the BASE_PMTU and the state is changed to PROBE_BASE.

Arrival of an ICMP message: The active probing of the path can be supported by the arrival of PTB messages sent by routers or middleboxes with a link MTU that is smaller than the probe packet size. If the PTB message includes the router link MTU, three cases can be distinguished:

1. The indicated link MTU in the PTB message is between the already probed and effective MTU and the probe that triggered the PTB message.
2. The indicated link MTU in the PTB message is smaller than the effective PMTU.
3. The indicated link MTU in the PTB message is equal to the BASE_PMTU.

In first case, the PROBE_BASE state transitions to the PROBE_ERROR state. In the PROBE_SEARCH state, a new probe packet is sent with the sized reported by the PTB message. Its result is handled according to the former events.

The second case could be a result of a network re-configuration. If the reported link MTU in the PTB message is greater than the BASE_MTU, the probing starts again with a value of PROBE_BASE. Otherwise, the method enters the state PROBE_ERROR.

In the third case, the maximum possible PMTU has been reached. This is probed again, because there could be a link further along the path with a still smaller MTU.

Note: Not all routers include the link MTU size when they send a PTB message. If the PTB message does not indicate the link MTU, the probe is handled in the same way as condition 2 of Figure 5.

Appendix B. Revision Notes

Note to RFC-Editor: please remove this entire section prior to publication.

Individual draft -00:

- o Comments and corrections are welcome directly to the authors or via the IETF TSVWG working group mailing list.
- o This update is proposed for WG comments.

Individual draft -01:

- o Contains the first representation of the algorithm, showing the states and timers
- o The text describing when to set the effective PMTU has not yet been verified by the authors
- o The text describing how to handle a PTB message indicating a link MTU larger than the probe has yet not been verified by the authors
- o No text currently describes how to handle inconsistent results from arbitrary re-routing along different parallel paths
- o Some middleboxes lie about the MTU they report in PTB messages.
- o Some constants and times do not yet have recommended values
- o To determine security to off-path-attacks: We need to decide whether a received PTB message SHOULD be verified or MUST be verified?
- o This update is proposed for WG comments.

Authors' Addresses

Godred Fairhurst
University of Aberdeen
School of Engineering
Fraser Noble Building
Aberdeen, AB24 3UE
UK

Email: gorry@erg.abdn.ac.uk

Tom Jones
University of Aberdeen
School of Engineering
Fraser Noble Building
Aberdeen, AB24 3UE
UK

Email: tom@erg.abdn.ac.uk

Michael Tuexen
Muenster University of Applied Sciences
Stegerwaldstrasse 39
Steinfurt, 48565
DE

Email: tuexen@fh-muenster.de

Irene Ruengeler
Muenster University of Applied Sciences
Stegerwaldstrasse 39
Steinfurt, 48565
DE

Email: i.ruengeler@fh-muenster.de

INTERNET-DRAFT
Intended Status: Informational
Expires: April 22, 2018

Tom Herbert
Quantonium
Petr Lapukhov
Facebook
October 19, 2017

Identifier-locator addressing for IPv6
draft-herbert-intarea-ila-00

Abstract

This specification describes identifier-locator addressing (ILA) for IPv6. Identifier-locator addressing differentiates between location and identity of a network node. Part of an address expresses the immutable identity of the node, and another part indicates the location of the node which can be dynamic. Identifier-locator addressing can be used to efficiently implement overlay networks for network virtualization as well as solutions for use cases in mobility.

Status of this Memo

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

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

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

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

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

Copyright and License Notice

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

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

Table of Contents

1	Introduction	4
1.1	Terminology	4
1.2	Use cases	6
2	Architectural overview	6
2.1	Addressing	6
2.2	Network topology	7
2.3	Translations and mappings	8
2.4	ILA routing	8
2.5	ILA domains	9
2.6	ILA control plane	9
3	Address formats	10
3.1	ILA address format	10
3.2	Locators	10
3.3	Identifiers	10
3.4	Standard identifier representation addresses	11
4	Optional identifier formats	12
4.1	Checksum neutral mapping	12
4.2	Identifier types	12
4.2.1	Interface identifiers	15
4.2.2	Locally unique identifiers	15
4.2.3	Virtual networking identifiers for IPv4	15
4.2.4	Virtual networking identifiers for IPv6 unicast	16
4.2.5	Virtual networking identifiers for IPv6 multicast	17
4.2.6	Non-local address identifiers	18
4.3	SIR addresses with formatted identifiers	19
4.3.1	SIR for locally unique identifiers	19
4.3.2	SIR for virtual addresses	20
4.3.2	SIR for non-local address identifiers	20
5	Operation	20
5.1	Identifier to locator mapping	20
5.2	Address translations	21
5.2.1	SIR to ILA address translation	21
5.2.2	ILA to SIR address translation	21
5.3	Virtual networking operation	22

5.3.1	Crossing virtual networks	22
5.3.2	IPv4/IPv6 protocol translation	22
5.4	Transport layer checksums	22
5.4.1	Checksum-neutral mapping	23
5.4.2	Sending an unmodified checksum	25
5.5	Non-local address mapping	25
5.6	Address selection	26
5.7	Duplicate identifier detection	26
5.8	ICMP error handling	26
5.8.1	Handling ICMP errors by ILA capable hosts	26
5.8.2	Handling ICMP errors by non-ILA capable hosts	27
5.9	Multicast	27
6	Motivation for ILA	28
6.1	Use cases	28
6.1.1	Multi-tenant virtualization	28
6.1.2	Datacenter virtualization	28
6.1.3	Mobile networks	29
6.2	Alternative methods	29
6.2.1	ILNP	29
6.2.2	Flow label as virtual network identifier	30
6.2.3	Extension headers	30
6.2.4	Encapsulation techniques	31
7	IANA Considerations	31
8	References	32
8.1	Normative References	32
8.2	Informative References	32
9	Acknowledgments	33
	Appendix A: Communication scenarios	34
A.1	Terminology for scenario descriptions	34
A.2	Identifier objects	35
A.3	Reference network for scenarios	35
A.4	Scenario 1: Object to task	36
A.5	Scenario 2: Object to Internet	36
A.6	Scenario 3: Internet to object	36
A.7	Scenario 4: Tenant system to service	37
A.8	Scenario 5: Object to tenant system	37
A.9	Scenario 6: Tenant system to Internet	38
A.10	Scenario 7: Internet to tenant system	38
A.11	Scenario 8: IPv4 tenant system to object	38
A.12	Tenant to tenant system in the same virtual network	39
A.12.1	Scenario 9: TS to TS in the same VN using IPV6	39
A.12.2	Scenario 10: TS to TS in same VN using IPv4	39
A.13	Tenant system to tenant system in different virtual networks	39
A.13.1	Scenario 11: TS to TS in different VNs using IPV6	39
A.13.2	Scenario 12: TS to TS in different VNs using IPv4	40
A.13.3	Scenario 13: IPv4 TS to IPv6 TS in different VNs	40
A.14	Scenario 14: Non-local address to tenant system	40

Appendix B: unique identifier generation	41
B.1 Globally unique identifiers method	41
B.2 Universally Unique Identifiers method	42
Appendix C: Datacenter task virtualization	42
C.1 Address per task	42
C.2 Job scheduling	43
C.3 Task migration	43
C.3.1 Address migration	44
C.3.2 Connection migration	44
Appendix D: Mobility in wireless networks	45

1 Introduction

This specification describes the address formats, protocol operation, and communication scenarios of identifier-locator addressing (ILA). In identifier-locator addressing, an IPv6 address is split into a locator and an identifier component. The locator indicates the topological location in the network for a node, and the identifier indicates the node's identity which refers to the logical or virtual node in communications. Locators are routable within a network, but identifiers typically are not. An application addresses a peer destination by identifier. Identifiers are mapped to locators for transit in the network. The on-the-wire address is composed of a locator and an identifier: the locator is sufficient to route the packet to a physical host, and the identifier allows the receiving host to translate and forward the packet to the application.

With identifier-locator addressing, network virtualization and addressing for mobility can be implemented in an IPv6 network without any additional encapsulation headers. Packets sent with identifier-locator addresses look like plain unencapsulated packets (e.g. TCP/IP packets). This method is transparent to the network, so protocol specific mechanisms in network hardware work seamlessly. These mechanisms include hash calculation for ECMP, NIC large segment offload, checksum offload, etc.

Some of the concepts for ILA are adapted from Identifier-Locator Network Protocol (ILNP) ([RFC6740], [RFC6741]) which defines a protocol and operations model for identifier-locator addressing in IPv6.

Section 6 provides a motivation for ILA and comparison of ILA with alternative methods that achieve similar functionality.

1.1 Terminology

ILA Identifier-locator addressing.

ILA host	An end host that is capable of performing ILA translations on transmit or receive.
ILA router	A network node that performs ILA translation and forwarding of translated packets.
ILA node	A network node capable of performing ILA translations. This can be an ILA router or ILA host.
Locator	A network prefix that routes to a physical host. Locators provide the topological location of an addressed node. ILA locators are a sixty-four bit prefixes.
Identifier	A number that identifies an addressable node in the network independent of its location. ILA identifiers are sixty-four bit values.
Identifier address	An IP address that contains an identifier.
ILA address	An IPv6 address composed of a locator (upper sixty-four bits) and an identifier (low order sixty-four bits).
ILA domain	A unique identifier namespace indicated by a SIR prefix. Each SIR prefix maps to ILA domain.
ILA translation	The process of translating the upper sixty-four bits of an IPv6 address. Translations may be from a SIR prefix to a locator or a locator to a SIR prefix.
SIR	Standard identifier representation.
SIR prefix	A sixty-four bit network prefix used to identify a SIR address.
SIR address	An IPv6 address composed of a SIR prefix (upper sixty-four bits) and an identifier (lower sixty-four bits). SIR addresses are visible to applications and provide a means to address nodes independent of their location.
Virtual address	An IPv6 or IPv4 address that resides in the address space of a virtual network. Such addresses may be translated to SIR addresses as an external

representation of the address outside of the virtual network, or they may be translated to ILA addresses for transit over an underlay network.

Topological address

An address that refers to a non-virtual node in a network topology. These address physical hosts in a network.

Checksum-neutral mapping

A method to preserve a correct transport layer checksum when performing ILA translation. When the upper sixty-four bits of an address are overwritten in an ILA translation, a modification can be made to the low order bits of the identifier to offset the checksum difference.

1.2 Use cases

ILA use cases include datacenter virtualization, network virtualization, and mobility in cellular and other mobile networks. Section 6 provides details on these use cases. ILA operates at the network layer so it works with any transport layer protocol and can be used at intermediate devices or end nodes. An ILA implementation may include optimizations depending on where in the network it runs.

2 Architectural overview

Identifier-locator addressing allows a data plane method to implement network virtualization without encapsulation and its related overheads. The service ILA provides is effectively layer 3 over layer 3 network virtualization (IPv4 or IPv6 over IPv6).

2.1 Addressing

ILA performs translations on IPv6 address. There are two types of addresses introduced for ILA: ILA addresses and SIR addresses.

ILA addresses are IPv6 addresses that are composed of a locator (upper sixty-four bits) and an identifier (low order sixty-four bits). The identifier serves as the logical addresses of a node, and the locator indicates the location of a node on the network.

A SIR address (standard identifier representation) is an IPv6 address that contains an identifier and an application visible SIR prefix. SIR addresses are visible to the application and can be used as connection endpoints. When a packet is sent to a SIR address, an ILA router or host overwrites the SIR prefix with a locator corresponding

to the identifier. When a peer receives the packet, the locator is overwritten with the original SIR prefix before delivery to the application. In this manner applications only see SIR addresses, they do not have visibility into ILA addresses.

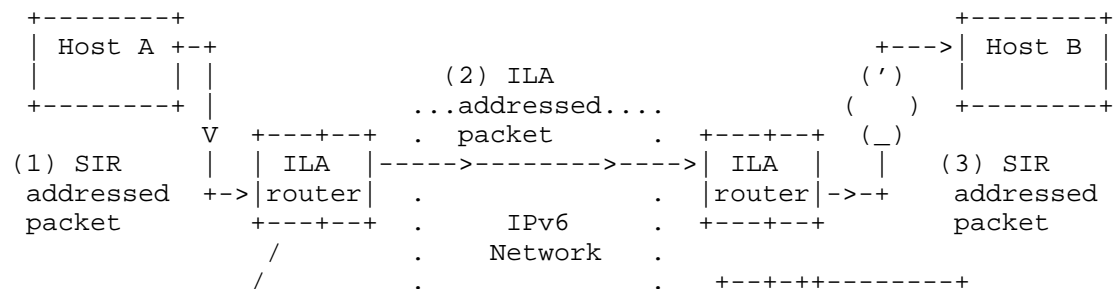
ILA translations can transform addresses from one type to another. In network virtualization, virtual addresses can be translated into ILA and SIR addresses, and conversely ILA and SIR addresses can be translated to virtual addresses.

2.2 Network topology

ILA nodes are nodes in the network that perform ILA translations. An ILA router is a node that performs ILA address translation and packet forwarding to implement overlay network functionality. ILA routers perform translations on packets sent by end nodes for transport across an underlay network. Packets received by ILA routers on the underlay network have their addresses reversed translated for reception at an end node. An ILA host is an end node that implements ILA functionality for transmitting or receiving packets.

ILA nodes are responsible for transit of packets over an underlay network. On ingress, an ILA node (host or router) will translate the virtual or SIR address of a destination to an ILA address. At a peer ILA node, the reverse translation is performed before handing packets to an application.

The figure below provides an example topology using ILA. ILA translations performed in one direction between Host A and Host B are denoted. Host A sends a packet with a destination SIR address (step (1)). An ILA router in the path translates the SIR address to an ILA address with a locator. The locator is set to a value that will route packets to a peer ILA node that Host B is downstream of. The packet is forwarded over the network and delivered to the peer ILA node (step 2). The peer ILA node, in this case another ILA router, translates the destination address back to a SIR address and forwards to the final destination (step 3).





2.3 Translations and mappings

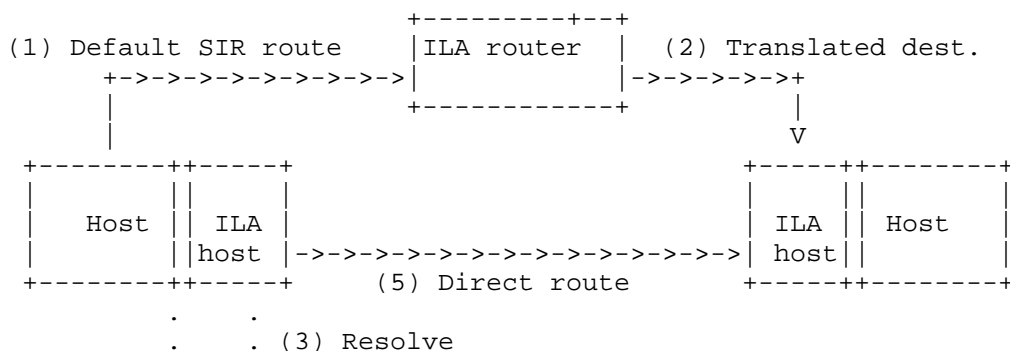
Address translation is the mechanism employed by ILA. Logical or virtual addresses are translated to topological IPv6 addresses for transport to the proper destination. Translation occurs in the upper sixty-four bits of an address, the low order sixty-four bits contains an identifier that is immutable and is not used to route a packet.

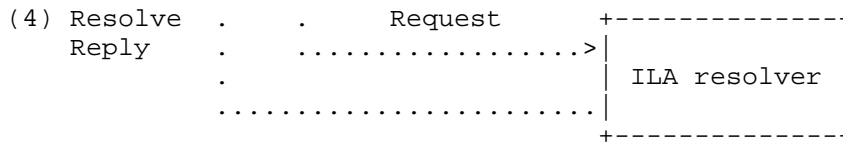
Each ILA node maintains a mapping table. This table maps identifiers to locators. The mappings are dynamic as nodes with identifiers can be created, destroyed, or move in the network. Mappings are propagated amongst ILA routers or hosts in a network using mapping propagation protocols (mapping propagation protocols will be described in other specifications).

Identifiers are not statically bound to a host on the network, and in fact their binding (or location) may change. This is the basis for network virtualization and device mobility. An identifier is mapped to a locator at any given time, and a set of identifier to locator mappings is propagated throughout a network to allow communications. The mappings are kept synchronized so that if an identifier migrates to a new location, its identifier to locator mapping is updated.

2.4 ILA routing

ILA is intended to be sufficiently lightweight so that all the hosts in a network could potentially send and receive ILA addressed packets. In order to scale this model and allow for hosts that do not participate in ILA, a routing topology may be applied. A simple routing topology is illustrated below.





An ILA router can be addressed by an "anycast" SIR prefix so that it receives packets sent on the network with SIR addresses. When an ILA router receives a SIR addressed packet (step (1) in the diagram) it will perform the ILA translation and send the ILA addressed packet to the destination ILA node (step (2)).

If a sending host is ILA capable the triangular routing can be eliminated by performing an ILA resolution protocol. This entails a host sending an ILA resolve request that specifies the SIR address to resolve (step (3) in the figure). An ILA resolver can respond to a resolve request with the identifier to locator mapping (step (4)). Subsequently, the ILA host can perform ILA translation and send directly to the destination specified in the locator (step (5) in the figure). The ILA resolution protocol will be specified in a companion document.

In this model an ILA host maintains a cache of identifier mappings for identifiers that it is currently communicating with. ILA routers are expected to maintain a complete list of identifier to locator mappings within the SIR domains that they service.

2.5 ILA domains

An ILA domain defines a namespace for identifiers. Identifiers must be unique within an ILA domain. Each SIR prefix maps to one ILA domain so that the combination of a SIR prefix and an identifier (a SIR address) uniquely identifies a node. More than one SIR prefix may be associated with a domain in which case SIR addresses with different SIR prefixes but the same identifier would refer to the same node.

Locators **MUST** map to only one SIR domain in order to ensure that translation from a locator to SIR prefix is unambiguous.

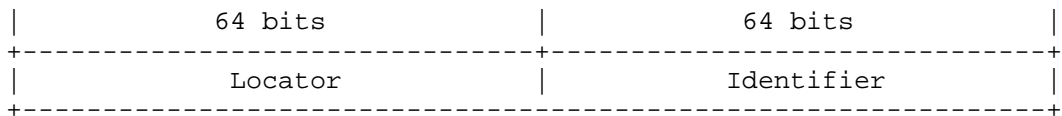
2.6 ILA control plane

ILA routers and ILA hosts assume a control plane that propagates the tables that map SIR addresses to ILA address (or just identifier to locator mappings). There are several possible methods for control planes that have been proposed including synchronized configuration, BGP, DNS, and NoSQL databases. Defining a specific control plane for ILA is out of scope of this document.

3 Address formats

3.1 ILA address format

An ILA address is composed of a locator and an identifier where each occupies sixty-four bits (similar to the encoding in ILNP [RFC6741]).

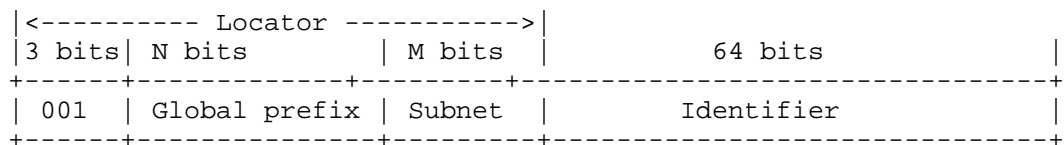


Note that there is no technical reason why identifiers and locators must be sixty-four bits. Different sizes could be used. The split is somewhat arbitrary, however it does simplify the description and implementation. For instance, sixty-four bits is the size of a "long long" native data type in several computer architectures. It is conceivable that a different arrangement could be used for some ILA domain. However, for the purposes of this document we assume the 64/64 split.

3.2 Locators

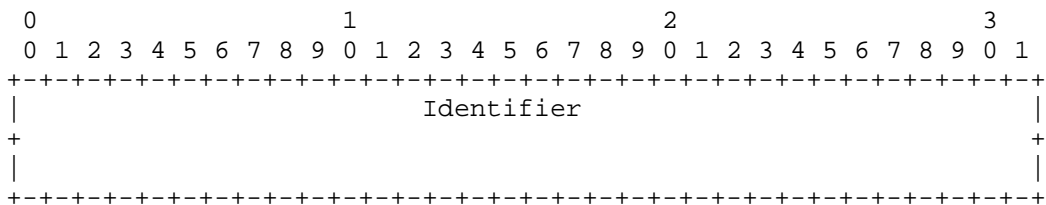
Locators are routable network address prefixes that create topological addresses for physical hosts within the network. They SHOULD be assigned from a global address block [RFC3587].

The format of an ILA address with a global unicast locator is:



3.3 Identifiers

Identifiers uniquely identify logical nodes in an ILA domain. The format of an ILA identifier is:



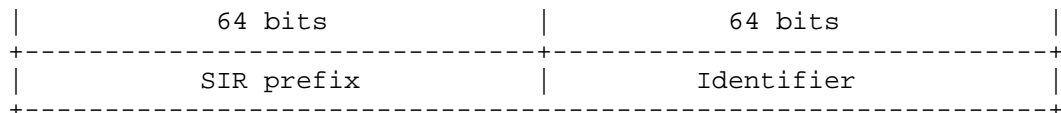
Identifiers are specified to be sixty-four bit values that are unstructured. A structure and format for identifiers MAY be defined for a domain; for instance the operator of an ILA domain may define the use of prefixes for its identifiers in order to facilitate hierarchies of its identifiers. Section 4 defines optional ILA formats that an ILA domain might impose locally that allow different types of identifiers as well as an indication of checksum neutral mapping.

3.4 Standard identifier representation addresses

An identifier identifies objects or nodes in a network. For instance, an identifier may refer to a specific host, virtual machine, or tenant system. When a host initiates a connection or sends a packet, it uses an identifier to indicate the peer endpoint of the communication. The endpoints of an established connection context are also referenced by identifiers (encoded in SIR addresses). It is only when the packet is actually being sent over a network that the locator for the identifier needs to be resolved.

In order to maintain compatibility with existing networking stacks and applications, identifiers are encoded in IPv6 addresses using a standard identifier representation (SIR) address. A SIR address is a combination of a prefix which occupies what would be the locator portion of an ILA address, and the identifier in its usual location.

The format of a SIR address is:



A SIR prefix SHOULD be a globally routable prefix per [RFC3587]. A globally routable SIR prefix facilitates connectivity between hosts on the Internet and ILA nodes. An ILA router between a site's network and the Internet can translate between SIR prefix and locator for an identifier. A network may have multiple SIR prefixes where each prefix defines a unique identifier space.

Locators MUST only be associated with one SIR prefix. This ensures that if a translation from a SIR address to an ILA address is performed when sending a packet, the reverse translation at the receiver yields the same SIR address that was seen at the transmitter. This also ensures that a reverse checksum-neutral mapping can be performed at a receiver to restore the addresses that were included in a pseudo header for setting a transport checksum.

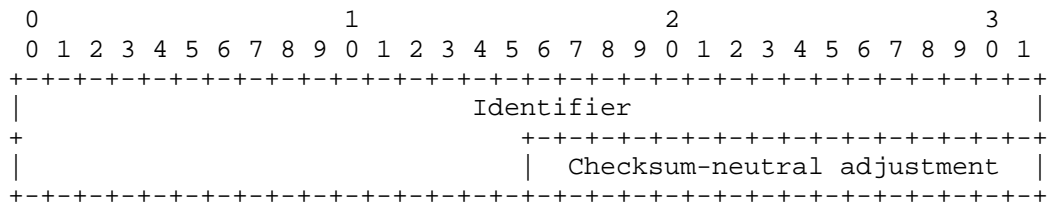
A standard identifier representation address can be used as the externally visible address for a node. This can be used throughout the network, returned in DNS AAAA records [RFC3363], used in logging, etc. An application can use a SIR address without knowledge that it encodes an identifier.

4 Optional identifier formats

This section describes optional identifier formats that allow for different types of identifiers, groups of identifiers, and checksum neutral mapping being applied. Note that identifiers are defined as unstructured fields, there is no required structure imposed on them. An administrator MAY impose an identifier format within an ILA domain. Any imposed structure is local only to the domain and all ILA nodes within the domain must agree on the format. A format might include optional elements as described below, or may include other elements customized for a domain.

4.1 Checksum neutral mapping

Checksum neutral mapping is an optional mechanism that may be applied to an ILA address (see section 5.4.1 for description of checksum-neutral mapping). When employed the checksum neutral mapping occupies the low order sixteen bits of the identifier in a locator address.



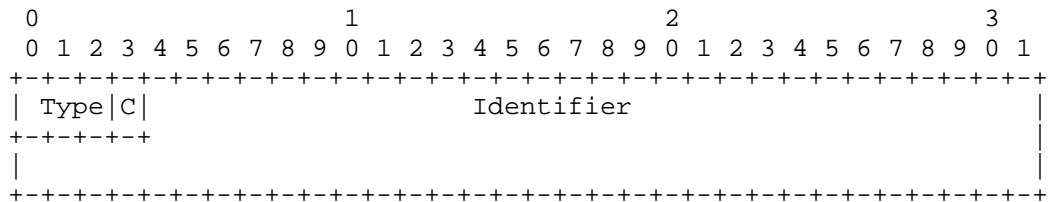
The presence of the checksum-neutral adjustment field must be unambiguous. An optional C-bit flag could be used in the identifier to indicate the checksum-neutral field is valid. The use of the C-bit is demonstrated below. Alternatively, within an ILA domain an operator could require it to be assumed that all ILA addresses have the checksum-neutral field set so that an explicit flag is not needed. Note that checksum-neutral adjustment is not used with SIR addresses.

4.2 Identifier types

This section describes an optional identifier format that allows for different types of identifiers and an indication of checksum neutral mapping being applied.

Note that the identifier type format is optional. If this is not used within an ILA domain then all ILA nodes assume that all identifiers are of the same type (locally unique identifier for instance).

The optional type format of an ILA identifier with the checksum adjust flag is:



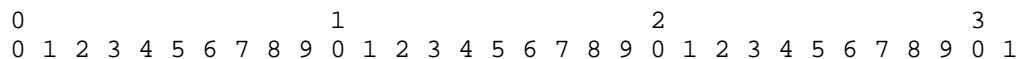
Fields are:

- o Type: Type of the identifier (see below).
- o C: The C-bit. This indicates that checksum-neutral mapping applied (see below). Presence of this field is optional.
- o Identifier: Identifier value.

Identifier types allow standard encodings for common uses of identifiers. Defined identifier types are:

- 0: interface identifier
- 1: locally unique identifier
- 2: virtual networking identifier for IPv4 address
- 3: virtual networking identifier for IPv6 unicast address
- 4: virtual networking identifier for IPv6 multicast address
- 5: non-local address identifier
- 6-7: Reserved

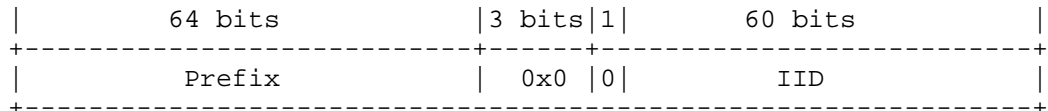
If the C-bit is set then the low order sixteen bits of an identifier contain the adjustment for checksum-neutral mapping (see section 4.4.1 for description of checksum-neutral mapping). The format of an identifier with checksum neutral mapping is:



```
+-----+
| Type|1|                               Identifier                               |
+-----+                               +-----+
|                               | Checksum-neutral adjustment |
+-----+
```

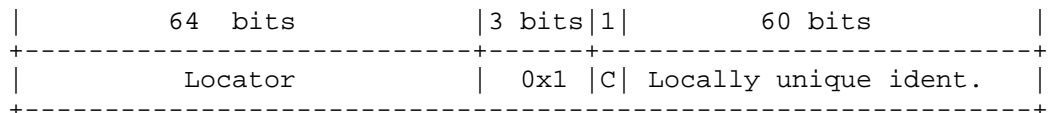
4.2.1 Interface identifiers

The interface identifier type indicates a plain local scope interface identifier. When this type is used the address is a normal IPv6 address without identifier-locator semantics. The purpose of this type is to allow normal IPv6 addresses to be defined within the same networking prefix as ILA addresses. Type bits and C-bit MUST be zero. The format of an ILA interface identifier address is:

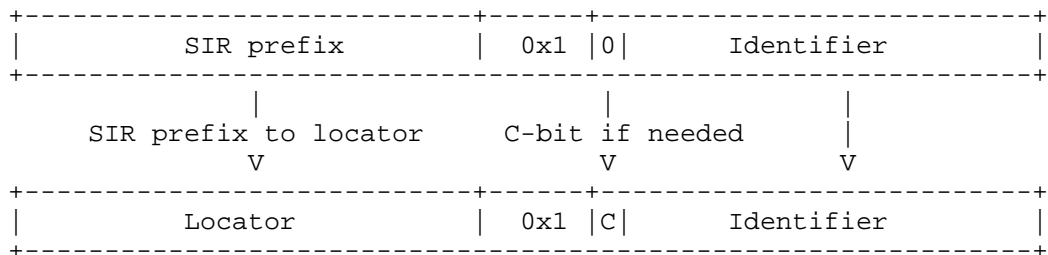


4.2.2 Locally unique identifiers

Locally unique identifiers (LUI) can be created for various addressable objects within a network. These identifiers are in a flat space and must be unique within a SIR domain (unique within a site for instance). To simplify administration, hierarchical allocation of locally unique identifiers may be performed. The format of an ILA address with locally unique identifiers is:



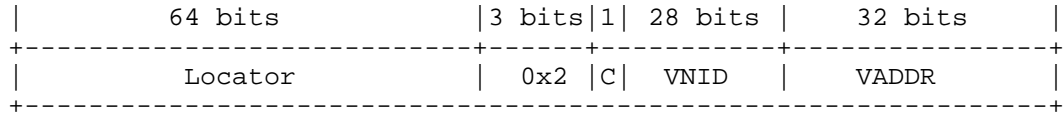
The figure below illustrates the translation from SIR address to an ILA address as would be performed when a node sends to a SIR address. Note the low order 16 bits of the identifier may be modified as the checksum-neutral adjustment. The reverse translation of ILA address to SIR address is symmetric.



4.2.3 Virtual networking identifiers for IPv4

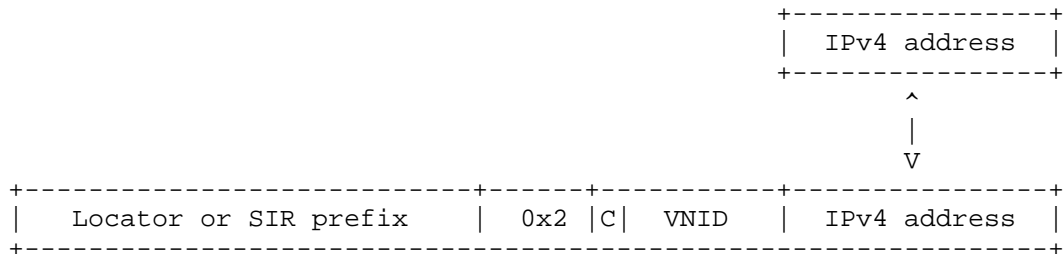
This type defines a format for encoding an IPv4 virtual address and virtual network identifier within an identifier. The format of an ILA

address for IPv4 virtual networking is:



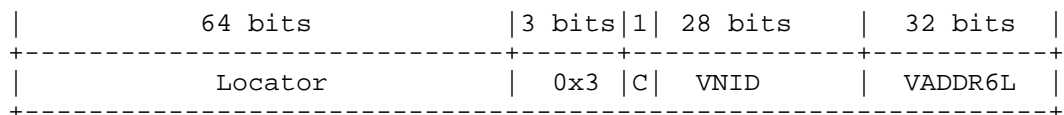
VNID is a virtual network identifier and VADDR is a virtual address within the virtual network indicated by the VNID. The VADDR can be an IPv4 unicast or multicast address, and may often be in a private address space (i.e. [RFC1918]) used in the virtual network.

Translating a virtual IPv4 address into an ILA or SIR address and the reverse translation are straight forward. Note that the low order 16 bits of the IPv6 address may be modified as the checksum-neutral adjustment and that this translation implies protocol translation between IPv4 and IPv6.



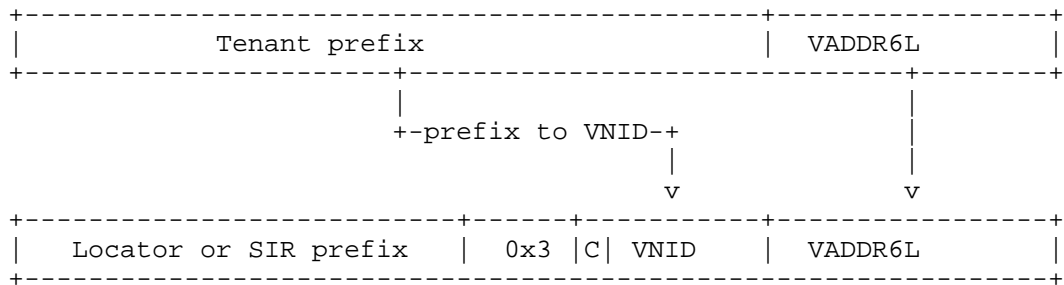
4.2.4 Virtual networking identifiers for IPv6 unicast

In this format, a virtual network identifier and virtual IPv6 unicast address are encoded within an identifier. To facilitate encoding of virtual addresses, there is a unique mapping between a VNID and a ninety-six bit prefix of the virtual address. The format an IPv6 unicast encoding with VNID in an ILA address is:

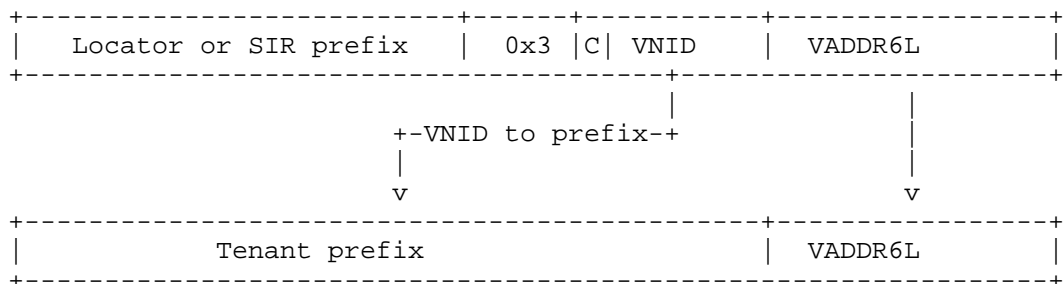


VADDR6L contains the low order 32 bits of the IPv6 virtual address. The upper 96 bits of the virtual address inferred from the VNID to prefix mapping. Note that for ILA translations the low order sixteen of the VADDR6L may be modified for checksum-neutral adjustment.

The figure below illustrates encoding a tenant IPv6 virtual unicast address into a ILA or SIR address.

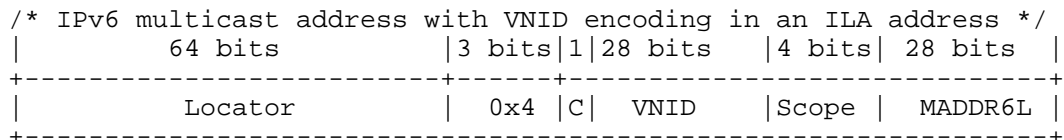


This encoding is reversible, given an ILA address, the virtual address visible to the tenant can be deduced:



4.2.5 Virtual networking identifiers for IPv6 multicast

In this format, a virtual network identifier and virtual IPv6 multicast address are encoded within an identifier.



This format encodes an IPv6 multicast address in an identifier. The scope indicates multicast address scope as defined in [RFC7346]. MADDR6L is the low order 28 bits of the multicast address. The full multicast address is thus:

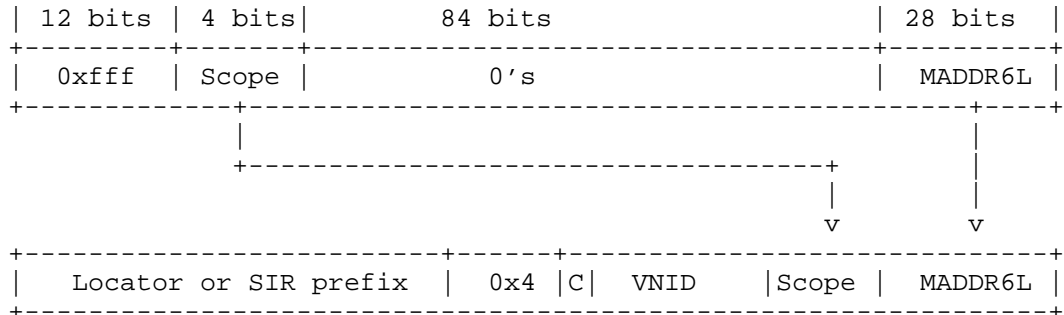
ff0<Scope>::<MADDR6L high 12 bits>:<MADDR6L low 16 bits>

And so can encode multicast addresses of the form:

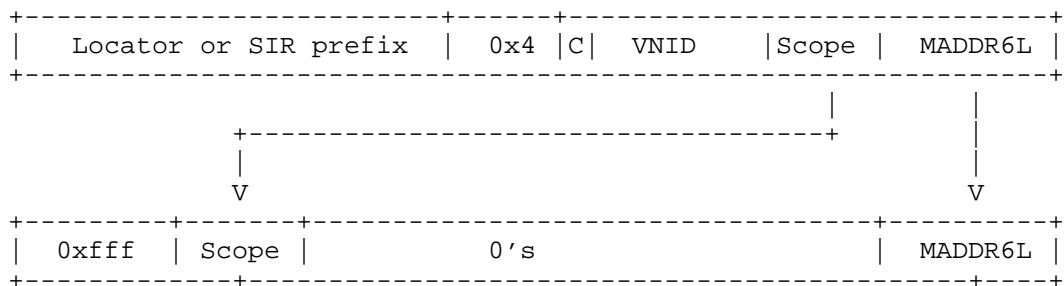
ff0X::0 to ff0X::0fff:ffff

The figure below illustrates encoding a tenant IPv6 virtual multicast

address in an ILA or SIR address. Note that low order sixteen bits of MADDR6L may be modified to be the checksum-neutral adjustment.



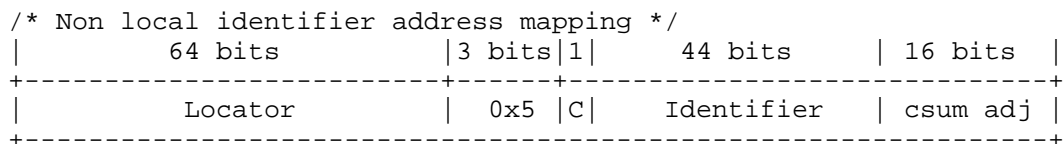
This translation is reversible:



4.2.6 Non-local address identifiers

Non-local address identifiers allow mapping an arbitrary address to an ILA address. The mapping system contains an entry that associates an IPv6 address with an identifier. The associated IP address does not need to be a SIR address or even in the same routing domain.

The format of a non-local address identifier is

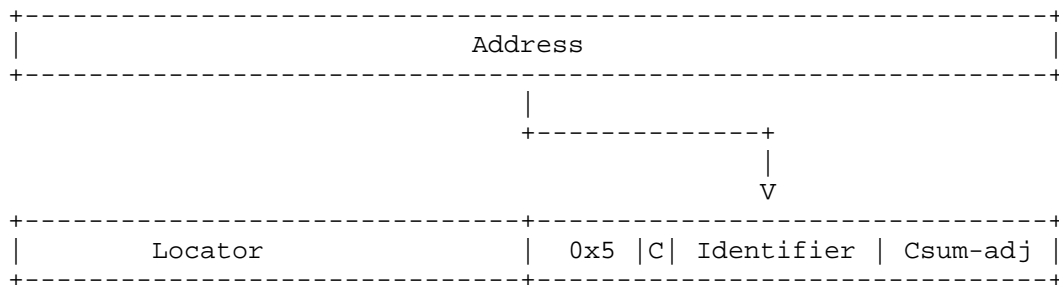


If the checksum adjust field is present it is not part of the identifier that is used in the mapping lookup. The high order bits of the address were originally not a SIR prefix, so it cannot be assumed the checksum adjustment is based on a SIR prefix. The identifier is taken to be the forty-four bits that precede the checksum adjustment

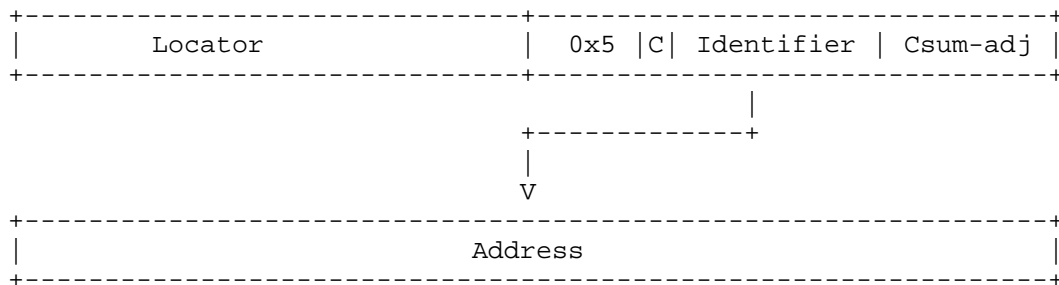
field. When creating the ILA address, the checksum adjustment field is initialized to zero and then set based on checksum difference between the original non-local address and the ILA address.

The figure below illustrates encoding an address into a locator address.

```
/* Non local address identifier */
```

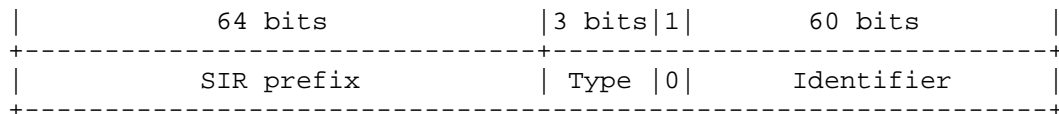


A reverse translation is performed based on a lookup in the mapping table on the identifier (44 bits as shown above). The result of the lookup provides the original address.



4.3 SIR addresses with formatted identifiers

The format of a SIR address with a formatted identifier is:



The C-bit (checksum-neutral mapping) MUST be zero for a SIR address. Type may be any identifier type except zero (interface identifiers)

4.3.1 SIR for locally unique identifiers

The SIR address for a locally unique identifier has format:

	64 bits	3 bits 1	60 bits	
+-----+		+-----+		+-----+
	SIR prefix	0x1 0	Locally unique ident.	
+-----+		+-----+		+-----+

4.3.2 SIR for virtual addresses

A virtual address can be encoded using the standard identifier representation. For example, the SIR address for an IPv6 virtual address may be:

	64 bits	3 bits 1	28 bits		32 bits	
+-----+		+-----+		+-----+		+-----+
	SIR prefix	0x3 0	VNID		VADDR6	
+-----+		+-----+		+-----+		+-----+

Note that this allows three representations of the same address in the network: as a virtual address, a SIR address, and an ILA address.

4.3.2 SIR for non-local address identifiers

A non-local address identifier can be encoded using the standard identifier representation. For example, an encoding may be:

	64 bits	3 bits 1	44 bits		16 bits	
+-----+		+-----+		+-----+		+-----+
	SIR prefix	0x5 0	Identifier		0	
+-----+		+-----+		+-----+		+-----+

Note that lower order sixteen bits are set to zero since that would be the checksum adjustment value bits if translated to an ILA address.

5 Operation

This section describes operation methods for using identifier-locator addressing.

5.1 Identifier to locator mapping

An application initiates a communication or flow using a SIR address or virtual address for a destination. In order to send a packet on the network, the destination address is translated by an ILA node in the path. An ILA node maintains a list of mappings from identifier to locator to perform this translation.

The mechanisms of propagating and maintaining identifier to locator mappings are outside the scope of this document.

5.2 Address translations

With ILA, address translation is performed to convert SIR addresses to ILA addresses, and ILA addresses to SIR addresses. Translation is usually done on a destination address as a form of source routing, however translation on source virtual addresses to SIR addresses can also be done to support some network virtualization scenarios (see section Appendix A for examples).

5.2.1 SIR to ILA address translation

When translating a SIR address to an ILA address, the SIR prefix in the address is overridden with a locator, and checksum neutral mapping may be performed. Since this operation is potentially done for every packet the process should be very efficient (particularly the lookup and checksum processing operations).

The typical steps to transmit a packet using ILA are:

- 1) Host stack creates a packet with source address set to a local address (possibly a SIR address) for the local identity, and the destination address is set to the SIR address or virtual address for the peer. The peer address may have been discovered through DNS or other means.
- 2) An ILA node translates the packet to use the locator. If the original destination address is a SIR address then the SIR prefix is overwritten with the locator. If the original packet is a virtually addressed tenant packet then the virtual address is translated per section 4.2. The locator is discovered by a lookup in the locator to identifier mappings.
- 3) The ILA node performs checksum-neutral mapping if configured for that (section 5.4).
- 4) Packet is forwarded on the wire. The network routes the packet to the node indicated by the locator.

5.2.2 ILA to SIR address translation

When a destination node (ILA router or end host) receives an ILA addressed packet, the ILA address **MUST** be translated back to a SIR address (or virtual address) before upper layer processing.

The steps of receive processing are:

- 1) Packet is received. The destination locator is verified to match a locator assigned to the node.
- 2) A lookup is performed on the destination identifier to find if it addresses a local identifier. If match is found, either the locator is overwritten with SIR prefix (for locally unique identifier type) or the address is translated back to a tenant virtual address.
- 3) Perform reverse checksum-neutral mapping if C-bit is set (section 5.4).
- 4) Perform any optional policy checks; for instance that the source may send a packet to the destination address, that packet is not illegitimately crossing virtual networks, etc.
- 5) Forward packet to the application.

5.3 Virtual networking operation

When using ILA with virtual networking identifiers, address translation is performed to convert tenant virtual network and virtual addresses to ILA addresses, and ILA addresses back to a virtual network and tenant's virtual addresses. Translation may occur on either source address, destination address, or both (see scenarios for virtual networking in Appendix A). Address translation is performed similar to the SIR translation cases described above.

5.3.1 Crossing virtual networks

With explicit configuration, virtual network hosts may communicate directly with virtual hosts in another virtual network by using SIR addresses for virtualization in both the source and destination addresses. This might be done to allow services in one virtual network to be accessed from another (by prior agreement between tenants). See appendix A.13 for example of ILA addressing for such a scenario.

5.3.2 IPv4/IPv6 protocol translation

An IPv4 tenant may send a packet that is converted to an IPv6 packet with ILA addresses. Similarly, an IPv6 packet with ILA addresses may be converted to an IPv4 packet to be received by an IPv4-only tenant. These are IPv4/IPv6 stateless protocol translations as described in [RFC6144] and [RFC6145]. See appendix A.12 for a description of these scenarios.

5.4 Transport layer checksums

Packets undergoing ILA translation may encapsulate transport layer checksums (e.g. TCP or UDP) that include a pseudo header that is affected by the translation.

ILA provides two alternatives do deal with this:

- o Perform a checksum-neutral mapping to ensure that an encapsulated transport layer checksum is kept correct on the wire.
- o Send the checksum as-is, that is send the checksum value based on the pseudo header before translation.

Some intermediate devices that are not the actual end point of a transport protocol may attempt to validate transport layer checksums. In particular, many Network Interface Cards (NICs) have offload capabilities to validate transport layer checksums (including any pseudo header) and return a result of validation to the host. Typically, these devices will not drop packets with bad checksums, they just pass a result to the host. Checksum offload is a performance benefit, so if packets have incorrect checksums on the wire this benefit is lost. With this incentive, using checksum-neutral mapping is recommended. If it is known that the addresses of a packet are not included in a transport checksum, for instance a GRE packet is being encapsulated, then a source may choose not to perform checksum-neutral mapping.

5.4.1 Checksum-neutral mapping

When a change is made to one of the IP header fields in the IPv6 pseudo-header checksum (such as one of the IP addresses), the checksum field in the transport layer header may become invalid. Fortunately, an incremental change in the area covered by the Internet standard checksum [RFC1071] will result in a well-defined change to the checksum value [RFC1624]. So, a checksum change caused by modifying part of the area covered by the checksum can be corrected by making a complementary change to a different 16-bit field covered by the same checksum.

ILA can perform a checksum-neutral mapping when a SIR prefix or virtual address is translated to a locator in an IPv6 address, and performs the reverse mapping when translating a locator back to a SIR prefix or virtual address. The low order sixteen bits of the identifier contain the checksum adjustment value for ILA.

On transmission, the translation process is:

- 1) Compute the one's complement difference between the SIR prefix

and the locator. Fold this value to 16 bits (add-with-carry four 16-bit words of the difference).

- 2) If the C-bit is to be used then add-with-carry the bit-wise not of the 0x1000 (i.e. 0xffff) to the value from #1. This compensates the checksum for setting the C-bit.
- 3) Add-with-carry the value from #2 to the low order sixteen bits of the identifier.
- 4) Set the resultant value from #3 in the low order sixteen bits of the identifier and set the C-bit if it is to be present.

Note that the "adjustment" (the 16-bit value set in the identifier) is fixed for a given SIR to locator mapping, so the adjustment value can be saved in an associated data structure for a mapping to avoid computing it for each translation.

On reception of an ILA addressed packet, if checksum-neutral mapping is applied to the packet (either the C-bit is set or its used is assumed for the ILA domain):

- 1) Compute the one's complement difference between the locator in the address and the SIR prefix that the locator is being translated to. Fold this value to 16 bits (add-with-carry four 16-bit words of the difference).
- 2) If the C-bit is used then add-with-carry 0x1000 to the value from #1. This compensates the checksum for clearing the C-bit.
- 3) Add-with-carry the value from #2 to the low order sixteen bits of the identifier.
- 4) Set the resultant value from #3 in the low order sixteen bits of the identifier and clear the C-bit if its present. This restores the original identifier sent in the packet.

Note that receive checksum-neutral mapping process requires that the original upper sixty four bits in the address can be deduced. The method for this is different based on identifier type:

- o interface identifier: checksum-neutral mapping is not used.
- o locally unique identifier: the SIR prefix is inferred from the one to one mapping with a locator.
- o virtual network identifier for IPv4: the original upper sixty-four bits are assumed to be zero.

- o virtual network identifier for IPv6 unicast: the VNID in the identifier is mapped to a tenant prefix that includes the original upper sixty-four bits.
- o virtual network identifier for IPv6 multicast: the original upper sixty-four bits can be deduced by from the scope field in the identifier and fixed field of the multicast address.
- o non-local address identifier: the identifier, not including the low order sixteen bits of the address, is used to lookup the original address. Since the full address is provided by the lookup, the process to undo a checksum-neutral mapping can be obviated in this case

5.4.2 Sending an unmodified checksum

When sending an unmodified checksum, the checksum is incorrect as viewed in the packet on the wire. At the receiver, ILA translation of the destination ILA address back to the SIR address occurs before transport layer processing. This ensures that the checksum can be verified when processing the transport layer header containing the checksum. Intermediate devices are not expected to drop packets due to a bad transport layer checksum.

5.5 Non-local address mapping

Non-local addresses may be mapped into ILA addresses using non-local address identifiers. This allows transit of such addresses across the underlay of an ILA domain. This would be useful for handling addresses in a network that originate from an external source. An example of this would be roaming in cellular network so that a device can continue using addresses that are part of its home network.

A packet may be forwarded to an ILA router that has a non-local destination address which is not a SIR address for the domain. An ILA router can perform a lookup on the full address in an alternate mapping table. If there is a match, an identifier is returned that reverses maps to the address. This identifier is in the ILA domain space and identifies the node with the non-local address. A normal mapping table lookup can then be done to get the locator for the node in the ILA domain.

At a peer ILA router, a lookup is performed on the destination identifier in a table that maps the non-local address identifier to the original non-local address. If an entry is found, the address is set in the destination address and the packet is forward to the destination.

Note that the non-local address to identifier mapping and its reverse mapping must be set in the table before hand.

5.6 Address selection

There may be multiple possibilities for creating either a source or destination address. A node may be associated with more than one identifier, and there may be multiple locators for a particular identifier. The choice of locator or identifier is implementation or configuration specific. The selection of an identifier occurs at flow creation and must be invariant for the duration of the flow. Locator selection must be done at least once per flow, and the locator associated with the destination of a flow may change during the lifetime of the flow (for instance in the case of a migrating connection it will change). ILA address selection should follow specifications in Default Address Selection for Internet Protocol Version 6 (IPv6) [RFC6724].

5.7 Duplicate identifier detection

As part of implementing the locator to identifier mapping, duplicate identifier detection should be implemented in a centralized control plane. A registry of identifiers could be maintained (possibly in association with the identifier to locator mapping database). When a node creates an identifier it registers the identifier, and when the identifier is no longer in use the identifier is unregistered. The control plane should be able to detect a registration attempt for an existing identifier and deny the request.

5.8 ICMP error handling

A packet that contains an ILA address may cause ICMP errors within the network. In this case the ICMP data contains an IP header with an ILA address. ICMP messages are sent back to the source address in the packet. Upon receiving an ICMP error the host will process it differently depending on whether it is ILA capable.

5.8.1 Handling ICMP errors by ILA capable hosts

If a host is ILA capable it can attempt to reverse translate the ILA address in the destination of a header in the ICMP data back to a SIR address that was originally used to transmit the packet. The steps are:

- 1) Assume that the upper sixty-four bits of the destination address in the ICMP data is a locator. Match these bits to a SIR address. If the host is only in one SIR domain, then the mapping to SIR address is implicit. If the host is in multiple

domains then a locator to SIR addresses table can be maintained for this lookup.

- 2) If the identifier includes checksum-neutral mapping, undo the checksum-neutral mapping using the SIR address found in #1 and the process in section 5.4.1. The resulting identifier address is potentially the original address used to send the packet.
- 3) Lookup the identifier in the identifier to locator mapping table. If an entry is found compare the locator in the entry to the locator (upper sixty-four bits) of the destination address in the IP header of the ICMP data. If these match then proceed to next step.
- 4) Overwrite the upper sixty-four bits of the destination address in the ICMP data with the found SIR address and overwrite the low order sixty-four bits with the found identifier (the result of undoing checksum-neutral mapping). The resulting address should be the original SIR address used in sending. The ICMP error packet can then be received by the stack for further processing.

5.8.2 Handling ICMP errors by non-ILA capable hosts

A non-ILA capable host may receive an ICMP error generated by the network that contains an ILA address in IP header contained in the ICMP data. This would happen in the case that an ILA router performed translation on a packet the host sent and that packet subsequently generated an ICMP error. In this case the host receiving the error message will attempt to find the connection state corresponding to the packet header in the ICMP data. Since the host is unaware of ILA the lookup for connection state should fail. Because the host cannot recover the original addresses it used to send the packet, it won't be able any to derive any useful information about the original destination of the packet that it sent.

If packets for a flow are always routed through an ILA router in both directions, for example ILA routers are coincident with edge routes in a network, then ICMP errors could be intercepted by an intermediate node which could translate the destination addresses in ICMP data back to the original SIR addresses. A receiving host would then see the destination address in the packet of the ICMP data to be that it used to transmit the original packet.

5.9 Multicast

ILA is generally not intended for use with multicast. In the case of multicast, routing of packets is based on the source address. Neither

the SIR address nor an ILA address is suitable for use as a source address in a multicast packet. A SIR address is unroutable and hence would make a multicast packet unroutable if used as a source address. Using an ILA address as the source address makes the multicast packet routable, but this exposes ILA address to applications which is especially problematic on a multicast receiver that doesn't support ILA.

If all multicast receivers are known to support ILA, a local locator address may be used in the source address of the multicast packet. In this case, each receiver will translate the source address from an ILA address to a SIR address before delivering packets to an application.

6 Motivation for ILA

6.1 Use cases

6.1.1 Multi-tenant virtualization

In multi-tenant virtualization overlay networks are established for tenants to provide virtual networks. Each tenant may have one or more virtual networks and a tenant's nodes are assigned virtual addresses within virtual networks. Identifier-locator addressing may be used as an alternative to traditional network virtualization encapsulation protocols used to create overlay networks (e.g. VXLAN [RFC7348]).

Tenant systems (e.g. VMs) run on physical hosts and may migrate to different hosts. A tenant system is identified by a virtual address and virtual networking identifier of a corresponding virtual network. ILA can encode the virtual address and a virtual networking identifier in an ILA identifier. Each identifier is mapped to a locator that indicates the current host where the tenant system resides. Nodes that send to the tenant system set the locator per the mapping. When a tenant system migrates its identifier to locator mapping is updated and communicating nodes will use the new mapping.

6.1.2 Datacenter virtualization

Datacenter virtualization virtualizes networking resources. Various objects within a datacenter can be assigned addresses and serve as logical endpoints of communication. A large address space, for example that of IPv6, allows addressing to be used beyond the traditional concepts of host based addressing. Addressed objects can include tasks, virtual IP addresses (VIPs), pieces of content, disk blocks, etc. Each object has a location which is given by the host on which an object resides. Some objects may be migratable between hosts such that their location changes over time.

Objects are identified by a unique identifier within a namespace for the datacenter (appendix B discusses methods to create unique identifiers for ILA). Each identifier is mapped to a locator that indicates the current host where the object resides. Nodes that send to an object set the locator per the mapping. When an object migrates its identifier to locator mapping is updated and communicating nodes will use the new mapping.

A datacenter object of particular interest is tasks, units of execution for applications. The goal of virtualizing tasks is to maximize resource efficiency and job scheduling. Tasks share many properties of tenant systems, however they are finer grained objects, may have a shorter lifetimes, and are likely created in greater numbers. Appendix C provides more detail and motivation for virtualizing tasks using ILA.

6.1.3 Mobile networks

ILA may be applied as a solution for mobility in mobile networks (such as cellular networks). In mobile networks, devices such as smart phones move physically within the network. When a device moves it changes its point of attachment in the network. The goal of mobility is to provide a seamless transition when a device moves from one attachment point to another. Appendix D provides more detail and motivation for ILA in wireless networks.

Each mobile device in a network may be assigned one or more identifiers to use in communications. The ILA mapping table has an entry for each identifier that maps to a locator indicating the current network point of attachment for the device. Nodes that send to the device set the locator per the mapping. When a mobile device moves to a new attachment point, then mapping table entries all of its associated identifiers are updated with a new locator.

6.2 Alternative methods

This section discusses the merits of alternative solution that have been proposed to provide network virtualization or mobility in IPv6.

6.2.1 ILNP

ILNP splits an address into a locator and identifier in the same manner as ILA. ILNP has characteristics, not present in ILA, that prevent it from being a practical solution:

- o ILNP requires that transport layer protocol implementations must be modified to work over ILNP.

- o ILNP can only be implemented in end hosts, not within the network. This essentially requires that all end hosts need to be modified to participate in mobility.

6.2.2 Flow label as virtual network identifier

The IPv6 flow label could conceptually be used as a 20-bit virtual network identifier in order to indicate a packet is sent on an overlay network. In this model the addresses may be virtual addresses within the specified virtual network. Presumably, the tuple of flow-label and addresses could be used by switches to forward virtually addressed packets.

This approach has some issues:

- o Forwarding virtual packets to their physical location would require specialized switch support.
- o The flow label is only twenty bits, this is too small to be a discriminator in forwarding a virtual packet to a specific destination. Conceptually, the flow label might be used in a type of label switching to solve that.
- o The flow label is not considered immutable in transit, intermediate devices may change it.
- o The flow label is not part of the pseudo header for transport checksum calculation, so it is not covered by any transport (or other) checksums.

6.2.3 Extension headers

To accomplish network virtualization an extension header, as a destination or routing option, could be used that contains the virtual destination address of a packet. The destination address in the IPv6 header would be the topological address for the location of the virtual node. Conceivably, segment routing could be used to implement network virtualization in this manner.

This technique has some issues:

- o Intermediate devices must not insert extension headers [RFC8200].
- o Extension headers introduce additional packet overhead which may impact performance.
- o Extension headers are not covered by transport checksums (as the

addresses would be) nor any other checksum.

- o Extension headers are not widely supported in network hardware or devices. For instance, several NIC offloads don't work in the presence of extension headers.

6.2.4 Encapsulation techniques

Various encapsulation techniques have been proposed for implementing network virtualization and mobility. LISP is an example of an encapsulation that is based on locator identifier separation similar to ILA. The primary drawback of encapsulation is complexity and per packet overhead. For, instance when LISP is used with IPv6 the encapsulation overhead is fifty-six bytes and two IP headers are present in every packet. This adds considerable processing costs, requires considerations to handle path MTU correctly, and certain network accelerations may be lost.

7 IANA Considerations

There are no IANA considerations in this specification.

8 References

8.1 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>>.
- [RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, February 2006.
- [RFC6296] Wasserman, M. and F. Baker, "IPv6-to-IPv6 Network Prefix Translation", RFC 6296, June 2011.
- [RFC1071] Braden, R., Borman, D., Partridge, C., and W. Plummer, "Computing the Internet checksum", RFC 1071, September 1988.
- [RFC1624] Rijssinghani, A., "Computation of the Internet Checksum via Incremental Update", RFC 1624, May 1994.
- [RFC6724] Thaler, D., Ed., Draves, R., Matsumoto, A., and T. Chown, "Default Address Selection for Internet Protocol Version 6 (IPv6)", RFC 6724, September 2012.

8.2 Informative References

- [RFC6740] RJ Atkinson and SN Bhatti, "Identifier-Locator Network Protocol (ILNP) Architectural Description", RFC 6740, November 2012.
- [RFC6741] RJ Atkinson and SN Bhatti, "Identifier-Locator Network Protocol (ILNP) Engineering Considerations", RFC 6741, November 2012.
- [RFC1918] Rekhter, Y., Moskowitz, B., Karrenberg, D., de Groot, G., and E. Lear, "Address Allocation for Private Internets", BCP 5, RFC 1918, February 1996.
- [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, August 2002.
- [RFC3587] Hinden, R., Deering, S., and E. Nordmark, "IPv6 Global Unicast Address Format", RFC 3587, August 2003.

- [RFC6144] Baker, F., Li, X., Bao, C., and K. Yin, "Framework for IPv4/IPv6 Translation", RFC 6144, April 2011.
- [RFC8014] Black, D., Hudson, J., Kreeger, L., Lasserre, M., and T. Narten, "An Architecture for Data-Center Network Virtualization over Layer 3 (NVO3)", RFC 8014, DOI 10.17487/RFC8014, December 2016, <<https://www.rfc-editor.org/info/rfc8014>>.
- [GUE] Herbert, T., and Yong, L., "Generic UDP Encapsulation", draft-ietf-intarea-gue-04, work in progress.
- [GUESEC] Yong, L., and Herbert, T. "Generic UDP Encapsulation (GUE) for Secure Transport", draft-hy-gue-4-secure-transport-03, work in progress

9 Acknowledgments

The authors would like to thank Mark Smith, Lucy Yong, Erik Kline, Saleem Bhatti, Blake Matheny, Doug Porter, Pierre Pfister, and Fred Baker for their insightful comments for this draft; Roy Bryant, Lorenzo Colitti, Mahesh Bandewar, and Erik Kline for their work on defining and applying ILA; Kalyani Bogineni, Niranjan Avula, and Ratul Guha for insights regarding the mobility use case.

Appendix A: Communication scenarios

This section describes the use of identifier-locator addressing in several scenarios.

A.1 Terminology for scenario descriptions

A formal notation for identifier-locator addressing with ILNP is described in [RFC6740]. We extend this to include for network virtualization cases.

Basic terms are:

- A = IP Address
- I = Identifier
- L = Locator
- LUI = Locally unique identifier
- VNI = Virtual network identifier
- VA = An IPv4 or IPv6 virtual address
- VAX = An IPv6 networking identifier (IPv6 VA mapped to VAX)
- SIR = Prefix for standard identifier representation
- VNET = IPv6 prefix for a tenant (assumed to be globally routable)
- Iaddr = IPv6 address of an Internet host

An ILA IPv6 address is denoted by

L:I

A SIR address with a locally unique identifier and SIR prefix is denoted by

SIR:LUI

A virtual identifier with a virtual network identifier and a virtual IPv4 address is denoted by

VNI:VA

An ILA IPv6 address with a virtual networking identifier for IPv4 would then be denoted

L:(VNI:VA)

The local and remote address pair in a packet or endpoint is denoted

A,A

An address translation sequence from SIR addresses to ILA addresses

for transmission on the network and back to SIR addresses at a receiver has notation:

A,A -> L:I,A -> A,A

A.2 Identifier objects

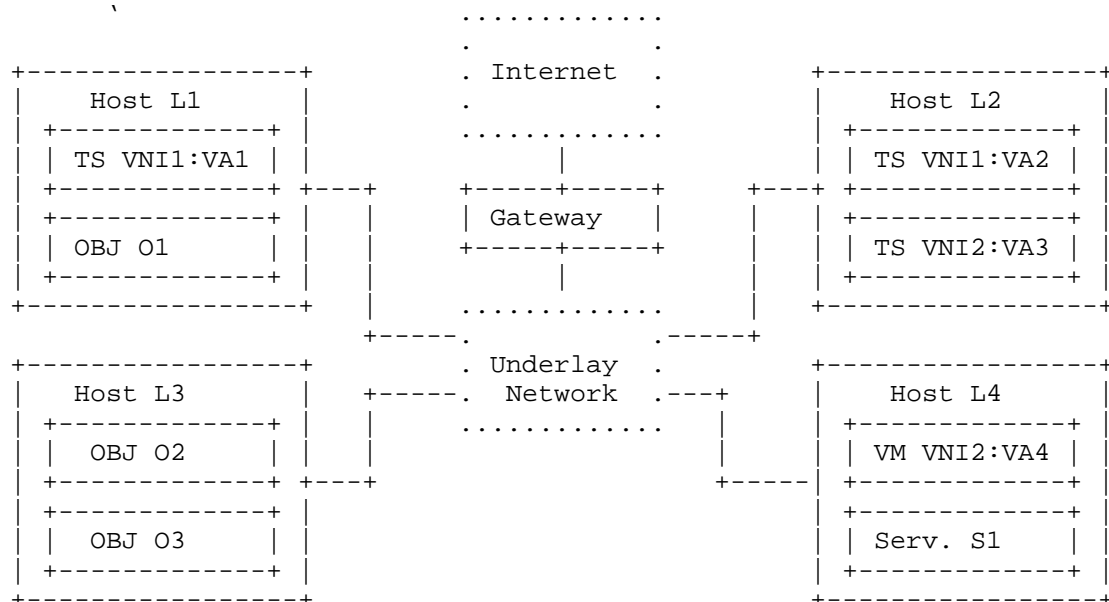
Identifier-locator addressing is broad enough in scope to address many different types of networking entities. For the purposes of this section we classify these as "objects" and "tenant systems".

Objects encompass uses where nodes are address by local unique identifiers (LUI). In the scenarios below objects are denoted by OBJ.

Tenant systems are those associated with network virtualization that have virtual addresses (that is they are addressed by VNI:VA). In the scenarios below tenant systems are denoted by TS.

A.3 Reference network for scenarios

The figure below provides an example network topology with ILA addressing in use. In this example, there are four hosts in the network with locators L1, L2, L3, and L4. There three objects with identifiers O1, O2, and O3, as well as a common networking service with identifier S1. There are two virtual networks VNI1 and VNI2, and four tenant systems addressed as: VA1 and VA2 in VNI1, VA3 and VA4 in VNI2. The network is connected to the Internet via a gateway.



Several communication scenarios can be considered:

- 1) Object to object
- 2) Object to Internet
- 3) Internet to object
- 4) Tenant system to local service
- 5) Object to tenant system
- 6) Tenant system to Internet
- 7) Internet to tenant system
- 8) IPv4 tenant system to service
- 9) Tenant system to tenant system same virtual network using IPv6
- 10) Tenant system to tenant system in same virtual network using IPv4
- 11) Tenant system to tenant system in different virtual network using IPv6
- 12) Tenant system to tenant system in different virtual network using IPv4
- 13) IPv4 tenant system to IPv6 tenant system in different virtual networks
- 14) Non-local address to tenant system

A.4 Scenario 1: Object to task

The transport endpoints for object to object communication are the SIR addresses for the objects. When a packet is sent on the wire, the locator is set in the destination address of the packet. On reception the destination addresses is converted back to SIR representation for processing at the transport layer.

If task T1 is communicating with task T2, the ILA translation sequence would be:

```
SIR:O1,SIR:O2 ->           // Transport endpoints on O1
SIR:O1,L3:O2 ->           // ILA used on the wire
SIR:O1,SIR:O2             // Received at O2
```

A.5 Scenario 2: Object to Internet

Communication from an object to the Internet is accomplished through use of a SIR address (globally routable) in the source address of packets. No ILA translation is needed in this path.

If object O1 is sending to an address Iaddr on the Internet, the packet addresses would be:

```
SIR:O1,Iaddr
```

A.6 Scenario 3: Internet to object

An Internet host transmits a packet to a task using an externally routable SIR address. The SIR prefix routes the packet to a gateway for the datacenter. The gateway translates the destination to an ILA address.

If a host on the Internet with address Iaddr sends a packet to object O3, the ILA translation sequence would be:

```
Iaddr,SIR:O3 ->                // Transport endpoint at Iaddr
Iaddr,L1:O3 ->                 // On the wire in datacenter
Iaddr,SIR:O3                   // Received at O3
```

A.7 Scenario 4: Tenant system to service

A tenant can communicate with a datacenter service using the SIR address of the service.

If TS VA1 is communicating with service S1, the ILA translation sequence would be:

```
VNET:VA1,Saddr->                // Transport endpoints in TS
SIR:(VNET:VA1):Saddr->          // On the wire
SIR:(VNET:VA1):Saddr            // Received at S1
```

Where VNET is the address prefix for the tenant and Saddr is the IPv6 address of the service.

The ILA translation sequence in the reverse path, service to tenant system, would be:

```
Saddr,SIR:(VNET:VA1)            // Transport endpoints in S1
Saddr,L1:(VNET:VA1)             // On the wire
Saddr,VNET:VA1                  // Received at the TS
```

Note that from the point of view of the service task there is no material difference between a peer that is a tenant system versus one which is another task.

A.8 Scenario 5: Object to tenant system

An object can communicate with a tenant system through it's externally visible address.

If object O2 is communicating with TS VA4, the ILA translation sequence would be:

```
SIR:O2,VNET:VA4 ->                // Transport endpoints at T2
SIR:O2,L4:(VNI2:VAX4) ->          // On the wire
```

```
SIR:O2,VNET:VA4
```

```
// Received at TS
```

A.9 Scenario 6: Tenant system to Internet

Communication from a TS to the Internet assumes that the VNET for the TS is globally routable, hence no ILA translation would be needed.

If TS VA4 sends a packet to the Internet, the addresses would be:

```
VNET:VA4,Iaddr
```

A.10 Scenario 7: Internet to tenant system

An Internet host transmits a packet to a tenant system using an externally routable tenant prefix and address. The prefix routes the packet to a gateway for the datacenter. The gateway translates the destination to an ILA address.

If a host on the Internet with address Iaddr is sending to TS VA4, the ILA translation sequence would be:

```
Iaddr,VNET:VA4 ->
```

```
// Endpoint at Iaddr
```

```
Iaddr,L4:(VNI2:VAX4) ->
```

```
// On the wire in datacenter
```

```
Iaddr,VNET:VA4
```

```
// Received at TS
```

A.11 Scenario 8: IPv4 tenant system to object

A TS that is IPv4-only may communicate with an object using protocol translation. The object would be represented as an IPv4 address in the tenant's address space, and stateless NAT64 should be usable as described in [RFC6145].

If TS VA2 communicates with object O3, the ILA translation sequence would be:

```
VA2,ADDR3 ->
```

```
// IPv4 endpoints at TS
```

```
SIR:(VNI1:VA2),L3:O3 ->
```

```
// On the wire in datacenter
```

```
SIR:(VNI1:VA2),SIR:O3
```

```
// Received at task
```

VA2 is the IPv4 address in the tenant's virtual network, ADDR4 is an address in the tenant's address space that maps to the network service.

The reverse path, task sending to a TS with an IPv4 address, requires a similar protocol translation.

For object O3 communicate with TS VA2, the ILA translation sequence would be:

```
SIR:O3,SIR:(VNI1:VA2) ->           // Endpoints at T4
SIR:O3,L2:(VNI1:VA2) ->           // On the wire in datacenter
ADDR4,VA2                          // IPv4 endpoint at TS
```

A.12 Tenant to tenant system in the same virtual network

ILA may be used to allow tenants within a virtual network to communicate without the need for explicit encapsulation headers.

A.12.1 Scenario 9: TS to TS in the same VN using IPV6

If TS VA1 sends a packet to TS VA2, the ILA translation sequence would be:

```
VNET:VA1,VNET:VA2 ->               // Endpoints at VA1
VNET:VA1,L2:(VNI1,VAX2) ->         // On the wire
VNET:VA1,VNET:VA2 ->               // Received at VA2
```

A.12.2 Scenario 10: TS to TS in same VN using IPv4

For two tenant systems to communicate using IPv4 and ILA, IPv4/IPv6 protocol translation is done both on the transmit and receive.

If TS VA1 sends an IPv4 packet to TS VA2, the ILA translation sequence would be:

```
VA1,VA2 ->                         // Endpoints at VA1
SIR:(VNI1:VA1),L2:(VNI1,VA2) ->    // On the wire
VA1,VA2                             // Received at VA2
```

Note that the SIR is chosen by an ILA node as an appropriate SIR prefix in the underlay network. Tenant systems do not use SIR address for this communication, they only use virtual addresses.

A.13 Tenant system to tenant system in different virtual networks

A tenant system may be allowed to communicate with another tenant system in a different virtual network. This should only be allowed with explicit policy configuration.

A.13.1 Scenario 11: TS to TS in different VNs using IPV6

For TS VA4 to communicate with TS VA1 using IPv6 the translation sequence would be:

```
VNET2:VA4,VNET1:VA1->              // Endpoint at VA4
VNET2:VA4,L1:(VNI1,VAX1)->         // On the wire
VNET2:VA4,VNET1:VA1                 // Received at VA1
```

Note that this assumes that VNET1 and VNET2 are globally routable between the two virtual networks.

A.13.2 Scenario 12: TS to TS in different VNs using IPv4

To allow IPv4 tenant systems in different virtual networks to communicate with each other, an address representing the peer would be mapped into each tenant's address space. IPv4/IPv6 protocol translation is done on transmit and receive.

For TS VA4 to communicate with TS VA1 using IPv4 the translation sequence may be:

```
VA4,SADDR1 ->                // IPv4 endpoint at VA4
SIR:(VNI2:VA4),L1:(VNI1,VA1)-> // On the wire
SADDR4,VA1                    // Received at VA1
```

SADDR1 is the mapped address for VA1 in VA4's address space, and SADDR4 is the mapped address for VA4 in VA1's address space.

A.13.3 Scenario 13: IPv4 TS to IPv6 TS in different VNs

Communication may also be mixed so that an IPv4 tenant system can communicate with an IPv6 tenant system in another virtual network. IPv4/IPv6 protocol translation is done on transmit.

For TS VA4 using IPv4 to communicate with TS VA1 using IPv6 the translation sequence may be:

```
VA4,SADDR1 ->                // IPv4 endpoint at VA4
SIR:(VNI2:VA4),L1:(VNI1,VAX1)-> // On the wire
SIR:(VNI2:VA4),VNET1:VA1        // Received at VA1
```

SADDR1 is the mapped IPv4 address for VA1 in VA4's address space.

In the reverse direction, TS VA1 using IPv6 would communicate with TS VA4 with the translation sequence:

```
VNET1:VA1,SIR:(VNI2:VA4)      // Endpoint at VA1
VNET1:VA1,L4:(VNI2:VA4)      // On the wire
SADDR1,VA4                    // Received at VA4
```

A.14 Scenario 14: Non-local address to tenant system

A tenant system may have a global address that is non-local to the network. A host on the Internet or a tenant system may send packet to this address. The packet is forwarded by some means to a gateway or other ILA node (tunneling could be used to accomplish this). An ILA

node can create an ILA address for this using a non-local address identifier.

For a node sending to a non-local address that is an address of task T2, the ILA translation sequence would be:

```
SADDR,A           // Endpoint at a host
SADDR,L3:X        // On the wire
SADDR,A           // Received by TS 2
```

Note that A is the non-local address, and X is an identifier that maps to the non-local address.

Appendix B: unique identifier generation

The unique identifier type of ILA identifiers can address 2^{60} objects (assuming the typed identifier format is used as described in section 4). This appendix describes some method to perform allocation of identifiers for objects to avoid duplicated identifiers being allocated.

B.1 Globally unique identifiers method

For small to moderate sized deployments the technique for creating locally assigned global identifiers described in [RFC4193] could be used. In this technique a SHA-1 digest of the time of day in NTP format and an EUI-64 identifier of the local host is performed. N bits of the result are used as the globally unique identifier.

The probability that two or more of these IDs will collide can be approximated using the formula:

$$P = 1 - \exp(-N^2 / 2^{L+1})$$

where P is the probability of collision, N is the number of identifiers, and L is the length of an identifier.

The following table shows the probability of a collision for a range of identifiers using a 60-bit length.

Identifiers	Probability of Collision
1000	4.3368×10^{-13}
10000	4.3368×10^{-11}
100000	4.3368×10^{-09}
1000000	4.3368×10^{-07}

Note that locally unique identifiers may be ephemeral, for instance a task may only exist for a few seconds. This should be considered when

determining the probability of identifier collision.

B.2 Universally Unique Identifiers method

For larger deployments, hierarchical allocation may be desired. The techniques in Universally Unique Identifier (UUID) URN ([RFC4122]) can be adapted for allocating unique object identifiers in sixty bits. An identifier is split into two components: a registrar prefix and sub-identifier. The registrar prefix defines an identifier block which is managed by an agent, the sub-identifier is a unique value within the registrar block.

For instance, each host in a network could be an agent so that unique identifiers for objects could be created autonomously by the host. The identifier might be composed of a twenty-four bit host identifier followed by a thirty-six bit timestamp. Assuming that a host can allocate up to 100 identifiers per second, this allows about 21.8 years before wrap around.

```

/* LUI identifier with host registrar and timestamp */
|3 bits|1|      24 bits      |              36 bits              |
+-----+-----+-----+-----+
| 0x1  |C| Host identifier |              Timestamp Identifier    |
+-----+-----+-----+-----+

```

Appendix C: Datacenter task virtualization

This section describes some details to apply ILA to virtualizing tasks in a datacenter.

C.1 Address per task

Managing the port number space for services within a datacenter is a nontrivial problem. When a service task is created, it may run on arbitrary hosts. The typical scenario is that the task will be started on some machine and will be assigned a port number for its service. The port number must be chosen dynamically to not conflict with any other port numbers already assigned to tasks on the same machine (possibly even other instances of the same service). A canonical name for the service is entered into a database with the host address and assigned port. When a client wishes to connect to the service, it queries the database with the service name to get both the address of an instance as well as its port number. Note that DNS is not adequate for the service lookup since it does not provide port numbers.

With ILA, each service task can be assigned its own IPv6 address and therefore will logically be assigned the full port space for that

address. This a dramatic simplification since each service can now use a publicly known port number that does not need to be unique between services or instances. A client can perform a lookup on the service name to get an IP address of an instance and then connect to that address using a well known port number. In this case, DNS is sufficient for directing clients to instances of a service.

C.2 Job scheduling

In the usual datacenter model, jobs are scheduled to run as tasks on some number of machines. A distributed job scheduler provides the scheduling which may entail considerable complexity since jobs will often have a variety of resource constraints. The scheduler takes these constraints into account while trying to maximize utility of the datacenter in terms of utilization, cost, latency, etc. Datacenter jobs do not typically run in virtual machines (VMs), but may run within containers. Containers are mechanisms that provide resource isolation between tasks running on the same host OS. These resources can include CPU, disk, memory, and networking.

A fundamental problem arises in that once a task for a job is scheduled on a machine, it often needs to run to completion. If the scheduler needs to schedule a higher priority job or change resource allocations, there may be little recourse but to kill tasks and restart them on a different machine. In killing a task, progress is lost which results in increased latency and wasted CPU cycles. Some tasks may checkpoint progress to minimize the amount of progress lost, but this is not a very transparent or general solution.

An alternative approach is to allow transparent job migration. The scheduler may migrate running jobs from one machine to another.

C.3 Task migration

Under the orchestration of the job scheduler, the steps to migrate a job may be:

- 1) Stop running tasks for the job.
- 2) Package the runtime state of the job. The runtime state is derived from the containers for the jobs.
- 3) Send the runtime state of the job to the new machine where the job is to run.
- 4) Instantiate the job's state on the new machine.
- 5) Start the tasks for the job continuing from the point at which it was stopped.

This model is similar to virtual machine (VM) migration except that the runtime state is typically much less data-- just task state as

opposed to a full OS image. Task state may be compressed to reduce latency in migration.

C.3.1 Address migration

ILA facilitates address (specifically SIR address) migration between hosts as part of task migration or for other purposes. The steps in migrating an address might be:

- 1) Configure address on the target host.
- 2) Suspend use of the address on the old host. This includes handling established connections (see next section). A state may be established to drop packets or send ICMP destination unreachable when packets to the migrated address are received.
- 3) Update the identifier to locator mapping database. Depending on the control plane implementation this may include pushing the new mapping to hosts.
- 4) Communicating hosts will learn of the new mapping via a control plane either by participation in a protocol for mapping propagation or by the ILA resolution protocol.

C.3.2 Connection migration

When a task and its addresses are migrated between machines, the disposition of existing TCP connections needs to be considered.

The simplest course of action is to drop TCP connections across a migration. Since migrations should be relatively rare events, it is conceivable that TCP connections could be automatically closed in the network stack during a migration event. If the applications running are known to handle this gracefully (i.e. reopen dropped connections) then this may be viable.

For seamless migration, open connections may be migrated between hosts. Migration of these entails pausing the connection, packaging connection state and sending to target, instantiating connection state in the peer stack, and restarting the connection. From the time the connection is paused to the time it is running again in the new stack, packets received for the connection should be silently dropped. For some period of time, the old stack will need to keep a record of the migrated connection. If it receives a packet, it should either silently drop the packet or forward it to the new location.

Appendix D: Mobility in wireless networks

ILA can be used in public wireless networks to provide a solution for mobility.

Devices in a carrier network are referred to as User Equipment (UE) and can include smart phones, automobiles, and other IoT devices. UEs attach to provider network at base stations (eNodeB in carrier terminology). As the device moves, it may change its point of attachment to a geographically close based station. A cellular network is composed of cells each of which has an eNodeB.

A node may change cells several times over a time period. In order to provide seamless communications it is desirable that the existing connections of the device are preserved. ILA provides for this by assigning SIR addresses to UEs and deploying ILA routers in the network infrastructure.

In a canonical architecture each base station (eNodeB) would have an ILA router, and there would be a number of ILA routers that serve as gateways between a provider's network and the Internet. When a host on the Internet sends to a UE's SIR address, a gateway ILA router will translate the address. The locator addresses the base station that is the current point of attachment. At the base station ILA router, the destination is translated back to a SIR address and delivered to a UE. A similar process can happen when two UEs in the network communicate.

The wireless network use case is conceptually similar to network virtualization. In both scenarios, nodes have a point of attachment and can move to other points of attachment. The difference is that in network virtualization it is virtual machines that are mobile, in wireless networks it is real devices.

The wireless use case has some unique properties:

- o These are often public networks so that privacy is a consideration. It is likely that devices may have many addresses assigned to promote privacy.
- o A single device might have many identifiers assigned to it. When a device moves, all of the identifiers must change to map to the same locator.
- o Devices move on their own accord so that mobility is unpredictable.
- o There are mostly real humans using devices so that human

identity and exposing geo location are concerns.

Author's Address

Tom Herbert
Quantonium
4701 Patrick Henry Dr.
Santa Clara, CA
EMail: tom@herbertland.com

Petr Lapukhov
1 Hacker Way
Menlo Parck, CA
EMail: petr@fb.com

intarea
Internet-Draft
Intended status: Standards Track
Expires: May 3, 2018

P. Pfister
E. Vyncke, Ed.
Cisco
T. Pauly
D. Schinazi
Apple
M. Keane
Microsoft
October 30, 2017

Discovering Provisioning Domain Names and Data
draft-ietf-intarea-provisioning-domains-00

Abstract

An increasing number of hosts and networks are connected to the Internet through multiple interfaces, some of which may provide multiple ways to access the internet by means of multiple IPv6 prefix configurations.

This document describes a way for hosts to retrieve additional information about their network access characteristics. The set of configuration items required to access the Internet is called a Provisioning Domain (PvD). The PvD is identified by a Fully Qualified Domain Name (FQDN). This identifier, retrieved using a new Router Advertisement (RA) option, is associated with the set of information included within the RA and may later be used to retrieve additional information associated with the PvD by way of an HTTP-over-TLS request.

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 May 3, 2018.

Copyright Notice

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

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<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
2. Terminology	3
3. Provisioning Domain Identification using Router Advertisements	4
3.1. PvD ID Option for Router Advertisements	4
3.2. Router Behavior	5
3.3. Host Behavior	5
3.3.1. DHCPv6 configuration association	6
3.3.2. DHCPv4 configuration association	7
3.3.3. Interconnection Sharing by the Host	7
4. Provisioning Domain Additional Information	7
4.1. Retrieving the PvD Additional Information	7
4.2. Providing the PvD Additional Information	9
4.3. PvD Additional Information Format	9
4.3.1. Connectivity Characteristics Information	10
4.3.2. Private Extensions	11
4.3.3. Example	11
4.4. Detecting misconfiguration and misuse	12
5. Security Considerations	12
6. Privacy Considerations	12
7. IANA Considerations	13
8. Acknowledgements	13
9. Contributor	14
10. References	14
10.1. Normative references	14
10.2. Informative references	15
Appendix A. Changelog	16
A.1. Version 00	16
A.2. Version 01	16
A.3. Version 02	17
A.4. WG Document version 00	18

Appendix B. Connection monetary cost	18
B.1. Conditions	18
B.2. Price	19
B.3. Examples	20
Authors' Addresses	21

1. Introduction

It has become very common in modern networks for hosts to access the network through different network interfaces, tunnels, or next-hop routers. To describe the set of network configurations associated with %% each access method, the concept of Provisioning Domain (PvD) was defined in [RFC7556].

This specification provides a way to identify explicit PvDs with Fully Qualified Domain Names (FQDN). The FQDN is thus called PvD ID in this document. The PvD IDs is included in a Router Advertisement [RFC4861] option. This new option, when present, associates the set of configurations with the PvD ID in the same RA message. It is worth noting that multiple PvDs (with different PvD IDs) could be provisioned on any host interface, as well as noting that the same PvD ID could be used on different interfaces in order to inform the host that all PvDs with the same PvD ID, on different interfaces, ultimately provide identical services.

This document also introduces a way for hosts to retrieve additional information related to a specific PvD by the mean of an HTTP-over-TLS query using an URI derived from the PvD ID. The retrieved JSON object contains additional network information that would typically be considered unfit, or too large, to be directly included in the Router Advertisements. This information can be used by the networking stack, the applications, or even be partially displayed to the users (e.g., by displaying a localized network service name).

2. Terminology

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

In addition, this document uses the following terminology:

PvD: A Provisioning Domain, a set of network configuration information; for more information, see [RFC7556].

PvD ID: A Fully Qualified Domain Name (FQDN) used to identify a PvD.

Explicit PvD: A PvD uniquely identified with a PvD ID. for more information, see [RFC7556].

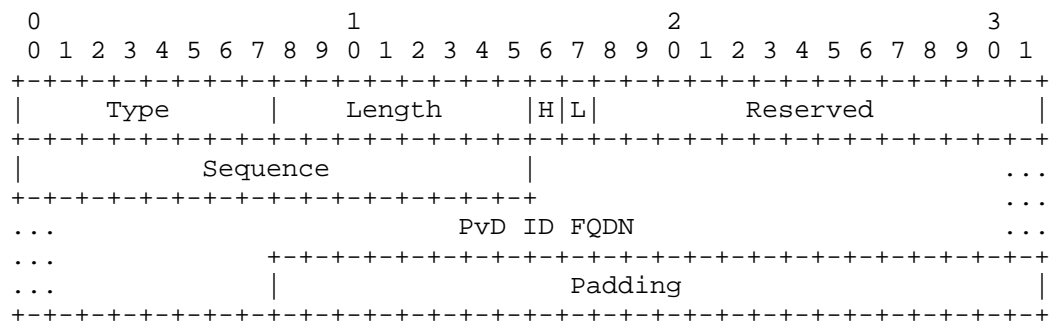
Implicit PvD: A PvD associated with a set of configuration information that, in the absence of a PvD ID, is associated with the advertising router.

3. Provisioning Domain Identification using Router Advertisements

Each provisioning domain is identified by a PvD ID. The PvD ID is a Fully Qualified Domain Name (FQDN) which MUST belong to the network operator in order to avoid ambiguity. The same PvD ID MAY be used in several access networks when the set of configuration information is identical (e.g. in all home networks subscribed to the same service).

3.1. PvD ID Option for Router Advertisements

This document introduces a Router Advertisement (RA) option called the PvD ID Router Advertisement Option, used to convey the FQDN identifying a given PvD.



PvD ID Router Advertisements Option format

Type : (8 bits) To be defined by IANA. Current experimentation uses the value of 253.

Length : (8 bits) The length of the option (including the Type and Length fields) in units of 8 octets.

H-flag : (1 bit) Whether some PvD Additional Information is made available through HTTP over TLS, as described in Section 4.

L-flag : (1 bit) Whether the router is also providing IPv4 information using DHCPv4 (see Section 3.3.2).

Reserved : (14 bits) Reserved for later use. It MUST be set to zero by the sender and ignored by the receiver.

Sequence : (16 bits) Sequence number for the PvD Additional Information, as described in Section 4.

PvD ID FQDN : The FQDN used as PvD ID encoded as described in Section 3.1 of RFC1035 [RFC1035]. Note that for simple decoding, the domain names MUST NOT be encoded in the compressed form described in Section 4.1.4 of RFC1035 [RFC1035]. This encoding is the same as the one used in RFC8106 [RFC8106]. The encoding MUST end with a null (zero-length) label.

Padding : Zero or more padding octets such as to set the option length (Type and Length fields included) to eight times the value of the Length field. It MUST be set to zero by the sender and ignored by the receiver.

Routers MUST NOT include more than one PvD ID Router Advertisement Option in each RA. In case multiple PvD ID options are found in a given RA, hosts MUST ignore all but the first PvD ID option.

3.2. Router Behavior

A router MAY insert only one PvD ID Option in an RA. The included PvD ID is associated with all the other options included in the same RA (for example, and not limited to: Prefix Information [RFC4861], Recursive DNS Server [RFC8106], Routing Information [RFC4191] options).

In order to provide multiple independent PvDs, a router MUST send multiple RAs using different source link-local addresses (LLA) (as proposed in [I-D.bowbakova-rtgwg-enterprise-pa-multihoming]), each of which MAY include a PvD ID option. In such cases, routers MAY originate the different RAs using the same datalink layer address.

If the router is actually a VRRP instance [RFC5798], then the procedure is identical except that the virtual datalink layer address is used as well as the virtual IPv6 LLA.

3.3. Host Behavior

RAs provide configuration information for IPv6 hosts. When a host receives an RA message including a PvD ID Option, it MUST associate all the configuration objects which are updated by the received RA (the same types as in Section 3.3) with the PvD identified by the PvD ID Option, even if some objects are already associated with a different explicit or implicit PvD. PvD ID are compared in a case-

insensitive manner (i.e., A=a), assuming ASCII with zero parity. Non-alphabetic codes must match exactly (see also Section 3.1 of [RFC1035]).

If the received RA does not include a PvD ID Option, the host MUST associate the configuration objects which are updated by the received RA with an implicit PvD, even if some objects were already associated with a different explicit or implicit PvD. This implicit PvD MUST be identified by the LLA of the router sending the RA and the interface on which the RA was received.

This document does not update the way Router Advertisement options are processed. But in addition to the option processing defined in other documents, hosts implementing this specification MUST associate each created or updated object (e.g. address, default route, more specific route, DNS server list) with the PvD associated with the received RA.

While resolving names, executing the default address selection algorithm [RFC6724] or executing the default router selection algorithm ([RFC2461], [RFC4191] and [RFC8028]), hosts MAY consider only the configuration associated with an arbitrary set of PvDs.

For example, a host MAY associate a given process with a specific PvD, or a specific set of PvDs, while associating another process with another PvD. A PvD-aware application might also be able to select, on a per-connection basis, which PvDs should be used for a given connection. In particular, constrained devices such as small battery operated devices (e.g. IoT), or devices with limited CPU or memory resources may purposefully use a single PvD while ignoring some received RAs containing different PvD IDs.

The way an application expresses its desire to use a given PvD, or a set of PvDs, or the way this selection is enforced, is out of the scope of this document. Useful insights about these considerations can be found in [I-D.kline-mif-mpvd-api-reqs].

3.3.1. DHCPv6 configuration association

When a host retrieves configuration elements using DHCPv6, they MUST be associated with the explicit or implicit PvD of the RA received on the same interface, sent from the same LLA, and with the O-flag set [RFC4861]. If no such PvD is found, or whenever multiple different PvDs are found, the host behavior is unspecified.

This process requires hosts to keep track of received RAs, associated PvD IDs, and routers LLA; it also assumes that the router either acts as a DHCPv6 server or relay and uses the same LLA for DHCPv6 and RA

traffic (which may not be the case when the router uses VRRP to send its RA).

3.3.2. DHCPv4 configuration association

When a host retrieves configuration elements from DHCPv4, they MUST be associated with the explicit PvD received on the same interface, whose PVD ID Options L-flag is set and, in the case of a non point-to-point link, using the same datalink address. If no such PvD is found, or whenever multiple different PvDs are found, the configuration elements coming from DHCPv4 MUST be associated with an IPv4-only implicit PvD identified by the interface on which the DHCPv4 transaction happened. The case of multiple explicit PvD for an IPv4 interface is undefined.

3.3.3. Interconnection Sharing by the Host

The situation when a node receives RA on one interface (e.g. cellular) and shares this connectivity by also acting as a router by transmitting RA on another interface (e.g. WiFi) is known as 'tethering'. It can be done as ND proxy. The exact behavior is TBD but it is expected that the one or several PvD associated to the shared interface (e.g. cellular) will also be advertised to the clients on the other interface (e.g. WiFi).

4. Provisioning Domain Additional Information

Once a new PvD ID is discovered, it may be used to retrieve additional information about the characteristics of the provided connectivity. This set of information is called PvD Additional Information, and is encoded as a JSON object [RFC7159].

The purpose of this additional set of information is to securely provide additional information to hosts about the connectivity that is provided using a given interface and source address pair. It typically includes data that would be considered too large, or not critical enough, to be provided within an RA option. The information contained in this object MAY be used by the operating system, network libraries, applications, or users, in order to decide which set of PvDs should be used for which connection, as described in Section 3.3.

4.1. Retrieving the PvD Additional Information

When the H-flag of the PvD ID Option is set, hosts MAY attempt to retrieve the PvD Additional Information associated with a given PvD by performing an HTTP over TLS [RFC2818] GET query to `https://<PvD-`

ID>/well-known/pvd [RFC5785]. Inversely, hosts MUST NOT do so whenever the H-flag is not set.

Note that the DNS name resolution of <PvD-ID> as well as the actual query MUST be performed using the PvD associated with the PvD ID. In other words, the name resolution, source address selection, as well as the next-hop router selection MUST be performed while using exclusively the set of configuration information attached with the PvD, as defined in Section 3.3. In some cases, it may therefore be necessary to wait for an address to be available for use (e.g., once the Duplicate Address Detection or DHCPv6 processes are complete) before initiating the HTTP over TLS query. If the PvD allows for temporary address per [RFC4941], then host SHOULD use a temporary address to fetch the PvD Additional Information and SHOULD deprecate the used temporary address and generate a new temporary address.

If the HTTP status of the answer is greater than or equal to 400 the host MUST abandon and consider that there is no additional PvD information. If the HTTP status of the answer is between 300 and 399, inclusive, it MUST follow the redirection(s). If the HTTP status of the answer is between 200 and 299, inclusive, the host MAY get a file containing a single JSON object. When a JSON object could not be retrieved, an error message SHOULD be logged and/or displayed in a rate-limited fashion.

After retrieval of the PvD Additional Information, hosts MUST watch the PvD ID Sequence field for change. In case a different value than the one in the RA Sequence field is observed, or whenever the validity time included in the PvD Additional Information JSON object is expired, hosts MUST either perform a new query and retrieve a new version of the object, or, failing that, deprecate the object and stop using it.

Hosts retrieving a new PvD Additional Information object MUST check for the presence and validity of the mandatory fields Section 4.3. A retrieved object including an outdated expiration time or missing a mandatory element MUST be ignored. In order to avoid traffic spikes toward the server hosting the PvD Additional Information when an object expires, a host which last retrieved an object at a time A, including a validity time B, SHOULD renew the object at a uniformly random time in the interval $[(B-A)/2, A]$.

The PvD Additional Information object includes a set of IPv6 prefixes (under the key "prefixes") which MUST be checked against all the Prefix Information Options advertised in the RA. If any of the prefixes included in the PIO is not covered by at least one of the listed prefixes, the PvD associated with the tested prefix MUST be considered unsafe and MUST NOT be used. While this does not prevent

a malicious network provider, it does complicate some attack scenarios, and may help detecting misconfiguration.

The server providing the JSON files SHOULD also check whether the client address is part of the prefixes listed into the additional information and SHOULD return a 403 response code if there is no match. The server MAY also use the client address to select the right JSON object to be returned.

4.2. Providing the PvD Additional Information

Whenever the H-flag is set in the PvD RA Option, a valid PvD Additional Information object MUST be made available to all hosts receiving the RA by the network operator. In particular, when a captive portal is present, hosts MUST still be allowed to access the object, even before logging into the captive portal.

Routers MAY increment the PVD ID Sequence number in order to inform host that a new PvD Additional Information object is available and should be retrieved.

4.3. PvD Additional Information Format

The PvD Additional Information is a JSON object.

The following array presents the mandatory keys which MUST be included in the object:

JSON key	Description	Type	Example
name	Human-readable service name	UTF-8 string [RFC3629]	"Awesome Wifi"
expires	Date after which this object is not valid	[RFC3339]	"2017-07-23T06:00:00Z"
prefixes	Array of IPv6 prefixes valid for this PVD	Array of strings	["2001:db8:1::/48", "2001:db8:4::/48"]

A retrieved object which does not include a valid string associated with the "name" key at the root of the object, or a valid date associated with the "expires" key, also at the root of the object, MUST be ignored. In such cases, an error message SHOULD be logged and/or displayed in a rate-limited fashion. If the PIO of the

received RA is not included in the "prefixes" key, the retrieved object SHOULD be ignored.

The following table presents some optional keys which MAY be included in the object.

JSON key	Description	Type	Example
localizedName	Localized user-visible service name, language can be selected based on the HTTP Accept-Language header in the request.	UTF-8 string	"Wifi Genial"
dnsZones	DNS zones searchable and accessible	array of DNS zones	["example.com", "sub.example.org"]
noInternet	No Internet, set when the PvD only provides restricted access to a set of services	boolean	true
characteristics	Connectivity characteristics	JSON object	See Section 4.3.1
metered	metered, when the access volume is limited	boolean	false

It is worth noting that the JSON format allows for extensions. Whenever an unknown key is encountered, it MUST be ignored along with its associated elements.

4.3.1. Connectivity Characteristics Information

The following set of keys can be used to signal certain characteristics of the connection towards the PvD.

They should reflect characteristics of the overall access technology which is not limited to the link the host is connected to, but rather a combination of the link technology, CPE upstream connectivity, and further quality of service considerations.

JSON key	Description	Type	Example
maxThroughput	Maximum achievable throughput	object({down(int), up(int)}) in kbit/s	{"down": 10000, "up": 5000}
minLatency	Minimum achievable latency	object({down(int), up(int)}) in msec	{"down": 10, "up": 20}
rl	Maximum achievable reliability	object({down(int), up(int)}) in losses every 1000 packets	{"down": 0.1, "up": 1}

4.3.2. Private Extensions

JSON keys starting with "x-" are reserved for private use and can be utilized to provide information that is specific to vendor, user or enterprise. It is RECOMMENDED to use one of the patterns "x-FQDN-KEY" or "x-PEN-KEY" where FQDN is a fully qualified domain name or PEN is a private enterprise number [PEN] under control of the author of the extension to avoid collisions.

4.3.3. Example

Here are two examples based on the keys defined in this section.

```
{
  "name": "Foo Wireless",
  "localizedName": "Foo-France Wifi",
  "expires": "2017-07-23T06:00:00Z",
  "prefixes": ["2001:db8:1::/48", "2001:db8:4::/48"],
  "characteristics": {
    "maxThroughput": { "down": 200000, "up": 50000 },
    "minLatency": { "down": 0.1, "up": 1 }
  }
}

{
  "name": "Bar 4G",
  "localizedName": "Bar US 4G",
  "expires": "2017-07-23T06:00:00Z",
  "prefixes": ["2001:db8:1::/48", "2001:db8:4::/48"],
  "metered": true,
  "characteristics": {
    "maxThroughput": { "down": 80000, "up": 20000 }
  }
}
```


4.4. Detecting misconfiguration and misuse

Although some solutions such as IPsec or SEND [RFC3971] can be used in order to secure the IPv6 Neighbor Discovery Protocol, actual deployments largely rely on link layer or physical layer security mechanisms (e.g. 802.1x [IEEE8021X]) in conjunction with RA Guard [RFC6105].

This specification does not improve the Neighbor Discovery Protocol security model, but extends the purely link-local configuration retrieval mechanisms with HTTP-over-TLS communications and some checks to detect misconfiguration and some misuses.

When a host retrieves the PvD Additional Information, it MUST verify that the HTTPS server certificate is valid and that the Subject Name is equal to the PvD ID expressed as an FQDN. This authentication creates a secure binding between the information provided by the trusted Router Advertisement, and the HTTPS server. But this does not mean the Advertising Router and the PvD server belong to the same entity.

When the "prefixes" key is included in the PvD Additional Information, then host MUST verify that all prefixes in the RA PIO are covered by a prefixes from the PvD Additional Information. An adversarial router willing to fake the use of a given explicit PvD, without any access to the actual PvD Additional Information, would need to perform NAT66 in order to circumvent this check.

It is also RECOMMENDED that the HTTPS server checks the source addresses of incoming connections (see Section 4.1). This checks give reasonable assurance that NAT66 was not used and also restrict the information to the valid network users.

5. Security Considerations

It must be noted that the Section 4.4 of this document only provides reasonable assurance against misconfiguration but does not prevent an hostile network access provider to wrong information that could lead applications or hosts to select an hostile PvD. Users should always apply caution when connecting to an unknown network.

6. Privacy Considerations

When a host retrieves via HTTPS the additional information, all nodes on the path (including the HTTPS server) can detect that the node is active.

As it can be expected that the HTTPS server is located in the same management domain as the client (usually, it will be within an enterprise network, WiFi hotspot, or Service Provider network), the network operator as usually other means to also detect the new active node (DHCP, Neighbor Discovery Protocol cache inspection or DNS request logging). In this case, privacy is not worsened by using PvD.

It must also be noted that most operating systems implement a system to detect the presence of a captive portal and also connect to a well-known web site over the Internet, for example to <http://captive.example.com/hotspot-detect.html>. This detection mechanism is exposing the activity of the detecting node not only within the management domain but also to all nodes outside this domain on the path to the captive.example.com server. As PvD can also be used to detect captive portal, then the PvD actually preserves privacy.

Finally, the fetching of additional information is an option and could be disabled by the host.

7. IANA Considerations

IANA is asked to assign the value TBD from the IPv6 Neighbor Discovery Option Formats registry for the PvD ID Router Advertisement option.

IANA is asked to assign the value "pvd" from the Well-Known URIs registry.

IANA is asked to create and maintain a new registry entitled "Additional Information PvD Keys" containing ASCII strings. The initial content of this registry are given below; future assignments are to be made through Expert Review [BCP36].

8. Acknowledgements

Many thanks to M. Stenberg and S. Barth for their earlier work: [I-D.stenberg-mif-mpvd-dns].

Thanks also to Mikael Abrahamson, Ray Bellis, Lorenzo Colitti, Thierry Danis, Bob Hinden, Tatuya Jinmei, Erik Kline, Ted Lemon, Jen Lenkova, Mark Townsley, James Woodyatt for useful and interesting discussions.

Finally, many thanks to Thierry Danis for his implementation work ([github]), Tom Jones for his integration effort into the Neat project and Rigil Salim for his implementation work.

9. Contributor

Basile Bruneau was a co-author of this document while he was studying at the Polytechnique Paris.

10. References

10.1. Normative references

- [I-D.bowbakova-rtgwg-enterprise-pa-multihoming]
Baker, F., Bowers, C., and J. Linkova, "Enterprise Multihoming using Provider-Assigned Addresses without Network Prefix Translation: Requirements and Solution", draft-bowbakova-rtgwg-enterprise-pa-multihoming-01 (work in progress), October 2016.
- [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>>.
- [RFC2461] Narten, T., Nordmark, E., and W. Simpson, "Neighbor Discovery for IP Version 6 (IPv6)", RFC 2461, DOI 10.17487/RFC2461, December 1998, <<https://www.rfc-editor.org/info/rfc2461>>.
- [RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, DOI 10.17487/RFC2818, May 2000, <<https://www.rfc-editor.org/info/rfc2818>>.
- [RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, DOI 10.17487/RFC3629, November 2003, <<https://www.rfc-editor.org/info/rfc3629>>.
- [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>>.
- [RFC7159] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", RFC 7159, DOI 10.17487/RFC7159, March 2014, <<https://www.rfc-editor.org/info/rfc7159>>.

- [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>>.

10.2. Informative references

- [github] Cisco, "IPv6-mPvD github repository", <<https://github.com/IPv6-mPvD>>.
- [I-D.kline-mif-mpvd-api-reqs] Kline, E., "Multiple Provisioning Domains API Requirements", draft-kline-mif-mpvd-api-reqs-00 (work in progress), November 2015.
- [I-D.stenberg-mif-mpvd-dns] Stenberg, M. and S. Barth, "Multiple Provisioning Domains using Domain Name System", draft-stenberg-mif-mpvd-dns-00 (work in progress), October 2015.
- [IEEE8021X] IEEE, "IEEE Standards for Local and Metropolitan Area Networks: Port based Network Access Control, IEEE Std".
- [PEN] IANA, "Private Enterprise Numbers", <<https://www.iana.org/assignments/enterprise-numbers>>.
- [RFC3339] Klyne, G. and C. Newman, "Date and Time on the Internet: Timestamps", RFC 3339, DOI 10.17487/RFC3339, July 2002, <<https://www.rfc-editor.org/info/rfc3339>>.
- [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>>.
- [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>>.
- [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>>.

- [RFC5785] Nottingham, M. and E. Hammer-Lahav, "Defining Well-Known Uniform Resource Identifiers (URIs)", RFC 5785, DOI 10.17487/RFC5785, April 2010, <<https://www.rfc-editor.org/info/rfc5785>>.
- [RFC5798] Nadas, S., Ed., "Virtual Router Redundancy Protocol (VRRP) Version 3 for IPv4 and IPv6", RFC 5798, DOI 10.17487/RFC5798, March 2010, <<https://www.rfc-editor.org/info/rfc5798>>.
- [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>>.
- [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>>.
- [RFC7556] Anipko, D., Ed., "Multiple Provisioning Domain Architecture", RFC 7556, DOI 10.17487/RFC7556, June 2015, <<https://www.rfc-editor.org/info/rfc7556>>.
- [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>>.
- [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>>.

Appendix A. Changelog

Note to RFC Editors: Remove this section before publication.

A.1. Version 00

Initial version of the draft. Edited by Basile Bruneau + Eric Vyncke and based on Basile's work.

A.2. Version 01

Major rewrite intended to focus on the the retained solution based on corridors, online, and WG discussions. Edited by Pierre Pfister. The following list only includes major changes.

PvD ID is an FQDN retrieved using a single RA option. This option contains a sequence number for push-based updates, a new H-flag, and a L-flag in order to link the PvD with the IPv4 DHCP server.

A lifetime is included in the PvD ID option.

Detailed Hosts and Routers specifications.

Additional Information is retrieved using HTTP-over-TLS when the PvD ID Option H-flag is set. Retrieving the object is optional.

The PvD Additional Information object includes a validity date.

DNS-based approach is removed as well as the DNS-based encoding of the PvD Additional Information.

Major cut in the list of proposed JSON keys. This document may be extended later if need be.

Monetary discussion is moved to the appendix.

Clarification about the 'prefixes' contained in the additional information.

Clarification about the processing of DHCPv6.

A.3. Version 02

The FQDN is now encoded with ASCII format (instead of DNS binary) in the RA option.

The PvD ID option lifetime is removed from the object.

Use well known URI "https://<PvD-ID>/.well-known/pvd"

Reference RFC3339 for JSON timestamp format.

The PvD ID Sequence field has been extended to 16 bits.

Modified host behavior for DHCPv4 and DHCPv6.

Removed IKEv2 section.

Removed mention of RFC7710 Captive Portal option. A new I.D. will be proposed to address the captive portal use case.

A.4. WG Document version 00

Document has been accepted as INTAREA working group document

IANA considerations follow RFC8126 [RFC8126]

PvD ID FQDN is encoded as per RFC 1035 [RFC1035]

PvD ID FQDN is prepended by a one-byte length field

Marcus Keane added as co-author

dnsZones key is added back

draft of a privacy consideration section and added that a temporary address should be used to retrieve the PvD additional information

per Bob Hinden's request: the document is now aiming at standard track and security considerations have been moved to the main section

Appendix B. Connection monetary cost

NOTE: This section is included as a request for comment on the potential use and syntax.

The billing of a connection can be done in a lot of different ways. The user can have a global traffic threshold per month, after which his throughput is limited, or after which he/she pays each megabyte. He/she can also have an unlimited access to some websites, or an unlimited access during the weekends.

An option is to split the bill in elementary billings, which have conditions (a start date, an end date, a destination IP address...). The global billing is an ordered list of elementary billings. To know the cost of a transmission, the host goes through the list, and the first elementary billing whose conditions are fulfilled gives the cost. If no elementary billing conditions match the request, the host MUST make no assumption about the cost.

B.1. Conditions

Here are the potential conditions for an elementary billing. All conditions MUST be fulfill.

Key	Description	Type	JSON Example
beginDate	Date before which the billing is not valid	ISO 8601	"1977-04-22T06:00:00Z"
endDate	Date after which the billing is not valid	ISO 8601	"1977-04-22T06:00:00Z"
domains	FQDNs whose the billing is limited	array(string)	["deezer.com", "spotify.com"]
prefixes4	IPv4 prefixes whose the billing is limited	array(string)	["78.40.123.182/32", "78.40.123.183/32"]
prefixes6	IPv6 prefixes whose the billing is limited	array(string)	["2a00:1450:4007:80e::200e/64"]

B.2. Price

Here are the different possibilities for the cost of an elementary billing. A missing key means "all/unlimited/unrestricted". If the elementary billing selected has a trafficRemaining of 0 kb, then it means that the user has no access to the network. Actually, if the last elementary billing has a trafficRemaining parameter, it means that when the user will reach the threshold, he/she will not have access to the network anymore.

Key	Description	Type	JSON Example
pricePerGb	The price per Gigabit	float (currency per Gb)	2
currency	The currency used	ISO 4217	"EUR"
throughputMax	The maximum achievable throughput	float (kb/s)	100000
trafficRemaining	The traffic remaining	float (kB)	12000000

B.3. Examples

Example for a user with 20 GB per month for 40 EUR, then reach a threshold, and with unlimited data during weekends and to example.com:

```
[
  {
    "domains": ["example.com"]
  },
  {
    "prefixes4": ["78.40.123.182/32", "78.40.123.183/32"]
  },
  {
    "beginDate": "2016-07-16T00:00:00Z",
    "endDate": "2016-07-17T23:59:59Z",
  },
  {
    "beginDate": "2016-06-20T00:00:00Z",
    "endDate": "2016-07-19T23:59:59Z",
    "trafficRemaining": 12000000
  },
  {
    "throughputMax": 100000
  }
]
```

If the host tries to download data from example.com, the conditions of the first elementary billing are fulfilled, so the host takes this elementary billing, finds no cost indication in it and so deduces that it is totally free. If the host tries to exchange data with foobar.com and the date is 2016-07-14T19:00:00Z, the conditions of the first, second and third elementary billing are not fulfilled.

But the conditions of the fourth are. So the host takes this elementary billing and sees that there is a threshold, 12 GB are remaining.

Another example for a user abroad, who has 3 GB per year abroad, and then pay each MB:

```
[
  {
    "beginDate": "2016-02-10T00:00:00Z",
    "endDate": "2017-02-09T23:59:59Z",
    "trafficRemaining": 3000000
  },
  {
    "pricePerGb": 30,
    "currency": "EUR"
  }
]
```

Authors' Addresses

Pierre Pfister
Cisco
11 Rue Camille Desmoulins
Issy-les-Moulineaux 92130
France

Email: ppfister@cisco.com

Eric Vyncke (editor)
Cisco
De Kleetlaan, 6
Diegem 1831
Belgium

Email: evyncke@cisco.com

Tommy Pauly
Apple

Email: tpauly@apple.com

David Schinazi
Apple

Email: dschinazi@apple.com

Marcus Keane
Microsoft
Sandyford Industrial Estate
Dublin 18
Ireland

Email: Marcus.Keane@microsoft.com

Transport Area Working Group
Internet-Draft
Updates: 3168, 4341, 4342, 5622, 6679
(if approved)
Intended status: Standards Track
Expires: April 23, 2018

D. Black
Dell EMC
October 20, 2017

Relaxing Restrictions on Explicit Congestion Notification (ECN)
Experimentation
draft-ietf-tsvwg-ecn-experimentation-07

Abstract

This memo updates RFC 3168, which specifies Explicit Congestion Notification (ECN) as an alternative to packet drops for indicating network congestion to endpoints. It relaxes restrictions in RFC 3168 that hinder experimentation towards benefits beyond just removal of loss. This memo summarizes the anticipated areas of experimentation and updates RFC 3168 to enable experimentation in these areas. An Experimental RFC in the IETF document stream is required to take advantage of any of these enabling updates. In addition, this memo makes related updates to the ECN specifications for RTP in RFC 6679 and for DCCP in RFC 4341, RFC 4342 and RFC 5622. This memo also records the conclusion of the ECN nonce experiment in RFC 3540, and provides the rationale for reclassification of RFC 3540 as Historic; this reclassification enables new experimental use of the ECT(1) codepoint.

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 23, 2018.

Copyright Notice

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

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

This document may contain material from IETF Documents or IETF Contributions published or made publicly available before November 10, 2008. The person(s) controlling the copyright in some of this material may not have granted the IETF Trust the right to allow modifications of such material outside the IETF Standards Process. Without obtaining an adequate license from the person(s) controlling the copyright in such materials, this document may not be modified outside the IETF Standards Process, and derivative works of it may not be created outside the IETF Standards Process, except to format it for publication as an RFC or to translate it into languages other than English.

Table of Contents

1. Introduction	3
1.1. ECN Terminology	3
1.2. Requirements Language	4
2. ECN Experimentation: Overview	4
2.1. Effective Congestion Control is Required	5
2.2. Considerations for Other Protocols	5
2.3. Operational and Management Considerations	6
3. ECN Nonce and RFC 3540	7
4. Updates to RFC 3168	8
4.1. Congestion Response Differences	8
4.2. Congestion Marking Differences	9
4.3. TCP Control Packets and Retransmissions	12
5. ECN for RTP Updates to RFC 6679	13
6. ECN for DCCP Updates to RFCs 4341, 4342 and 5622	14
7. Acknowledgements	15
8. IANA Considerations	15
9. Security Considerations	16
10. References	16
10.1. Normative References	16

10.2. Informative References	17
Author's Address	21

1. Introduction

This memo updates RFC 3168 [RFC3168] which specifies Explicit Congestion Notification (ECN) as an alternative to packet drops for indicating network congestion to endpoints. It relaxes restrictions in RFC 3168 that hinder experimentation towards benefits beyond just removal of loss. This memo summarizes the proposed areas of experimentation and updates RFC 3168 to enable experimentation in these areas. An Experimental RFC in the IETF document stream [RFC4844] is required to take advantage of any of these enabling updates. Putting all of these updates into a single document enables experimentation to proceed without requiring a standards process exception for each Experimental RFC that needs changes to RFC 3168, a Proposed Standard RFC.

There is no need for this memo to update RFC 3168 to simplify standardization of protocols and mechanisms that are documented in Standards Track RFCs, as any Standards Track RFC can update RFC 3168 directly without either relying on updates in this memo or using a standards process exception.

In addition, this memo makes related updates to the ECN specification for RTP [RFC6679] and for three DCCP profiles ([RFC4341], [RFC4342] and [RFC5622]) for the same reason. Each experiment is still required to be documented in one or more separate RFCs, but use of Experimental RFCs for this purpose does not require a process exception to modify any of these Proposed Standard RFCs when the modification falls within the bounds established by this memo (RFC 5622 is an Experimental RFC; it is modified by this memo for consistency with modifications to the other two DCCP RFCs).

Some of the anticipated experimentation includes use of the ECT(1) codepoint that was dedicated to the ECN nonce experiment in RFC 3540 [RFC3540]. This memo records the conclusion of the ECN nonce experiment and provides the explanation for reclassification of RFC 3540 as Historic in order to enable new experimental use of the ECT(1) codepoint.

1.1. ECN Terminology

ECT: ECN-Capable Transport. One of the two codepoints ECT(0) or ECT(1) in the ECN field [RFC3168] of the IP header (v4 or v6). An ECN-capable sender sets one of these to indicate that both transport end-points support ECN.

Not-ECT: The ECN codepoint set by senders that indicates that the transport is not ECN-capable.

CE: Congestion Experienced. The ECN codepoint that an intermediate node sets to indicate congestion. A node sets an increasing proportion of ECT packets to CE as the level of congestion increases.

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 RFC 2119 [RFC2119].

2. ECN Experimentation: Overview

Three areas of ECN experimentation are covered by this memo; the cited Internet-Drafts should be consulted for the detailed goals and rationale of each proposed experiment:

Congestion Response Differences: An ECN congestion indication communicates a higher likelihood that a shorter queue exists at the network bottleneck node by comparison to a longer queue that is more likely when a packet drop occurs that indicates congestion [I-D.ietf-tcpm-alternativebackoff-ecn]. This difference suggests that for congestion indicated by ECN, a different sender congestion response (e.g., sender backs off by a smaller amount) may be appropriate by comparison to the sender response to congestion indicated by loss. Two examples of proposed sender congestion response changes are described in [I-D.ietf-tcpm-alternativebackoff-ecn] and [I-D.ietf-tsvwg-ecn-l4s-id] - the proposal in the latter draft couples the sender congestion response change to Congestion Marking Differences changes (see next paragraph). This is at variance with RFC 3168's requirement that a sender's congestion control response to ECN congestion indications be the same as to drops. IETF approval, e.g., via an Experimental RFC in the IETF document stream, is required for any sender congestion response used in this area of experimentation. See Section 4.1 for further discussion.

Congestion Marking Differences: Congestion marking at network nodes can be configured to maintain very shallow queues in conjunction with a different sender response to congestion indications (CE marks), e.g., as proposed in [I-D.ietf-tsvwg-ecn-l4s-id]. The traffic involved needs to be identified by the senders to the network nodes in order to avoid damage to other network traffic whose senders do not expect the more frequent congestion marking

used to maintain very shallow queues. Use of different ECN codepoints, specifically ECT(0) and ECT(1), is a promising means of traffic identification for this purpose, but that technique is at variance with RFC 3168's requirement that ECT(0)-marked traffic and ECT(1)-marked traffic not receive different treatment in the network. IETF approval, e.g., via an Experimental RFC in the IETF document stream, is required for any sender congestion response used in this area of experimentation. See Section 4.2 for further discussion.

TCP Control Packets and Retransmissions: RFC 3168 limits the use of ECN with TCP to data packets, excluding retransmissions. With the successful deployment of ECN in large portions of the Internet, there is interest in extending the benefits of ECN to TCP control packets (e.g., SYNs) and retransmitted packets, e.g., as proposed in [I-D.bagnulo-tcpm-generalized-ecn]. This is at variance with RFC 3168's prohibition of use of ECN for TCP control packets and retransmitted packets. See Section 4.3 for further discussion.

The scope of this memo is limited to these three areas of experimentation. This memo expresses no view on the likely outcomes of the proposed experiments and does not specify the experiments in detail. Additional experiments in these areas are possible, e.g., on use of ECN to support deployment of a protocol similar to DCTCP [I-D.ietf-tcpm-dctcp] beyond DCTCP's current applicability that is limited to data center environments. The purpose of this memo is to remove constraints in standards track RFCs that stand in the way of these areas of experimentation.

2.1. Effective Congestion Control is Required

Congestion control remains an important aspect of the Internet architecture [RFC2914]. Any Experimental RFC in the IETF document stream that takes advantage of this memo's updates to any RFC is required to discuss the congestion control implications of the experiment(s) in order to provide assurance that deployment of the experiment(s) does not pose a congestion-based threat to the operation of the Internet.

2.2. Considerations for Other Protocols

ECN is widely deployed in the Internet and is being designed into additional protocols such as TRILL [I-D.ietf-trill-ecn-support]. While the responsibility for coexistence with other protocols and transition from current ECN functionality falls primary upon the designers of experimental changes to ECN, this subsection provides some general guidelines for designers and users of other protocols

that minimize the likelihood of interaction with the areas of ECN experimentation enabled by this memo.

1. RFC 3168's forwarding behavior remains the preferred approach for routers that are not involved in ECN experiments, in particular continuing to treat the ECT(0) and ECT(1) codepoints as equivalent, as specified in Section 4.2 below.
2. The ECN CE codepoint SHOULD NOT be assumed to indicate that the packet would have been dropped if ECN were not in use, as that is not the case for either Congestion Response Differences experiments (see Section 4.1 below) or Congestion Marking Differences experiments (see Section 4.2 below). This is already the case when the ECN field is used for Pre-Congestion Notification (PCN) [RFC6660].
3. Traffic marked with ECT(1) MUST NOT be originated, as specified in Section 4.2 below.
4. ECN may now be used on packets where it has not been used previously, specifically TCP control packets and retransmissions, see Section 4.3 below, and in particular its new requirements for middlebox behavior. In general, any system or protocol that inspects or monitors network traffic SHOULD be prepared to encounter ECN usage on packets and traffic that currently do not use ECN.
5. Requirements for handling of the ECN field by tunnel encapsulation and decapsulation are specified in [RFC6040]. Additional related guidance can be found in [I-D.ietf-tsvwg-ecn-encap-guidelines] and [I-D.ietf-tsvwg-rfc6040update-shim].

2.3. Operational and Management Considerations

Changes in network traffic behavior that result from ECN experimentation are likely to impact network operations and management. Designers of ECN experiments are expected to anticipate possible impacts and consider how they may be dealt with. Specific topics to consider include possible network management changes or extensions, monitoring of the experimental deployment, collection of data for evaluation of the experiment and possible interactions with other protocols, particularly protocols that encapsulate network traffic.

For further discussion, see [RFC5706]; the questions in Appendix A provide a concise survey of some important aspects to consider.

3. ECN Nonce and RFC 3540

As specified in RFC 3168, ECN uses two ECN Capable Transport (ECT) codepoints to indicate that a packet supports ECN, ECT(0) and ECT(1). The second codepoint, ECT(1), is used to support ECN nonce functionality that discourages receivers from exploiting ECN to improve their throughput at the expense of other network users, as specified in Experimental RFC 3540 [RFC3540]. This section explains why RFC 3540 is being reclassified as Historic and makes associated updates to RFC 3168.

While the ECN nonce works as specified, and has been deployed in limited environments, widespread usage in the Internet has not materialized. A study of the ECN behaviour of the top one million web servers using 2014 data [Trammell15] found that after ECN was negotiated, none of the 581,711 IPv4 servers tested were using both ECT codepoints, which would have been a possible sign of ECN nonce usage. Of the 17,028 IPv6 servers tested, 4 set both ECT(0) and ECT(1) on data packets. This might have been evidence of use of the ECN nonce by these 4 servers, but might equally have been due to erroneous re-marking of the ECN field by a middlebox or router.

With the emergence of new experimental functionality that depends on use of the ECT(1) codepoint for other purposes, continuing to reserve that codepoint for the ECN nonce experiment is no longer justified. In addition, other approaches to discouraging receivers from exploiting ECN have emerged, see Appendix B.1 of [I-D.ietf-tsvwg-ecn-l4s-id]. Therefore, in support of ECN experimentation with the ECT(1) codepoint, this memo:

- o Declares that the ECN nonce experiment [RFC3540] has concluded, and notes the absence of widespread deployment.
- o Updates RFC 3168 [RFC3168] to remove discussion of the ECN nonce and use of ECT(1) for that nonce.

The four primary updates to RFC 3168 that remove discussion of the ECN nonce and use of ECT(1) for that nonce are:

1. Remove the paragraph in Section 5 that immediately follows Figure 1; this paragraph discusses the ECN nonce as the motivation for two ECT codepoints.
2. Remove Section 11.2 "A Discussion of the ECN nonce." in its entirety.
3. Remove the last paragraph of Section 12, which states that ECT(1) may be used as part of the implementation of the ECN nonce.

4. Remove the first two paragraphs of Section 20.2, which discuss the ECN nonce and alternatives. No changes are made to the rest of Section 20.2, which discusses alternate uses for the fourth ECN codepoint.

In addition, other less substantive RFC 3168 changes are required to remove all other mentions of the ECN nonce and to remove implications that ECT(1) is intended for use by the ECN nonce; these specific text updates are omitted for brevity.

4. Updates to RFC 3168

The following subsections specify updates to RFC 3168 to enable the three areas of experimentation summarized in Section 2.

4.1. Congestion Response Differences

RFC 3168 specifies that senders respond identically to packet drops and ECN congestion indications. ECN congestion indications are predominately originated by Active Queue Management (AQM) mechanisms in intermediate buffers. AQM mechanisms are usually configured to maintain shorter queue lengths than non-AQM based mechanisms, particularly non-AQM drop-based mechanisms such as tail-drop, as AQM mechanisms indicate congestion before the queue overflows. While the occurrence of loss does not easily enable the receiver to determine if AQM is used, the receipt of an ECN Congestion Experienced (CE) mark conveys a strong likelihood that AQM was used to manage the bottleneck queue. Hence an ECN congestion indication communicates a higher likelihood that a shorter queue exists at the network bottleneck node by comparison to a packet drop that indicates congestion [I-D.ietf-tcpm-alternativebackoff-ecn]. This difference suggests that for congestion indicated by ECN, a different sender congestion response (e.g., sender backs off by a smaller amount) may be appropriate by comparison to the sender response to congestion indicated by loss. However, section 5 of RFC 3168 specifies that:

Upon the receipt by an ECN-Capable transport of a single CE packet, the congestion control algorithms followed at the end-systems MUST be essentially the same as the congestion control response to a *single* dropped packet.

This memo updates this RFC 3168 text to allow the congestion control response (including the TCP Sender's congestion control response) to a CE-marked packet to differ from the response to a dropped packet, provided that the changes from RFC 3168 are documented in an Experimental RFC in the IETF document stream. The specific change to RFC 3168 is to insert the words "unless otherwise specified by an

Experimental RFC in the IETF document stream" at the end of the sentence quoted above.

RFC 4774 [RFC4774] quotes the above text from RFC 3168 as background, but does not impose requirements based on that text. Therefore no update to RFC 4774 is required to enable this area of experimentation.

Section 6.1.2 of RFC 3168 specifies that:

If the sender receives an ECN-Echo (ECE) ACK packet (that is, an ACK packet with the ECN-Echo flag set in the TCP header), then the sender knows that congestion was encountered in the network on the path from the sender to the receiver. The indication of congestion should be treated just as a congestion loss in non-ECN-Capable TCP. That is, the TCP source halves the congestion window "cwnd" and reduces the slow start threshold "ssthresh".

This memo also updates this RFC 3168 text to allow the congestion control response (including the TCP Sender's congestion control response) to a CE-marked packet to differ from the response to a dropped packet, provided that the changes from RFC 3168 are documented in an Experimental RFC in the IETF document stream. The specific change to RFC 3168 is to insert the words "Unless otherwise specified by an Experimental RFC in the IETF document stream" at the beginning of the second sentence quoted above.

4.2. Congestion Marking Differences

Taken to its limit, an AQM algorithm that uses ECN congestion indications can be configured to maintain very shallow queues, thereby reducing network latency by comparison to maintaining a larger queue. Significantly more aggressive sender responses to ECN are needed to make effective use of such very shallow queues; Datacenter TCP (DCTCP) [I-D.ietf-tcpm-dctcp] provides an example. In this case, separate network node treatments are essential, both to prevent the aggressive low latency traffic from starving conventional traffic (if present) and to prevent any conventional traffic disruption to any lower latency service that uses the very shallow queues. Use of different ECN codepoints is a promising means of identifying these two classes of traffic to network nodes, and hence this area of experimentation is based on the use of the ECT(1) codepoint to request ECN congestion marking behavior in the network that differs from ECT(0) counterbalanced by use of a different IETF-approved congestion response to CE marks at the sender, e.g., as proposed in [I-D.ietf-tsvwg-ecn-l4s-id].

Section 5 of RFC 3168 specifies that:

Routers treat the ECT(0) and ECT(1) codepoints as equivalent.

This memo updates RFC 3168 to allow routers to treat the ECT(0) and ECT(1) codepoints differently, provided that the changes from RFC 3168 are documented in an Experimental RFC in the IETF document stream. The specific change to RFC 3168 is to insert the words "unless otherwise specified by an Experimental RFC in the IETF document stream" at the end of the above sentence.

When an AQM is configured to use ECN congestion indications to maintain a very shallow queue, congestion indications are marked on packets that would not have been dropped if ECN was not in use. Section 5 of RFC 3168 specifies that:

For a router, the CE codepoint of an ECN-Capable packet SHOULD only be set if the router would otherwise have dropped the packet as an indication of congestion to the end nodes. When the router's buffer is not yet full and the router is prepared to drop a packet to inform end nodes of incipient congestion, the router should first check to see if the ECT codepoint is set in that packet's IP header. If so, then instead of dropping the packet, the router MAY instead set the CE codepoint in the IP header.

This memo updates RFC 3168 to allow congestion indications that are not equivalent to drops, provided that the changes from RFC 3168 are documented in an Experimental RFC in the IETF document stream. The specific change is to change "For a router," to "Unless otherwise specified by an Experimental RFC in the IETF document stream" at the beginning of the first sentence of the above paragraph.

A larger update to RFC 3168 is necessary to enable sender usage of ECT(1) to request network congestion marking behavior that maintains very shallow queues at network nodes. When using loss as a congestion signal, the number of signals provided should be reduced to a minimum and hence only presence or absence of congestion is communicated. In contrast, ECN can provide a richer signal, e.g., to indicate the current level of congestion, without the disadvantage of a larger number of packet losses. A proposed experiment in this area, Low Latency Low Loss Scalable throughput (L4S) [I-D.ietf-tsvwg-ecn-l4s-id] significantly increases the CE marking probability for ECT(1)-marked traffic in a fashion that would interact badly with existing sender congestion response functionality because that functionality assumes that the network marks ECT packets as frequently as it would drop Not-ECT packets. If network traffic that uses such a conventional sender congestion response were to encounter L4S's increased marking probability (and hence rate) at a network bottleneck queue, the resulting traffic throughput is likely

to be much less than intended for the level of congestion at the bottleneck queue.

This memo updates RFC 3168 to remove that interaction for ECT(1). The specific update to Section 5 of RFC 3168 is to replace the following two paragraphs:

Senders are free to use either the ECT(0) or the ECT(1) codepoint to indicate ECT, on a packet-by-packet basis.

The use of both the two codepoints for ECT, ECT(0) and ECT(1), is motivated primarily by the desire to allow mechanisms for the data sender to verify that network elements are not erasing the CE codepoint, and that data receivers are properly reporting to the sender the receipt of packets with the CE codepoint set, as required by the transport protocol. Guidelines for the senders and receivers to differentiate between the ECT(0) and ECT(1) codepoints will be addressed in separate documents, for each transport protocol. In particular, this document does not address mechanisms for TCP end-nodes to differentiate between the ECT(0) and ECT(1) codepoints. Protocols and senders that only require a single ECT codepoint SHOULD use ECT(0).

with this paragraph:

Protocols and senders MUST use the ECT(0) codepoint to indicate ECT unless otherwise specified by an Experimental RFC in the IETF document stream. Protocols and senders MUST NOT use the ECT(1) codepoint to indicate ECT unless otherwise specified by an Experimental RFC in the IETF document stream. Guidelines for senders and receivers to differentiate between the ECT(0) and ECT(1) codepoints will be addressed in separate documents, for each transport protocol. In particular, this document does not address mechanisms for TCP end-nodes to differentiate between the ECT(0) and ECT(1) codepoints.

Congestion Marking Differences experiments SHOULD modify the network behavior for ECT(1)-marked traffic rather than ECT(0)-marked traffic if network behavior for only one ECT codepoint is modified. Congestion Marking Differences experiments MUST NOT modify the network behavior for ECT(0)-marked traffic in a fashion that requires changes to sender congestion response to obtain desired network behavior. If a Congestion Marking Differences experiment modifies the network behavior for ECT(1)-marked traffic, e.g., CE-marking behavior, in a fashion that requires changes to sender congestion response to obtain desired network behavior, then the Experimental RFC in the IETF document stream for that experiment MUST specify:

- o The sender congestion response to CE marking in the network, and
- o Router behavior changes, or the absence thereof, in forwarding CE-marked packets that are part of the experiment.

In addition, this memo updates RFC 3168 to remove discussion of the ECN nonce, as noted in Section 3 above.

4.3. TCP Control Packets and Retransmissions

With the successful use of ECN for traffic in large portions of the Internet, there is interest in extending the benefits of ECN to TCP control packets (e.g., SYNs) and retransmitted packets, e.g., as proposed by ECN++ [I-D.bagnulo-tcpm-generalized-ecn].

RFC 3168 prohibits use of ECN for TCP control packets and retransmitted packets in a number of places:

- o "To ensure the reliable delivery of the congestion indication of the CE codepoint, an ECT codepoint MUST NOT be set in a packet unless the loss of that packet in the network would be detected by the end nodes and interpreted as an indication of congestion." (Section 5.2)
- o "A host MUST NOT set ECT on SYN or SYN-ACK packets." (Section 6.1.1)
- o "pure acknowledgement packets (e.g., packets that do not contain any accompanying data) MUST be sent with the not-ECT codepoint." (Section 6.1.4)
- o "This document specifies ECN-capable TCP implementations MUST NOT set either ECT codepoint (ECT(0) or ECT(1)) in the IP header for retransmitted data packets, and that the TCP data receiver SHOULD ignore the ECN field on arriving data packets that are outside of the receiver's current window." (Section 6.1.5)
- o "the TCP data sender MUST NOT set either an ECT codepoint or the CWR bit on window probe packets." (Section 6.1.6)

This memo updates RFC 3168 to allow the use of ECT codepoints on SYN and SYN-ACK packets, pure acknowledgement packets, window probe packets and retransmissions of packets that were originally sent with an ECT codepoint, provided that the changes from RFC 3168 are documented in an Experimental RFC in the IETF document stream. The specific change to RFC 3168 is to insert the words "unless otherwise specified by an Experimental RFC in the IETF document stream" at the end of each sentence quoted above.

In addition, beyond requiring TCP senders not to set ECT on TCP control packets and retransmitted packets, RFC 3168 is silent on whether it is appropriate for a network element, e.g. a firewall, to discard such a packet as invalid. For this area of ECN experimentation to be useful, middleboxes ought not to do that, therefore RFC 3168 is updated by adding the following text to the end of Section 6.1.1.1 on Middlebox Issues:

Unless otherwise specified by an Experimental RFC in the IETF document stream, middleboxes SHOULD NOT discard TCP control packets and retransmitted TCP packets solely because the ECN field in the IP header does not contain Not-ECT. An exception to this requirement occurs in responding to an attack that uses ECN codepoints other than Not-ECT. For example, as part of the response, it may be appropriate to drop ECT-marked TCP SYN packets with higher probability than TCP SYN packets marked with not-ECT. Any such exceptional discarding of TCP control packets and retransmitted TCP packets in response to an attack MUST NOT be done routinely in the absence of an attack and SHOULD only be done if it is determined that the use of ECN is contributing to the attack.

5. ECN for RTP Updates to RFC 6679

RFC 6679 [RFC6679] specifies use of ECN for RTP traffic; it allows use of both the ECT(0) and ECT(1) codepoints, and provides the following guidance on use of these codepoints in section 7.3.1 :

The sender SHOULD mark packets as ECT(0) unless the receiver expresses a preference for ECT(1) or for a random ECT value using the "ect" parameter in the "a=ecn-capable-rtp:" attribute.

The Congestion Marking Differences area of experimentation increases the potential consequences of using ECT(1) instead of ECT(0), and hence the above guidance is updated by adding the following two sentences:

Random ECT values MUST NOT be used, as that may expose RTP to differences in network treatment of traffic marked with ECT(1) and ECT(0) and differences in associated endpoint congestion responses. In addition, ECT(0) MUST be used unless otherwise specified in an Experimental RFC in the IETF document stream.

Section 7.3.3 of RFC 6679 specifies RTP's response to receipt of CE marked packets as being identical to the response to dropped packets:

The reception of RTP packets with ECN-CE marks in the IP header is a notification that congestion is being experienced. The default

reaction on the reception of these ECN-CE-marked packets MUST be to provide the congestion control algorithm with a congestion notification that triggers the algorithm to react as if packet loss had occurred. There should be no difference in congestion response if ECN-CE marks or packet drops are detected.

In support of Congestion Response Differences experimentation, this memo updates this text in a fashion similar to RFC 3168 to allow the RTP congestion control response to a CE-marked packet to differ from the response to a dropped packet, provided that the changes from RFC 6679 are documented in an Experimental RFC in the IETF document stream. The specific change to RFC 6679 is to insert the words "Unless otherwise specified by an Experimental RFC in the IETF document stream" and reformat the last two sentences to be subject to that condition, i.e.:

The reception of RTP packets with ECN-CE marks in the IP header is a notification that congestion is being experienced. Unless otherwise specified by an Experimental RFC in the IETF document stream:

- * The default reaction on the reception of these ECN-CE-marked packets MUST be to provide the congestion control algorithm with a congestion notification that triggers the algorithm to react as if packet loss had occurred.
- * There should be no difference in congestion response if ECN-CE marks or packet drops are detected.

The second sentence of the immediately following paragraph in RFC 6679 requires a related update:

Other reactions to ECN-CE may be specified in the future, following IETF Review. Detailed designs of such additional reactions MUST be specified in a Standards Track RFC and be reviewed to ensure they are safe for deployment under any restrictions specified.

The update is to change "Standards Track RFC" to "Standards Track RFC or Experimental RFC in the IETF document stream" for consistency with the first update.

6. ECN for DCCP Updates to RFCs 4341, 4342 and 5622

The specifications of the three DCCP Congestion Control IDs (CCIDs) 2 [RFC4341], 3 [RFC4342] and 4 [RFC5622] contain broadly the same wording as follows:

each DCCP-Data and DCCP-DataAck packet is sent as ECN Capable with either the ECT(0) or the ECT(1) codepoint set.

This memo updates these sentences in each of the three RFCs as follows:

each DCCP-Data and DCCP-DataAck packet is sent as ECN Capable. Unless otherwise specified by an Experimental RFC in the IETF document stream, such DCCP senders MUST set the ECT(0) codepoint.

In support of Congestion Marking Differences experimentation (as noted in Section 3), this memo also updates all three of these RFCs to remove discussion of the ECN nonce. The specific text updates are omitted for brevity.

7. Acknowledgements

The content of this draft, including the specific portions of RFC 3168 that are updated draws heavily from [I-D.khademi-tsvwg-ecn-response], whose authors are gratefully acknowledged. The authors of the Internet Drafts describing the experiments have motivated the production of this memo - their interest in innovation is welcome and heartily acknowledged. Colin Perkins suggested updating RFC 6679 on RTP and provided guidance on where to make the updates.

The draft has been improved as a result of comments from a number of reviewers, including Ben Campbell, Brian Carpenter, Benoit Claise, Spencer Dawkins, Gorrry Fairhurst, Sue Hares, Ingemar Johansson, Naeem Khademi, Mirja Kuehlewind, Karen Nielsen, Hilarie Orman, Eric Rescorla, Adam Roach and Michael Welzl. Bob Briscoe's thorough review of an early version of this memo resulted in numerous improvements including addition of the updates to the DCCP RFCs.

8. IANA Considerations

To reflect the reclassification of RFC 3540 as Historic, IANA is requested to update the Transmission Control Protocol (TCP) Header Flags registry (<https://www.iana.org/assignments/tcp-header-flags/tcp-header-flags.xhtml#tcp-header-flags-1>) to remove the registration of bit 7 as the NS (Nonce Sum) bit and add an annotation to the registry to state that bit 7 was used by Historic RFC 3540 as the NS (Nonce Sum) bit.

9. Security Considerations

As a process memo that only relaxes restrictions on experimentation, there are no protocol security considerations, as security considerations for any experiments that take advantage of the relaxed restrictions are discussed in the Internet-Drafts that propose the experiments.

However, effective congestion control is crucial to the continued operation of the Internet, and hence this memo places the responsibility for not breaking Internet congestion control on the experiments and the experimenters who propose them. This responsibility includes the requirement to discuss congestion control implications in an IETF document stream Experimental RFC for each experiment, as stated in Section 2.1; review of that discussion by the IETF community and the IESG prior to RFC publication is intended to provide assurance that each experiment does not break Internet congestion control.

See Appendix C.1 of [I-D.ietf-tsvwg-ecn-l4s-id] for discussion of alternatives to the ECN nonce.

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>>.
- [RFC2914] Floyd, S., "Congestion Control Principles", BCP 41, RFC 2914, DOI 10.17487/RFC2914, September 2000, <<https://www.rfc-editor.org/info/rfc2914>>.
- [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>>.
- [RFC3540] Spring, N., Wetherall, D., and D. Ely, "Robust Explicit Congestion Notification (ECN) Signaling with Nonces", RFC 3540, DOI 10.17487/RFC3540, June 2003, <<https://www.rfc-editor.org/info/rfc3540>>.

- [RFC4341] Floyd, S. and E. Kohler, "Profile for Datagram Congestion Control Protocol (DCCP) Congestion Control ID 2: TCP-like Congestion Control", RFC 4341, DOI 10.17487/RFC4341, March 2006, <<https://www.rfc-editor.org/info/rfc4341>>.
- [RFC4342] Floyd, S., Kohler, E., and J. Padhye, "Profile for Datagram Congestion Control Protocol (DCCP) Congestion Control ID 3: TCP-Friendly Rate Control (TFRC)", RFC 4342, DOI 10.17487/RFC4342, March 2006, <<https://www.rfc-editor.org/info/rfc4342>>.
- [RFC5622] Floyd, S. and E. Kohler, "Profile for Datagram Congestion Control Protocol (DCCP) Congestion ID 4: TCP-Friendly Rate Control for Small Packets (TFRC-SP)", RFC 5622, DOI 10.17487/RFC5622, August 2009, <<https://www.rfc-editor.org/info/rfc5622>>.
- [RFC6679] Westerlund, M., Johansson, I., Perkins, C., O'Hanlon, P., and K. Carlberg, "Explicit Congestion Notification (ECN) for RTP over UDP", RFC 6679, DOI 10.17487/RFC6679, August 2012, <<https://www.rfc-editor.org/info/rfc6679>>.

10.2. Informative References

- [I-D.bagnulo-tcpm-generalized-ecn]
Bagnulo, M. and B. Briscoe, "ECN++: Adding Explicit Congestion Notification (ECN) to TCP Control Packets", draft-bagnulo-tcpm-generalized-ecn-04 (work in progress), May 2017.
- [I-D.ietf-tcpm-alternativebackoff-ecn]
Khademi, N., Welzl, M., Armitage, G., and G. Fairhurst, "TCP Alternative Backoff with ECN (ABE)", draft-ietf-tcpm-alternativebackoff-ecn-01 (work in progress), May 2017.
- [I-D.ietf-tcpm-dctcp]
Bensley, S., Thaler, D., Balasubramanian, P., Eggert, L., and G. Judd, "Datacenter TCP (DCTCP): TCP Congestion Control for Datacenters", draft-ietf-tcpm-dctcp-10 (work in progress), August 2017.
- [I-D.ietf-trill-ecn-support]
Eastlake, D. and B. Briscoe, "TRILL: ECN (Explicit Congestion Notification) Support", draft-ietf-trill-ecn-support-03 (work in progress), May 2017.

- [I-D.ietf-tsvwg-ecn-encap-guidelines]
Briscoe, B., Kaippallimalil, J., and P. Thaler,
"Guidelines for Adding Congestion Notification to
Protocols that Encapsulate IP", draft-ietf-tsvwg-ecn-
encap-guidelines-09 (work in progress), July 2017.
- [I-D.ietf-tsvwg-ecn-l4s-id]
Schepper, K. and B. Briscoe, "Identifying Modified
Explicit Congestion Notification (ECN) Semantics for
Ultra-Low Queuing Delay", draft-ietf-tsvwg-ecn-l4s-id-00
(work in progress), May 2017.
- [I-D.ietf-tsvwg-rfc6040update-shim]
Briscoe, B., "Propagating Explicit Congestion Notification
Across IP Tunnel Headers Separated by a Shim", draft-ietf-
tsvwg-rfc6040update-shim-04 (work in progress), July 2017.
- [I-D.khademi-tsvwg-ecn-response]
Khademi, N., Welzl, M., Armitage, G., and G. Fairhurst,
"Updating the Explicit Congestion Notification (ECN)
Specification to Allow IETF Experimentation", draft-
khademi-tsvwg-ecn-response-01 (work in progress), July
2016.
- [RFC4774] Floyd, S., "Specifying Alternate Semantics for the
Explicit Congestion Notification (ECN) Field", BCP 124,
RFC 4774, DOI 10.17487/RFC4774, November 2006,
<<https://www.rfc-editor.org/info/rfc4774>>.
- [RFC4844] Daigle, L., Ed. and Internet Architecture Board, "The RFC
Series and RFC Editor", RFC 4844, DOI 10.17487/RFC4844,
July 2007, <<https://www.rfc-editor.org/info/rfc4844>>.
- [RFC5706] Harrington, D., "Guidelines for Considering Operations and
Management of New Protocols and Protocol Extensions",
RFC 5706, DOI 10.17487/RFC5706, November 2009,
<<https://www.rfc-editor.org/info/rfc5706>>.
- [RFC6040] Briscoe, B., "Tunnelling of Explicit Congestion
Notification", RFC 6040, DOI 10.17487/RFC6040, November
2010, <<https://www.rfc-editor.org/info/rfc6040>>.
- [RFC6660] Briscoe, B., Moncaster, T., and M. Menth, "Encoding Three
Pre-Congestion Notification (PCN) States in the IP Header
Using a Single Diffserv Codepoint (DSCP)", RFC 6660,
DOI 10.17487/RFC6660, July 2012,
<<https://www.rfc-editor.org/info/rfc6660>>.

[Trammell15]

Trammell, B., Kuehlewind, M., Boppart, D., Learmonth, I., Fairhurst, G., and R. Scheffenegger, "Enabling Internet-Wide Deployment of Explicit Congestion Notification".

In Proc Passive & Active Measurement (PAM'15) Conference (2015)

Appendix A. Change History

[To be removed before RFC publication.]

Changes from draft-ietf-tsvwg-ecn-experimentation-00 to -01:

- o Add mention of DCTCP as another protocol that could benefit from ECN experimentation (near end of Section 2).

Changes from draft-ietf-tsvwg-ecn-experimentation-01 to -02:

- o Generalize to describe rationale for areas of experimentation, with less focus on individual experiments
- o Add ECN terminology section
- o Change name of "ECT Differences" experimentation area to "Congestion Marking Differences"
- o Add overlooked RFC 3168 modification to Section 4.1
- o Clarify text for Experimental RFC exception to ECT(1) non-usage requirement
- o Add explanation of exception to "SHOULD NOT drop" requirement in 4.3
- o Rework RFC 3540 status change text to provide rationale for a separate status change document that makes RFC 3540 Historic. Don't obsolete RFC 3540.
- o Significant editorial changes based on reviews by Mirja Kuehlewind, Michael Welzl and Bob Briscoe.

Changes from draft-ietf-tsvwg-ecn-experimentation-02 to -03:

- o Remove change history prior to WG adoption.
- o Update L4S draft reference to reflect TSVWG adoption of draft.

- o Change the "SHOULD" for DCCP sender use of ECT(0) to a "MUST" (overlooked in earlier editing).

- o Other minor edits.

Changes from draft-ietf-tsvwg-ecn-experimentation-03 to -04:

- o Change name of "Generalized ECN" experimentation area to "TCP Control Packets and Retransmissions."
- o Add IANA Considerations text to request removal of the registration of the NS bit in the TCP header.

Changes from draft-ietf-tsvwg-ecn-experimentation-04 to -05:

- o Minor editorial changes from Area Director review

Changes from draft-ietf-tsvwg-ecn-experimentation-05 to -06:

- o Add summary of RFC 3168 changes to remove the ECN nonce, and use lower-case "nonce" instead of "Nonce" to match RFC 3168 usage.
- o Add security considerations sentence to indicate that review of Experimental RFCs prior to publication approval is the means to ensure that congestion control is not broken by experiments.
- o Other minor editorial changes from IETF Last Call

Changes from draft-ietf-tsvwg-ecn-experimentation-06 to -07:

- o Change draft title to make scope clear - this only covers relaxing of restrictions on ECN experimentation.
- o Any Experimental RFC that takes advantage of this memo has to be in the IETF document stream.
- o Added sections 2.2 and 2.3 on considerations for other protocols and O&M, relocated discussion of congestion control requirement to section 2.1 from section 4.4
- o Remove text indicating that ECT(1) may be assigned to L4S - the requirement for an Experimental RFC suffices to ensure that coordination with L4S will occur.
- o Improve explanation of attack response exception to not dropping packets "solely because the ECN field in the IP header does not contain Not-ECT" in Section 4.3

- o Fix L4S draft reference for discussion of ECN Nonce alternatives - it's Appendix C.1, not B.1.
- o Numerous additional editorial changes from IESG Evaluation

Author's Address

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

Email: david.black@dell.com

Internet Engineering Task Force
Internet-Draft
Obsoletes: 3662 (if approved)
Updates: 4594 (if approved)
Intended status: Standards Track
Expires: January 1, 2018

R. Bless
Karlsruhe Institute of Technology (KIT)
June 30, 2017

A Lower Effort Per-Hop Behavior (LE PHB)
draft-ietf-tsvwg-le-phb-02

Abstract

This document specifies properties and characteristics of a Lower Effort (LE) per-hop behavior (PHB). The primary objective of this LE PHB is to protect best-effort (BE) traffic (packets forwarded with the default PHB) from LE traffic in congestion situations, i.e., when resources become scarce, best-effort traffic has precedence over LE traffic and may preempt it. There are numerous uses for this PHB, e.g., for background traffic of low precedence, such as bulk data transfers with low priority in time, non time-critical backups, larger software updates, web search engines while gathering information from web servers and so on. This document recommends a standard DSCP value for the LE PHB.

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 January 1, 2018.

Copyright Notice

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

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

This document may contain material from IETF Documents or IETF Contributions published or made publicly available before November 10, 2008. The person(s) controlling the copyright in some of this material may not have granted the IETF Trust the right to allow modifications of such material outside the IETF Standards Process. Without obtaining an adequate license from the person(s) controlling the copyright in such materials, this document may not be modified outside the IETF Standards Process, and derivative works of it may not be created outside the IETF Standards Process, except to format it for publication as an RFC or to translate it into languages other than English.

Table of Contents

1. Introduction	3
1.1. Applicability	3
1.2. Deployment Considerations	5
1.3. Requirements Language	6
2. PHB Description	6
3. Traffic Conditioning Actions	7
4. Recommended DS Codepoint	7
5. Remarking to other DSCPs/PHBs	7
6. Changes to RFC 4594	8
7. IANA Considerations	9
8. Security Considerations	9
9. References	9
9.1. Normative References	9
9.2. Informative References	10
Appendix A. History of the LE PHB	11
Appendix B. Acknowledgments	11
Appendix C. Change History	11
Appendix D. Note to RFC Editor	12
Author's Address	12

1. Introduction

This document defines a Differentiated Services per-hop behavior [RFC2474] called "Lower Effort" (LE) which is intended for traffic of sufficiently low urgency that all other traffic takes precedence over LE traffic in consumption of network link bandwidth. Low urgency traffic has a low priority for timely forwarding, which does not necessarily imply that it is generally of minor importance. From this viewpoint, it can be considered as a network equivalent to a background priority for processes in an operating system. There may or may not be memory (buffer) resources allocated for this type of traffic.

Some networks carry traffic for which delivery is considered optional; that is, packets of this type of traffic ought to consume network resources only when no other traffic is present. Alternatively, the effect of this type of traffic on all other network traffic is strictly limited ("no harm" property). This is distinct from "best-effort" (BE) traffic since the network makes no commitment to deliver LE packets. In contrast, BE traffic receives an implied "good faith" commitment of at least some available network resources. This document proposes a Lower Effort Differentiated Services per-hop behavior (LE PHB) for handling this "optional" traffic in a differentiated services node.

1.1. Applicability

A Lower Effort PHB is applicable for many applications that otherwise use best-effort delivery. More specifically, it is suitable for traffic and services that can tolerate strongly varying throughput for their data flows, especially periods of very low throughput or even starvation (i.e., long interruptions due to significant or even complete packet loss). Therefore, an application sending an LE marked flow must be able to tolerate short or (even very) long interruptions due to the presence of severe congestion conditions during the transmission of the flow. Thus, there should be an expectation that packets of the LE PHB may be excessively delayed or dropped when any other traffic is present. The LE PHB is suitable for sending traffic of low urgency across a Differentiated Services (DS) domain or DS region.

LE traffic SHOULD be congestion controlled. Since LE traffic may be starved completely for a longer period of time, transport protocols or applications (and their related congestion control mechanisms) SHOULD be able to detect and react to such a situation and should resume the transfer as soon as possible. Congestion control is not only useful to let the flows within the LE behavior aggregate adapt to the available bandwidth that may be highly fluctuating, but also

in case that LE traffic is mapped to the default PHB in DS domains that do not support LE.

Use of the LE PHB might assist a network operator in moving certain kinds of traffic or users to off-peak times. Alternatively, or in addition, packets can be designated for the LE PHB when the goal is to protect all other packet traffic from competition with the LE aggregate while not completely banning LE traffic from the network. An LE PHB SHOULD NOT be used for a customer's "normal internet" traffic nor should packets be "downgraded" to the LE PHB instead of being dropped, particularly when the packets are unauthorized traffic. The LE PHB is expected to have applicability in networks that have at least some unused capacity at certain periods.

The LE PHB allows networks to protect themselves from selected types of traffic as a complement to giving preferential treatment to other selected traffic aggregates. LE should not be used for the general case of downgraded traffic, but may be used by design, e.g., to protect an internal network from untrusted external traffic sources. In this case there is no way for attackers to preempt internal (non LE) traffic by flooding. Another use case in this regard is forwarding of multicast traffic from untrusted sources. Multicast forwarding is currently enabled within domains only for specific sources within a domain, but not for sources from anywhere in the Internet. A main problem is that multicast routing creates traffic sources at (mostly) unpredictable branching points within a domain, potentially leading to congestion and packet loss. In case multicast packets from untrusted sources are forwarded as LE traffic, they will not harm traffic from non-LE behavior aggregates. A further related use case is mentioned in [RFC3754]: preliminary forwarding of non-admitted multicast traffic.

There is no intrinsic reason to limit the applicability of the LE PHB to any particular application or type of traffic. It is intended as an additional traffic engineering tool for network administrators. For instance, it can be used to fill protection capacity of transmission links that is otherwise unused. Some network providers keep link utilization below 50% to ensure that all traffic is forwarded without loss after rerouting caused by a link failure. LE marked traffic can utilize the normally unused capacity and will be preempted automatically in case of link failure when 100% of the link capacity is required for all other traffic. Ideally, applications mark their packets as LE traffic, since they know the urgency of flows.

Example uses for the LE PHB:

- o For traffic caused by world-wide web search engines while they gather information from web servers.
- o For software updates or dissemination of new releases of operating systems.
- o For backup traffic or non-time critical synchronization or mirroring traffic.
- o For content distribution transfers between caches.
- o For preloading or prefetching objects from web sites.
- o For Netnews and other "bulk mail" of the Internet.
- o For "downgraded" traffic from some other PHB when this does not violate the operational objectives of the other PHB or the overall network.
- o For multicast traffic from untrusted (e.g., non-local) sources.

1.2. Deployment Considerations

In order to enable LE support, DS nodes typically only need

- o A BA classifier (Behavior Aggregate classifier, see [RFC2475]) that classifies packets according to the LE DSCP
- o A dedicated LE queue
- o A suitable scheduling discipline, e.g., simple priority queueing

Alternatively, implementations may use active queue management mechanisms instead of a dedicated LE queue, e.g., dropping all arriving LE packets when certain queue length or sojourn time thresholds are exceeded.

Internet-wide deployment of the LE PHB is eased by the following properties:

- o No harm to other traffic: since the LE PHB has the lowest forwarding priority it does not consume resources from other PHBs. Deployment across different provider domains with LE support causes no trust issues or attack vectors to existing (non LE) traffic. Thus, providers can trust LE markings from end-systems, i.e., there is no need to police or remark incoming LE traffic.

- o No PHB parameters or configuration of traffic profiles: the LE PHB itself possesses no parameters that need to be set or configured. Similarly, since LE traffic requires no admission or policing, it is not necessary to configure traffic profiles.
- o No traffic conditioning mechanisms: the LE PHB requires no traffic meters, droppers, or shapers. See also Section 3 for further discussion.

DS domains that cannot or do not want to support the LE PHB should be aware that they violate the "no harm" property of LE. DS domains without LE PHB support SHOULD NOT drop LE marked packets, but rather map them to the default PHB and keep the LE DSCP. See also Section 5 for further discussion of forwarding LE traffic with the default PHB instead.

1.3. Requirements Language

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

2. PHB Description

The LE PHB is defined in relation to the default PHB (best-effort). A packet forwarded with the LE PHB SHOULD have lower precedence than packets forwarded with the default PHB, i.e., in case of congestion, LE marked traffic SHOULD be dropped prior to dropping any default PHB traffic. Ideally, LE packets SHOULD be forwarded only if no packet with any other PHB is awaiting transmission.

A straightforward implementation could be a simple priority scheduler serving the default PHB queue with higher priority than the lower-effort PHB queue. Alternative implementations may use scheduling algorithms that assign a very small weight to the LE class. This, however, may sometimes cause better service for LE packets compared to BE packets in cases when the BE share is fully utilized and the LE share not.

If a dedicated LE queue is not available, an active queue management mechanism within a common BE/LE queue could also be used. This could drop all arriving LE packets as soon as certain queue length or sojourn time thresholds are exceeded.

Since congestion control is also useful within the LE traffic class, Explicit Congestion Notification [RFC3168] SHOULD be used for LE packets, too.

3. Traffic Conditioning Actions

If possible, packets SHOULD be pre-marked in DS-aware end systems by applications due to their specific knowledge about the particular precedence of packets. There is no incentive for DS domains to distrust this initial marking, because letting LE traffic enter a DS domain causes no harm. Thus, any policing such as limiting the rate of LE traffic is not necessary at the DS boundary.

As for most other PHBs an initial classification and marking can be also performed at the first DS boundary node according to the DS domain's own policies (e.g., as protection measure against untrusted sources). However, non-LE traffic (e.g., BE traffic) SHOULD NOT be remarked to LE on a regular basis without consent or knowledge of the user. See also remarks with respect to downgrading in Section 1.1.

4. Recommended DS Codepoint

The RECOMMENDED codepoint for the LE PHB is '000010'.

Earlier specifications [RFC4594] recommended to use CS1 as codepoint (as mentioned in [RFC3662]). This is problematic since it may cause a priority inversion in DiffServ domains that treat CS1 as originally proposed in [RFC2474], resulting in forwarding LE packets with higher precedence than BE packets. Existing implementations SHOULD therefore use the unambiguous LE codepoint '000010' whenever possible.

5. Remarking to other DSCPs/PHBs

"DSCP bleaching", i.e., setting the DSCP to '000000' (default PHB) is NOT RECOMMENDED for this PHB. This may cause effects that are in contrast to the original intent in protecting BE traffic from LE traffic (no harm property). In case DS domains do not support the LE PHB, they SHOULD treat LE marked packets with the default PHB instead (by mapping the LE DSCP to the default PHB), but they SHOULD do so without remarking to DSCP '000000'. The reason for this is that later traversed DS domains may then have still the possibility to treat such packets according the LE PHB. However, operators of DS domains that forward LE traffic within the BE aggregate should be aware of the implications, i.e., induced congestion situations and quality-of-service degradation of the original BE traffic. In this case, the LE property of not harming other traffic is no longer fulfilled. In order to limit the impact in such cases, traffic policing of the LE aggregate may be used.

In case LE marked packets are effectively carried within the default PHB (i.e., forwarded as best-effort traffic) they get a better

forwarding treatment than expected. For some applications and services, it is favorable if the transmission is finished earlier than expected. However, in some cases it may be against the original intention of the LE PHB user to strictly send the traffic only if otherwise unused resources are available, i.e., LE traffic may compete with BE traffic for the same resources and thus adversely affect the original BE aggregate. In some cases users want to be sure that their LE marked traffic actually fulfills the "no harm" property.

One possible solution for a clear distinction in such cases would be to use two different codepoints, "LE-min = LE, better treatment allowed", "LE-strict = LE, better treatment NOT allowed". However, since DSCPs are a scarce resource, applications that want to ensure the lower precedence compared to BE traffic SHOULD use additionally a corresponding Lower-than-Best-Effort transport protocol [RFC6297], e.g., LEDBAT [RFC6817].

A DS domain that still uses DSCP CS1 for marking LE traffic (including Low Priority-Data as defined in [RFC4594] or the old definition in [RFC3662]) MUST remark traffic to the LE DSCP '000010' at the egress to the next DS domain. This increases the probability that the DSCP is preserved end-to-end, whereas a CS1 marked packet may be remarked by the default DSCP if the next domain is applying DiffServ-intercon [RFC8100].

6. Changes to RFC 4594

[RFC4594] recommended to use CS1 as codepoint in section 4.10, whereas CS1 was defined in [RFC2474] to have a higher precedence than CS0, i.e., the default PHB. Consequently, DiffServ domains implementing CS1 according to [RFC2474] will cause a priority inversion for LE packets that contradicts with the original purpose of LE. Therefore, every occurrence of the CS1 DSCP is replaced by the LE DSCP.

Changes:

- o The Low-Priority Data row in Figure 3 is updated as follows:

Low-Priority Data	LE	000010	Any flow that has no BW assurance
----------------------	----	--------	--------------------------------------

- o The Low-Priority Data row in Figure 4 is updated as follows:

Low-Priority Data	LE	Not applicable	RFCXXXX	Rate	Yes
----------------------	----	----------------	---------	------	-----

- o Section 4.10: The RECOMMENDED DSCP marking is LE (Lower Effort).
- o [RFC4594] recommended to remark Low-Priority Data to DSCP '000001' inside a DS domain that uses IP precedence marking. By using the herein defined LE DSCP such remarking is not necessary, so even if Low-Priority Data is unsupported (i.e., mapped to the default PHB) the LE DSCP should be kept across the domain as RECOMMENDED in Section 5.

7. IANA Considerations

This document assigns the Differentiated Services Field Codepoint (DSCP) '000010' from the Differentiated Services Field Codepoints (DSCP) registry (<https://www.iana.org/assignments/dscp-registry/dscp-registry.xml>) to the LE PHB. IANA is requested to update the registry as follows:

- o Name: LE
- o Value (Binary): 000010
- o Value (Decimal): 2
- o Reference: [RFC number of this memo]

8. Security Considerations

There are no specific security exposures for this PHB. Since it defines a new class of low forwarding priority, remarking other traffic as LE traffic may lead to quality-of-service degradation of such traffic. Thus, any attacker that is able to modify the DSCP of a packet to LE may carry out a downgrade attack. See the general security considerations in [RFC2474] and [RFC2475].

9. References

9.1. 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, <<http://www.rfc-editor.org/info/rfc2119>>.

- [RFC2474] Nichols, K., Blake, S., Baker, F., and D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", RFC 2474, DOI 10.17487/RFC2474, December 1998, <<http://www.rfc-editor.org/info/rfc2474>>.
- [RFC2475] Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., and W. Weiss, "An Architecture for Differentiated Services", RFC 2475, DOI 10.17487/RFC2475, December 1998, <<http://www.rfc-editor.org/info/rfc2475>>.

9.2. Informative References

- [draft-bless-diffserv-lbe-phb-00]
Bless, R. and K. Wehrle, "A Lower Than Best-Effort Per-Hop Behavior", draft-bless-diffserv-lbe-phb-00 (work in progress), September 1999, <<https://tools.ietf.org/html/draft-bless-diffserv-lbe-phb-00>>.
- [RFC3168] Ramakrishnan, K., Floyd, S., and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", RFC 3168, DOI 10.17487/RFC3168, September 2001, <<http://www.rfc-editor.org/info/rfc3168>>.
- [RFC3662] Bless, R., Nichols, K., and K. Wehrle, "A Lower Effort Per-Domain Behavior (PDB) for Differentiated Services", RFC 3662, DOI 10.17487/RFC3662, December 2003, <<http://www.rfc-editor.org/info/rfc3662>>.
- [RFC3754] Bless, R. and K. Wehrle, "IP Multicast in Differentiated Services (DS) Networks", RFC 3754, DOI 10.17487/RFC3754, April 2004, <<http://www.rfc-editor.org/info/rfc3754>>.
- [RFC4594] Babiarz, J., Chan, K., and F. Baker, "Configuration Guidelines for DiffServ Service Classes", RFC 4594, DOI 10.17487/RFC4594, August 2006, <<http://www.rfc-editor.org/info/rfc4594>>.
- [RFC6297] Welzl, M. and D. Ros, "A Survey of Lower-than-Best-Effort Transport Protocols", RFC 6297, DOI 10.17487/RFC6297, June 2011, <<http://www.rfc-editor.org/info/rfc6297>>.
- [RFC6817] Shalunov, S., Hazel, G., Iyengar, J., and M. Kuehlewind, "Low Extra Delay Background Transport (LEDBAT)", RFC 6817, DOI 10.17487/RFC6817, December 2012, <<http://www.rfc-editor.org/info/rfc6817>>.

[RFC8100] Geib, R., Ed. and D. Black, "Diffserv-Interconnection Classes and Practice", RFC 8100, DOI 10.17487/RFC8100, March 2017, <<http://www.rfc-editor.org/info/rfc8100>>.

Appendix A. History of the LE PHB

A first version of this PHB was suggested by Roland Bless and Klaus Wehrle in 1999 [draft-bless-diffserv-lbe-phb-00]. After some discussion in the DiffServ Working Group Brian Carpenter and Kathie Nichols proposed a bulk handling per-domain behavior and believed a PHB was not necessary. Eventually, Lower Effort was specified as per-domain behavior and finally became [RFC3662]. More detailed information about its history can be found in Section 10 of [RFC3662].

Appendix B. Acknowledgments

Since text is borrowed from earlier Internet-Drafts and RFCs the co-authors of previous specifications are acknowledged here: Kathie Nichols and Klaus Wehrle. David Black and Ruediger Geib provided helpful comments and suggestions.

Appendix C. Change History

This section briefly lists changes between Internet-Draft versions for convenience.

Changes in Version 02:

- o Applied many editorial suggestions from David Black
- o Added Multicast traffic use case
- o Clarified what is required for deployment in section 1.2 (Deployment Considerations)
- o Added text about implementations using AQMs and ECN usage
- o Updated IANA section according to David Black's suggestions
- o Revised text in the security section
- o Changed copyright Notice to pre5378Trust200902

Changes in Version 01:

- o Now obsoletes RFC 3662.

- o Tried to be more precise in section 1.1 (Applicability) according to R. Geib's suggestions, so rephrased several paragraphs. Added text about congestion control
- o Change section 2 (PHB Description) according to R. Geib's suggestions.
- o Added RFC 2119 language to several sentences.
- o Detailed the description of remarking implications and recommendations in Section 5.
- o Added Section 6 to explicitly list changes with respect to RFC 4594, because this document will update it.

Appendix D. Note to RFC Editor

This section lists actions for the RFC editor during final formatting.

- o Please replace the occurrence of RFCXXXX in Section 6 with the assigned RFC number for this document.
- o Delete Appendix C.
- o Delete this section.

Author's Address

Roland Bless
Karlsruhe Institute of Technology (KIT)
Kaiserstr. 12
Karlsruhe 76131
Germany

Phone: +49 721 608 46413
Email: roland.bless@kit.edu

Network Working Group
Internet-Draft
Intended status: Informational
Expires: March 28, 2018

T. Mizrahi
Marvell
J. Fabini
Vienna University of Technology
A. Morton
AT&T Labs
September 24, 2017

Guidelines for Defining Packet Timestamps
draft-mizrahi-intarea-packet-timestamps-01

Abstract

This document specifies guidelines for defining binary packet timestamp formats in networking protocols at various layers. It also presents three recommended timestamp formats. The target audience of this memo includes network protocol designers. It is expected that a new network protocol that requires a packet timestamp will, in most cases, use one of the recommended timestamp formats. If none of the recommended formats fits the protocol requirements, the new protocol specification should specify the format of the packet timestamp according to the guidelines in this document.

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 March 28, 2018.

Copyright Notice

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

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

(<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
2.1. Requirements Language	3
2.2. Abbreviations	3
3. Packet Timestamp Format Specification	3
4. Recommended Timestamp Formats	4
4.1. Using a Recommended Timestamp Format	5
4.2. NTP Timestamp Formats	5
4.2.1. NTP 64-bit Timestamp Format	5
4.2.2. NTP 32-bit Timestamp Format	6
4.3. The PTP Truncated Timestamp Format	8
5. Timestamp Use Cases	9
5.1. Example 1	9
5.2. Example 2	10
6. Packet Timestamp Control Field	10
7. IANA Considerations	11
8. Security Considerations	11
9. Acknowledgments	12
10. References	12
10.1. Normative References	12
10.2. Informative References	12
Authors' Addresses	14

1. Introduction

Timestamps are widely used in network protocols for various purposes, including delay measurement, clock synchronization, and logging or reporting the time of an event.

Timestamps are represented in the RFC series in one of two forms: text-based timestamps, and packet timestamps. Text-based timestamps [RFC3339] are represented as user-friendly strings, and are widely used in the RFC series, for example in information objects and data models, e.g., [RFC5646], [RFC6991], and [RFC7493]. Packet timestamps, on the other hand, are represented by a compact binary field that has a fixed size, and are not intended to have a human-friendly format. Packet timestamps are also very common in the RFC

series, and are used for example for measuring delay and for synchronizing clocks, e.g., [RFC5905], [RFC4656], and [RFC1323].

This memo presents guidelines for defining a packet timestamp format in network protocols. Three recommended timestamp formats are presented. It is expected that a new network protocol that requires a packet timestamp will, in most cases, use one of the recommended timestamp formats. If none of the recommended formats fits the protocol requirements, the new protocol specification should specify the format of the packet timestamp according to the guidelines in this document.

2. Terminology

2.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2.2. Abbreviations

NTP	Network Time Protocol [RFC5905]
PTP	Precision Time Protocol [IEEE1588]

3. Packet Timestamp Format Specification

This memo recommends to use the timestamp formats defined in Section 4. In cases where these timestamp formats do not satisfy the protocol requirements, the timestamp specification should clearly state the reasons for defining a new format. Moreover, it is recommended to derive the new timestamp format from an existing timestamp format, either a timestamp format from this memo, or any other previously defined timestamp format.

This section defines a template for specifying packet timestamp formats. A timestamp format specification **MUST** include the following aspects:

Timestamp field format:

The format of the timestamp field consists of:

- + Size: The number of bits (or octets) used to represent the packet timestamp field.
- + Units: The units used to represent the timestamp.

If the timestamp is comprised of more than one field, the format of each field is specified.

Epoch:

The origin of the timescale used for the timestamp; the moment in time used as a reference for the timestamp value.

Resolution:

The timestamp resolution; the resolution is equal to the timestamp field unit. If the timestamp consists of two or more fields using different time units, then the resolution is the smallest time unit.

Wraparound:

The wraparound period of the timestamp; any further wraparound-related considerations should be described here.

4. Recommended Timestamp Formats

This memo recommends to use one of the timestamp formats specified below.

Clearly, different network protocols may have different requirements and constraints, and consequently may use different timestamp formats. The choice of the specific timestamp format for a given protocol may depend on a various factors. A few examples of factors that may affect the choice of the timestamp format:

- o Timestamp size: while some network protocols may allow a large timestamp fields, in other cases there may be constraints with respect to the timestamp size, affecting the choice of the timestamp format.
- o Resolution: the time resolution is another factor that may directly affect the selected timestamp format. Similarly, the wraparound periodicity of the timestamp may also affect the selected format.
- o Wraparound period: the length of the time interval in which the timestamp is unique may also be an important factor in choosing the timestamp format. Along with the timestamp resolution, these two factors determine the required number of bits in the timestamp.

- o Common format for multiple protocols: if there are two or more network protocols that use timestamps and are often used together in typical systems, using a common timestamp format should be preferred if possible. Specifically, if the network protocol that is being defined typically runs on a PC, then an NTP-based timestamp format may allow easier integration with an NTP-synchronized timer. In contrast, a protocol that is typically deployed on a hardware-based platform, may make better use of a PTP-based timestamp, allowing more efficient integration with a PTP-synchronized timer.

4.1. Using a Recommended Timestamp Format

A specification that uses one of the recommended timestamp formats should specify explicitly that this is a recommended timestamp format, and point to the relevant section in the current memo.

A specification that uses one of the recommended timestamp formats should also include a section on Synchronization Aspects. Any assumptions or requirements related to synchronization should be specified in this section. For example, the synchronization aspects may specify whether nodes populating the timestamps should be synchronized among themselves, and whether the timestamp is measured with respect to a central reference clock such as an NTP server. If time is assumed to be synchronized to a time standard such as UTC or TAI, it should be specified in this section. Further considerations may be discussed in this section, such as required accuracy, or leap second handling.

4.2. NTP Timestamp Formats

4.2.1. NTP 64-bit Timestamp Format

The Network Time Protocol (NTP) 64-bit timestamp format is defined in [RFC5905]. This timestamp format is used in several network protocols, including [RFC6374], [RFC4656], and [RFC5357]. Since this timestamp format is used in NTP, this timestamp format should be preferred in network protocols that are typically deployed in concert with NTP.

The format is presented in this section according to the template defined in Section 3.

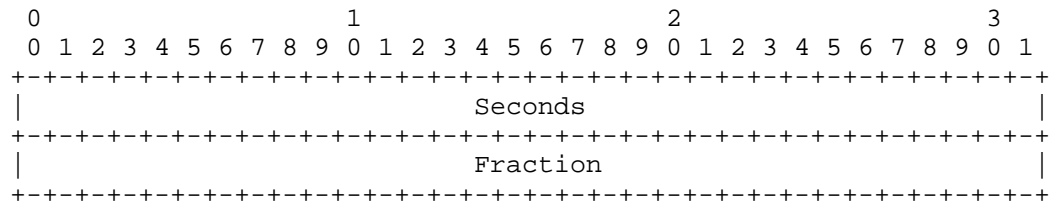


Figure 1: NTP [RFC5905] 64-bit Timestamp Format

Timestamp field format:

Seconds: specifies the integer portion of the number of seconds since the epoch.

+ Size: 32 bits.

+ Units: seconds.

Fraction: specifies the fractional portion of the number of seconds since the epoch.

+ Size: 32 bits.

+ Units: the unit is 2^{-32} seconds, which is roughly equal to 233 picoseconds.

Epoch:

The epoch is 1 January 1900 at 00:00 UTC.

Resolution:

The resolution is 2^{-32} seconds.

Wraparound:

This time format wraps around every 2^{32} seconds, which is roughly 136 years. The next wraparound will occur in the year 2036.

4.2.2. NTP 32-bit Timestamp Format

The Network Time Protocol (NTP) 32-bit timestamp format is defined in [RFC5905]. This timestamp format is used in [I-D.morton-ippm-mbm-registry]. This timestamp format should be preferred in network protocols that are typically deployed in concert with NTP. The 32-bit format can be used either when space

constraints do not allow the use of the 64-bit format, or when the 32-bit format satisfies the resolution and wraparound requirements.

The format is presented in this section according to the template defined in Section 3.

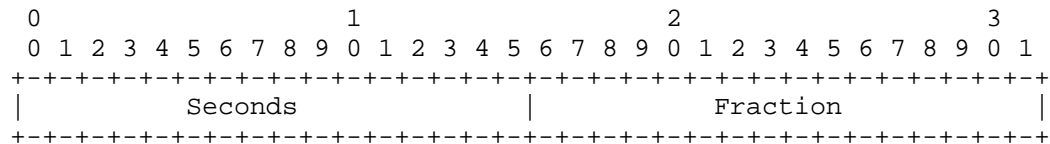


Figure 2: NTP [RFC5905] 32-bit Timestamp Format

Timestamp field format:

Seconds: specifies the integer portion of the number of seconds since the epoch.

+ Size: 16 bits.

+ Units: seconds.

Fraction: specifies the fractional portion of the number of seconds since the epoch.

+ Size: 16 bits.

+ Units: the unit is 2^{-16} seconds, which is roughly equal to 15.3 microseconds.

Epoch:

The epoch is 1 January 1900 at 00:00 UTC.

Resolution:

The resolution is 2^{-16} seconds.

Wraparound:

This time format wraps around every 2^{16} seconds, which is roughly 18 hours.

4.3. The PTP Truncated Timestamp Format

The Precision Time Protocol (PTP) [IEEE1588] uses an 80-bit timestamp format. The truncated timestamp format is a 64-bit field, which is the 64 least significant bits of the 80-bit PTP timestamp. Since this timestamp format is similar to the one used in PTP, this timestamp format should be preferred in network protocols that are typically deployed in PTP-capable devices.

The PTP truncated timestamp format is used in several protocols, such as [RFC6374], [RFC7456], [RFC8186] and [ITU-T-Y.1731].

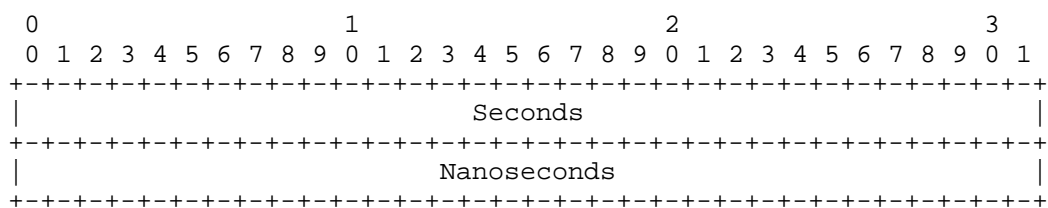


Figure 3: PTP [IEEE1588] Truncated Timestamp Format

Timestamp field format:

Seconds: specifies the integer portion of the number of seconds since the epoch.

+ Size: 32 bits.

+ Units: seconds.

Nanoseconds: specifies the fractional portion of the number of seconds since the epoch.

+ Size: 32 bits.

+ Units: nanoseconds. The value of this field is in the range 0 to $(10^9)-1$.

Epoch:

The PTP [IEEE1588] epoch is 1 January 1970 00:00:00 TAI, which is 31 December 1969 23:59:51.999918 UTC.

Resolution:

The resolution is 1 nanosecond.

Wraparound:

This time format wraps around every 2^{32} seconds, which is roughly 136 years. The next wraparound will occur in the year 2106.

5. Timestamp Use Cases

Packet timestamps are used in various network protocols. Typical applications of packet timestamps include delay measurement, clock synchronization, and others. The following table presents a (non-exhaustive) list of protocols that use packet timestamps, and the timestamp formats used in each of these protocols.

Protocol	Recommended formats			Other format
	NTP 64-bit	NTP 32-bit	PTP Trunc.	
NTP [RFC5905]	+			
OWAMP [RFC4656]	+			
TWAMP [RFC5357]	+			
TWAMP [RFC8186]			+	
TRILL [RFC7456]			+	
MPLS [RFC6374]			+	
TCP [RFC1323]				+
RTP [RFC3550]	+			+

Figure 4: Protocols that use Packet Timestamps

The rest of this section presents two hypothetical examples of network protocol specifications that use one of the recommended timestamp formats. The examples include the text that specifies the information related to the timestamp format.

5.1. Example 1

Timestamp:

The timestamp format used in this specification is the NTP [RFC5905] 64-bit format, as specified in Section 4.2.1 of [I-D.mizrahi-intarea-packet-timestamps].

Synchronization aspects:

It is assumed that nodes that run this protocol are synchronized to UTC using a synchronization mechanism that is outside the scope of this document. In typical deployments this protocol will be run on a machine that uses NTP [RFC5905] for synchronization. Thus, the timestamp may be derived from the NTP-synchronized clock, allowing the timestamp to be measured with respect to the clock of an NTP server.

5.2. Example 2

Timestamp:

The timestamp format used in this specification is the PTP [IEEE1588] Truncated format, as specified in Section 4.2.3 of [I-D.mizrahi-intarea-packet-timestamps].

Synchronization aspects:

It is assumed that nodes that run this protocol are synchronized among themselves. Nodes may be synchronized to a global reference time. Note that if PTP [IEEE1588] is used for synchronization, the timestamp may be derived from the PTP-synchronized clock, allowing the timestamp to be measured with respect to the clock of an PTP Grandmaster clock.

6. Packet Timestamp Control Field

In some cases it is desirable to have a control field that includes information about the timestamp format. This section defines a recommended format of a timestamp-related control field that is intended for network protocols that require such timestamp-related control information.

The recommended control field includes the following sub-fields:

- o Timestamp format.
- o Precision - the resolution or granularity of the system clock.
- o Epoch.

- o Era - the number of times the time has wrapped around since the epoch.

7. IANA Considerations

This memo includes no request to IANA.

8. Security Considerations

A network protocol that uses a packet timestamp MUST specify the security considerations that result from using the timestamp. This section provides an overview of some of the common security considerations of using timestamps.

Any metadata that is attached to control or data packets, and specifically packet timestamps, can facilitate network reconnaissance; by passively eavesdropping to timestamped packets an attacker can gather information about the network performance, and about the level of synchronization between nodes.

Timestamps can be spoofed or modified by on-path attackers, thus attacking the application that uses the timestamps. For example, if timestamps are used in a delay measurement protocol, an attacker can modify en route timestamps in a way that manipulates the measurement results. Integrity protection mechanisms, such as Hashed Message Authentication Codes (HMAC), can mitigate such attacks. The specification of an integrity protection mechanism is outside the scope of this document, as typically integrity protection will be defined on a per-network-protocol basis, and not specifically for the timestamp field.

Another potential threat that can have a similar impact is delay attacks. An attacker can maliciously delay some or all of the en route messages, with the same harmful implications as described in the previous paragraph. Mitigating delay attacks is a significant challenge; in contrast to spoofing and modification attacks, the delay attack cannot be prevented by cryptographic integrity protection mechanisms. In some cases delay attacks can be mitigated by sending the timestamped information through multiple paths, allowing to detect and to be resilient to an attacker that has access to one of the paths.

In many cases timestamping relies on an underlying synchronization mechanism. Thus, any attack that compromises the synchronization mechanism can also compromise protocols that use timestamping. Attacks on time protocols are discussed in detail in [RFC7384].

9. Acknowledgments

The authors thank Yaakov Stein and other members of the TICTOC and NTP working groups for many helpful comments.

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>>.

10.2. Informative References

- [I-D.mizrahi-intarea-packet-timestamps]
Mizrahi, T., Fabini, J., and A. Morton, "Guidelines for Defining Packet Timestamps", draft-mizrahi-intarea-packet-timestamps-00 (work in progress), June 2017.
- [I-D.morton-ippm-mbm-registry]
Morton, A. and M. Mathis, "Initial Performance Metric Registry Entries Part 2: MBM", draft-morton-ippm-mbm-registry-01 (work in progress), March 2017.
- [IEEE1588]
IEEE, "IEEE 1588 Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems Version 2", 2008.
- [ITU-T-Y.1731]
ITU-T, "OAM functions and mechanisms for Ethernet based Networks", 2013.
- [RFC1323] Jacobson, V., Braden, R., and D. Borman, "TCP Extensions for High Performance", RFC 1323, DOI 10.17487/RFC1323, May 1992, <<https://www.rfc-editor.org/info/rfc1323>>.
- [RFC3339] Klyne, G. and C. Newman, "Date and Time on the Internet: Timestamps", RFC 3339, DOI 10.17487/RFC3339, July 2002, <<https://www.rfc-editor.org/info/rfc3339>>.
- [RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, DOI 10.17487/RFC3550, July 2003, <<https://www.rfc-editor.org/info/rfc3550>>.

- [RFC4656] Shalunov, S., Teitelbaum, B., Karp, A., Boote, J., and M. Zekauskas, "A One-way Active Measurement Protocol (OWAMP)", RFC 4656, DOI 10.17487/RFC4656, September 2006, <<https://www.rfc-editor.org/info/rfc4656>>.
- [RFC5357] Hedayat, K., Krzanowski, R., Morton, A., Yum, K., and J. Babiarz, "A Two-Way Active Measurement Protocol (TWAMP)", RFC 5357, DOI 10.17487/RFC5357, October 2008, <<https://www.rfc-editor.org/info/rfc5357>>.
- [RFC5646] Phillips, A., Ed. and M. Davis, Ed., "Tags for Identifying Languages", BCP 47, RFC 5646, DOI 10.17487/RFC5646, September 2009, <<https://www.rfc-editor.org/info/rfc5646>>.
- [RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, DOI 10.17487/RFC5905, June 2010, <<https://www.rfc-editor.org/info/rfc5905>>.
- [RFC6374] Frost, D. and S. Bryant, "Packet Loss and Delay Measurement for MPLS Networks", RFC 6374, DOI 10.17487/RFC6374, September 2011, <<https://www.rfc-editor.org/info/rfc6374>>.
- [RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <<https://www.rfc-editor.org/info/rfc6991>>.
- [RFC7384] Mizrahi, T., "Security Requirements of Time Protocols in Packet Switched Networks", RFC 7384, DOI 10.17487/RFC7384, October 2014, <<https://www.rfc-editor.org/info/rfc7384>>.
- [RFC7456] Mizrahi, T., Senevirathne, T., Salam, S., Kumar, D., and D. Eastlake 3rd, "Loss and Delay Measurement in Transparent Interconnection of Lots of Links (TRILL)", RFC 7456, DOI 10.17487/RFC7456, March 2015, <<https://www.rfc-editor.org/info/rfc7456>>.
- [RFC7493] Bray, T., Ed., "The I-JSON Message Format", RFC 7493, DOI 10.17487/RFC7493, March 2015, <<https://www.rfc-editor.org/info/rfc7493>>.
- [RFC8186] Mirsky, G. and I. Meilik, "Support of the IEEE 1588 Timestamp Format in a Two-Way Active Measurement Protocol (TWAMP)", RFC 8186, DOI 10.17487/RFC8186, June 2017, <<https://www.rfc-editor.org/info/rfc8186>>.

Authors' Addresses

Tal Mizrahi
Marvell
6 Hamada st.
Yokneam
Israel

Email: talmi@marvell.com

Joachim Fabini
Vienna University of Technology
Gusshausstrasse 25/E389
Vienna 1040
Austria

Phone: +43 1 58801 38813
Fax: +43 1 58801 38898
Email: Joachim.Fabini@tuwien.ac.at
URI: <http://www.tc.tuwien.ac.at/about-us/staff/joachim-fabini/>

Al Morton
AT&T Labs
200 Laurel Avenue South
Middletown,, NJ 07748
USA

Phone: +1 732 420 1571
Fax: +1 732 368 1192
Email: acmorton@att.com
URI: <http://home.comcast.net/~acmacm/>

Internet Area Working Group
Internet-Draft
Intended status: Experimental
Expires: May 3, 2018

V. Olteanu
D. Niculescu
University Politehnica of Bucharest
October 30, 2017

SOCKS Protocol Version 6
draft-olteanu-intarea-socks-6-01

Abstract

The SOCKS protocol is used primarily to proxy TCP connections to arbitrary destinations via the use of a proxy server. Under the latest version of the protocol (version 5), it takes 2 RTTs (or 3, if authentication is used) before data can flow between the client and the server.

This memo proposes SOCKS version 6, which reduces the number of RTTs used, takes full advantage of TCP Fast Open, and adds support for 0-RTT authentication.

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 May 3, 2018.

Copyright Notice

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

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<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
1.1. Revision log	3
2. Requirements language	4
3. Mode of operation	4
4. Connection Requests	5
5. Version Mismatch Replies	8
6. Authentication Replies	8
7. Operation Replies	9
7.1. Handling CONNECT	11
7.2. Handling BIND	11
7.3. Handling UDP ASSOCIATE	11
8. SOCKS Options	11
8.1. Authentication options	11
8.2. Idempotence options	12
8.2.1. Requesting a fresh token window	13
8.2.2. Spending a token	14
8.2.3. Handling Token Window Advertisements	16
9. Security Considerations	16
9.1. Large requests	16
9.2. Replay attacks	16
10. IANA Considerations	16
11. Acknowledgements	17
12. References	17
12.1. Normative References	17
12.2. Informative References	17
Authors' Addresses	17

1. Introduction

Versions 4 and 5 [RFC1928] of the SOCKS protocol were developed two decades ago and are in widespread use for circuit level gateways or as circumvention tools, and enjoy wide support and usage from various software, such as web browsers, SSH clients, and proxifiers. However, their design needs an update in order to take advantage of the new features of transport protocols, such as TCP Fast Open [RFC7413], or to better assist newer transport protocols, such as MPTCP [RFC6824].

One of the main issues faced by SOCKS version 5 is that, when taking into account the TCP handshake, method negotiation, authentication, connection request and grant, it may take up to 5 RTTs for a data

exchange to take place at the application layer. This is especially costly in networks with a large delay at the access layer, such as 3G, 4G, or satellite.

The desire to reduce the number of RTTs manifests itself in the design of newer security protocols. TLS version 1.3 [I-D.ietf-tls-tls13] defines a zero round trip (0-RTT) handshake mode for connections if the client and server had previously communicated.

TCP Fast Open [RFC7413] is a TCP option that allows TCP to send data in the SYN and receive a response in the first ACK, and aims at obtaining a data response in one RTT. The SOCKS protocol needs to concern itself with at least two TFO deployment scenarios: First, when TFO is available end-to-end (at the client, at the proxy, and at the server); second, when TFO is active between the client and the proxy, but not at the server.

This document describes the SOCKS protocol version 6. The key improvements over SOCKS version 5 are:

- o The client sends as much information upfront as possible, and does not wait for the authentication process to conclude before requesting the creation of a socket.
- o The connection request also mimics the semantics of TCP Fast Open [RFC7413]. As part of the connection request, the client can supply the payload for the initial SYN that is sent out to the server.
- o The protocol can be extended via options without breaking backward-compatibility.
- o The protocol can leverage the aforementioned options to support 0-RTT authentication schemes.

1.1. Revision log

draft-01

- o Added this section.
- o Support for idempotent commands.
- o Removed version numbers from operation replies.
- o Request port number for SOCKS over TLS. Deprecate encryption/encapsulation within SOCKS.

- o Added Version Mismatch Replies.
- o Renamed the AUTH command to NOOP.
- o Shifted some fields to make requests and operation replies easier to parse.

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 [RFC2119].

3. Mode of operation

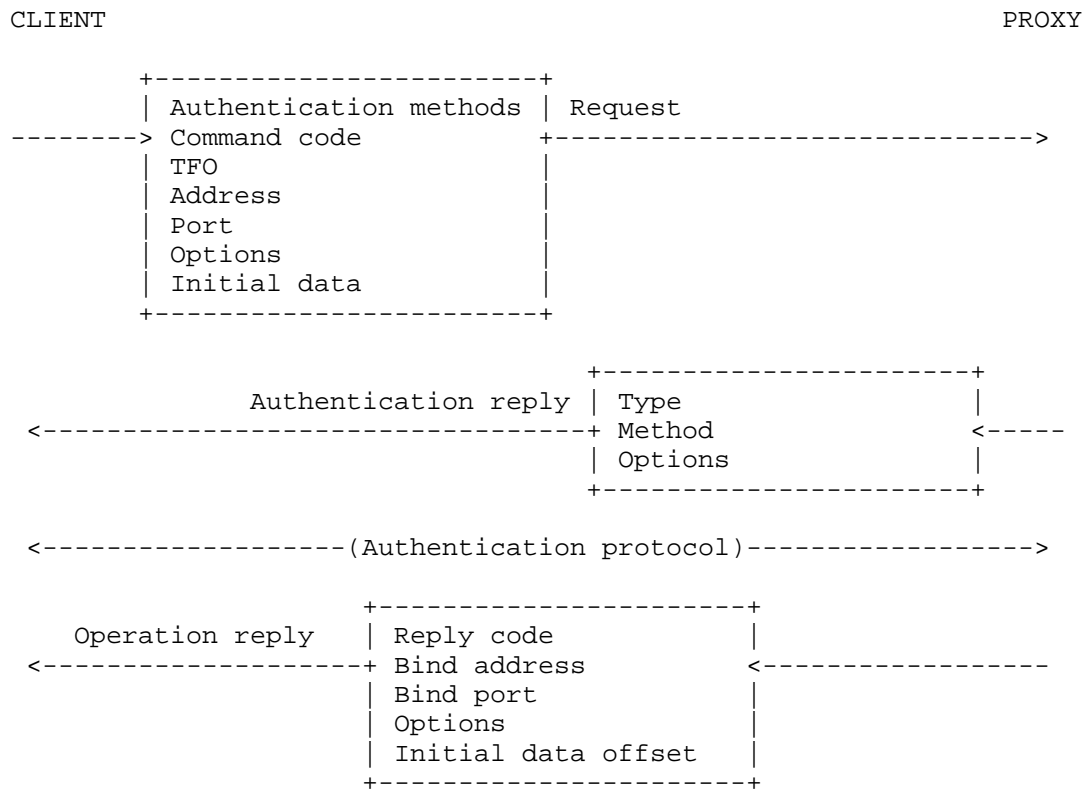


Figure 1: The SOCKS version 6 protocol message exchange

When a TCP-based client wishes to establish a connection to a server, it must open a TCP connection to the appropriate SOCKS port on the SOCKS proxy. The client then enters a negotiation phase, by sending the request in figure Figure 1, that contains, in addition to fields present in SOCKS 5 [RFC1928], fields that facilitate low RTT usage and faster authentication negotiation.

Next, the server sends an authentication reply. If the request did not contain the necessary authentication information, the proxy indicates an authentication method that must proceed. This may trigger a longer authentication sequence that could include tokens for ulterior faster authentications. The part labeled "Authentication protocol" is specific to the authentication method employed and is not expected to be employed for every connection between a client and its proxy server. The authentication protocol typically takes up 1 RTT or more.

If the authentication is successful, an operation reply is generated by the proxy. It indicates whether the proxy was successful in creating the requested socket or not.

In the fast case, when authentication is properly set up, the proxy attempts to create the socket immediately after the receipt of the request, thus achieving an operational connection in one RTT (provided TFO functionality is available at the client, proxy, and server).

4. Connection Requests

The client starts by sending a request to the proxy.

Version		Number of		Methods
Major	Minor	Methods		
1	1	1		Variable
Command		TFO	Address	Port
Code			Type	Address
1	1	1	2	Variable
Number of		Options	Initial Data	
Options			Size	
1		Variable	2	
			Variable	

Figure 2: SOCKS 6 Request

- o Version: The major byte MUST be set to 0x06, and the minor byte MUST be set to 0x00.
- o Number of Methods: The number of supported authentication methods that the client wishes to advertise.
- o Methods: One byte per advertised method. Method numbers are assigned by IANA.
- o Command Code:
 - * 0x00 NOOP: authenticate the client and do nothing.
 - * 0x01 CONNECT: requests the establishment of a TCP connection.
 - * 0x02 BIND: requests the establishment of a TCP port binding.
 - * 0x03 UDP ASSOCIATE: requests a UDP port association.
- o TFO:
 - * 0x00 indicates that the proxy MUST NOT attempt to use TFO in case of a CONNECT command, or accept TFO in case of a BIND command. In case of an AUTH or UDP ASSOCIATE command, this field MUST be set to 0x00.

- * 0x01 indicates that the proxy SHOULD attempt to use TFO in case of a CONNECT command, or accept TFO in case of a BIND command.
- o Address Type:
 - * 0x01: IPv4
 - * 0x03: Domain Name
 - * 0x04: IPv6
- o Address: this field's format depends on the address type:
 - * IPv4: a 4-byte IPv4 address
 - * Domain Name: one byte that contains the length of the FQDN, followed by the FQDN itself. The string is not NUL-terminated.
 - * IPv6: a 16-byte IPv6 address
- o Port: the port in network byte order.
- o Number of Options: the number of SOCKS options that appear in the Options field.
- o Options: see section Section 8.
- o Initial Data Size: A two-byte number in network byte order. In case of AUTH, BIND or UDP ASSOCIATE, this field MUST be set to 0. In case of CONNECT, this is the number of bytes of initial data that are supplied in the following field.
- o Initial Data: The first octets of the data stream.

Clients MUST support the "No authentication required" method. Clients MAY omit advertising the "No authentication required" option.

Clients SHOULD NOT issue AUTH commands unless they advertise authentication methods with support for 0-RTT authentication.

The server MAY truncate the initial data to an arbitrary size and disregard the rest. This is will be communicated later to the client, should the authentication process be successful (see section Section 7). As such, server implementations do not have to buffer the initial data while waiting for the (potentially malicious) client to authenticate.

5. Version Mismatch Replies

Upon receipt of a request starting with a version number other than 6.0, the proxy sends the following response:

Version	
Major	Minor
1	1

Figure 3: SOCKS 6 Version Mismatch Reply

- o Version: The major byte MUST be set to 0x06, and the minor byte MUST be set to 0x00.

A client MUST close the connection after receiving such a reply.

6. Authentication Replies

Upon receipt of a valid request, the proxy sends an Authentication Reply:

Version		Type	Method	Number of Options	Options
Major	Minor				
1	1	1	1	1	Variable

Figure 4: SOCKS 6 Authentication Reply

- o Version: The major byte MUST be set to 0x06, and the minor byte MUST be set to 0x00.
- o Type:
 - * 0x00: authentication successful.
 - * 0x01: further authentication needed.
- o Method: The chosen authentication method.
- o Number of Options: the number of SOCKS options that appear in the Options field.

- o Options: see section Section 8.

Multihomed clients SHOULD cache the chosen method on a per-interface basis and SHOULD NOT include authentication options related to any other methods in further requests originating from the same interface.

If the server signals that further authentication is needed and selects "No Acceptable Methods", the client MUST close the connection.

The client and proxy begin a method-specific negotiation. During such negotiations, the proxy MAY supply information that allows the client to authenticate a future request using an authentication option. Descriptions of such negotiations are beyond the scope of this memo.

If the cliend issued an AUTH command, the client MUST close the connection after the negotiation is complete.

7. Operation Replies

After the authentication negotiations are complete, the server sends an Operation Reply:

Reply Code	Address Type	Bind Port	Bind Address	Initial Data Offset
1	1	2	Variable	2

Number of Options	Options
1	Variable

Figure 5: SOCKS 6 Operation Reply

- o Reply Code:

- * 0x00: Succes
- * 0x01: General SOCKS server failure
- * 0x02: Connection not allowed by ruleset

- * 0x03: Network unreachable
- * 0x04: Host unreachable
- * 0x05: Connection refused
- * 0x06: TTL expired
- * 0x07: Command not supported
- * 0x08: Address type not supported
- o Address Type:
 - * 0x01: IPv4
 - * 0x03: Domain Name
 - * 0x04: IPv6
- o Bind Address: the proxy bound address in the following format:
 - * IPv4: a 4-byte IPv4 address
 - * Domain Name: one byte that contains the length of the FQDN, followed by the FQDN itself. The string is not NUL-terminated.
 - * IPv6: a 16-byte IPv6 address
- o Bind Port: the proxy bound port in network byte order.
- o Number of Options: the number of SOCKS options that appear in the Options field.
- o Options: see section Section 8.
- o Initial Data Offset: A two-byte number in network byte order. In case of BIND or UDP ASSOCIATE, this field MUST be set to 0. In case of CONNECT, it represents the offset in the plain data stream from which the client is expected to continue sending data.

If the proxy returns a reply code other than "Success", the client MUST close the connection.

7.1. Handling CONNECT

In case the client has issued a CONNECT request, data can now pass. The client **MUST** resume the data stream at the offset indicated by the Initial Data Offset field.

7.2. Handling BIND

In case the client has issued a BIND request, it must wait for a second Operation reply from the proxy, which signifies that a host has connected to the bound port. The Bind Address and Bind Port fields contain the address and port of the connecting host. Afterwards, application data may pass.

7.3. Handling UDP ASSOCIATE

The relay of UDP packets is handled exactly as in SOCKS 5 [RFC1928].

8. SOCKS Options

SOCKS options have the following format:

Kind	Length	Option Data
1	1	Variable

Figure 6: SOCKS 6 Option

- o Kind: **MUST** be allocated by IANA. (See section Section 10.)
- o Length: The length of the option.
- o Option Data: The contents are specific to each option kind.

8.1. Authentication options

Authentication options carry method-specific authentication data. They can be part of SOCKS Requests and Authentication Replies.

Authentication options have the following format:

Kind	Length	Method	Authentication Data
1	1	1	Variable

Figure 7: Authentication Option

- o Kind: MUST be allocated by IANA. (See section Section 10.)
- o Length: The length of the option.
- o Method: The number of the authentication method. These numbers are assigned by IANA.
- o Authentication Data: The contents are specific to each method.

All proxy implementations MUST support authentication method options. Clients MAY omit advertising authentication methods for which they have included at least an authentication option.

8.2. Idempotence options

To protect against duplicate SOCKS Requests, authenticated clients can request, and then spend, idempotence tokens. A token can only be spent on a single SOCKS request.

Tokens are 4-byte unsigned integers in a modular 4-byte space. Therefore, if x and y are tokens, x is smaller than y if $(y - x) < 2^{31}$ in unsigned 32-bit arithmetic.

Proxies grant contiguous ranges of tokens called token windows. Token windows are defined by their base (the first token in the range) and size. Windows can be shifted (i. e. have their base increased, while retaining their size) unilaterally by the proxy.

Requesting and spending tokens is done via Idempotence options:

Kind	Length	Type	Option Data
1	1	1	Variable

Figure 8: Idempotence Option

- o Kind: MUST be allocated by IANA. (See section Section 10.)
- o Length: The length of the option.
- o Type:
 - * 0x00: Token Request
 - * 0x01: Token Window Advertisement
 - * 0x02: Token Expenditure
 - * 0x03: Token Expenditure Reply
- o Option Data: The contents are specific to each type.

8.2.1. Requesting a fresh token window

A client can obtain a fresh window of tokens by sending a Token Request option as part of a SOCKS Request:

Kind	Length	Type	Window Size
1	1	1	4

Figure 9: Token Request

- o Kind: MUST be allocated by IANA. (See section Section 10.)
- o Length: 7
- o Type: 0x00 (Token Request)
- o Window Size: The requested window size.

The proxy then includes a Token Window Advertisement option in the corresponding Operation Reply:

Kind	Length	Type	Window Base	Window Size
1	1	1	4	4

Figure 10: Token Window Advertisement

- o Kind: MUST be allocated by IANA. (See section Section 10.)
- o Length: 11
- o Type: 0x01 (Token Grant)
- o Window Base: The first token in the window.
- o Window Size: The window size. This value SHOULD be lower or equal to the requested window size.

8.2.2. Spending a token

The client can attempt to spend a token by including a Token Expenditure option in its SOCKS request:

Kind	Length	Type	Token
1	1	1	4

Figure 11: Token Expenditure

- o Kind: MUST be allocated by IANA. (See section Section 10.)
- o Length: 7
- o Type: 0x02 (Token Expenditure)
- o Token: The token being spent.

Clients SHOULD prioritize spending the smaller tokens.

The server responds by sending a Token Expenditure Reply option as part of the Operation Reply:

Kind	Length	Type	Response Code
1	1	1	1

Figure 12: Token Expenditure Response

- o Kind: MUST be allocated by IANA. (See section Section 10.)
- o Length: 4
- o Type: 0x03 (Token Expenditure Response)
- o Response Code:
 - * 0x00: Success: The token was spent successfully.
 - * 0x01: No Window: The proxy does not have a token window associated with the client.
 - * 0x02: Out of Window: The token is not within the window.
 - * 0x03: Duplicate: The token has already been spent.

If eligible, the token is spent as soon as the client authenticates. If the token is not eligible for spending, the proxy MUST NOT attempt to honor the client's SOCKS Request; further, it MUST indicate a General SOCKS server failure in the Operation Reply.

Proxy implementations SHOULD also send a Token Window Advertisement if:

- o the token is out of window, or
- o by the proxy's internal logic, successfully spending the token caused the window to shift.

Proxy implementations SHOULD NOT shift the window's base beyond the highest unspent token.

Proxy implementations MAY include a Token Window Advertisement in any Operation Reply.

8.2.3. Handling Token Window Advertisements

Even though the proxy increases the window's base monotonically, there is no mechanism whereby a SOCKS client can receive the Token Window Advertisements in order. As such, clients SHOULD disregard unsolicited Token Window Advertisements with a Window Base less than the previously known value.

9. Security Considerations

9.1. Large requests

Given the format of the request message, a malicious client could craft a request that is in excess of 100 KB and proxies could be prone to DDoS attacks.

To mitigate such attacks, proxy implementations SHOULD be able to incrementally parse the requests. Proxies MAY close the connection to the client if:

- o the request is not fully received after a certain timeout, or
- o the number of options exceeds an imposed hard cap, or
- o the total size of the options exceeds an imposed hard cap, or
- o the size of the initial data exceeds a hard cap.

Further, the server MAY choose not to buffer any initial data beyond what would be expected to fit in a TFO SYN's payload.

9.2. Replay attacks

In TLS 1.3, early data (which is likely to contain a full SOCKS request) is prone to replay attacks.

While Token Expenditure options can be used to mitigate replay attacks, the initial Token Request is still vulnerable. As such, client implementations SHOULD NOT make use of TLS early data when sending a Token Request.

10. IANA Considerations

This document requests that IANA allocate option codes for SOCKS 6 options. Further, this document requests option codes for authentication and idempotence options.

This document also requests that IANA allocate a port for SOCKS over TLS.

11. Acknowledgements

The protocol described in this draft builds upon and is a direct continuation of SOCKS 5 [RFC1928].

12. References

12.1. 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>>.

12.2. Informative References

- [I-D.ietf-tls-tls13] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", draft-ietf-tls-tls13-21 (work in progress), July 2017.
- [RFC1928] Leech, M., Ganis, M., Lee, Y., Kuris, R., Koblas, D., and L. Jones, "SOCKS Protocol Version 5", RFC 1928, DOI 10.17487/RFC1928, March 1996, <<https://www.rfc-editor.org/info/rfc1928>>.
- [RFC6824] Ford, A., Raiciu, C., Handley, M., and O. Bonaventure, "TCP Extensions for Multipath Operation with Multiple Addresses", RFC 6824, DOI 10.17487/RFC6824, January 2013, <<https://www.rfc-editor.org/info/rfc6824>>.
- [RFC7413] Cheng, Y., Chu, J., Radhakrishnan, S., and A. Jain, "TCP Fast Open", RFC 7413, DOI 10.17487/RFC7413, December 2014, <<https://www.rfc-editor.org/info/rfc7413>>.

Authors' Addresses

Vladimir Olteanu
University Politehnica of Bucharest

Email: vladimir.olteanu@cs.pub.ro

Dragos Niculescu
University Politehnica of Bucharest

Email: dragos.niculescu@cs.pub.ro

Internet Draft
Intended status: Informational
Expires: April 2018

J. Xia
Huawei
October 30, 2017

Architectural Considerations for
Delivering Latency Critical Communication over the Internet
draft-xia-latency-critical-communication-00.txt

Abstract

There is clear demand for Internet applications requiring critical low-latency and reliable communication - Latency Critical Communication (LCC).

This document is intended to stimulate LCC discussion and is not expected to be published as an RFC.

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/ietf/lid-abstracts.txt>

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

This Internet-Draft will expire on April 30, 2018.

Copyright Notice

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

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

Table of Contents

1. Introduction	2
2. The Need for Low Latency Communications	3
3. Quantifying Latency	4
3.1. Determinism	4
3.2. Network KPIs	4
3.3. Service KQIs	6
3.4. Correlating KQI and KPI	6
3.5. Application Use Cases	7
3.5.1. Cloud-based Virtual Reality	8
3.5.1.1. Quality of Experience Requirements	8
3.5.2. Remote Surgery	8
3.5.2.1. Quality of Experience Requirements	8
3.5.3. Live-TV Distribution in Virtualized CDN environments	9
3.5.3.1. Quality of Experience Requirements	9
4. Measurement of Latency	10
4.1. End-to-end Latency	11
4.2. Per Link Latency	12
4.3. Per Node Latency	12
4.4. Reporting Per Link and Per Node Latency	12
4.5. Isolating Latency Disruption	13
5. Mechanisms to achieve low latency flows	13
5.1. Path Computation	13
5.2. Traffic Engineering	13
5.3. Coloring	14
5.4. Queue Management	14
5.5. Latency Management	15
6. Functional Architecture for LCC	15
6.1. LCC Functional Components	16
7. Alternatives to Low Latency Networking	16
7.1. Mitigation	16
7.2. Resource Placement	16
7.3. Application and Resource Pipelining	17
7.4. Prediction	17
7.5. Buffering	17
8. Security Considerations	17
8.1 Privacy and Regulatory Issues	17
9. IANA Considerations	17
10. References	18
10.1. Normative References	18
10.2. Informative References	18
11. Acknowledgments	18

Latency Critical Communication (LCC) applications are increasingly important, requiring guaranteed low-latency communication high-reliability and ensuring quality of user experience.

Several on-going mechanisms exist for delivering LCC services within multiple Standards Development Organizations, including: Time-Sensitive Networking Task Group [TSN8021] in IEEE 802.1, 5G requirements for next-generation access technology [TS38913] in 3GPP and Broadband Assured IP Service [BAS-Architecture] in the BBF.

This draft identifies common service requirements in heterogeneous networks for delivering LCC services, and outlines specific uses across a spectrum of applications, specifically: cloud-based virtual reality, remote surgery, and live-TV distribution in virtualized CDN environments.

We may scope LCC application requirements by explicitly focusing on end-to-end (E2E) service characteristics and capability requirements for delivering each specific use case. Furthermore, as the E2E service usually traverses multiple domains and involves multiple layers. Yet, existing standards and current discussion typically focuses on a specific layer, protocol, or link layer technology. This focused view lacks an integrated approach or system view on solving the LCC problem space.

This document is intended to stimulate discussion and outlines specific LCC application requirements, and proposes an architecture and enabling functional components to address the common requirements discussed in each use case.

2. The Need for Low Latency Communications

Fundamentally, latency is a time interval between the stimulation and response, or, from a more general point of view, a time delay between the cause and the effect of change in the system being observed.

Network latency in packet networks is measured either one-way (the time from the source sending a packet to the destination receiving it), or round-trip delay time (the one-way latency from source to destination plus the one-way latency from the destination back to the source). Some packets will be dropped, i.e., never delivered, due to buffer overflows, synchronization failures, etc. Moreover, we assume that packets that are decoded in error are also dropped either by the protocol itself or by higher layers. Using the convention that dropped packets have infinite latency, we can define

Our community has recognized low latency networking as an important research problem, and devoted much attention to tackle the issue from various perspectives, these include:

- o Processing delays
- o Buffer delays
- o Transmission delays
- o Packet loss
- o Propagation delays

There are a number of common requirements across low latency use cases (including 3GPP on Cloud RAN, front haul, back haul and by various application layers use cases. Additional useful documents exist that provide background and motivation for low latency networks, including [I-D.arkko-arch-low-latency] and [I-D.dunbar-e2e-latency-arch-view-and-gaps].

3. Quantifying Latency

LCC Applications exist for a variety of deployments, use cases are assigned into the following categories:

- o Extreme Mobile Broadband (xMBB): high speed and low latency mobile broadband;
- o Ultra-reliable Machine-type Communication (uMTC): reliability is the key service requirement of these services;
- o Massive Machine-Type Communication (mMTC) and Massive IoT (mIoT): massive M2M and IoT connectivity;
- o Critical Connections/ Ultra Reliable Low Latency Connections (Cric/URLLC): low latency and ultra-reliable communications.

The focus of this document is to outline requirements for Cric/URLLC use cases, specifically:

- o Cloud-based virtual reality;
- o Remote surgery;
- o Live-TV distribution in virtualized CDN environments.

- o Difference between predictable and reliable bounds.
- o Behavior of packet flow, and loss, and/or packets allowed outside of the bounds.

3.2. Network KPIs

For each category of use case, specific KPIs are identified for clustering requirements:

Device density:

- o High: 10000 devices per km²
- o Medium: 1000 - 10000 devices per km²
- o Low: < 1000 devices per km²

Mobility:

- o No: static users
- o Low: pedestrians (0-3 km/h)
- o Medium: slow moving vehicles (3 - 50 km/h)
- o High: fast moving vehicles, e.g. cars and trains (> 50 km/h)

Infrastructure:

- o Limited: no infrastructure available or only macro cell coverage
- o Medium density: Small number of small cells
- o Highly available infrastructure: Big number of small cells available

Traffic type:

- o Continuous
- o Bursty
- o Event driven
- o Periodic
- o All types

User data rate:

- o Very high data rate: 1 Gbps
- o High: 100 Mbps - 1 Gbps
- o Medium: 50 - 100 Mbps
- o Low: < 50 Mbps

Latency

- o High: > 50 ms
- o Medium: 10 - 50 ms
- o Low: 1 - 10 ms

Reliability

- o Low: < 95%

- o Medium: 95 - 99%
- o High: > 99%

Availability (related to coverage)

- o Low: < 95%
- o Medium: 95 - 99%
- o High: > 99%

3.3. Service KQIs

Application requirements, can be modelled by user experience (QoE), and qualified by service KQI. From users' point of view, QoE is the overall performance of a system. It is a measure of E2E performance at the services level from the user perspective and an indication of how well the system meets the user's needs. There are many factors affecting QoE, such as user expectations, end-to-end system effects, etc. It is essential to establish a relation between user expectations and QoS, considered as the ability of the network to provide a service at a guaranteed performance level.

Network's performance can be evaluated with network KPIs such as delay, jitter, packet loss, etc. For URLLC services, existing KPIs are insufficient to forecast the service quality and reflect end-users' QoE. Hence, it is important to identify useful KPIs to quantify end-users' experiences and build the connections between network KPI and service KQI, as shown in Figure 1. The KQI for a given service can be expressed as a function/combination of the KPIs, and can be expressed as follow: $KQI=f(KPI1, KPI2, \dots, KPI_n)$.

3.4. Correlating KQI and KPI

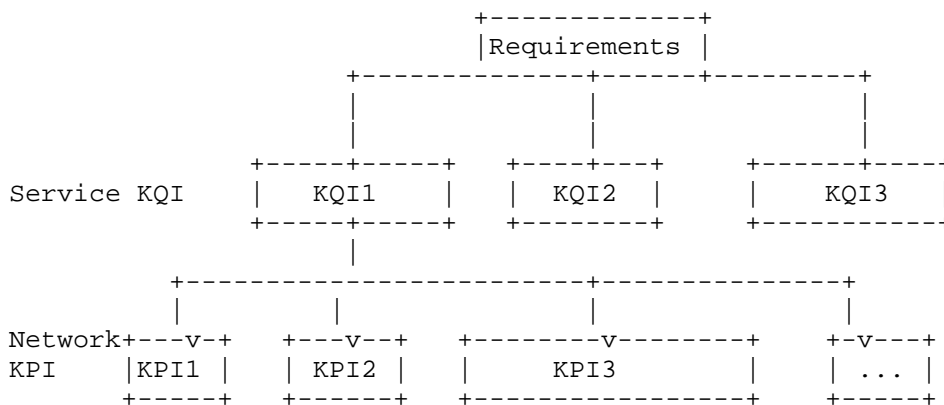


Figure 1: KQI-KPI Correlation

The emerging LCC application services have led to composite KQIs use, providing network measurement of specific application service aspects (i.e., the performance of the application or service). As there is limited experience to guide how to deliver the new LCC services, the mapping between the KPI and KQI will require specific

3.5. Application Use Cases

3.5.1. Cloud-based Virtual Reality

Virtual Reality (VR), also known as immersive multimedia or computer-simulated reality, is a computer technology that replicates an environment, real or imagined, and simulates a user's physical presence and environment to allow for user interaction.

Although some aspects of VR are becoming promising, there is still bottleneck that prevents it from being popular. High cost, especially for higher-end systems that try to reproduce a high-quality experience, is one barrier to success for VR. One way to reduce the cost of local VR computing, to make it more affordable and increase its popularity and general usage, is to offload the computations to a cloud-based server. This especially fits the cloud-based VR gaming environment when connecting with multiple parties.

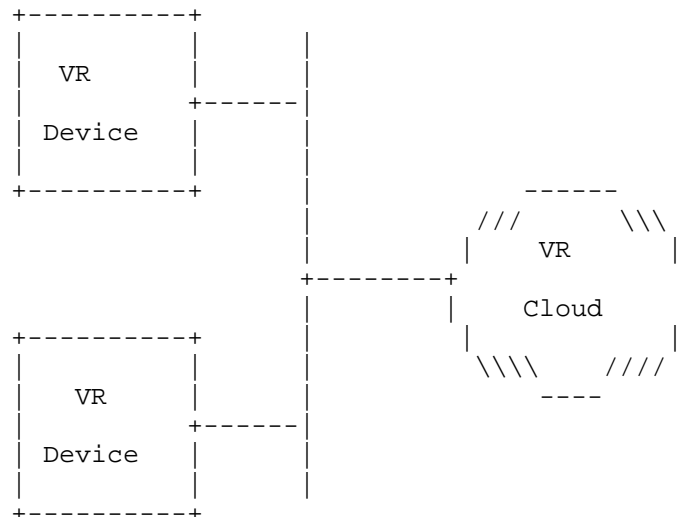


Figure 2: Cloud-based VR Scenario

But then, additional stringent requirements for the underlying network are being introduced into the system, including high bandwidth, low latency and low packet loss ratio.

To make the VR world realistic, the VR motion-to-photon latency is recommended to be less than 20ms. However, the network transmission latency is limited within 5ms because other VR processing (i.e., tracking, rendering, and displaying) latency almost consumes 15ms. To achieve this, the VR cloud is proposed to be deployed at the edge of operator network.

Regarding bandwidth requirements, high-end VR systems typically use a display frame rate of 75-90 frames per second on dual HD or 4K displays, which can result in traffic rates four to eight times that for regular HD or 4K video respectively. Of course, low packet loss is required to prevent video freezes or dropouts.

Name	Elements	
Service type	CriC/URLLC	
Bandwidth [Mb/s]	4K	25Mb/s
	8K	100 Mb/s
	12K	418 Mb/s
Bitrate(Mbps)	4K	16 Mbps
	8K	64 Mbps
	12K	279 Mbps
Latency	5 ms	
Reliability	High (five 9)	

Figure 3: Cloud VR Service Type

3.5.2. Remote Surgery

Remote surgery (also known as telesurgery) is the ability for a doctor to perform surgery on a patient even though they are not physically in the same location. It further includes the high-speed communication networks, connecting the surgical robot in one location to the surgeon console at another location manipulating the robot through an operation.

Remote telesurgery allows the specialized surgeons to be available to the patients worldwide, without the need for patients to travel beyond their local hospital. Imagine a doctor in an urban city, performing an urgent procedure on a patient in an inaccessible rural area.

3.5.2.1. Quality of Experience Requirements

In order to ensure a telesurgery procedure to be highly safe, a particularly unique demand is required on the network, at least including very reliable connection (99.999% availability), sufficient bandwidth to ensure adequate video resolution as required by the remote surgeon controlling the robot, as little as possible latency allowing the feel of instantaneous reaction to the actions of the surgeons and of course as little as possible latency variation (i.e., jitter) allowing system or human correction of the latency.

Name	Elements
Service type	CriC/URLLC
Bandwidth [Mb/s]	Up to 1Mb/s for control commands
Bitrate(Mbps)	8K 64 Mbps
Latency	30 ms
Reliability	High (five 9)

Figure 4: Remote Surgery Service Type

3.5.3. Live-TV Distribution in Virtualized CDN environments

Live-TV signal distribution is a growing service that a network operator needs to support. The bandwidth needed to convey a video stream is determined by its quality. Evolution from standard definition (SD) and high definition (HD) quality formats towards Ultra-High Definition (UHD) formats, including 2K and 4K UHD will have to be carried across an IP network, thus requiring the migration from traditional Serial Digital Interfaces (SDI) transmission to all-IP environments.

Various paradigms exist to provide cost-effective scalable live-TV distribution. Specifically, in live-TV distribution, uncompressed video stream formats are used before the video is produced. Once the video has been produced, distribution to end-users is based on compressed video streams, which quality is adapted to the one that fits better the user's device (i.e., compressed SD, HD or UHD formats).

Content Delivery Networks (CDN) can be considered as a suitable option for live-TV content delivery by means of the standardized MPEG Dynamic Adaptive Streaming over HTTP (MPEG-DASH) technique.

3.5.3.1. Quality of Experience Requirements

Internet-Draft Delivering Low Latency Services October 2017

Transport quality (packet loss, jitter) highly impacts on users' quality of experience (QoE). Undesired effects such as pixelization, tiling, frame freezing, or blue screen can appear if transport quality is slightly degraded.

Monitoring at different levels (network, computing, service) and applying local/global KDD procedures enable dynamic adaptive CDN reconfiguration, i.e. scaling up/down HTTP servers, reassigning users, increasing CDN links capacity, etc.

Name	Elements
Service type	CriC/URLLC
Bandwidth [Mb/s]	Low 1-4 Mb/s SD Med 10 Mb/s HD High 25 Mb/s UHD
Latency	High 50-100s ms
Jitter	Stringent <1 ms
Reliability	High (five 9)
Availability	Moderate (99%)
Mobility - UE Speed	Up to 50km/h
Area Traffic	Normal 1s Gb/s Hotspot 10s Gb/s
Sensor Network	No
Massive Type	No
Device Direct	No
Coverage Required	Standard
Energy Consumption Critical	No
Type of Use Equip.	Conventional

4. Measurement of Latency

Various Internet measurement methods have been proposed to identify latency between end hosts. Active network measurements, which involve sending a stream of measurement data traversed along arbitrary paths over a network, including the Internet, are amongst the more popular methods to provision end-to-end quality-of-service.

Accurate network measurement would require mesh measurement of all network links to collect sufficient network latency information for network path construction based on active measurement methods. It takes a longer time; thus, it may be possible for a small group of nodes but not for larger number of nodes. Inaccurate measurement over lossy network with long inter-packet delays would become an issue, and not support real-time applications that require time sensitive information for network path decisions.

In the [I-D.dunbar-e2e-latency-arch-view-and-gaps], several key latency factors are listed as below:

- o Generation: delay between physical event and availability of data
- o Transmission: signal propagation, initial signal encoding
- o Processing: Forwarding, encoding/decoding, NAT, encryption, authentication, compress, error coding, signal translation
- o Multiplexing: Delays needed to support sharing; Shared channel acquisition, output queuing, connection establishment
- o Grouping: Reduces frequency of control information and processing; Packetization, message aggregation

From the network point of view, only the last four latency factors are highly relevant to the network characteristic and need to be measured.

The E2E performance has been focused on connection-less technologies, the requirements of measuring and maintaining "flow" state for end-user have gaps.

Measurement of network delay, performance guarantees, dynamic path adaption, and throughput optimization, all exist but are generally technology specific.

4.1. End-to-end Latency

A One-way Delay Metric for IPPM is defined for packets across Internet paths based on notions introduced in the IPPM framework with detailed introduction on the measurement methodology, error analysis and relevant statistics. The result can be used to indicate the performance of specific application and the congestion state on the path traversed.

IP Flow Information Export (IPFIX) Protocol serves as a means for transmitting Traffic Flow information over the network from an IPFIX Exporting Process to an IPFIX Collecting Process.

IPPM or IPFIX should be sufficient for the controller of distributed control plane to make the necessary optimization or bandwidth control, assuming IPFIX and IPPM can measure segment, interface, and chassis/fabric time. But if not, the extension of existing IPPM (metrics) may be needed.

In addition, other existing technologies, such as One-Way Active Measurement Protocol (OWAMP) and Two-Way Active Measurement Protocol (TWAMP), are focused on providing one way and two way IP performance metrics. Latency is one of metrics that can be used for End-to-End deterministic latency provisioning.

Using OWAMP/TWAMP protocols or extension on that to support measurement of flow latency performance is also feasible.

4.2. Per Link Latency

Latency related to link can be computed as the ratio between the link length and the propagation speed over the specific medium of the link.

The link capacities along the path as well as the way in which the available capacity is used can have impact on the latency. Whenever the link capacity is low, the time of getting data out of network card to onto the medium will be high. Furthermore, capacity is often shared and only a small proportion of the capacity may be available to a specific flow, this is the case when links are congested.

4.3. Per Node Latency

The links along a network path are connected by network nodes, such as core switches and routers. Transit through each node adds to the path latency, this type of latency is referred to as switching/forwarding latency.

To achieve optimized end-to-end low latency services, each network node along the path needs to measure the latency metric on it. Using OWAMP /TWAMP and/or extension on that, each network node needs to record accurate measurements and thus requires accurate time synchronization, which also contributes latency on the network node.

4.4. Reporting Per Link and Per Node Latency

Basically, the latency information needs to be reported from the network node to the controller or OAM handler [RFC7491] to keep the end-to-end latency under bound. A template that defines the LCC connection attributes, latency, loss and etc, must be used.

In addition, an interface or mechanism to report such latency performance information is necessary. A simple approach can be an interface from network device to controller, which collects all the latency performance information of each network node, and then make a decision how to serve the flow at each network node.

4.5. Isolating Latency Disruption

When congestion occurs, it is often not being detected until it has already induced latency. Early detection of the onset of congestion allows the controllers to reduce their transmission rate quickly. This could use delay based inference of congestion or early explicit notification of congestion by the network.

However, the congestion occurred link should be separated with other links and thus will not disrupt the other links. One feasible way is to reserve dedicated network resources to the specific link (for a specific application) and thus isolate the usage of the dedicated network resources from other links.

5. Mechanisms to achieve low latency flows

The network infrastructure will need advanced interaction with LLC applications. The network will need insight into which types of applications are being transported, and traffic classification and path control to ensure SLAs expected by the applications are met. Several techniques exist to achieve this, and are discussed in the following sub-sections.

5.1. Path Computation

The Path Computation Element (PCE) was developed to provide path computation services for path controlled networks. The may be used to provide path computation and policy enforcement for LCC applications and services.

The PCE operates on a view of the network topology stored in the Traffic Engineering Database (TED). The TED is a data store of topology information about network nodes and links, and capacity and metrics such as link performance (latency, latency-variation, and packet loss). The TED may be further augmented with status information about existing services as well.

The PCE would facilitate the setting up of LCC application paths by computing a path based on the end-to-end network performance criteria.

5.2. Traffic Engineering

MPLS-TE allows for a TE scheme where the ingress node of a label
switched path (LSP) can calculate the most efficient route
(including latency minimization) through the network toward the
egress router of the LSP.

The operator typically has a pre-planning task to monitor the
physical layout of the network for capacity planning and network
visualization followed by estimation of possible TE settings of the
links, knowing how much an IGP setting affects the traffic flow and
path. Modification of TE settings to reduce latency based on network
conditions is possible, but introduces potential network instability
if changes are frequent.

Overall, TE technologies come with limitations such as scalability,
operational complexity, protocol overhead, and supervised network
optimization. Although, recent enhancements to MPLS-TE exist, the
interaction between applications and network infrastructure is still
not sufficient for the LLC challenges.

5.3. Coloring

It is possible to build colored paths through the network with the
colors representing low bandwidth, low delay, high cost, affinities.
Application traffic can then be assigned to those paths based on
traffic placement profile.

Link coloring could be used to classify specific low latency links
for LLC applications, and assigned to a logical topology for the
delay-sensitive application traffic.

MPLS-TE also supports this function, often described as
administrative groups "colors" or "link colors". Furthermore, link
coloring is supported in IP networks with the use of MT-aware IGPs.

5.4. Queue Management

Deploying queue management techniques, such as Active Queue
Management (AQM), in the network may facilitate latency reduction,
reduce network latency. It may be useful to distinguish between two
related classes of algorithms: "queue management" versus
"scheduling" algorithms.

- o Queue management algorithms manage the length of packet queues by
marking or dropping packets when necessary or appropriate

The two mechanisms are loosely related, they address different performance issues and operate on different timescales.

As interactive applications (e.g. voice over IP, real time video streaming and financial transactions) run in the Internet, high latency and latency variation degrade application performance.

Deploying intelligent queue management and scheduling schemes to control latency and latency variation, would provide desirable and predictable behavior to end-to-end connections for applications

5.5. Latency Management

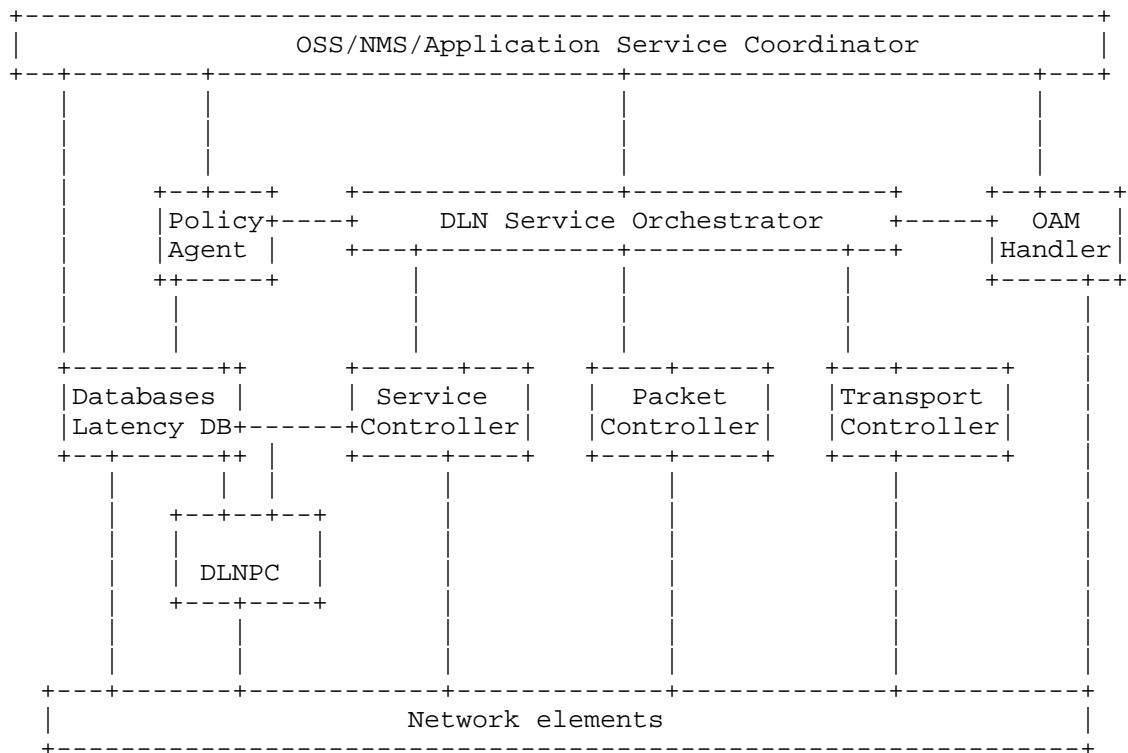
Latency management techniques include:

o XoDel (Controlled Delay) and FQ-CoDel (FlowQueue-CoDel) Controlled Delay (CoDel) are queue management technologies to set limits per packet for delay

o FlowQueue-CoDel (FQ-CoDel) is a hybrid packet scheduler/AQM algorithm for fighting application latency across the Internet. It is based on a two-tier Deficit Round Robin (DRR) queue scheduler, with the CoDel AQM algorithm operating on each sub-queue.

6. Functional Architecture for LCC

A basic architecture for LCC operation will be required. These will include the necessary components to manage the latency service, underlay packet and transport communications infrastructure.



6.1. LCC Functional Components

7. Alternatives to Low Latency Networking

7.1. Mitigation

Several strategies and techniques exist for reducing or negating network latency for some time sensitive applications.

7.2. Resource Placement

There is a trend of placing resources in locations that would reduce or negate service and application latency.

One approach to support more dynamic placement of functions, enabling the LLC application, close to the user is to introduce Virtualized Network Functions (NFV) at the edge of the network, near the LCC application users to reduce end-to-end latency, time-to-response, and unnecessary utilization of the core network infrastructure.

7.3. Application and Resource Pipelining

To be discussed.

7.4. Prediction

To be discussed.

7.5. Buffering

Reducing switch queue length, or buffer occupancy, is the most direct way to tackle the latency problem. Packet forwarders could use deep buffers to handle bursty traffic. However, they must ensure that this does not become detrimental to latency performance. As TCP relies on packet drops for congestion control, it introduces overhead for the application.

8. Security Considerations

The following list provides some security challenges and considerations in designing and building network infrastructure for LLC applications:

- o Identification and authentication of the entities involved in the LLC service
- o Access control to the different entities that need to be connected and capable of creating LLC services
- o Processes and mechanisms to guarantee availability of LLC network resources and protect them against attack

8.1 Privacy and Regulatory Issues

- o Identification of endpoints
- o Data protection to guarantee the security (confidentiality, integrity, availability, authenticity) and privacy of the information carried by the network for the LCC service

9. IANA Considerations

10. References

10.2. Informative References

[TSN8021] "Time-Sensitive Networking Task Group", IEEE
(<http://www.ieee802.org/1/pages/tsn.html>).

[BAS-Architecture] Y.L. Jiang, "Broadband Assured IP Services
Architecture", draft WT-387-00, broadband forum (BBF), July, 2016.

[TS38913] "3rd Generation Partnership Project; Technical
Specification Group Radio Access Network; Study on Scenarios and
Requirements for Next Generation Access Technologies; (Release 14)"

[I-D.arkko-arch-low-latency] J. Arkko, "Low Latency Applications and
the Internet Architecture", draft-arkko-arch-low-latency (work in
progress), 2017.

[I-D.dunbar-e2e-latency-arch-view-and-gaps] Dunbar, L.,
"Architectural View of E2E Latency and Gaps", draft-dunbar-e2e-
latency-arch-view-and-gaps (work in progress), 2017.

[MEC_White_Paper] ETSI, "Mobile-Edge Computing - Introductory
Technical White Paper", 2014.

[RFC7491] D. King and A. Farrel, "A PCE-Based Architecture for
Application-Based Network Operations ", RFC 7491, March 2015,
<<http://www.rfc-editor.org/info/rfc7491>>.

11. Acknowledgments

Authors' Addresses

Jinwei Xia
Huawei
101 Software Avenue, Yuhua District
Nanjing, Jiangsu 210012
China

Email: xiajinwei@huawei.com

Contributors

Ning Zong
Huawei Technologies

Email: zongning@huawei.com

Daniel King
Lancaster University

Email: d.king@lancaster.ac.uk