Fast Recovery for EVPN Designated Forwarder Election

draft-ietf-bess-evpn-fast-df-recovery-09

Abstract

The Ethernet Virtual Private Network (EVPN) solution provides Designated Forwarder (DF) election procedures for multihomed Ethernet Segments. These procedures have been enhanced further by applying Highest Random Weight (HRW) algorithm for Designated Forwarder election in order to avoid unnecessary DF status changes upon a failure. This document improves these procedures by providing a fast Designated Forwarder election upon recovery of the failed link or node associated with the multihomed Ethernet Segment. This document updates Section 2.1 of [RFC8584] by optionally introducing delays between some of the events therein.

The solution is independent of the number of EVPN Instances (EVIs) associated with that Ethernet Segment and it is performed via a simple signaling between the recovered node and each of the other nodes in the multihoming group.

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 9 January 2025.
1. Introduction

The Ethernet Virtual Private Network (EVPN) solution [RFC7432] is becoming pervasive in data center (DC) applications for Network Virtualization Overlay (NVO) and DC interconnect (DCI) services, and in service provider (SP) applications for next generation virtual private LAN services.

[RFC7432] describes Designated Forwarder (DF) election procedures for multihomed Ethernet Segments. These procedures are enhanced further in [RFC8584] by applying the Highest Random Weight (HRW) algorithm for DF election in order to avoid unnecessary DF status changes upon a link or node failure associated with the multihomed Ethernet Segment. This document makes further improvements to the DF election
procedures in [RFC8584] by providing an option for a fast DF election upon recovery of the failed link or node associated with the multihomed Ethernet Segment. This DF election is achieved independent of the number of EVPN Instances (EVIs) associated with that Ethernet Segment and it is performed via straightforward signaling between the recovered node and each of the other nodes in the multihomed group.

This document updates the DF Election Finite State Machine (FSM) described in Section 2.1 of [RFC8584], by optionally introducing delays between some events, as further detailed in Section 2.2. The solution is based on a simple one-way signaling mechanism.

1.1. Requirements Language

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

1.2. Terminology

PE: Provider Edge device.

Designated Forwarder (DF): A PE that is currently forwarding (encapsulating/decapsulating) traffic for a given VLAN in and out of a site.

EVI: An EVPN instance spanning the Provider Edge (PE) devices participating in that EVPN.

1.3. Challenges with Existing Mechanism

In EVPN technology, multiple Provider Edge (PE) devices have the ability to encapsulate and decapsulate data belonging to the same VLAN. Under certain conditions, this may cause Layer2 duplicates and potential loops if there is a momentary overlap in forwarding roles between two or more PE devices, consequently leading to broadcast storms.

EVPN [RFC7432] currently specifies timer-based synchronization among PE devices within a redundancy group. This approach can lead to duplications and potential loops due to multiple Designated Forwarders (DFs) if the timer interval is too short, or to packet drops if the timer interval is too long.
Split-horizon filtering, as described in Section 8.3 of [RFC7432], can prevent loops but does not address duplicates. However, if there are overlapping Designated Forwarders (DFs) of two different sites simultaneously for the same VLAN, the site identifier will differ when the packet re-enters the Ethernet Segment. Consequently, the split-horizon check will fail, resulting in Layer 2 loops.

The updated Designated Forwarder (DF) procedures outlined in [RFC8584] use the well-known Highest Random Weight (HRW) algorithm to prevent the reshuffling of VLANs among PE devices within the redundancy group during failure or recovery events. This approach minimizes the impact on VLANs not assigned to the failed or recovered ports and eliminates the occurrence of loops or duplicates during such events.

However, upon PE insertion or a port being newly added to a multihomed Ethernet Segment, HRW also cannot help as a transfer of DF role to the new port must occur while the old DF is still active.

In Figure 1, when PE2 is inserted in the Ethernet Segment or its CE1-facing interface recovered, PE1 will transfer the DF role of some VLANs to PE2 to achieve load balancing. However, because there is no handshake mechanism between PE1 and PE2, overlapping of DF roles for a given VLAN is possible which leads to duplication of traffic as well as Layer 2 loops.

Current EVPN specifications [RFC7432] and [RFC8584] rely on a timer-based approach for transferring the DF role to the newly inserted device. This can cause the following issues:

* Loops/Duplicates if the timer value is too short
1.4. Design Principles for a Solution

The clock-synchronization solution for fast DF recovery presented in this document follows several design principles and presents multiple advantages, namely:

* Complex handshake signaling mechanisms and state machines are avoided in favor of a simple uni-directional signaling approach.

* The fast DF recovery solution maintains backwards-compatibility (see Section 4) by ensuring that PEs any unrecognized new BGP Extended Community.

* Existing DF Election algorithms remain supported.

* The fast DF recovery solution is independent of any BGP delays in propagation of Ethernet Segment routes (Route Type 4).

* The fast DF recovery solution is agnostic of the actual time synchronization mechanism used, and normalizes to NTP for EVPN signalling only.

2. DF Election Synchronization Solution

The fast DF recovery solution relies on the concept of common clock alignment between partner PEs participating in a common Ethernet Segment i.e. PE1 and PE2 in Figure 1. The main idea is to have all peering PEs of that Ethernet Segment perform DF election, and apply the result at the same pre-announced time.

The DF Election procedure, as described in [RFC7432] and as optionally signalled in [RFC8584], is applied. All PEs attached to a given Ethernet Segment are clock-synchronized using a networking protocol for clock synchronization (e.g., NTP, PTP). When a new PE is inserted in an Ethernet Segment or a failed PE device of the Ethernet Segment recovers, that PE communicates to peering partners the current time plus the value of the timer for partner discovery from step 2 in Section 8.5 of [RFC7432]. This constitutes an "end time" or "absolute time" as seen from the local PE. That absolute time is called the "Service Carving Time" (SCT).

A new BGP Extended Community, the Service Carving Timestamp is advertised along with the Ethernet Segment route (RT-4) to communicate the Service Carving Time to other partners.
Upon receipt of the new BGP Extended Community, partner PEs can determine the service carving time of the newly inserted PE. To eliminate any potential for duplicate traffic or loops, the concept of skew is introduced: a small time offset to ensure a controlled and orderly transition when multiple Provider Edge (PE) devices are involved. The receiving partner PEs add a skew (default = -10ms) to the Service Carving Time to enforce this mechanism. The previously inserted PE(s) must perform service carving first, followed shortly by the newly inserted PE, after the specified skew delay.

To summarize, all peering PEs perform service carving almost simultaneously at the time announced by the newly added/recovered PE. The newly inserted PE initiates the SCT, and triggers service carving immediately on its local timer expiry. The previously inserted PE(s) receiving Ethernet Segment route (RT-4) with a SCT BGP extended community, perform service carving shortly before Service Carving Time.

2.1. BGP Encoding

A new BGP extended community is defined to communicate the Service Carving Timestamp for each Ethernet Segment.

A new transitive extended community where the Type field is 0x06, and the Sub-Type is 0x0F is advertised along with the Ethernet Segment route. The expected Service Carving Time is encoded as an 8-octet value as follows:

```
 1 2 3
+----------+----------+
| Type = 0x06 | Sub-Type(0x0F) |
| Timestamp Seconds |
+----------+----------+
| Timestamp Seconds |
| Timestamp Fractional Seconds |
```

Figure 2: Service Carving Time

The timestamp exchanged uses the NTP prime epoch of January 1, 1900 [RFC5905] and the 64-bit NTP Timestamp Format. The NTP Era value is not exchanged and Era 0 is assumed as of the writing of this document. A DF Election operation occurring exactly at the Era transition boundary some time in 2036 is outside of the scope of this document.

The 64-bit NTP Timestamp Format consists of a 32-bit part for Seconds and a 32-bit part for Fraction, which are encoded in the Service Carving Time as follows:
* Timestamp Seconds: 32-bit NTP seconds are encoded in this field.

* Timestamp Fractional Seconds: the high order 16 bits of the NTP 'Fraction' field are encoded in this field.

When rebuilding a 64-bit NTP Timestamp Format using the values from a received SCT BGP extended community, the lower order 16 bits of the Fractional field are set to 0. The use of a 16-bit fractional seconds yields adequate precision of 15 microseconds (2^-16 s).

This document introduces a new flag called "T" (for Time Synchronization) to the bitmap field of the DF Election Extended Community defined in [RFC8584].

```
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Type = 0x06   | Sub-Type(0x06) | RSV |  DF Alg | |A| |T|       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
         Bitmap    |            Reserved = 0                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: DF Election Extended Community

* Bit 3: Time Synchronization (corresponds to Bit 27 of the DF Election Extended Community). When set to 1, it indicates the desire to use Time Synchronization capability with the rest of the PEs in the Ethernet Segment.

This capability is utilized in conjunction with the agreed-upon DF Election Type. For instance, if all the PE devices in the Ethernet Segment indicate possessing Time Synchronization capability and request the DF Election Type to be Highest Random Weight (HRW), then the HRW algorithm is edused in conjunction with this capability. A PE which does not support the procedures set out in this document, or receives a route from another PE in which the capability is not set MUST NOT delay Designated Forwarder election as this could lead to duplicate traffic in some instances (overlapping Designated Forwarders).

2.2. Updates to RFC8584

This document introduces an additional delay to the events and transitions defined for the default DF election algorithm FSM in Section 2.1 of [RFC8584] without changing the FSM state or event definitions themselves.
Upon receiving a RECV_ES message, the peering PE’s Finite State Machine (FSM) transitions from the DF_DONE (indicating the DF election process was complete) state to the DF_CALC (indicating that a new DF calculation is needed) state. Due to the Service Carving Time (SCT) included in the Ethernet-Segment update, the completion of the DF_CALC state and the subsequent transition back to the DF_DONE state are delayed. This delay ensures proper synchronization and prevents conflicts. Consequently, the accompanying forwarding updates to the Designated Forwarder (DF) and Non-Designated Forwarder (NDF) states are also deferred.

The corresponding actions when transitions are performed or states are entered/exited are modified as follows:

9. DF_CALC on CALCULATED: Mark the election result for the VLAN or VLAN Bundle.

  9.1 If an SCT timestamp is present during the RECV_ES event of Action 11, wait until the time indicated by the SCT before proceeding to step 9.2.

  9.2 Assume the role of DF or NDF for the local PE concerning the VLAN or VLAN Bundle, and transition to the DF_DONE state.

This revised approach ensures proper timing and synchronization in the DF election process, avoiding conflicts and ensuring accurate forwarding updates.

3. Synchronization Scenarios

Consider Figure 1 as an example, where initially PE2 has failed and PE1 has taken over. This scenario illustrates the problem with the DF-Election mechanism described in Section 8.5 of [RFC7432], specifically in the context of the timer value configured for all PEs on the Ethernet Segment.

Procedure based on Section 8.5 of [RFC7432] with the default 3 second timer in step 2:

1. Initial state: PE1 is in a steady-state and PE2 is recovering

2. Recovery: PE2 recovers at an absolute time of t=99.

3. Advertisement: PE2 advertises RT-4, sent at t=100, to partner PE1.

4. Timer Start: PE2 starts a 3 second timer to allow the reception of RT-4 from other PE nodes.
5. Immediate carving: PE1 performs service carving immediately upon RT-4 reception, i.e. t=100 plus some BGP propagation delay.

6. Delayed Carving: PE2 performs service carving at time t=103

[RFC7432] favors traffic drops over duplicate traffic. With the above procedure, traffic drops will occur as part of each PE recovery sequence since PE1 transitions some VLANs to Non-Designated Forwarder (NDF) immediately upon RT-4 reception. The timer value (default = 3 seconds) directly affects the duration of the packet drops. A shorter (or zero) timer may result in duplicate traffic or traffic loops.

Procedure based on the Service Carving Time (SCT) approach:

1. Initial state: PE1 is in a steady state, and PE2 is recovering
2. Recovery: PE2 recovers at an absolute time of t=99.
3. Advertisement: PE2 advertises RT-4, sent at t=100, with a target SCT value of t=103 to partner PE1.
4. Timer Start: PE2 starts a 3 second timer to allow the reception of RT-4 from other PE nodes.
5. Service Carving Timer: PE1 starts the service carving timer, with the remaining time until t=103
6. Simultaneous Carving: Both PE1 and PE2 carve at an absolute time of t=103

To maintain the preference for minimal loss over duplicate traffic, PE1 should carve slightly before PE2 (with skew). The recovering PE2 performs both DF to NDF and NDF to DF transitions per VLAN at the timer’s expiry. The original PE1, which received the SCT, applies the following:

* DF to NDF Transition(s): at t=SCT minus skew, where both PEs are NDF for the skew duration.
* NDF to DF Transition(s): at t=SCT

This split-behavior ensures a smooth DF role transition with minimal loss.
Using the SCT approach, the negative effect of the timer to allow the reception of RT-4 from other PE nodes is mitigated. Furthermore, the BGP Ethernet Segment route (RT-4) transmission delay (from PE2 to PE1) becomes a non-issue. The SCT approach shortens the 3-second timer window to the order of milliseconds.

3.1. Concurrent Recoveries

In the eventuality 2 or more PEs in a peering Ethernet Segment group are recovering concurrently or roughly the same time, each will advertise a Service Carving Timestamp. This SCT value would correspond to what each recovering PE considers the "end time" for DF Election. A similar situation arises in sequentially recovering PEs, when a second PE recovers approximately at the time of the first PE’s advertised SCT expiry, and with its own new SCT-2 outside of the initial SCT window.

In the case of multiple concurrent DF elections, each initiated by one of the recovering PEs, the SCTs must be ordered chronologically. All PEs shall execute only a single DF Election at the service carving time corresponding to the largest (latest) received timestamp value. This DF Election will involve all active PEs in a unified DF Election update.

Example:

1. Initial State: PE1 is in a steady state, with services elected at PE1.

2. Recovery of PE2: PE2 recovers at time t=100 and advertises RT-4 with a target SCT value of t=103 to its partners (PE1)

3. Timer Initiation by PE2: PE2 starts a 3 second timer to allow the reception of RT-4 from other PE nodes.

4. Timer Initiation by PE1: PE1 starts the service carving timer, with the remaining time until t=103.

5. Recovery of PE3: PE3 recovers at time t=102 and advertises RT-4 with a target SCT value of t=105 to its partners (PE1, PE2).

6. Timer Initiation by PE3: PE3 starts a 3 second timer to allow the reception of RT-4 from other PE nodes.

7. Timer Update by PE2: PE2 cancels the running timer and starts the service carving timer with the remaining time until t=105.
8. Timer Update by PE1: PE1 updates its service carving timer, with the remaining time until t=105.

9. Service Carving: PE1, PE2, and PE3 perform service carving at the absolute time of t=105.

In the eventuality a PE in an Ethernet Segment group recovers during the discovery window specified in Section 8.5 of [RFC7432], and does not support or advertise the T-bit, then all PEs in the current peering sequence SHALL immediately revert to the default [RFC7432] behavior.

4. Backwards Compatibility

For the DF election procedures to achieve global convergence and unanimity within a redundancy group, it is essential that all participating PEs agree on the DF election algorithm to be employed. However, it is possible that some PEs may continue to use the existing modulo-based DF election algorithm from [RFC7432] and not utilize the new Service Carving Time (SCT) BGP extended community. PEs that operate using the baseline DF election mechanism will simply discard the new SCT BGP extended community as unrecognized. [RFC7432] and do not rely on the new SCT BGP extended community.

A PE can indicate its willingness to support clock-synchronized carving by signaling the new ‘T’ DF Election Capability and including the new SCT BGP extended community along with the Ethernet Segment Route (Type-4). If one or more PEs attached to the Ethernet Segment do not signal T=1, then all PEs in the Ethernet Segment SHALL revert to the timer-based approach as specified in [RFC7432]. This reversion is particularly crucial in preventing VLAN shuffling when more than two PEs are involved.

5. Security Considerations

The mechanisms in this document use EVPN control plane as defined in [RFC7432]. Security considerations described in [RFC7432] are equally applicable.

For the new SCT Extended Community, attack vectors may be setting the value to zero, to a value in the past or to large times in the future. The procedures in this document address implicitly what occurs with a carving time in the past, as this would be a naturally occurring event with a large BGP propagation delay: the receiving PE SHALL treat the DF Election at the peer as having occurred already, and proceed without starting any timer to further delay service carving. For timestamp values in the future, a rogue PE may be advertising a value inconsistent with its local behavior. This is no
different than a rogue PE setting all its DF Election results inconsistently to its peers using (or ignoring adherence to) the procedures from [RFC7432], and the result would similarly be duplicate or dropped traffic. It is left to implementations to decide what consists an "unreasonably large" SCT value.

This document uses MPLS and IP-based tunnel technologies to support data plane transport. Security considerations described in [RFC7432] and in [RFC8365] are equally applicable.

6. IANA Considerations

IANA maintains the "EVPN Extended Community Sub-Types" registry set up by [RFC7153]. IANA is requested to confirm the First Come First Served assignment as follows:

<table>
<thead>
<tr>
<th>Sub-Type Value</th>
<th>Name</th>
<th>Reference</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x0F</td>
<td>Service Carving Timestamp</td>
<td>This document</td>
<td>TBD</td>
</tr>
</tbody>
</table>

IANA should replace the field TBD with the date of publication of this document as an RFC.

IANA maintains the "DF Election Capabilities" registry set up by [RFC8584]. IANA is requested to make the following assignment from this registry:

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Reference</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>Time Synchronization</td>
<td>This document</td>
<td>TBD</td>
</tr>
</tbody>
</table>

IANA should replace the field TBD with the date of publication of this document as an RFC.

7. Normative References


Brissette, et al. Expires 9 January 2025
Appendix A. Contributors

In addition to the authors listed on the front page, the following co-authors have also contributed substantially to this document:

Gaurav Badoni  
Cisco  
Email: gbadoni@cisco.com

Dhananjaya Rao  
Cisco  
Email: dhrao@cisco.com

Appendix B. Acknowledgements

Authors would like to acknowledge helpful comments and contributions of Satya Mohanty and Bharath Vasudevan. Also thank you to Anoop Ghanwani and Gunter van de Velde for their thorough review with valuable comments and corrections.
Authors’ Addresses

Patrice Brissette (editor)
Cisco
Email: pbrisset@cisco.com

Ali Sajassi
Cisco
Email: sajassi@cisco.com

Luc Andre Burdet
Cisco
Email: lburdet@cisco.com

John Drake
Independent
Email: je_drake@yahoo.com

Jorge Rabadan
Nokia
Email: jorge.rabdan@nokia.com
Initializing a DNS Resolver with Priming Queries
draft-ietf-dnsop-rfc8109bis-06

Abstract

This document describes the queries that a DNS resolver should emit
to initialize its cache. The result is that the resolver gets both a
current NS Resource Record Set (RRset) for the root zone and the
necessary address information for reaching the root servers.

This document, when published, obsoletes RFC 8109. See Section 1.1
for the list of changes from RFC 8109.

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 30 January 2025.

Copyright Notice

Copyright (c) 2024 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
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
Recursive DNS resolvers need a starting point to resolve queries. [RFC1034] describes a common scenario for recursive resolvers: they begin with an empty cache and some configuration for finding the names and addresses of the DNS root servers. [RFC1034] describes that configuration as a list of servers that will give authoritative answers to queries about the root. This has become a common implementation choice for recursive resolvers, and is the topic of this document.

This document describes the steps needed for this common implementation choice. Note that this is not the only way to start a recursive name server with an empty cache, but it is the only one described in [RFC1034]. Some implementers have chosen other directions, some of which work well and others of which fail (sometimes disastrously) under different conditions. For example, an implementation that only gets the addresses of the root name servers from configuration, not from the DNS as described in this document, will have stale data that could cause slower resolution.
This document only deals with recursive name servers (recursive resolvers, resolvers) for the IN class.

1.1. Changes from RFC 8109

This document obsoletes [RFC8109]. The significant changes from RFC 8109 are:

* Added section on the content of priming information.

* Added paragraph about no expectation that the TC bit in responses will be set.

* Added paragraph about RFC 9471 and requirements on authoritative servers and the TC bit. This clarified the role of glue records and truncation for responses from the root zone.

* Changed "man-in-the-middle" to "machine-in-the-middle" to be both less sexist and more technically accurate.

* Clarified that there are other effects of machine-in-the-middle attacks.

* Clarified language for root server domain names as "root server identifiers".

* Added short discussion of post-priming strategies.

* Added informative references to RSSAC documents.

* Added short discussion about this document and private DNS.

* Clarified that machine-in-the-middle attacks could be successful for non-signed TLDs.

* Added discussion of where resolvers that pre-fetch should get the root NS addresses.

* Elevated the expectations in "Expected Properties of the Priming Response" to MUST-level.

* Clarified that "currently" means at the time that this document is published.

* Added a note about priming and RFC 8806.

* Added a reference to research about discontinued root server addresses.
1.2. 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 BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

See [RSSAC026v2] for terminology that relates to the root server system.

2. Description of Priming

Priming is the act of finding the list of root servers from a configuration that lists some or all of the purported IP addresses of some or all of those root servers. In priming, a recursive resolver starts with no cached information about the root servers, and finishes with a full list of their names and their addresses in its cache.

Priming is described in Sections 5.3.2 and 5.3.3 of [RFC1034]. (It is called "SBELT", a "safety belt" structure, in that document.) The scenario used in that description, that of a recursive server that is also authoritative, is no longer as common.

The configured list of IP addresses for the root servers usually comes from the vendor or distributor of the recursive server software. This list is usually correct and complete when shipped, but may become out of date over time.

The domain names for the root servers are called the "root server identifiers". This list has been stable since 1997, but the IPv4 and IPv6 addresses for the root server identifiers sometimes change. Research shows that after those addresses change, some resolvers never get the new addresses; for example, see [OLD-J].

Therefore, it is important that resolvers be able to cope with change, even without relying upon configuration updates to be applied by their operator. Root server identifier and address changes are the main reasons that resolvers need to use priming to get a full and accurate list of root servers, instead of just using a statically configured list.

See [RSSAC023v2] for a history of the root server system.

Although this document is targeted at the global DNS, it also could apply to a private DNS as well. These terms are defined in [RFC9499].
Some systems serve a copy of the full root zone on the same server as the resolver, such as is described in [RFC8806]. In such a setup, the resolver primes its cache using the same methods as described in the rest of this document.

2.1. Content of Priming Information

As described above, the configuration for priming is a list of IP addresses. The priming information in software may be in any format that gives the software the addresses associated with at least some of the root server identifiers.

Some software has configuration that also contains the root server identifiers (such as "L.ROOT-SERVERS.NET"), sometimes as comments and sometimes as data consumed by the software. For example, IANA’s "Root Hints File" at <https://www.internic.net/domain/named.root> is derived directly from the root zone and contains all of the addresses of the root server identifiers found in the root zone. It is in DNS zone file presentation format, and includes the root server identifiers. Although there is no harm to adding these names, they are not useful in the root priming process.

3. Priming Queries

A priming query is a DNS query whose response provides root server names and addresses. It has a QNAME of ".", a QTYPE of NS, and a QCLASS of IN; it is sent to one of the addresses in the configuration for the recursive resolver. The priming query can be sent over either UDP or TCP. If the query is sent over UDP, the source port SHOULD be randomly selected (see [RFC5452]). The Recursion Desired (RD) bit MAY be set to 0 or 1, although the meaning of it being set to 1 is undefined for priming queries.

The recursive resolver SHOULD use EDNS0 [RFC6891] for priming queries and SHOULD announce and handle a reassembly size of at least 1024 octets [RFC3226]. Doing so allows responses that cover the size of a full priming response (see Section 4.2) for the current set of root servers. See Section 3.3 for discussion of setting the DNSSEC OK (DO) bit (defined in [RFC4033]).

3.1. Repeating Priming Queries

The recursive resolver SHOULD send a priming query only when it is needed, such as when the resolver starts with an empty cache or when the NS RRset for the root zone has expired. Because the NS records for the root zone are not special, the recursive resolver expires those NS records according to their TTL values. (Note that a recursive resolver MAY pre-fetch the NS RRset before it expires.)
If a resolver chooses to pre-fetch the root NS RRset before that RRset has expired in its cache, it needs to choose whether to use the addresses for the root NS RRset that it already has in its cache or to use the addresses it has in its configuration. Such a resolver SHOULD send queries to the addresses in its cache in order to reduce the chance of delay due to out-of-date addresses in its configuration.

If a priming query does not get a response, the recursive resolver MUST retry the query with a different target address from the configuration.

3.2. Target Selection

In order to spread the load across all the root server identifiers, the recursive resolver SHOULD select the target for a priming query randomly from the list of addresses. The recursive resolver might choose either IPv4 or IPv6 addresses based on its knowledge of whether the system on which it is running has adequate connectivity on either type of address.

Note that this recommended method is not the only way to choose from the list in a recursive resolver’s configuration. Two other common methods include picking the first from the list, and remembering which address in the list gave the fastest response earlier and using that one. There are probably other methods in use today. However, the random method listed above SHOULD be used for priming.

3.3. DNSSEC with Priming Queries

The root NS RRset is signed and can be validated by a DNSSEC validating resolver. At the time this document is published, the addresses for the names in the root NS RRset are in the "root-servers.net" zone. All root servers are also authoritative for the "root-servers.net" zone, which allows priming responses to include the appropriate root name server A and AAAA RRsets. However, because at the time this document is published the "root-servers.net" zone is not signed, the root name server A and AAAA RRsets cannot be validated. An attacker that is able to provide a spoofed priming response can provide alternative A and AAAA RRsets and thus fool a resolver into considering addresses under the control of the attacker to be authoritative for the root zone.

A rogue root name server can view all queries from the resolver to the root and alter all unsigned parts of responses, such as the parent side NS RRsets and glue in referral responses. A resolver can be fooled into trusting child (TLD) NS addresses that are under the control of the attacker as being authoritative if the resolver:
* follows referrals from a rogue root server,

* and does not explicitly query the authoritative NS RRset at the apex of the child (TLD) zone,

* and does not explicitly query for the authoritative A and AAAA RRsets for the child (TLD) NS RRsets.

With such resolvers, an attacker that controls a rogue root server effectively controls the entire domain name space and can view all queries and alter all unsigned data undetected.

An attacker controlling a rogue root name server also has complete control over all unsigned delegations, and over the entire domain name space in case of non DNSSEC validating resolvers.

If the "root-servers.net" zone is later signed, or if the root servers are named in a different zone and that zone is signed, having DNSSEC validation for the priming queries might be valuable. The benefits and costs of resolvers validating the responses will depend heavily on the naming scheme used.

4. Priming Responses

A priming query is a normal DNS query. Thus, a root server cannot distinguish a priming query from any other query for the root NS RRset. Thus, the root server's response will also be a normal DNS response.

4.1. Expected Properties of the Priming Response

The priming response MUST have an RCODE of NOERROR, and MUST have the Authoritative Answer (AA) bit set. Also, it MUST have an NS RRset in the Answer section (because the NS RRset originates from the root zone), and an empty Authority section (because the NS RRset already appears in the Answer section). There will also be an Additional section with A and/or AAAA RRsets for the root servers pointed at by the NS RRset.

Resolver software SHOULD treat the response to the priming query as a normal DNS response, just as it would use any other data fed to its cache. Resolver software SHOULD NOT expect 13 NS RRs because, historically, some root servers have returned fewer.
4.2. Completeness of the Response

At the time this document is published, there are 13 root server operators operating a total of more than 1500 root server instances. Each has one IPv4 address and one IPv6 address. The combined size of all the A and AAAA RRsets exceeds the original 512-octet payload limit from [RFC1035].

In the event of a response where the Additional section omits certain root server address information, re-issuing of the priming query does not help with those root name servers that respond with a fixed order of addresses in the Additional section. Instead, the recursive resolver needs to issue direct queries for A and AAAA RRsets for the remaining names. At the time this document is published, these RRsets would be authoritatively available from the root name servers.

If some root server addresses are omitted from the Additional section, there is no expectation that the TC bit in the response will be set to 1. At the time that this document is written, many of the root servers are not setting the TC bit when omitting addresses from the Additional section.

Note that [RFC9471] updates [RFC1035] with respect to the use of the TC bit. It says "If message size constraints prevent the inclusion of all glue records for in-domain name servers, the server must set the TC (Truncated) flag to inform the client that the response is incomplete and that the client should use another transport to retrieve the full response." Because the priming response is not a referral, root server addresses in the priming response are not considered glue records. Thus, [RFC9471] does not apply to the priming response and root servers are not required to set the TC bit if not all root server addresses fit within message size constraints. There are no requirements on the number of root server addresses that a root server must include in a priming response.

5. Post-Priming Strategies

When a resolver has a zone’s NS RRset in cache, and it gets a query for a domain in that zone that cannot be answered from its cache, the resolver has to choose which NS to send queries to. (This statement is as true for the root zone as for any other zone in the DNS.) Two common strategies for choosing are "determine the fastest name server and always use it" and "create buckets of fastness and pick randomly in the buckets". This document gives no preference to any particular strategy other than to suggest that resolvers not treat the root zone as special for this decision.
6. Security Considerations

Spoofing a response to a priming query can be used to redirect all of the queries originating from a victim recursive resolver to one or more servers for the attacker. Until the responses to priming queries are protected with DNSSEC, there is no definitive way to prevent such redirection.

An on-path attacker who sees a priming query coming from a resolver can inject false answers before a root server can give correct answers. If the attacker’s answers are accepted, this can set up the ability to give further false answers for future queries to the resolver. False answers for root servers are more dangerous than, say, false answers for Top-Level Domains (TLDs), because the root is the highest node of the DNS. See Section 3.3 for more discussion.

In both of the scenarios above, a validating resolver will be able to detect the attack if its chain of queries comes to a zone that is signed, but not for those that are unsigned.

7. IANA Considerations

This document does not require any IANA actions.

8. References

8.1. Normative References


8.2. Informative References

[OLD-J] Wessels, D., "Thirteen Years of 'Old J Root'", 2015,
<https://indico.dns-oarc.net/event/24/contributions/378/>.

[RFC8806] Kumari, W. and P. Hoffman, "Running a Root Server Local to
a Resolver", RFC 8806, DOI 10.17487/RFC8806, June 2020,

[RSSAC023v2]
"History of the Root Server System", 2016,

Koch, et al.  Expires 30 January 2025
Appendix A. Acknowledgements

RFC 8109 was the product of the DNSOP WG and benefitted from the reviews done there. This document also benefitted from review by Duane Wessels.

Authors’ Addresses

Peter Koch
DENIC eG
Kaiserstrasse 75-77
60329 Frankfurt
Germany
Phone: +49 69 27235 0
Email: pk@DENIC.DE

Matt Larson
ICANN
Email: matt.larson@icann.org

Paul Hoffman
ICANN
Email: paul.hoffman@icann.org
IMAP MESSAGELIMIT Extension
draft-ietf-extra-imap-messagelimit-10

Abstract

The MESSAGELIMIT extension of the Internet Message Access Protocol (RFC 3501/RFC 9051) allows servers to announce a limit on the number of messages that can be processed in a single FETCH/SEARCH/STORE/COPY/MOVE (or their UID variants), APPEND or UID EXPUNGE command. This helps servers to control resource usage when performing various IMAP operations. This helps clients to know the message limit enforced by corresponding IMAP server and avoid issuing commands that would exceed such limit.

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 30 January 2025.

Copyright Notice

Copyright (c) 2024 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 Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised 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 and Overview .................................. 3
2. Document Conventions ........................................ 3
3. The MESSAGELIMIT extension .................................. 3
   3.1. Returning limits on the number of messages processed in a single SEARCH/FETCH/STORE/COPY/MOVE/APPEND/EXPUNGE command ......................................................... 4
   3.2. UIDAFTER and UIDBEFORE SEARCH criteria .................. 7
   3.3. Interaction with SORT and THREAD extensions .......... 8
   3.4. Interaction with SEARCHRES extension and IMAP4rev2 ... 8
4. Interoperability Considerations ............................... 8
   4.1. Effects of MESSAGELIMIT/SAVELIMIT extensions on non compliant clients ........................................ 8
   4.2. Maintaining Compatibility ................................. 9
5. Formal syntax .................................................. 10
6. Security Considerations ....................................... 10
7. IANA Considerations ........................................... 10
   7.1. Changes/additions to the IMAP4 capabilities registry ... 10
8. Acknowledgments ............................................... 11
9. References ..................................................... 11
   9.1. Normative References ..................................... 11
   9.2. Informative References ................................... 12
Index .......................................................... 12
Authors’ Addresses .............................................. 12
1. Introduction and Overview

This document defines an extension to the Internet Message Access Protocol [RFC3501] for announcing a server limit on the number of messages that can be processed in a single FETCH/SEARCH/STORE/COPY/MOVE (or their UID variants), APPEND or UID EXPUNGE command. This extension is compatible with both IMAP4rev1 [RFC3501] and IMAP4rev2 [RFC9051].

2. Document Conventions

In protocol examples, this document uses a prefix of "C: " to denote lines sent by the client to the server, and "S: " for lines sent by the server to the client. Lines prefixed with "// " are comments explaining the previous protocol line. These prefixes and comments are not part of the protocol. Lines without any of these prefixes are continuations of the previous line, and no line break is present in the protocol unless specifically mentioned.

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

Other capitalised words are IMAP key words [RFC3501][RFC9051] or key words from this document.

3. The MESSAGELIMIT extension

An IMAP server advertises support for the MESSAGELIMIT extension by including "MESSAGELIMIT=<limit>" capability in the CAPABILITY response/response code, where "<limit>" is a positive integer that conveys the maximum number of messages that can be processed in a single [UID] SEARCH/FETCH/STORE/COPY/MOVE, APPEND or UID EXPUNGE command.

An IMAP server that only enforces message limit on [UID] COPY/APPEND commands would include the "SAVELIMIT=<limit>" capability (instead of the "MESSAGELIMIT=<limit>") in the CAPABILITY response/response code.

The limit advertised in the MESSAGELIMIT or SAVELIMIT capability SHOULD NOT be lower than 1000 messages.
3.1. Returning limits on the number of messages processed in a single
SEARCH/FETCH/STORE/COPY/MOVE/APPEND/EXPUNGE command

If a server implementation doesn't allow more than \(<N>\) messages to be
operated on by a single COPY/UID COPY command, it MUST fail the
command by returning a tagged NO response with the MESSAGELIMIT
response code defined below. No messages are copied in this case.
If a server implementation doesn't allow more than \(<N>\) messages to be
operated on by a single SEARCH/FETCH/STORE/MOVE (or their UID
variants), APPEND or UID EXPUNGE command, it MUST return the
MESSAGELIMIT response code defined below:

MESSAGELIMIT The server doesn’t allow more than \(<N>\) messages to be operated
on by a single SEARCH/FETCH/STORE/COPY/MOVE command (or their
UID variants). The lowest processed UID is \(<LastUID>\). The
client needs to repeat the operation for remaining messages, if
required.

The server doesn’t allow more than \(<N>\) deleted messages to be
operated on by a single UID EXPUNGE command. The lowest
processed UID is \(<LastUID>\). The client needs to repeat the
operation for remaining messages, if required.

Note that when the MESSAGELIMIT response code is returned, the
server is REQUIRED to process messages from highest to lowest
UIDs.

Note that when the MESSAGELIMIT response code is similar to the
LIMIT ([RFC9051]) response code, but it provides more details
on the exact type of the limit and how to resume the command
once the limit is exceeded.

In the following example the \(<N>\) value is 1000 and the lowest
processed UID \(<LastUID>\) is 23221.

C: 03 FETCH 10000:14589 (UID FLAGS)
S: * 14589 FETCH (FLAGS (\Seen) UID 25000)
S: * 14588 FETCH (FLAGS (\Answered) UID 24998)
S: ... further 997 fetch responses
S: * 13590 FETCH (FLAGS () UID 23221)
S: 03 OK [MESSAGELIMIT 1000 23221] FETCH completed with 1000 partial
results

In the following example the client searches for UNDELETED UIDs
between 22000:25000. The total number of searched messages
(note, NOT the number of matched messages) exceeds the server’s
published 1000 messages limit.
C: 04 UID SEARCH UID 22000:25000 UNDELETED
S: * SEARCH 25000 24998 (... UIDs ...) 23221
S: 04 OK [MESSAGELIMIT 1000 23221] SEARCH completed with 1000 partial results

The following example demonstrates copy of messages with UIDs between 18000:21000. The total message count exceeds the server’s published 1000 messages limit. As COPY/UID COPY needs to atomic (as per [RFC3501]/[RFC9051]), no messages are copied.

C: 05 UID COPY 18000:21000 "Trash"
S: 05 NO [MESSAGELIMIT 1000 20001] Too many messages to copy, try a smaller subset

The following example shows MOVE of messages with UIDs between 18000:21000. The total message count exceeds the server’s published 1000 messages limit. (Unlike COPY/UID COPY, MOVE/UID MOVE don’t need to be atomic.) The client that wants to move all messages in the range and observes a MESSAGELIMIT response code, can repeat the UID MOVE command with the same parameter. (For the MOVE command, the message set parameter need to be updated before repeating the command.) The client needs to keep doing this until the MESSAGELIMIT response is not returned (or until a tagged NO/BAD is returned).

C: 06 UID MOVE 18000:21000 "Archive/2021/2021-12"
S: * OK [COPYUID 1397597919 20001:21000 22363:23362] Some messages were not moved
S: * 12336 EXPUNGE
S: * 12335 EXPUNGE
... 
S: * 11337 EXPUNGE
S: 06 OK [MESSAGELIMIT 1000 20001] MOVE completed for the last 1000 messages

The following example shows update of flags for messages with UIDs between 18000:20000. The total number of existing messages in the UID range exceeds the server’s published 1000 messages limit. The client that wants to change flags for all messages in the range and observes a MESSAGELIMIT response code, can repeat the UID STORE command with the updated UID range that doesn’t include the UID returned in the MESSAGELIMIT response code. (For the STORE command, the message set parameter also need to be updated before repeating the command.) The client needs to keep doing this until the MESSAGELIMIT response is not returned (or until a tagged NO/BAD is returned).
The following example shows removal of messages (using UID EXPUNGE) that have \Deleted flag set with UIDs between 11000:13000. The total message count of messages with \Deleted flag set exceeds the server’s published 1000 messages limit. The client that wants to remove all messages marked as \Deleted in the range and observes a MESSAGELIMIT response code, can repeat the UID EXPUNGE command with the same parameter. The client needs to keep doing this until the MESSAGELIMIT response is not returned (or until a tagged NO/BAD is returned).

```
C: 08 UID EXPUNGE 11000:13000
S: * 4306 EXPUNGE
S: * 4305 EXPUNGE
... 
S: * 3307 EXPUNGE
S: 08 OK [MESSAGELIMIT 1000 11627] UID EXPUNGE completed for the last 1000 messages
```

The following example shows removal of messages (using EXPUNGE) that have \Deleted flag set. Unlike UID EXPUNGE, the server MUST NOT impose any message limit when processing EXPUNGE.

```
C: 09 EXPUNGE
S: * 4306 EXPUNGE
S: * 4305 EXPUNGE
... 
S: * 3307 EXPUNGE
S: * 112 EXPUNGE
S: 09 OK EXPUNGE completed
```

Similarly, the server MUST NOT impose any message limit when processing a "CLOSE" or a "STATUS UNSEEN" command.

The following example shows use of MESSAGELIMIT response code together with the PARTIAL [RFC9394] extension. The total message count (as specified by the PARTIAL range) exceeds the server’s published 1000 messages limit, so the server refuses to do any work in this case.

```
C: 07 UID STORE 18000:20000 +FLAGS (\Seen)
S: * 11215 FETCH (FLAGS (\Seen \Deleted) UID 20000)
S: * 11214 FETCH (FLAGS (\Seen \Answered \Deleted) UID 19998)
... 
S: * 10216 FETCH (FLAGS (\Seen) UID 19578)
S: 07 OK [MESSAGELIMIT 1000 19578] STORE completed for the last 1000 messages
```
C: 10 UID FETCH 22000:25000 (UID FLAGS MODSEQ) (PARTIAL -1:-1500)
S: 10 NO [MESSAGELIMIT 1000] FETCH exceeds the maximum 1000 message limit

Without the PARTIAL parameter the above UID FETCH can look like this:

C: 10 UID FETCH 22000:25000 (UID FLAGS MODSEQ)
S: * 12367 FETCH (FLAGS (\Seen \Deleted) UID 23007)
S: * 12366 FETCH (FLAGS (\Seen \Answered \Deleted) UID 23114)
... 
S: * 13366 FETCH (FLAGS (\Seen) UID 24598)
S: 10 OK [MESSAGELIMIT 1000 23007] FETCH exceeds the maximum 1000 message limit

Note that when the server needs to return both EXPUNGEISSUED ([RFC9051]) and MESSAGELIMIT response codes, the former MUST be returned in the tagged OK response, while the latter MUST be returned in an untagged NO response. The following example demonstrates that:

C: 11 FETCH 10000:14589 (UID FLAGS)
S: * 14589 FETCH (FLAGS (\Seen) UID 25000)
S: * 14588 FETCH (FLAGS (\Answered) UID 24998)
S: ... further 997 fetch responses
S: * 13590 FETCH (FLAGS () UID 23221)
S: * NO [MESSAGELIMIT 1000 23221] FETCH completed with 1000 partial results
S: 11 OK [EXPUNGEISSUED] Some messages were also expunged

When IMAP MULTIAPPEND [RFC3502] extension is also supported by the server, the message limit also applies to the APPEND command. As MULTIAPPEND APPEND needs to atomic (as per [RFC3502]), no messages are appended when the server MESSAGELIMIT is exceeded.

3.2. UIDAFTER and UIDBEFORE SEARCH criteria

The MESSAGELIMIT extension also defines 2 extra SEARCH keys: UIDAFTER and UIDBEFORE, which make it easier to convert a single UID to a range of UIDs.

"UIDAFTER <uid>" - Messages that have a UID greater than the specified UID. This is semantically the same as "UID <uid>+1:".

"UIDBEFORE <uid>" - Messages that have a UID less than the specified UID. This is semantically the same as "UID 1:<uid>-1" (or if <uid> has the value 1, then the empty set).
These 2 SEARCH keys are particularly useful when the SEARCHRES [RFC5182] extension is also supported, but they can be used without it. For example, this allows a SEARCH that sets the "$" marker to be converted to a range of messages in a subsequent SEARCH, and both SEARCH requests can be pipelined.

C: 12 UID SEARCH UIDAFTER 25000 UNDELETED
S: * SEARCH 27800 27798 (... 250 UIDs ...) 25001
S: 12 OK SEARCH completed

3.3. Interaction with SORT and THREAD extensions

Servers that advertise MESSAGELIMIT N will be unable to execute a THREAD [RFC5256] command in a mailbox with more than N messages.

Servers that advertise MESSAGELIMIT N might be unable to execute a SORT [RFC5256] command in a mailbox with more than N messages, unless they maintain indices for different SORT orders they support. In absence of such indeces server implementors will need to decide whether their server advertises SORT or MESSAGELIMIT capability.

3.4. Interaction with SEARCHRES extension and IMAP4rev2

Servers that support both MESSAGELIMIT and SEARCHRES [RFC5182] extensions MUST truncate SEARCH SAVE result stored in the $ variable when the SEARCH command succeeds, but the MESSAGELIMIT response code is returned. For example, if the following SEARCH would have returned 1200 results in absence of MESSAGELIMIT, and the MESSAGELIMIT is 1000, only 1000 matching results will be saved in the $ variable:

C: D0004 UID SEARCH RETURN (SAVE) SINCE 1-Jan-2004 NOT FROM "Smith" UID 22000:2 5000 UNDELETED
S: D0004 OK [MESSAGELIMIT 1000 1179] SEARCH completed with 1000 partial results saved

4. Interoperability Considerations

4.1. Effects of MESSAGELIMIT/SAVELIMIT extensions on non compliant clients

A server that advertises the MESSAGELIMIT=N capability would have the following effect on clients that don’t support this capability:

- Operations on a mailbox that has <= N messages are not affected.

- In a mailbox with more than N messages:

  - An attempt to COPY/UID COPY more than N messages will always fail.
- EXPUNGE and CLOSE will always operate on the full mailbox, so they are not affected.

- Other commands like FETCH, SEARCH and MOVE will be effectively restricted to the last N messages of the mailbox. In particular unextended SEARCHes intended for counting of messages with or without a particular set of flags would return incorrect counts.

4.2. Maintaining Compatibility

It is important to understand that the above effects essentially abandon existing clients with respect to operation on large mailboxes. Suppose, for example, that a user is accessing a large and active mailing list via IMAP – the mailing list gets on the order of 1500 posts per week. When the user returns from a week-long vacation and catches up on the mailing list, the user’s client will be fetching information about 1500 messages. If the server has a MESSAGELIMIT of 1000, the client will only be able to download 1000 of most recent messages; the client will not understand why, will not be prepared to recover from the situation, and will act as if it is interacting with a broken server.

In order to give clients time to implement this extension, servers should not be strict about applying the MESSAGELIMIT at first. One possible approach is to advertise a MESSAGELIMIT but not enforce it at all for a while. Clients that understand this extension will comply, reducing load on the server, but clients that do not understand the limit will continue to work in all situations.

Another approach, perhaps phased in later, is to advertise one limit but to treat it as a soft limit and to begin enforcement at a higher, unadvertised hard limit. In the above example, perhaps the server might advertise 1000 but actually enforce a limit of 10,000. Again, clients that understand MESSAGELIMIT will comply with the limit of 1000, but other clients will still interoperate up to the higher threshold.

Attempts to go beyond the advertised limit can be logged so that client understanding of MESSAGELIMIT can be tracked. If implementation and deployment of this extension becomes common, it may at some point be acceptable to strictly enforce the advertised limit and to accept that the remaining clients will, indeed, no longer work properly with mailboxes above that limit.
5. Formal syntax

The following syntax specification uses the Augmented Backus-Naur Form (ABNF) notation as specified in [ABNF].

Non-terminals referenced but not defined below are as defined by IMAP4 [RFC3501].

Except as noted otherwise, all alphabetic characters are case-insensitive. The use of upper or lower case characters to define token strings is for editorial clarity only. Implementations MUST accept these strings in a case-insensitive fashion.

\[
\text{capability} = / \text{"MESSAGELIMIT=}" \text{message-limit} /
\text{"SAVELIMIT=}" \text{message-limit} / ;; <\text{capability}> \text{from [RFC3501]}
\text{message-limit} = \text{nz-number}
\text{resp-text-code} = / \text{"MESSAGELIMIT" SP message-limit [SP uniqueid]} \\
;; No more than \text{nz-number} messages can be processed \\
;; by any command at a time. The last (lowest) processed \\
;; UID is uniqueid. \\
;; The last parameter is omitted, when not known.
\]

6. Security Considerations

This document defines an additional IMAP4 capability. As such, it does not change the underlying security considerations of [RFC3501] and IMAP4rev2 [RFC9051].

This document defines an optimization that can both reduce the amount of work performed by the server, as well as the amount of data returned to the client. Use of this extension is likely to cause the server and the client to use less memory than when the extension is not used, which can in turn help to protect from Denial-of-Service attacks. However, as this is going to be new code in both the client and the server, rigorous testing of such code is required in order to avoid introducing of new implementation bugs.

7. IANA Considerations

7.1. Changes/additions to the IMAP4 capabilities registry

IMAP4 capabilities are registered by publishing a standards track or IESG approved Informational or Experimental RFC. The registry is currently located at:
IANA is requested to add registrations of "MESSAGELIMIT=" and "SAVELIMIT=" capabilities to this registry, both pointing to this document.

8. Acknowledgments

This document was motivated by the Yahoo! team and their questions about best client practices for dealing with large mailboxes.

Editor of this document would like to thank the following people who provided useful comments, contributed text or participated in discussions of this document: Timo Sirainen, Barry Leiba, Ken Murchison and Arnt Gulbrandsen.

9. References

9.1. Normative References


9.2. Informative References


Index

MESSAGELIMIT (response code)
Section 3.1, Paragraph 2.2.1

Authors’ Addresses

Alexey Melnikov
Isode Limited
Email: alexey.melnikov@isode.com
URI:   https://www.isode.com

Arun Prakash Achuthan
Yahoo!
Email: arunprakash@myyahoo.com

Vikram Nagulakonda
Yahoo!
Email: nvikram_imap@yahoo.com

Luis Alves
Email: luis.alves@lafaspot.com
Compression Dictionary Transport

draft-ietf-httpbis-compression-dictionary-12

Abstract

This specification defines a mechanism for using designated HTTP responses as an external dictionary for future HTTP responses for compression schemes that support using external dictionaries (e.g., Brotli (RFC 7932) and Zstandard (RFC 8878)).

About This Document

This note is to be removed before publishing as an RFC.

Status information for this document may be found at https://datatracker.ietf.org/doc/draft-ietf-httpbis-compression-dictionary/.

Discussion of this document takes place on the HTTP Working Group mailing list (mailto:ietf-http-wg@w3.org), which is archived at https://lists.w3.org/Archives/Public/ietf-http-wg/. Working Group information can be found at https://httpwg.org/.

Source for this draft and an issue tracker can be found at https://github.com/httpwg/http-extensions/labels/compression-dictionary.

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."
1.  Introduction ............................................. 3
2.  Dictionary Negotiation ................................. 6
3.  The ‘compression-dictionary’ Link Relation Type ...... 10
4.  Dictionary-Compressed Brotlı .......................... 11
5.  Dictionary-Compressed Zstandard ........................ 11
6.  Negotiating the content encoding ................... 12
7.  IANA Considerations .................................... 13
8.  Compatibility Considerations .......................... 15
9.  Security Considerations ............................... 15
1. Introduction

This specification defines a mechanism for using designated [HTTP] responses as an external dictionary for future HTTP responses for compression schemes that support using external dictionaries (e.g., Brotli [RFC7932] and Zstandard [RFC8878]).

This document describes the HTTP headers used for negotiating dictionary usage and registers media types for content encoding Brotli and Zstandard using a negotiated dictionary.

The negotiation of dictionary usage leverages HTTP’s content negotiation (see Section 12 of [HTTP]) and is usable with all versions of HTTP.

1.1. Use Cases

Any HTTP response can be specified to be used as a compression dictionary for future HTTP requests which provides a lot of flexibility. There are two common use cases that are seen frequently:

1.1.1. Version Upgrade

Using a previous version of a resource as a dictionary for a newer version enables delivery of a delta-compressed version of the changes, usually resulting in significantly smaller responses than can be achieved by compression alone.

For example:
<table>
<thead>
<tr>
<th>Client</th>
<th>Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET /app.v1.js</td>
<td></td>
</tr>
<tr>
<td>----------------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>200 OK</td>
<td></td>
</tr>
<tr>
<td>Use-As-Dictionary: match=&quot;/app*js&quot;</td>
<td></td>
</tr>
<tr>
<td>&lt;full app.v1.js resource - 100KB compressed&gt;</td>
<td></td>
</tr>
</tbody>
</table>

Some time later ...

<table>
<thead>
<tr>
<th>Client</th>
<th>Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET /app.v2.js</td>
<td></td>
</tr>
<tr>
<td>Available-Dictionary: :pZGm1A...2a2fFG4=:</td>
<td></td>
</tr>
<tr>
<td>Accept-Encoding: gzip, br, zstd, dcb, dcz</td>
<td></td>
</tr>
<tr>
<td>----------------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>200 OK</td>
<td></td>
</tr>
<tr>
<td>Content-Encoding: dcb</td>
<td></td>
</tr>
<tr>
<td>&lt;delta-compressed app.v2.js resource - 1KB&gt;</td>
<td></td>
</tr>
</tbody>
</table>

Figure 1: Version Upgrade Example

1.1.2. Common Content

If several resources share common patterns in their responses then it can be useful to reference an external dictionary that contains those common patterns, effectively compressing them out of the responses. Some examples of this are common template HTML for similar pages across a site and common keys and values in API calls.

For example:
Figure 2: Common Content Example

1.2. Notational Conventions

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

This document uses the following terminology from Section 3 of [STRUCTURED-FIELDS] to specify syntax and parsing: Dictionary, String, Inner List, Token, and Byte Sequence.
This document uses the line folding strategies described in [FOLDING].

This document also uses terminology from [HTTP] and [HTTP-CACHING].

2. Dictionary Negotiation

2.1. Use-As-Dictionary

When responding to a HTTP Request, a server can advertise that the response can be used as a dictionary for future requests for URLs that match the rules specified in the Use-As-Dictionary response header.

The Use-As-Dictionary response header is a Structured Field [STRUCTURED-FIELDS] Dictionary with values for "match", "match-dest", "id", and "type".

2.1.1. match

The "match" value of the Use-As-Dictionary header is a String value that provides the URL Pattern [URLPattern] to use for request matching.

The URL Pattern used for matching does not support using regular expressions.

The following algorithm is used to validate that a given String value is a valid URL Pattern that does not use regular expressions and is for the same Origin (Section 4.3.1 of [HTTP]) as the dictionary. It will return TRUE for a valid match pattern and FALSE for an invalid pattern that MUST NOT be used:

1. Let MATCH be the value of "match" for the given dictionary.
2. Let URL be the URL of the dictionary request.
3. Let PATTERN be a URL Pattern [URLPattern] constructed by setting input=MATCH, and baseURL=URL.
4. If the has RegExpGroups attribute of PATTERN is TRUE then return FALSE.
5. Return TRUE.

The "match" value is required and MUST be included in the Use-As-Dictionary response header for the dictionary to be considered valid.
2.1.2. match-dest

The "match-dest" value of the Use-As-Dictionary header is an Inner List of String values that provides a list of [FETCH] request destinations for the dictionary to match.

An empty list for "match-dest" MUST match all destinations.

For clients that do not support request destinations, the client MUST treat it as an empty list and match all destinations.

The "match-dest" value is optional and defaults to an empty list.

2.1.3. id

The "id" value of the Use-As-Dictionary header is a String value that specifies a server identifier for the dictionary. If an "id" value is present and has a string length longer than zero then it MUST be sent to the server in a "Dictionary-ID" request header when the client sends an "Available-Dictionary" request header for the same dictionary (see Section 2.2).

The server identifier MUST be treated as an opaque string by the client.

The server identifier MUST NOT be relied upon by the server to guarantee the contents of the dictionary. The dictionary hash MUST be validated before use.

The "id" value string length (after any decoding) supports up to 1024 characters.

The "id" value is optional and defaults to the empty string.

2.1.4. type

The "type" value of the Use-As-Dictionary header is a Token value that describes the file format of the supplied dictionary.

"raw" is the only defined dictionary format which represents an unformatted blob of bytes suitable for any compression scheme to use.

If a client receives a dictionary with a type that it does not understand, it MUST NOT use the dictionary.

The "type" value is optional and defaults to "raw".
2.1.5. Examples

2.1.5.1. Path Prefix

A response that contained a response header:

NOTE: '\' line wrapping per RFC 8792

Use-As-Dictionary: \n   match="/product/*", match-dest=('document')

Would specify matching any document request for a URL with a path
prefix of /product/ on the same Origin (Section 4.3.1 of [HTTP]) as
the original request.

2.1.5.2. Versioned Directories

A response that contained a response header:

Use-As-Dictionary: match="/app/*/main.js"

Would match main.js in any directory under /app/.

2.2. Available-Dictionary

When a HTTP client makes a request for a resource for which it has an
appropriate dictionary, it can add a "Available-Dictionary" request
header to the request to indicate to the server that it has a
dictionary available to use for compression.

The "Available-Dictionary" request header is a Structured Field
[STRUCTURED-FIELDS] Byte Sequence containing the [SHA-256] hash of
the contents of a single available dictionary.

The client MUST only send a single "Available-Dictionary" request
header with a single hash value for the best available match that it
has available.

For example:

Available-Dictionary: :pZGm1Av0IEBKARczz7exkNYsZb8LzaMrV7J32a2fFG4=:

2.2.1. Dictionary freshness requirement

To be considered as a match, the dictionary resource MUST be either
fresh [HTTP-CACHING] or allowed to be served stale (see eg
[RFC5861]).

Meenan & Weiss          Expires 16 February 2025                [Page 8]
2.2.2. Dictionary URL matching

When a dictionary is stored as a result of a "Use-As-Dictionary" directive, it includes "match" and "match-dest" strings that are used to match an outgoing request from a client to the available dictionaries.

Dictionaries MUST have been served from the same Origin (Section 4.3.1 of [HTTP]) as the outgoing request to match.

To see if an outbound request matches a given dictionary, the following algorithm will return TRUE for a successful match and FALSE for no-match:

1. If the current client supports request destinations:
   * Let DEST be the value of "match-dest" for the given dictionary.
   * Let REQUEST_DEST be the value of the destination for the current request.
   * If DEST is not an empty list and if REQUEST_DEST is not in the DEST list of destinations, return FALSE
2. Let BASEURL be the URL of the dictionary request.
3. Let URL represent the URL of the outbound request being checked.
4. If the Origin of BASEURL and the Origin of URL are not the same, return FALSE.
5. Let MATCH be the value of "match" for the given dictionary.
6. Let PATTERN be a URL Pattern [URLPattern] constructed by setting input=MATCH, and baseURL=BASEURL.
7. Return the result of running the "test" method of PATTERN with input=URL.

2.2.3. Multiple matching dictionaries

When there are multiple dictionaries that match a given request URL, the client MUST pick a single dictionary using the following rules:

1. For clients that support request destinations, a dictionary that specifies and matches a "match-dest" takes precedence over a match that does not use a destination.
2. Given equivalent destination precedence, the dictionary with the longest "match" takes precedence.

3. Given equivalent destination and match length precedence, the most recently fetched dictionary takes precedence.

2.3. Dictionary-ID

When a HTTP client makes a request for a resource for which it has an appropriate dictionary and the dictionary was stored with a server-provided "id" in the Use-As-Dictionary response then the client MUST echo the stored "id" in a "Dictionary-ID" request header.

The "Dictionary-ID" request header is a Structured Field [STRUCTURED-FIELDS] String of up to 1024 characters (after any decoding) and MUST be identical to the server-provided "id".

For example, given a HTTP response that set a dictionary ID:

Use-As-Dictionary: match="/app/*/main.js", id="dictionary-12345"

A future request that matches the given dictionary will include both the hash and the ID:

Available-Dictionary: :pZGmlAvOIEBKARczz7ekNYszb8LzaMrV7J32a2fFG4=:
Dictionary-ID: "dictionary-12345"

3. The ‘compression-dictionary’ Link Relation Type

This specification defines the 'compression-dictionary' link relation type [WEB-LINKING] that provides a mechanism for a HTTP response to provide a URL for a compression dictionary that is related to, but not directly used by the current HTTP response.

The 'compression-dictionary' link relation type indicates that fetching and caching the specified resource is likely to be beneficial for future requests. The response to some of those future requests are likely to be able to use the indicated resource as a compression dictionary.

Clients can fetch the provided resource at a time that they determine would be appropriate.

The response to the fetch for the compression dictionary needs to include a "Use-As-Dictionary" and caching response headers for it to be usable as a compression dictionary. The link relation only provides the mechanism for triggering the fetch of the dictionary.
The following example shows a link to a resource at https://example.org/dict.dat that is expected to produce a compression dictionary:

```
Link: <https://example.org/dict.dat>; rel="compression-dictionary"
```

4. Dictionary-Compressed Brotli

The "dcb" content encoding identifies a resource that is a "Dictionary-Compressed Brotli" stream.

A "Dictionary-Compressed Brotli" stream has a fixed header that is followed by a Shared Brotli [SHARED-BROTLI] stream. The header consists of a fixed 4-byte sequence and a 32-byte hash of the external dictionary that was used. The Shared Brotli stream is created using the referenced external dictionary and a compression window that is at most 16 MB in size.

The dictionary used for the "dcb" content encoding is a "raw" dictionary type as defined in Section 2.1.4 and is treated as a prefix dictionary as defined in section 9.2 of the Shared Brotli Compressed Data Format draft. [SHARED-BROTLI]

The 36-byte fixed header is as follows:

```
| Magic_Number:  4 fixed bytes: 0xff, 0x44, 0x43, 0x42. |
| SHA_256_Hash:  32 bytes. SHA-256 hash digest of the dictionary [SHA-256]. |
```

Clients that announce support for dcb content encoding MUST be able to decompress resources that were compressed with a window size of up to 16 MB.

With Brotli compression, the full dictionary is available during compression and decompression independent of the compression window, allowing for delta-compression of resources larger than the compression window.

5. Dictionary-Compressed Zstandard

The "dcz" content encoding identifies a resource that is a "Dictionary-Compressed Zstandard" stream.

A "Dictionary-Compressed Zstandard" stream is a binary stream that starts with a 40-byte fixed header and is followed by a Zstandard [RFC8878] stream of the response that has been compressed with an external dictionary.
The dictionary used for the "dcz" content encoding is a "raw" dictionary type as defined in Section 2.1.4 and is treated as a raw dictionary as per section 5 of RFC 8878.

The 40-byte header consists of a fixed 8-byte sequence followed by the 32-byte SHA-256 hash of the external dictionary that was used to compress the resource:

Magic_Number: 8 fixed bytes: 0x5e, 0x2a, 0x4d, 0x18, 0x20, 0x00, 0x00, 0x00.

SHA_256_Hash: 32 bytes. SHA-256 hash digest of the dictionary [SHA-256].

The 40-byte header is a Zstandard skippable frame (little-endian 0x184D2A5E) with a 32-byte length (little-endian 0x00000020) that is compatible with existing Zstandard decoders.

Clients that announce support for dcz content encoding MUST be able to decompress resources that were compressed with a window size of at least 8 MB or 1.25 times the size of the dictionary, which ever is greater, up to a maximum of 128 MB.

The window size used will be encoded in the content (currently, this can be expressed in powers of two only) and it MUST be lower than this limit. An implementation MAY treat a window size that exceeds the limit as a decoding error.

With Zstandard compression, the full dictionary is available during compression and decompression until the size of the input exceeds the compression window. Beyond that point the dictionary becomes unavailable. Using a compression window that is 1.25 times the size of the dictionary allows for full delta compression of resources that have grown by 25% between releases while still giving the client control over the memory it will need to allocate for a given response.

6. Negotiating the content encoding

When a compression dictionary is available for use for a given request, the encoding to be used is negotiated through the regular mechanism for negotiating content encoding in HTTP through the "Accept-Encoding" request header and "Content-Encoding" response header.

The dictionary to use is negotiated separately and advertised in the "Available-Dictionary" request header.
6.1. Accept-Encoding

When a dictionary is available for use on a given request, and the client chooses to make dictionary-based content-encoding available, the client adds the dictionary-aware content encodings that it supports to the "Accept-Encoding" request header. e.g.:

Accept-Encoding: gzip, deflate, br, zstd, dcb, dcz

When a client does not have a stored dictionary that matches the request, or chooses not to use one for the request, the client MUST NOT send its dictionary-aware content-encodings in the "Accept-Encoding" request header.

6.2. Content-Encoding

If a server supports one of the dictionary encodings advertised by the client and chooses to compress the content of the response using the dictionary that the client has advertised then it sets the "Content-Encoding" response header to the appropriate value for the algorithm selected. e.g.:

Content-Encoding: dcb

If the response is cacheable, it MUST include a "Vary" header to prevent caches serving dictionary-compressed resources to clients that don’t support them or serving the response compressed with the wrong dictionary:

Vary: accept-encoding, available-dictionary

7. IANA Considerations

7.1. Content Encoding

IANA is asked to enter the following into the "HTTP Content Coding Registry" registry maintained at <https://www.iana.org/assignments/http-parameters/http-parameters.xhtml>:

* Name: dcb
* Description: "Dictionary-Compressed Brotli" data format.
* Reference: This document
* Notes: Section 4
IANA is asked to enter the following into the "HTTP Content Coding Registry" registry maintained at <https://www.iana.org/assignments/http-parameters/http-parameters.xhtml>:

* Name: dcz
* Description: "Dictionary-Compressed Zstandard" data format.
* Reference: This document
* Notes: Section 5

7.2. Header Field Registration

IANA is asked to update the "Hypertext Transfer Protocol (HTTP) Field Name Registry" registry maintained at <https://www.iana.org/assignments/http-fields/http-fields.xhtml> according to the table below:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Status</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use-As-Dictionary</td>
<td>permanent</td>
<td>Section 2.1 of this document</td>
</tr>
<tr>
<td>Available-Dictionary</td>
<td>permanent</td>
<td>Section 2.2 of this document</td>
</tr>
<tr>
<td>Dictionary-ID</td>
<td>permanent</td>
<td>Section 2.3 of this document</td>
</tr>
</tbody>
</table>

Table 1

7.3. Link Relation Registration

IANA is asked to update the "Link Relation Types" registry maintained at <https://www.iana.org/assignments/link-relations/link-relations.xhtml>:

* Relation Name: compression-dictionary
* Description: Refers to a compression dictionary used for content encoding.
* Reference: This document, Section 3
8. Compatibility Considerations

It is not unusual for there to be devices on the network path that intercept, inspect and process HTTP requests (web proxies, firewalls, intrusion detection systems, etc). To minimize the risk of these devices incorrectly processing dictionary-compressed responses, compression dictionary transport MUST only be used in secure contexts (HTTPS).

9. Security Considerations

The security considerations for Brotli [RFC7932], Shared Brotli [SHARED-BROTLI] and Zstandard [RFC8878] apply to the dictionary-based versions of the respective algorithms.

9.1. Changing content

The dictionary must be treated with the same security precautions as the content, because a change to the dictionary can result in a change to the decompressed content.

The dictionary is validated using a SHA-256 hash of the content to make sure that the client and server are both using the same dictionary. The strength of the SHA-256 hash algorithm isn’t explicitly needed to counter attacks since the dictionary is being served from the same origin as the content. That said, if a weakness is discovered in SHA-256 and it is determined that the dictionary negotiation should use a different hash algorithm, the "Use-As-Dictionary" response header can be extended to specify a different algorithm and the server would just ignore any "Available-Dictionary" requests that do not use the updated hash.

9.2. Reading content

The compression attacks in Section 2.6 of [RFC7457] show that it’s a bad idea to compress data from mixed (e.g. public and private) sources -- the data sources include not only the compressed data but also the dictionaries. For example, if you compress secret cookies using a public-data-only dictionary, you still leak information about the cookies.

Not only can the dictionary reveal information about the compressed data, but vice versa, data compressed with the dictionary can reveal the contents of the dictionary when an adversary can control parts of data to compress and see the compressed size. On the other hand, if the adversary can control the dictionary, the adversary can learn information about the compressed data.
9.3. Security Mitigations

If any of the mitigations do not pass, the client MUST drop the response and return an error.

9.3.1. Cross-origin protection

To make sure that a dictionary can only impact content from the same origin where the dictionary was served, the URL Pattern used for matching a dictionary to requests (Section 2.1.1) is guaranteed to be for the same origin that the dictionary is served from.

9.3.2. Response readability

For clients, like web browsers, that provide additional protection against the readability of the payload of a response and against user tracking, additional protections MUST be taken to make sure that the use of dictionary-based compression does not reveal information that would not otherwise be available.

In these cases, dictionary compression MUST only be used when both the dictionary and the compressed response are fully readable by the client.

In browser terms, that means that both are either same-origin to the context they are being fetched from or that the response is cross-origin and passes the CORS check as defined in [FETCH].

9.3.3. Server Responsibility

As with any usage of compressed content in a secure context, a potential timing attack exists if the attacker can control any part of the dictionary, or if it can read the dictionary and control any part of the content being compressed, while performing multiple requests that vary the dictionary or injected content. Under such an attack, the changing size or processing time of the response reveals information about the content, which might be sufficient to read the supposedly secure response.

In general, a server can mitigate such attacks by preventing variations per request, as in preventing active use of multiple dictionaries for the same content, disabling compression when any portion of the content comes from uncontrolled sources, and securing access and control over the dictionary content in the same way as the response content. In addition, the following requirements on a server are intended to disable dictionary-aware compression when the client provides CORS request header fields that indicate a cross-origin request context.
The following algorithm will return FALSE for cross-origin requests where precautions such as not using dictionary-based compression should be considered:

1. If there is no "Sec-Fetch-Site" request header then return TRUE.
2. if the value of the "Sec-Fetch-Site" request header is "same-origin" then return TRUE.
3. If there is no "Sec-Fetch-Mode" request header then return TRUE.
4. If the value of the "Sec-Fetch-Mode" request header is "navigate" or "same-origin" then return TRUE.
5. If the value of the "Sec-Fetch-Mode" request header is "cors":
   * If the response does not include an "Access-Control-Allow-Origin" response header then return FALSE.
   * If the request does not include an "Origin" request header then return FALSE.
   * If the value of the "Access-Control-Allow-Origin" response header is "*" then return TRUE.
   * If the value of the "Access-Control-Allow-Origin" response header matches the value of the "Origin" request header then return TRUE.
6. return FALSE.

10. Privacy Considerations

Since dictionaries are advertised in future requests using the hash of the content of the dictionary, it is possible to abuse the dictionary to turn it into a tracking cookie.

To mitigate any additional tracking concerns, clients MUST treat dictionaries in the same way that they treat cookies [RFC6265]. This includes partitioning the storage as cookies are partitioned as well as clearing the dictionaries whenever cookies are cleared.

11. References

11.1. Normative References

11.2. Informative References

Meenan & Weiss          Expires 16 February 2025


Authors’ Addresses

Patrick Meenan (editor)
Google LLC
Email: pmeenan@google.com

Yoav Weiss (editor)
Shopify Inc
Email: yoav.weiss@shopify.com
Abstract

Deployments of Zstandard, or "zstd", can use different window sizes to limit memory usage during compression and decompression. Some browsers and user agents limit window sizes to mitigate memory usage concerns, causing interoperability issues. This document updates the window size limit in RFC8878 from a recommendation to a requirement in HTTP contexts.

About This Document

This note is to be removed before publishing as an RFC.

The latest revision of this draft can be found at https://httpwg.org/http-extensions/draft-ietf-httpbis-zstd-window-size.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-ietf-httpbis-zstd-window-size/.

Discussion of this document takes place on the HTTP Working Group mailing list (mailto:ietf-http-wg@w3.org), which is archived at https://lists.w3.org/Archives/Public/ietf-http-wg/.

Source for this draft and an issue tracker can be found at https://github.com/httpwg/http-extensions/labels/zstd-window-size.

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/.
1. Introduction

Zstandard, or "zstd", specified in [RFC8878], is a lossless data compression mechanism similar to gzip. When used with HTTP, the "zstd" content coding token signals to the decoder that the content is Zstandard-compressed.

An important property of Zstandard-compressed content is its Window_Size ([RFC8878], Section 3.1.1.1.2), which describes the maximum distance for back-references and therefore how much of the content must be kept in memory during decompression.
The minimum Window_Size is 1 KB. The maximum Window_Size is \((1<<41) + 7*(1<<38)\) bytes, which is 3.75 TB. Larger Window_Size values tend to improve the compression ratio, but at the cost of increased memory usage.

To protect against unreasonable memory usage, some browsers and user agents limit the maximum Window_Size they will handle. This causes failures to decode responses when the content is compressed with a larger Window_Size than the recipient allows, leading to decreased interoperability.

[RFC8878], Section 3.1.1.1.2 recommends that decoders support a Window_Size of up to 8 MB, and that encoders not generate frames using a Window_Size larger than 8 MB. However, it imposes no requirements.

This document updates [RFC8878] to enforce Window_Size limits on the encoder and decoder for the "zstd" HTTP content coding.

2. Conventions and Definitions

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

3. Window Size

To ensure interoperability, when using the "zstd" content coding, decoders MUST support a Window_Size of up to and including 8 MB, and encoders MUST NOT generate frames requiring a Window_Size larger than 8 MB (see Section 5.1).

4. Security Considerations

This document introduces no new security considerations beyond those discussed in [RFC8878].

Note that decoders still need to take into account that they can receive oversized frames that do not follow the window size limit specified in this document and fail decoding when such invalid frames are received.

5. IANA Considerations
5.1. Content Encoding

This document updates the entry added in [RFC8878] to the "HTTP Content Coding Registry" within the "Hypertext Transfer Protocol (HTTP) Parameters" registry:

Name: zstd

Description: A stream of bytes compressed using the Zstandard protocol with a Window Size of not more than 8 MB.

Reference: This document and [RFC8878]

6. Normative References


Acknowledgments

Zstandard was developed by Yann Collet.

The authors would like to thank Yann Collet, Klaus Post, Adam Rice, and members of the Web Performance Working Group in the W3C for collaborating on the window size issue and helping to formulate a solution. Also, thank you to Nick Terrell for providing feedback that went into RFC 8478 and RFC 8878.

Authors’ Addresses

Nidhi Jaju (editor)
Google
Shibuya Stream, 3 Chome-21-3 Shibuya, Shibuya City, Tokyo 150-0002
Japan
Email: nidhijaju@google.com
Advertising Segment Routing Policies in BGP
draft-ietf-idr-sr-policy-safi-06

Abstract

A Segment Routing (SR) Policy is an ordered list of segments (i.e.,
instructions) that represent a source-routed policy. An SR Policy
consists of one or more candidate paths, each consisting of one or
more segment lists. A headend may be provisioned with candidate
paths for an SR Policy via several different mechanisms, e.g., CLI,
NETCONF, PCEP, or BGP.

This document specifies how BGP may be used to distribute SR Policy
candidate paths. It introduces a BGP SAFI to advertise a candidate
path of a Segment Routing (SR) Policy and defines sub-TLVs for the
Tunnel Encapsulation Attribute for signaling information about these
candidate paths.

This document updates RFC9012 with extensions to the Color Extended
Community to support additional steering modes over SR Policy.

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 27 January 2025.

Copyright Notice

Copyright (c) 2024 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 Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction .................................................. 3
   1.1. Requirements Language .................................. 6
2. SR Policy Encoding ............................................. 6
   2.1. SR Policy SAFI and NLRI ................................. 6
   2.2. SR Policy and Tunnel Encapsulation Attribute ............ 8
   2.3. Applicability of Tunnel Encapsulation Attribute Sub-TLVs ........................................ 10
   2.4. SR Policy Sub-TLVs ....................................... 10
      2.4.1. Preference Sub-TLV ................................ 11
      2.4.2. Binding SID Sub-TLV ................................ 11
      2.4.3. SRv6 Binding SID Sub-TLV .......................... 13
      2.4.4. Segment List Sub-TLV ................................ 15
      2.4.5. Explicit NULL Label Policy Sub-TLV ............... 21
      2.4.6. Policy Priority Sub-TLV ............................ 23
      2.4.7. Policy Candidate Path Name Sub-TLV ............... 23
      2.4.8. Policy Name Sub-TLV ................................. 24
3. Color Extended Community ...................................... 25
4. SR Policy Operations ........................................... 27
   4.1. Advertisement of SR Policies ............................. 27
   4.2. Reception of an SR Policy NLRI .......................... 27
      4.2.1. Validation of an SR Policy NLRI ................. 28
      4.2.2. Eligibility for Local Use of an SR Policy NLRI ... 28
      4.2.3. Propagation of an SR Policy .......................... 29
5. Error Handling and Fault Management .......................... 29
6. IANA Considerations ........................................... 30
   6.1. Subsequent Address Family Identifiers (SAFI) Parameters ................................... 31
   6.2. BGP Tunnel Encapsulation Attribute Tunnel Types ........ 31
   6.3. BGP Tunnel Encapsulation Attribute sub-TLVs .......... 32
   6.4. Color Extended Community Flags ......................... 32
1. Introduction

Segment Routing (SR) [RFC8402] allows a headend node to steer a packet flow along a specific path. Intermediate per-path states are eliminated thanks to source routing.

The headend node is said to steer a flow into an SR Policy [RFC8402].

The packets steered into an SR Policy carry an ordered list of segments associated with that SR Policy.

[RFC9256] further details the concepts of SR Policy and steering into an SR Policy. These apply equally to the SR-MPLS and Segment Routing for IPv6 (SRv6) data-plane instantiations of Segment Routing using SR-MPLS and SRv6 Segment Identifiers (SIDs) as described in [RFC8402]. [RFC8660] describes the representation and processing of this ordered list of segments as an MPLS label stack for SR-MPLS. While [RFC8754] and [RFC8986] describe the same for SRv6 with the use of the Segment Routing Header (SRH).

The SR Policy related functionality described in [RFC9256] can be conceptually viewed as being incorporated in an SR Policy Module (SRPM). Following is a reminder of the high-level functionality of SRPM:

* Learning multiple candidate paths (CP) for an SR Policy via various mechanisms (CLI, NETCONF, PCEP, or BGP).

* Selection of the best candidate path for an SR Policy.

* Associating a Binding SID (BSID) to the selected candidate path of an SR Policy.
This document specifies the use of BGP to distribute one or more of the candidate paths of an SR Policy to the headend of that policy. The document describes the functionality provided by BGP and, as appropriate, provides references for the functionality which is outside the scope of BGP (i.e. resides within SRPM on the headend node).

This document specifies a way of representing SR Policy candidate paths in BGP UPDATE messages. BGP can then be used to propagate the SR Policy candidate paths to the headend nodes in a network. The usual BGP rules for BGP propagation and best-path selection are used. At the headend of a specific policy, this will result in one or more candidate paths being installed into the "BGP table". These paths are then passed to the SRPM. The SRPM may compare them to candidate paths learned via other mechanisms and will choose one or more paths to be installed in the data plane. BGP itself does not install SR Policy candidate paths into the data plane.

This document introduces a BGP subsequent address family (SAFI) for IPv4 and IPv6 address families. In UPDATE messages of those AFI/SAFIs, the NLRI identifies an SR Policy Candidate Path while the attributes encode the segment lists and other details of that SR Policy Candidate Path.

While for simplicity we may write that BGP advertises an SR Policy, it has to be understood that BGP advertises a candidate path of an SR policy and that this SR Policy might have several other candidate paths provided via BGP (via an NLRI with a different distinguisher as defined in Section 2.1), PCEP, NETCONF, or local policy configuration.

Typically, a SR Policy Controller [RFC9256] defines the set of policies and advertises them to policy headend routers (typically ingress routers). These policy advertisements use the BGP extensions defined in this document. The policy advertisement is, in most but not all cases, tailored for a specific policy headend; such an advertisement may be sent on a BGP session to that headend and not propagated any further.

Alternatively, a router (i.e., a BGP egress router) advertises SR Policies representing paths to itself. In this case, it is possible to send the policy to each headend over a BGP session to that headend, without requiring any further propagation of the policy.
An SR Policy intended only for the receiver will, in most cases, not traverse any Route Reflector (RR, [RFC4456]) (see Section 4.2.3).

In some situations, it is undesirable for a controller or BGP egress router to have a BGP session to each policy headend. In these situations, BGP Route Reflectors may be used to propagate the advertisements. In certain other deployments, it may be necessary for the advertisement to propagate through a sequence of one or more ASes within an SR Domain (refer to Section 7 for the associated security considerations). To make this possible, an attribute needs to be attached to the advertisement that enables a BGP speaker to determine whether it is intended to be a headend for the advertised policy. This is done by attaching one or more Route Target Extended Communities to the advertisement [RFC4360].

The BGP extensions for the advertisement of SR Policies include following components:

* A Subsequent Address Family Identifier (SAFI) whose NLRIs identifies an SR Policy candidate path.

* A Tunnel Type identifier for SR Policy, and a set of sub-TLVs to be inserted into the Tunnel Encapsulation Attribute (as defined in [RFC9012]) specifying segment lists of the SR Policy candidate path, as well as other information about the SR Policy.

* One or more IPv4 address-specific format route target extended community ([RFC4360]) attached to the SR Policy Candidate Path advertisement and that indicates the intended headend of such an SR Policy Candidate Path advertisement.

The SR Policy SAFI route updates use the Tunnel Encapsulation Attribute to signal an SR Policy - which is a tunnel itself. Its usage of this attribute is hence very different from [RFC9012] where this attribute is associated with a BGP route update (e.g., for Internet or VPN routes) to specify the tunnel which is used for forwarding traffic for that route. This document does not update or change the usage of the Tunnel Encapsulation Attribute as specified in [RFC9012] for existing AFI/SAFIs as specified in that document. The details of processing of the Tunnel Encapsulation Attribute for the SR Policy SAFI are specified in Section 2.2 and Section 2.3.

The northbound advertisement of the operational state of the SR Policy Candidate Paths as part of BGP-LS [RFC9552] topology information is specified in [I-D.ietf-idr-bgp-ls-sr-policy].
The signaling of Dynamic and Composite Candidate Paths (sections 5.2 and 5.3 respectively of [RFC9256]) is outside the scope of this document.

The Color Extended Community (as defined in [RFC9012]) is used to steer traffic into an SR Policy, as described in section 8.8 of [RFC9256]. The Section 3 of this document updates [RFC9012] with modifications to the format of the Flags field of the Color Extended Community by using the two leftmost bits of that field.

1.1. Requirements Language

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

2. SR Policy Encoding

2.1. SR Policy SAFI and NLRI

A SAFI is introduced in this document: the SR Policy SAFI with codepoint 73. The AFI used MUST be IPv4(1) or IPv6(2).

The SR Policy SAFI uses the NLRI format defined as follows:

```
+------------------+
| NLRI Length     | 1 octet
+------------------+
| Distinguisher    | 4 octets
+------------------+
| Policy Color     | 4 octets
+------------------+
| Endpoint         | 4 or 16 octets
+------------------+
```

Figure 1: SR Policy SAFI Format

where:

* NLRI Length: 1 octet indicating the length expressed in bits as defined in [RFC4760]. When AFI = 1 the value MUST be 96 and when AFI = 2 the value MUST be 192.

* Distinguisher: 4-octet value uniquely identifying the policy in the context of <color, endpoint> tuple. The distinguisher has no semantic value and is solely used by the SR Policy originator to
make unique (from an NLRI perspective) both for multiple candidate paths of the same SR Policy as well as candidate paths of different SR Policies (i.e. with different segment lists) with the same Color and Endpoint but meant for different headends.

* Policy Color: 4-octet value identifying (with the endpoint) the policy. The color is used to match the color of the destination prefixes to steer traffic into the SR Policy as specified in section 8 of [RFC9256].

* Endpoint: value identifies the endpoint of a policy. The Endpoint may represent a single node or a set of nodes (e.g., an anycast address). The Endpoint is an IPv4 (4-octet) address or an IPv6 (16-octet) address according to the AFI of the NLRI. The address can be either a unicast or an unspecified address (0.0.0.0 for IPv4, :: for IPv6), known as null endpoint, as specified in section 2.1 of [RFC9256].

The color and endpoint are used to automate the steering of BGP service routes on SR Policy as described in section 8 of [RFC9256].

The NLRI containing an SR Policy candidate path is carried in a BGP UPDATE message [RFC4271] using BGP multi-protocol extensions [RFC4760] with an AFI of 1 or 2 (IPv4 or IPv6) and with a SAFI of 73. The fault management and error handling in the encoding of the NLRI is specified in Section 5.

An update message that carries the MP_REACH_NLRI or MP_UNREACH_NLRI attribute with the SR Policy SAFI MUST also carry the BGP mandatory attributes. In addition, the BGP update message MAY also contain any of the BGP optional attributes.

The next-hop network address field in SR Policy SAFI (73) updates may be either a 4-octet IPv4 address or a 16-octet IPv6 address, independent of the SR Policy AFI. The length field of the next-hop address specifies the next-hop address family. If the next-hop length is 4, then the next-hop is an IPv4 address; if the next-hop length is 16, then it is a global IPv6 address; if the next-hop length is 32, then it has a global IPv6 address followed by a link-local IPv6 address. The setting of the next-hop field and its attendant processing is governed by standard BGP procedures as described in section 3 of [RFC4760] and section 3 of [RFC2545].
It is important to note that any BGP speaker receiving a BGP message with an SR Policy NLRI, the SRPM will process it only if the NLRI is among the best paths as per the BGP best-path selection algorithm. In other words, this document leverages the existing BGP propagation and best-path selection rules. Details of the procedures are described in Section 4.

It has to be noted that if several candidate paths of the same SR Policy (endpoint, color) are signaled via BGP to a headend, then it is RECOMMENDED that each NLRI uses a different distinguisher. If BGP has installed into the BGP table two advertisements whose respective NLRI have the same color and endpoint, but different distinguishers, both advertisements are passed to the SRPM as different candidate paths along with their respective originator information (i.e., ASN and BGP Router-ID) as described in section 2.4 of [RFC9256]. The ASN would be the ASN of the origin and the BGP Router-ID is determined in the following order:

* From the Route Origin Community [RFC4360] if present and carrying an IP Address, or
* As the BGP Originator ID [RFC4456] if present, or
* As the BGP Router-ID of the peer from which the update was received as a last resort.

The Section 2.9 of [RFC9256] specifies the selection of the active candidate path of the SR Policy by the SRPM based on the information provided to it by BGP.

2.2. SR Policy and Tunnel Encapsulation Attribute

The content of the SR Policy Candidate Path is encoded in the Tunnel Encapsulation Attribute defined in [RFC9012] using a Tunnel-Type called SR Policy Type with codepoint 15. The use of SR Policy Tunnel-type is applicable only for the AFI/SAFI pairs of (1/73, 2/73). This document specifies the use of the Tunnel Encapsulation Attribute with the SR Policy Tunnel-Type and the use of any other Tunnel-Type with the SR Policy SAFI MUST be considered malformed and handled by the "Treat-as-Withdraw" strategy [RFC7606].

The SR Policy Encoding structure is as follows:
SR Policy SAFI NLRI: <Distinguisher, Policy-Color, Endpoint>
Attributes:
  Tunnel Encapsulation Attribute (23)
    Tunnel Type: SR Policy (15)
      Binding SID
      SRv6 Binding SID
      Preference
      Priority
      Policy Name
      Policy Candidate Path Name
      Explicit NULL Label Policy (ENLP)
      Segment List
        Weight
        Segment
        Segment
        ...

Figure 2: SR Policy Encoding

where:

* SR Policy SAFI NLRI is defined in Section 2.1.

* Tunnel Encapsulation Attribute is defined in [RFC9012].

* Tunnel-Type is set to 15.

* Preference, Binding SID, SRv6 Binding SID, Priority, Policy Name, Policy Candidate Path Name, ENLP, Segment-List, Weight, and Segment sub-TLVs are defined in Section 2.4.

* Additional sub-TLVs may be defined in the future.

A Tunnel Encapsulation Attribute MUST NOT contain more than one TLV of type "SR Policy"; such updates MUST be considered malformed and handled by the "Treat-as-Withdraw" strategy [RFC7606].

BGP does not need to perform the validation of the tunnel (i.e., SR Policy) itself as indicated in section 6 of [RFC9012]. The validation of the SR Policy information that is advertised using the sub-TLVs specified in Section 2.4 is performed by the SRPM.
2.3. Applicability of Tunnel Encapsulation Attribute Sub-TLVs

The Tunnel Egress Endpoint and Color Sub-TLVs of the Tunnel Encapsulation Attribute [RFC9012] are not used for SR Policy encodings and therefore their value is irrelevant in the context of the SR Policy SAFI NLRI. If present, the Tunnel Egress Endpoint sub-TLV and the Color sub-TLV MUST be ignored by the BGP speaker and MAY be removed from the Tunnel Encapsulation Attribute during propagation.

Similarly, any other sub-TLVs (including those defined in [RFC9012]) whose applicability is not specifically defined for the SR Policy SAFI MUST be ignored by the BGP speaker and MAY be removed from the Tunnel Encapsulation Attribute during propagation.

2.4. SR Policy Sub-TLVs

This section specifies the sub-TLVs defined for encoding the information about the SR Policy Candidate Path.

Preference, Binding SID, SRv6 Binding SID, Segment-List, Priority, Policy Name, Policy Candidate Path Name, and Explicit NULL Label Policy are all optional sub-TLVs introduced for the BGP Tunnel Encapsulation Attribute [RFC9012] being defined in this section.

Weight and Segment are sub-TLVs of the Segment-List sub-TLV mentioned above.

An early version of this document included only the Binding SID sub-TLV that could be used for both SR-MPLS and SRv6 Binding SIDs. The SRv6 Binding SID TLV was introduced in later versions to support the advertisement of additional SRv6 capabilities without affecting backward compatibility for early implementations.

The fault management and error handling in the encoding of the sub-TLVs defined in this section are specified in Section 5. For the TLVs/sub-TLVs that are specified as single instance, only the first instance of that TLV/sub-TLV is used and the other instances MUST be ignored and MUST NOT considered to be malformed.

None of the sub-TLVs defined in the following sub-sections have any effect on the BGP best-path selection or propagation procedures. These sub-TLVs are not used by the BGP path selection process and are instead passed on to SRPM as SR Policy Candidate Path information for further processing described in section 2 of [RFC9256].
The use of SR Policy Sub-TLVs is applicable only for the AFI/SAFI pairs of (1/73, 2/73). Future documents may extend their applicability to other AFI/SAFI.

2.4.1. Preference Sub-TLV

The Preference sub-TLV is used to carry the Preference of an SR Policy candidate path. The contents of this sub-TLV are used by the SRPM as described in section 2.7 of [RFC9256].

The Preference sub-TLV is OPTIONAL and it MUST NOT appear more than once in the SR Policy encoding.

The Preference sub-TLV has following format:

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |   Length      |     Flags     |   RESERVED    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Preference (4 octets)                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: Preference sub-TLV

where:

* Type: 12

* Length: Specifies the length of the value field (i.e., not including Type and Length fields) in terms of octets. The value MUST be 6.

* Flags: 1 octet of flags. No flags are defined in this document. The Flags field MUST be set to zero on transmission and MUST be ignored on receipt.

* RESERVED: 1 octet of reserved bits. This field MUST be set to zero on transmission and MUST be ignored on receipt.

* Preference: a 4-octet value indicating the Preference of the SR Policy Candidate Path as described in section 2.7 of [RFC9256].

2.4.2. Binding SID Sub-TLV

The Binding SID sub-TLV is used to signal the binding SID related information of the SR Policy candidate path. The contents of this sub-TLV are used by the SRPM as described in section 6 in [RFC9256].
The Binding SID sub-TLV is OPTIONAL and it MUST NOT appear more than once in the SR Policy encoding.

When the Binding SID sub-TLV is used to signal an SRv6 SID, the choice of its SRv6 Endpoint Behavior [RFC8986] to be instantiated is left to the headend node. It is RECOMMENDED that the SRv6 Binding SID sub-TLV defined in Section 2.4.3, that enables the specification of the SRv6 Endpoint Behavior, be used for signaling of an SRv6 Binding SID for an SR Policy candidate path.

The Binding SID sub-TLV has the following format:

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |   Length      |     Flags     |   RESERVED    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Binding SID (variable, optional)                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4: Binding SID sub-TLV

where:

* Type: 13

* Length: Specifies the length of the value field (i.e., not including Type and Length fields) in terms of octets. The value MUST be one of: 18 when a SRv6 BSID is present, 6 when a SR-MPLS BSID is present, or 2 when no BSID is present.

* Flags: 1 octet of flags. The following flags are defined in the registry "SR Policy Binding SID Flags" as described in Section 6.6:

```
0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|S|I|           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 5: Binding SID Flags

where:

- S-Flag: This flag encodes the "Specified-BSID-only" behavior. It is used by SRPM as described in section 6.2.3 in [RFC9256].
- **I-Flag**: This flag encodes the "Drop Upon Invalid" behavior. It is used by SRPM as described in section 8.2 in [RFC9256] to define a specific SR Policy forwarding behavior. The flag indicates that the SR Policy is to perform the "drop upon invalid" behavior when no valid candidate path (CP) is available for this SR Policy. In this situation, the CP with the highest preference amongst those with the "drop upon invalid" config is made active to drop traffic steered over the SR Policy.

- The unassigned bits in the Flag octet MUST be set to zero upon transmission and MUST be ignored upon receipt.

* **RESERVED**: 1 octet of reserved bits. MUST be set to zero on transmission and MUST be ignored on receipt.

* **Binding SID**: If the length is 2, then no Binding SID is present. If the length is 6 then the Binding SID is encoded in 4 octets using the format below. Traffic Class (TC), S, and TTL (Total of 12 bits) are RESERVED and MUST be set to zero and MUST be ignored.

```
| Label | TC | S | TTL |
```

Figure 6: Binding SID Label Encoding

The Label field is validated by the SRPM, but MUST NOT contain the reserved MPLS label values (0-15). If the length is 18 then the Binding SID contains a 16-octet SRv6 SID.

2.4.3. SRv6 Binding SID Sub-TLV

The SRv6 Binding SID sub-TLV is used to signal the SRv6 Binding SID related information of an SR Policy candidate path. It enables the specification of the SRv6 Endpoint Behavior [RFC8986] to be instantiated on the headend node. The contents of this sub-TLV are used by the SRPM as described in section 6 in [RFC9256].

The SRv6 Binding SID sub-TLV is OPTIONAL. More than one SRv6 Binding SID sub-TLVs MAY be signaled in the same SR Policy encoding to indicate one or more SRv6 SIDs, each with potentially different SRv6 Endpoint Behaviors to be instantiated.

The SRv6 Binding SID sub-TLV has the following format:
Figure 7: SRv6 Binding SID sub-TLV

where:

* Type: 20

* Length: Specifies the length of the value field (i.e., not including Type and Length fields) in terms of octets. The value MUST be 26 when the SRv6 Endpoint Behavior and SID Structure is present else it MUST be 18.

* Flags: 1 octet of flags. The following flags are defined in the registry "SR Policy Binding SID Flags" as described in Section 6.7:

```
0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
| S | I | B |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
```

Figure 8: SRv6 Binding SID Flags

where:

- S-Flag: This flag encodes the "Specified-BSID-only" behavior. It is used by SRPM as described in section 6.2.3 in [RFC9256].

- I-Flag: This flag encodes the "Drop Upon Invalid" behavior. It is used by SRPM as described in section 8.2 in [RFC9256].

- B-Flag: This flag, when set, indicates the presence of the SRv6 Endpoint Behavior and SID Structure encoding specified in Section 2.4.4.2.4.

- The unassigned bits in the Flag octet MUST be set to zero upon transmission and MUST be ignored upon receipt.
* RESERVED: 1 octet of reserved bits. This field MUST be set to zero on transmission and MUST be ignored on receipt.

* SRv6 Binding SID: Contains a 16-octet SRv6 SID.

* SRv6 Endpoint Behavior and SID Structure: Optional, as defined in Section 2.4.4.2.4.

2.4.4. Segment List Sub-TLV

The Segment List sub-TLV encodes a single explicit path towards the endpoint as described in section 5.1 of [RFC9256]. The Segment List sub-TLV includes the elements of the paths (i.e., segments) as well as an optional Weight sub-TLV.

The Segment List sub-TLV may exceed 255 bytes in length due to a large number of segments. A 2-octet length is thus required. According to section 2 of [RFC9012], the sub-TLV type defines the size of the length field. Therefore, for the Segment List sub-TLV, a code point of 128 or higher is used.

The Segment List sub-TLV is OPTIONAL and MAY appear multiple times in the SR Policy encoding. The ordering of Segment List sub-TLVs does not matter since each sub-TLV encodes a Segment List.

The Segment List sub-TLV contains zero or more Segment sub-TLVs and MAY contain a Weight sub-TLV.

The Segment List sub-TLV has the following format:

```
 0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1               Type     Length     RESERVED
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
//                           sub-TLVs                          //
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 9: Segment List sub-TLV

where:

* Type: 128.

* Length: the total length (not including the Type and Length fields) of the sub-TLVs encoded within the Segment List sub-TLV in terms of octets.
* RESERVED: 1 octet of reserved bits. This field MUST be set to zero on transmission and MUST be ignored on receipt.

* sub-TLVs currently defined:
  - An optional single Weight sub-TLV.
  - Zero or more Segment sub-TLVs.

Validation of an explicit path encoded by the Segment List sub-TLV is beyond the scope of BGP and performed by the SRPM as described in section 5 of [RFC9256].

2.4.4.1. Weight Sub-TLV

The Weight sub-TLV specifies the weight associated with a given segment list. The contents of this sub-TLV are used only by the SRPM as described in section 2.11 of [RFC9256].

The Weight sub-TLV is OPTIONAL and it MUST NOT appear more than once inside the Segment List sub-TLV.

The Weight sub-TLV has the following format:

```
   0                   1                   2                   3
   0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |   Length      |     Flags     |   RESERVED    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              Weight                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 10: Weight sub-TLV

where:

* Type: 9.

* Length: Specifies the length of the value field (i.e., not including Type and Length fields) in terms of octets. The value MUST be 6.

* Flags: 1 octet of flags. No flags are defined in this document. The Flags field MUST be set to zero on transmission and MUST be ignored on receipt.

* RESERVED: 1 octet of reserved bits. This field MUST be set to zero on transmission and MUST be ignored on receipt.
* Weight: 4 octets value indicating the weight associated with a segment list as described in section 2.11 of [RFC9256]. A weight value of zero is invalid.

2.4.4.2. Segment Sub-TLVs

A Segment sub-TLV describes a single segment in a segment list (i.e., a single element of the explicit path). One or more Segment sub-TLVs constitute an explicit path of the SR Policy candidate path. The contents of these sub-TLVs are used only by the SRPM as described in section 4 in [RFC9256].

The Segment sub-TLVs are OPTIONAL and MAY appear multiple times in the Segment List sub-TLV.

Section 4 of [RFC9256] defines several Segment Types:

Type A: SR-MPLS Label
Type B: SRv6 SID
Type C: IPv4 Prefix with optional SR Algorithm
Type D: IPv6 Global Prefix with optional SR Algorithm for SR-MPLS
Type E: IPv4 Prefix with Local Interface ID
Type F: IPv4 Addresses for link endpoints as Local, Remote pair
Type G: IPv6 Prefix and Interface ID for link endpoints as Local, Remote pair for SR-MPLS
Type H: IPv6 Addresses for link endpoints as Local, Remote pair for SR-MPLS
Type I: IPv6 Global Prefix with optional SR Algorithm for SRv6
Type J: IPv6 Prefix and Interface ID for link endpoints as Local, Remote pair for SRv6
Type K: IPv6 Addresses for link endpoints as Local, Remote pair for SRv6

The following sub-sections specify the sub-TLVs used for Segment Types A and B. The other segment types are specified in [I-D.ietf-idr-bgp-sr-segtypes-ext]. As specified in section 5.1 of [RFC9256], a mix of SR-MPLS and SRv6 segments make the segment-list invalid.

2.4.4.2.1. Segment Type A

The Type A Segment Sub-TLV encodes a single SR-MPLS SID. The format is as follows and is used to encode MPLS Label fields as specified in [RFC3032] [RFC5462].:
Figure 11: Type A Segment sub-TLV

| Type | Length | Flags | RESERVED | Label | TC | S | TTL |
| ++++++ | +++++++ | ++++++++ | +++++++ | ++++++++ | ++++++++ | ++++++++ | ++++++++ |

where:

* Type: 1.

* Length: Specifies the length of the value field (i.e., not including Type and Length fields) in terms of octets. The value MUST be 6.

* Flags: 1 octet of flags as defined in Section 2.4.4.2.3.

* RESERVED: 1 octet of reserved bits. This field MUST be set to zero on transmission and MUST be ignored on receipt.

* Label: 20 bits of label value.

* TC: 3 bits of traffic class.

* S: 1 bit of bottom-of-stack.

* TTL: 1 octet of TTL.

The following applies to the Type-1 Segment sub-TLV:

* The S bit MUST be zero upon transmission and MUST be ignored upon reception.

* If the originator wants the receiver to choose the TC value, it sets the TC field to zero.

* If the originator wants the receiver to choose the TTL value, it sets the TTL field to 255.

* If the originator wants to recommend a value for these fields, it puts those values in the TC and/or TTL fields.
The receiver MAY override the originator’s values for these fields. This would be determined by local policy at the receiver. One possible policy would be to override the fields only if the fields have the default values specified above.

2.4.4.2.2. Segment Type B

The Type B Segment Sub-TLV encodes a single SRv6 SID. The format is as follows:

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |   Length      |     Flags     |   RESERVED    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
//                       SRv6 SID (16 octets)                  //
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
//     SRv6 Endpoint Behavior and SID Structure (optional)     //
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 12: Type B Segment sub-TLV

where:

* Type: 13.

* Length: Specifies the length of the value field (i.e., not including Type and Length fields) in terms of octets. The value MUST be 26 when the SRv6 Endpoint Behavior and SID Structure is present else it MUST be 18.

* Flags: 1 octet of flags as defined in Section 2.4.4.2.3.

* RESERVED: 1 octet of reserved bits. This field MUST be set to zero on transmission and MUST be ignored on receipt.

* SRv6 SID: 16 octets of IPv6 address.

* SRv6 Endpoint Behavior and SID Structure: Optional, as defined in Section 2.4.4.2.4.

The Sub-TLV code point 2 defined for the advertisement of Segment Type B in the earlier versions of this document has been deprecated to avoid backward compatibility issues.
2.4.4.2.3. Segment Flags

The Segment Types sub-TLVs described above may contain the following flags in the "Flags" field defined in Section 6.8:

```
+----------------------------------+
| 0 1 2 3 4 5 6 7                  |
+----------------------------------+
   ^ |   | B |            |
   |   |   |     +------------------+
```

Figure 22: Segment Flags

where:

V-Flag: This flag, when set, is used by SRPM for "SID verification" as described in Section 5.1 of [RFC9256].

B-Flag: This flag, when set, indicates the presence of the SRv6 Endpoint Behavior and SID Structure encoding specified in Section 2.4.4.2.4.

The unassigned bits in the Flag octet MUST be set to zero upon transmission and MUST be ignored upon receipt.

The following applies to the Segment Flags:

* V-Flag applies to all Segment Types.

* B-Flag applies to Segment Type B. If B-Flag appears with Segment Type A it MUST be ignored.

2.4.4.2.4. SRv6 SID Endpoint Behavior and Structure

The Segment Types sub-TLVs described above MAY contain the SRv6 Endpoint Behavior and SID Structure [RFC8986] encoding as described below:

```
+----------------------------------+
|       Endpoint Behavior      |   Reserved        |
+----------------------------------+
|       LB Length   |  LN Length  | Fun. Length | Arg. Length |
+----------------------------------+
```

Figure 23: SRv6 SID Endpoint Behavior and Structure

where:
Endpoint Behavior: 2 octets. It carries the SRv6 Endpoint Behavior code point for this SRv6 SID as defined in section 9.2 of [RFC8986]. When set with the value 0xFFFF (i.e., Opaque), the choice of SRv6 Endpoint Behavior is left to the headend.

Reserved: 2 octets of reserved bits. This field MUST be set to zero on transmission and MUST be ignored on receipt.

Locator Block Length: 1 octet. SRv6 SID Locator Block length in bits.

Locator Node Length: 1 octet. SRv6 SID Locator Node length in bits.

Function Length: 1 octet. SRv6 SID Function length in bits.

Argument Length: 1 octet. SRv6 SID Arguments length in bits.

The total of the locator block, locator node, function, and argument lengths MUST be less than or equal to 128.

2.4.5. Explicit NULL Label Policy Sub-TLV

To steer an unlabeled IP packet into an SR policy, it is necessary to push a label stack of one or more labels on that packet.

The Explicit NULL Label Policy (ENLP) sub-TLV is used to indicate whether an Explicit NULL Label [RFC3032] must be pushed on an unlabeled IP packet before any other labels.

If an ENLP Sub-TLV is not present, the decision of whether to push an Explicit NULL label on a given packet is a matter of local configuration.

The ENLP sub-TLV is OPTIONAL and it MUST NOT appear more than once in the SR Policy encoding.

The contents of this sub-TLV are used by the SRPM as described in section 4.1 of [RFC9256].
Where:

Type: 14.

Length: Specifies the length of the value field (i.e., not including Type and Length fields) in terms of octets. The value MUST be 3.

Flags: 1 octet of flags. No flags are defined in this document. The Flags field MUST be set to zero on transmission and MUST be ignored on receipt.

RESERVED: 1 octet of reserved bits. This field MUST be set to zero on transmission and MUST be ignored on receipt.

ENLP (Explicit NULL Label Policy): Indicates whether Explicit NULL labels are to be pushed on unlabeled IP packets that are being steered into a given SR policy. The following values have been currently defined for this field:

- 1: Push an IPv4 Explicit NULL label on an unlabeled IPv4 packet, but do not push an IPv6 Explicit NULL label on an unlabeled IPv6 packet.


- 4: Do not push an Explicit NULL label.

This field can have one of the values as specified in Section 6.10. The ENLP unassigned values may be used for future extensions and implementations SHOULD ignore the ENLP Sub-TLV with these values. The behavior signaled in this Sub-TLV MAY be
overridden by local configuration. The section 4.1 of [RFC9256] describes the behavior on the headend for the handling of the explicit null label.

2.4.6. Policy Priority Sub-TLV

An operator MAY set the Policy Priority sub-TLV to indicate the order in which the SR policies are re-computed upon topological change. The contents of this sub-TLV are used by the SRPM as described in section 2.12 of [RFC9256].

The Priority sub-TLV is OPTIONAL and it MUST NOT appear more than once in the SR Policy encoding.

The Priority sub-TLV has following format:

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |   Length      |  Priority     |   RESERVED    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 25: Priority sub-TLV

Where:

Type: 15

Length: Specifies the length of the value field (i.e., not including Type and Length fields) in terms of octets. The value MUST be 2.

Priority: a 1-octet value.

RESERVED: 1 octet of reserved bits. This field MUST be set to zero on transmission and MUST be ignored on receipt.

2.4.7. Policy Candidate Path Name Sub-TLV

An operator MAY set the Policy Candidate Path Name sub-TLV to attach a symbolic name to the SR Policy candidate path.

Usage of Policy Candidate Path Name sub-TLV is described in section 2.6 of [RFC9256].
The Policy Candidate Path Name sub-TLV may exceed 255 bytes in length due to a long name. A 2-octet length is thus required. According to section 2 of [RFC9012], the sub-TLV type defines the size of the length field. Therefore, for the Policy Candidate Path Name sub-TLV a code point of 128 or higher is used.

It is **RECOMMENDED** that the size of the symbolic name for the candidate path is limited to 255 bytes. Implementations MAY choose to truncate long names to 255 bytes when signaling via BGP.

The Policy Candidate Path Name sub-TLV is **OPTIONAL** and it **MUST NOT** appear more than once in the SR Policy encoding.

The Policy Candidate Path Name sub-TLV has following format:

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |   Length                      |   RESERVED    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
//              Policy Candidate Path Name                     //
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

**Figure 26: Policy Candidate Path Name sub-TLV**

Where:

- **Type**: 129.
- **Length**: Specifies the length of the value field (i.e., not including Type and Length fields) in terms of octets. The value is variable.
- **RESERVED**: 1 octet of reserved bits. This field **MUST** be set to zero on transmission and **MUST** be ignored on receipt.
- **Policy Candidate Path Name**: Symbolic name for the SR Policy candidate path without a NULL terminator as specified in section 2.6 of [RFC9256].

### 2.4.8. Policy Name Sub-TLV

An operator **MAY** set the Policy Name sub-TLV to associate a symbolic name with the SR Policy for which the candidate path is being advertised via the SR Policy NLRI.

Usage of Policy Name sub-TLV is described in section 2.1 of [RFC9256].
The Policy Name sub-TLV may exceed 255 bytes in length due to a long policy name. A 2-octet length is thus required. According to section 2 of [RFC9012], the sub-TLV type defines the size of the length field. Therefore, for the Policy Name sub-TLV a code point of 128 or higher is used.

It is RECOMMENDED that the size of the symbolic name for the SR Policy is limited to 255 bytes. Implementations MAY choose to truncate long names to 255 bytes when signaling via BGP.

The Policy Name sub-TLV is OPTIONAL and it MUST NOT appear more than once in the SR Policy encoding.

The Policy Name sub-TLV has the following format:

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |   Length                      |   RESERVED    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
                  //                        Policy Name                          //
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 27: Policy Name sub-TLV

Where:

- **Type:** 130
- **Length:** Specifies the length of the value field (i.e., not including Type and Length fields) in terms of octets. The value is variable.
- **RESERVED:** 1 octet of reserved bits. This field MUST be set to zero on transmission and MUST be ignored on receipt.
- **Policy Name:** Symbolic name for the policy. It SHOULD be a string of printable ASCII characters, without a NULL terminator.

3. Color Extended Community

The Color Extended Community [RFC9012] is used to steer traffic corresponding to BGP routes into an SR Policy with matching color value. The Color Extended Community MAY be carried in any BGP UPDATE message whose AFI/SAFI is 1/1 (IPv4 Unicast), 2/1 (IPv6 Unicast), 1/4 (IPv4 Labeled Unicast), 2/4 (IPv6 Labeled Unicast), 1/128 (VPN-IPv4 Labeled Unicast), 2/128 (VPN-IPv6 Labeled Unicast), or 25/70 (Ethernet VPN, usually known as EVPN). Use of the Color Extended Community
Community in BGP UPDATE messages of other AFI/SAFIs is outside the scope of this document.

Two bits from the Flags field of the Color Extended Community are used as follows to support the requirements of Color-Only steering as specified in Section 8.8 of [RFC9256]:

```
  1 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
+----------------------------------+
| C O |        Unassigned         |
+----------------------------------+
```

Figure 28: Color Extended Community Flags

The CO bits together form the Color-Only Type field which indicates the various matching criteria between BGP NH and SR Policy endpoint in addition to the matching of the color value. Following types are defined:

* Type 0: Specific Endpoint Match: Request match for the endpoint that is the BGP NH

* Type 1: Specific or Null Endpoint Match: Request match for either the endpoint that is the BGP NH or a null endpoint (e.g., like a default gateway)

* Type 2: Specific, Null, or Any Endpoint Match: Request match for either the endpoint that is the BGP NH or with a null or any endpoint

* Type 3: reserved for future use and SHOULD NOT be used. Upon reception, an implementation MUST treat it like Type 0.

The details of the SR Policy steering mechanisms based on these Color-Only types are specified in section 8.8 of [RFC9256].

One or more Color Extended Communities MAY be associated with a BGP route update. Sections 8.4.1, 8.5.1, and 8.8.2 of [RFC9256] specify the steering behaviors over SR Policies when multiple Color Extended Communities are associated with a BGP route.
4. SR Policy Operations

As mentioned in Section 1, BGP is not the actual consumer of an SR Policy NLRI. BGP is in charge of the origination and propagation of the SR Policy NLRI but its installation and use are outside the scope of BGP. The details of SR Policy installation and use are specified in [RFC9256].

4.1. Advertisement of SR Policies

Typically, but not limited to, an SR Policy is computed by a controller or a path computation engine (PCE) and originated by a BGP speaker on its behalf.

Multiple SR Policy NLRIs may be present with the same <color, endpoint> tuple but with different distinguishers when these SR policies are intended for different headends.

The distinguisher of each SR Policy NLRI prevents undesired BGP route selection among these SR Policy NLRIs and allows their propagation across route reflectors [RFC4456].

Moreover, one or more route targets SHOULD be attached to the advertisement, where each route target identifies one or more intended headends for the advertised SR Policy update.

If no route target is attached to the SR Policy NLRI, then it is assumed that the originator sends the SR Policy update directly (e.g., through a BGP session) to the intended receiver. In such a case, the NO_ADVERTISE community [RFC1997] MUST be attached to the SR Policy update (see further details in Section 4.2.3).

4.2. Reception of an SR Policy NLRI

On reception of an SR Policy NLRI, a BGP speaker first determines if it is valid as described in Section 4.2.1 and then performs the decision process for selection of the best route (Section 9.1 of [RFC4271]). The key difference from the base BGP decision process is that BGP does not download the selected best routes of SR Policy SAFI into the forwarding and instead considers them "usable" for passing on to the SRPM for further processing as described in Section 4.2.2. The selected best route is "propagated" (Section 9.1.3 of [RFC4271]) as described in Section 4.2.3 irrespective of its "usability" by the local router.
4.2.1. Validation of an SR Policy NLRI

When a BGP speaker receives an SR Policy NLRI from a neighbor it MUST first perform validation based on the following rules in addition to the validation described in Section 5:

* The SR Policy NLRI MUST include a distinguisher, color, and endpoint field which implies that the length of the NLRI MUST be either 12 or 24 octets (depending on the address family of the endpoint).

* The SR Policy update MUST have either the NO_ADVERTISE community or at least one route target extended community in IPv4-address format or both. If a router supporting this specification receives an SR Policy update with no route target extended communities and no NO_ADVERTISE community, the update MUST be considered as malformed.

* The Tunnel Encapsulation Attribute MUST be attached to the BGP Update and MUST have a Tunnel Type TLV set to SR Policy (codepoint is 15).

A router that receives an SR Policy update that is not valid according to these criteria MUST treat the update as malformed and the SR Policy candidate path MUST NOT be passed to the SRPM.

4.2.2. Eligibility for Local Use of an SR Policy NLRI

An SR Policy NLRI update without any route target extended community but having the NO_ADVERTISE community is considered usable.

If one or more route targets are present, then at least one route target MUST match the BGP Identifier of the receiver for the update to be considered usable. The BGP Identifier is defined in [RFC4271] as a 4-octet IPv4 address. Therefore, the route target extended community MUST be of the same format.

If one or more route targets are present and none matches the local BGP Identifier, then, while the SR Policy NLRI is valid, it is not usable on the receiver node.

When the SR Policy tunnel type includes any sub-TLV that is unrecognized or unsupported, the update SHOULD NOT be considered usable. An implementation MAY provide an option for ignoring unsupported sub-TLVs.
Once BGP on the receiving node has determined that the SR Policy NLRI is usable, it passes the SR Policy candidate path to the SRPM. Note that, along with the candidate path details, BGP also passes the originator information for breaking ties in the candidate path selection process as described in section 2.4 of [RFC9256].

When an update for an SR Policy NLRI results in its becoming unusable, BGP MUST delete its corresponding SR Policy candidate path from the SRPM.

The SRPM applies the rules defined in section 2 of [RFC9256] to determine whether the SR Policy candidate path is valid and to select the active candidate path for a given SR Policy.

4.2.3. Propagation of an SR Policy

SR Policy NLRIs that have the NO_ADVERTISE community attached to them MUST NOT be propagated.

By default, a BGP node receiving an SR Policy NLRI MUST NOT propagate it to any EBGP neighbor. An implementation MAY provide an explicit configuration to override this and enable the propagation of valid SR Policy NLRIs to specific EBGP neighbors where the SR domain comprises multiple-ASes within a single service provider domain (see Section 7 for details).

A BGP node advertises a received SR Policy NLRI to its IBGP neighbors according to normal IBGP propagation rules.

By default, a BGP node receiving an SR Policy NLRI SHOULD NOT remove route target extended community before propagation. An implementation MAY provide support for configuration to filter and/or remove route target extended community before propagation.

A BGP node MUST NOT alter the SR Policy information carried in the Tunnel Encapsulation Attribute during propagation.

5. Error Handling and Fault Management

This section describes the error handling actions, as described in [RFC7606], that are to be performed for the handling of the BGP update messages for BGP SR Policy SAFI.
A BGP Speaker MUST perform the following syntactic validation of the SR Policy NLRI to determine if it is malformed. This includes the validation of the length of each NLRI and the total length of the MP_REACH_NLRI and MP_UNREACH_NLRI attributes. It also includes the validation of the consistency of the NLRI length with the AFI and the endpoint address as specified in Section 2.1.

When the error determined allows for the router to skip the malformed NLRI(s) and continue the processing of the rest of the update message, then it MUST handle such malformed NLRIs as ‘Treat-as-withdraw’. In other cases, where the error in the NLRI encoding results in the inability to process the BGP update message (e.g. length related encoding errors), then the router SHOULD handle such malformed NLRIs as ‘AFI/SAFI disable’ when other AFI/SAFI besides SR Policy are being advertised over the same session. Alternately, the router MUST perform ‘session reset’ when the session is only being used for SR Policy or when it ‘AFI/SAFI disable’ action is not possible.

The validation of the TLVs/sub-TLVs introduced in this document and defined in their respective sub-sections of Section 2.4 MUST be performed to determine if they are malformed or invalid. The validation of the Tunnel Encapsulation Attribute itself and the other TLVs/sub-TLVs specified in Section 13 of [RFC9012] MUST be done as described in that document. In case of any error detected, either at the attribute or its TLV/sub-TLV level, the ”treat-as-withdraw” strategy MUST be applied. This is because an SR Policy update without a valid Tunnel Encapsulation Attribute (comprising of all valid TLVs/sub-TLVS) is not usable.

An SR Policy update that is determined to be not valid, and therefore malformed, based on rules described in Section 4.2.1 MUST be handled by the ”treat-as-withdraw” strategy.

The validation of the individual fields of the TLVs/sub-TLVs defined in Section 2.4 are beyond the scope of BGP as they are handled by the SRPM as described in the individual TLV/sub-TLV sub-sections. A BGP implementation MUST NOT perform semantic verification of such fields nor consider the SR Policy update to be invalid or not usable based on such validation.

An implementation SHOULD log any errors found during the above validation for further analysis.

6. IANA Considerations

This document uses code point allocations from the following existing registries:
This document introduces a SAFI in the registry "Subsequent Address Family Identifiers (SAFI) Parameters" that has been assigned a code point by IANA. The entry needs to be updated as follows:

<table>
<thead>
<tr>
<th>Code Point</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>73</td>
<td>SR Policy SAFI</td>
<td>This document</td>
</tr>
</tbody>
</table>

Table 1: BGP SAFI Code Point

6.2. BGP Tunnel Encapsulation Attribute Tunnel Types

This document introduces a Tunnel-Type in the registry "BGP Tunnel Encapsulation Attribute Tunnel Types" that has been assigned a codepoint by IANA. The entry needs to be updated as follows:
Table 2: Tunnel Type Code Point

6.3. BGP Tunnel Encapsulation Attribute sub-TLVs

This document defines sub-TLVs in the registry "BGP Tunnel Encapsulation Attribute sub-TLVs" that have been assigned code points by IANA as follows via the early allocation process which needs to be made permanent:

<table>
<thead>
<tr>
<th>Code Point</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>Preference sub-TLV</td>
<td>This document</td>
</tr>
<tr>
<td>13</td>
<td>Binding SID sub-TLV</td>
<td>This document</td>
</tr>
<tr>
<td>14</td>
<td>ENLP sub-TLV</td>
<td>This document</td>
</tr>
<tr>
<td>15</td>
<td>Priority sub-TLV</td>
<td>This document</td>
</tr>
<tr>
<td>20</td>
<td>SRv6 Binding SID sub-TLV</td>
<td>This document</td>
</tr>
<tr>
<td>128</td>
<td>Segment List sub-TLV</td>
<td>This document</td>
</tr>
<tr>
<td>129</td>
<td>Policy Candidate Path Name sub-TLV</td>
<td>This document</td>
</tr>
<tr>
<td>130</td>
<td>Policy Name sub-TLV</td>
<td>This document</td>
</tr>
</tbody>
</table>

Table 3: BGP Tunnel Encapsulation Attribute Code Points

6.4. Color Extended Community Flags

This document defines the use of 2 bits in the registry called "Color Extended Community Flags" under the "BGP Tunnel Encapsulation" registry that have been assigned by IANA via the early allocation process to form the Color-Only Types field which needs to be made permanent:

<table>
<thead>
<tr>
<th>Bit Position</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-1</td>
<td>Color-only Types Field</td>
<td>This document</td>
</tr>
</tbody>
</table>

Table 4: Color Extended Community Flag Bits

6.5. SR Policy Segment List Sub-TLVs

This document requests the creation of a new registry called "SR Policy Segment List Sub-TLVs" under the "BGP Tunnel Encapsulation" registry. The allocation policy of this registry is "Standards Action" according to [RFC8126].
Following initial Sub-TLV codepoints are assigned by this document:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Reserved</td>
<td>This document</td>
</tr>
<tr>
<td>1</td>
<td>Segment Type A sub-TLV</td>
<td>This document</td>
</tr>
<tr>
<td>2</td>
<td>Deprecated</td>
<td>This document</td>
</tr>
<tr>
<td>3-8</td>
<td>Unassigned</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Weight sub-TLV</td>
<td>This document</td>
</tr>
<tr>
<td>10</td>
<td>Deprecated</td>
<td>This document</td>
</tr>
<tr>
<td>11</td>
<td>Deprecated</td>
<td>This document</td>
</tr>
<tr>
<td>12</td>
<td>Deprecated</td>
<td>This document</td>
</tr>
<tr>
<td>13</td>
<td>Segment Type B sub-TLV</td>
<td>This document</td>
</tr>
<tr>
<td>14-255</td>
<td>Unassigned</td>
<td></td>
</tr>
</tbody>
</table>

Table 5: SR Policy Segment List Code Points

6.6. SR Policy Binding SID Flags

This document requests the creation of a new registry called "SR Policy Binding SID Flags" under the "BGP Tunnel Encapsulation" registry. The allocation policy of this registry is "Standards Action" according to [RFC8126].

The following flags are defined:

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Specified-BSID-Only Flag (S-Flag)</td>
<td>This document</td>
</tr>
<tr>
<td>1</td>
<td>Drop Upon Invalid Flag (I-Flag)</td>
<td>This document</td>
</tr>
<tr>
<td>2-7</td>
<td>Unassigned</td>
<td></td>
</tr>
</tbody>
</table>

Table 6: SR Policy Binding SID Flags

6.7. SR Policy SRv6 Binding SID Flags

This document requests the creation of a new registry called "SR Policy SRv6 Binding SID Flags" under the "BGP Tunnel Encapsulation" registry. The allocation policy of this registry is "Standards Action" according to [RFC8126].

The following flags are defined:
6.8. SR Policy Segment Flags

This document requests the creation of a new registry called "SR Policy Segment Flags" under the "BGP Tunnel Encapsulation" registry. The allocation policy of this registry is "Standards Action" according to [RFC8126].

The following flags are defined:

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Segment Verification Flag (V-Flag)</td>
<td>This document</td>
</tr>
<tr>
<td>1-2</td>
<td>Unassigned</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>SRv6 Endpoint Behavior &amp; SID Structure Flag (B-Flag)</td>
<td>This document</td>
</tr>
<tr>
<td>4-7</td>
<td>Unassigned</td>
<td></td>
</tr>
</tbody>
</table>

Table 8: SR Policy Segment Flags

6.9. Color Extended Community Color-Only Types

This document requests the creation of a new registry called "Color Extended Community Color-Only Types" under the "BGP Tunnel Encapsulation" registry for assignment of codepoints (values 0 through 3) in the Color-Only Type field of the Color Extended Community Flags field. The allocation policy of this registry is "Standards Action" according to [RFC8126].

The following types are defined:

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Specific Endpoint Match</td>
<td>This document</td>
</tr>
<tr>
<td>1</td>
<td>Specific or Null Endpoint Match</td>
<td>This document</td>
</tr>
<tr>
<td>2</td>
<td>Specific, Null, or Any Endpoint Match</td>
<td>This document</td>
</tr>
<tr>
<td>3</td>
<td>Unassigned</td>
<td>This document</td>
</tr>
</tbody>
</table>

Table 9: Color Extended Community Color-Only Types
6.10. SR Policy ENLP Values

Note to IANA (RFC editor to remove this before publication): The new registry creation request below is also present in the draft-ietf-pce-segment-routing-policy-cp. IANA is requested to process the registry creation via the first of these two documents to reach publication stage and the authors of the other document would update the IANA considerations suitably.

This document requests IANA to maintain a new registry under "Segment Routing Parameters" registry group with the allocation policy of "Standards Action" [RFC8126]. The new registry is called "SR Policy ENLP Values" and contains the codepoints allocated to the "ENLP" field defined in Section 2.4.5. The registry contains the following codepoints, with initial values, to be assigned by IANA with the reference set to this document:

<table>
<thead>
<tr>
<th>Code Point</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Reserved (not to be used)</td>
</tr>
<tr>
<td>1</td>
<td>Push an IPv4 Explicit NULL label on an unlabeled IPv4 packet, but do not push an IPv6 Explicit NULL label on an unlabeled IPv6 packet</td>
</tr>
<tr>
<td>2</td>
<td>Push an IPv6 Explicit NULL label on an unlabeled IPv6 packet, but do not push an IPv4 Explicit NULL label on an unlabeled IPv4 packet</td>
</tr>
<tr>
<td>3</td>
<td>Push an IPv6 Explicit NULL label on an unlabeled IPv6 packet, and push an IPv4 Explicit NULL label on an unlabeled IPv4 packet</td>
</tr>
<tr>
<td>4</td>
<td>Do not push an Explicit NULL label</td>
</tr>
<tr>
<td>5-255</td>
<td>Unassigned</td>
</tr>
</tbody>
</table>

7. Security Considerations

The security mechanisms of the base BGP security model apply to the extensions described in this document as well. See the Security Considerations section of [RFC4271] for a discussion of BGP security. Also, refer to [RFC4272] and [RFC6952] for analysis of security issues for BGP.

The BGP SR Policy extensions specified in this document enable traffic engineering and service programming use-cases within an SR domain as described in [RFC9256]. SR operates within a trusted SR domain [RFC8402] and its security considerations also apply to BGP sessions when carrying SR Policy information. The SR Policies
distributed by BGP are expected to be used entirely within this trusted SR domain which comprises a single AS or multiple ASes/ domains within a single provider network. Therefore, precaution is necessary to ensure that the SR Policy information advertised via BGP sessions is limited to nodes in a secure manner within this trusted SR domain. BGP peering sessions for address-families other than SR Policy SAFI may be set up to routers outside the SR domain. The isolation of BGP SR Policy SAFI peering sessions may be used to ensure that the SR Policy information is not advertised by accident or error to an EBGP peering session outside the SR domain.

Additionally, it may be considered that the export of SR Policy information, as described in this document, constitutes a risk to confidentiality of mission-critical or commercially sensitive information about the network (more specifically endpoint/node addresses, SR SIDs, and the SR Policies deployed). BGP peerings are not automatic and require configuration; thus, it is the responsibility of the network operator to ensure that only trusted nodes (that include both routers and controller applications) within the SR domain are configured to receive such information.

8. Manageability Considerations

The specification of BGP models is an ongoing work based on [I-D.ietf-idr-bgp-model] and its future extensions are expected to cover the SR Policy SAFI. Existing BGP operational procedures also apply to the SAFI specified in this document. The management, operations, and monitoring of BGP speakers and the SR Policy SAFI sessions between them are not very different from other BGP sessions and can be managed using the same data models.

The YANG model for the operation and management of SR Policies [I-D.ietf-spring-sr-policy-yang] reports the SR Policies provisioned via BGP SR Policy SAFI along with their operational states.

9. Acknowledgments

The authors of this document would like to thank Shyam Sethuram, John Scudder, Przemyslaw Krol, Alex Bogdanov, Nandan Saha, Bruno Decraene, Gurusiddesh Nidasesi, Kausik Majumdar, Zafar Ali, Swadesh Agarwal, Jakob Heitz, Viral Patel, Peng Shaofu, Cheng Li, Martin Vigoureux, John Scudder, Vincent Roca, Brian Haberman, Mohamed Boucadair, Shunwan Zhuang, Andrew Alston, Jeffrey (Zhaohui) Zhang, Nagendra Nainar, Rajesh Melarcode Venkateswaran, Nat Kao, Boris Hassanov, and Vincent Roca for their comments and review of this document. The authors would like to thank Susan Hares for her detailed shepherd review that helped in improving the document.

10. Contributors

Eric Rosen  
Juniper Networks  
US  
Email: erosen@juniper.net

Arjun Sreekantiah  
Cisco Systems  
US  
Email: asreakan@cisco.com

Acee Lindem  
Cisco Systems  
US  
Email: acee@cisco.com

Siva Sivabalan  
Cisco Systems  
US  
Email: msiva@cisco.com

Imtiyaz Mohammad  
Arista Networks  
India  
Email: imtiyaz@arista.com

Gaurav Dawra  
Cisco Systems  
US  
Email: gdawra.ietf@gmail.com

Peng Shaofu  
ZTE Corporation  
China  
Email: peng.shaofu@zte.com.cn
11. References

11.1. Normative References


11.2. Informational References

[I-D.ietf-idr-bgp-ls-sr-policy]

[I-D.ietf-idr-bgp-model]

[I-D.ietf-idr-bgp-sr-segtypes-ext]

[I-D.ietf-spring-sr-policy-yang]


Authors’ Addresses

Stefano Previdi
Huawei Technologies
Italy
Email: stefano@previdi.net

Clarence Filsfils
Cisco Systems
Brussels
Belgium
Email: cfilsfil@cisco.com

Ketan Talaulikar (editor)
Cisco Systems
India
Email: ketant.ietf@gmail.com

Paul Mattes
Microsoft
One Microsoft Way
Redmond, WA 98052
United States of America
Email: pamattes@microsoft.com

Dhanendra Jain
Google
Email: dhanendra.ietf@gmail.com
Abstract

The Network File System v4 (NFSv4) allows a client to both open a file and be granted a delegation of that file. This delegation provides the client the right to authoritatively cache metadata on the file locally. This document presents several extensions for both the opening and delegating of the file to the client. This document extends both NFSv4.1 (see RFC8881) and NFSv4.2 (see RFC7863).

Note

This note is to be removed before publishing as an RFC.

Discussion of this draft takes place on the NFSv4 working group mailing list (nfsv4@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/nfsv4/. Working Group information can be found at https://datatracker.ietf.org/wg/nfsv4/about/.

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 14 February 2025.
In the Network File System version4 (NFSv4), a client may be granted a delegation for a file. This allows the client to act as the authority for the file’s metadata and data. This document presents a number of extensions which enhance the functionality of opens and delegations. These allow the client to:

* detect an offline file, which may require significant effort to obtain.
* determine which extensions of OPEN (see Section 18.16 of [RFC8881]) flags are supported by the server.

* during the OPEN procedure, get either the open or delegation stateids, but not both.

* cache both the access and modify times, reducing the number of times the client needs to go to the server to get that information.

Using the process detailed in [RFC8178], the revisions in this document become an extension of NFSv4.2 [RFC7862]. They are built on top of the external data representation (XDR) [RFC4506] generated from [RFC7863].

1.1. Definitions

offline file: A file which exists on a device which is not connected to the server. There is typically a cost associated with bringing the file to an online status. Historically this would be a file on tape media and the cost would have been finding and loading the tape. A more modern interpretation is that the file is in the cloud and the cost is a monetary one in downloading the file.

proxy: Proxying of attributes occurs when a client has the authority, as granted by the appropriate delegation, to represent the attributes normally maintained by the server. For read attributes, this occurs when the client has either a read or write delegations for the file. For write attributes, this occurs when the client has a write delegation for the file. The client having this authority is the "proxy" for those attributes.

1.2. Requirements Language

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

2. Offline Files

If a file is offline, then the server has immediate high-performance access to the file’s attributes, but not to the file’s content. The action of retrieving the data content is expensive, to the extent that the content should only be retrieved if it is going to be used. For example, a graphical file manager (such as OSX’s Finder) may want to access the beginning of the file to preview it for an user who is
hovering their pointer over the file name and not accessing it otherwise. If the file is retrieved, it will most likely either be immediately thrown away or returned.

A compound with a GETATTR or REaddir can report the file’s attributes without bringing the file online. However, either an OPEN or a LAYOUTGET might cause the file server to retrieve the archived data contents, bringing the file online. For non-pNFS systems, the OPEN operation requires a filehandle to the data content. For pNFS systems, the filehandle retrieved from an OPEN need not cause the data content to be retrieved. But when the LAYOUTGET operation is processed, a layout type specific mapping will cause the data content to be retrieved from offline storage.

If the client is not aware that the file is offline, it might inadvertently open the file to determine what type of file it is accessing. By interrogating the new attribute FATTR4_OFFLINE, a client can predetermine the availability of the file, avoiding the need to open it at all. Being offline might also involve situations in which the file is archived in the cloud, i.e., there can be an expense in both retrieving the file to bring online and in sending the file back to offline status.

2.1. XDR for Offline Attribute

```c
typedef bool fattr4_offline;
```

```c
const FATTR4_OFFLINE = 83;
```

3. Determining OPEN Feature Support

[RFC8178] (see Section 4.4.2) allows for extending a particular minor version of the NFSv4 protocol without requiring the definition of a new minor version. The client can probe the capabilities of the server and based on the result, determine if both it and the server support optional features not previously specified as part of the minor version.
The fattr4_open_arguments attribute is a new XDR extension which provides helpful support when the OPEN procedure is extended in such a fashion. It models all of the parameters via bitmap4 data structures, which allows for the addition of a new flag to any of the OPEN arguments (see Section 18.16.1 of [RFC8881]). The scope of this attribute applies to all objects with a matching fsid.

Two new flags are provided:

* OPEN4_SHARE_ACCESS_WANT_OPEN_XOR_DELEGATION (see Section 4)
* OPEN4_SHARE_ACCESS_WANT_DELEG_TIMESTAMPS (see Section 5)

Subsequent extensions can use this framework when introducing new OPTIONAL functionality to OPEN, by creating a new flag for each OPTIONAL parameter.

Since fattr4_open_arguments is a RECOMMENDED attribute, if the server informs the client via NFS4ERR_ATTRNOTSUPP that it does not support this new attribute, the client MUST take this to mean that the additional new OPTIONAL functionality to OPEN is also not supported.

Some other concerns are how to process both currently REQUIRED flags and OPTIONAL flags which become REQUIRED in the future. The server MUST mark REQUIRED flags as being supported. Note that as these flags MUST only change from OPTIONAL to REQUIRED when the NFSv4 minor version is incremented.

3.1. XDR for Open Arguments

```c
/// struct open_arguments4 {
///   bitmap4  oa_share_access;
///   bitmap4  oa_share_deny;
///   bitmap4  oa_share_access_want;
///   bitmap4  oa_open_claim;
///   bitmap4  oa_create_mode;
/// };
```

<CODE BEGINS>
///
/// enum open_args_share_access4 {
///    OPEN_ARGS_SHARE_ACCESS_READ = 1,
///    OPEN_ARGS_SHARE_ACCESS_WRITE = 2,
///    OPEN_ARGS_SHARE_ACCESS_BOTH = 3
/// };}
///
<CODE ENDS>

<CODE BEGINS>
///
/// enum open_args_share_deny4 {
///    OPEN_ARGS_SHARE_DENY_NONE = 0,
///    OPEN_ARGS_SHARE_DENY_READ = 1,
///    OPEN_ARGS_SHARE_DENY_WRITE = 2,
///    OPEN_ARGS_SHARE_DENY_BOTH = 3
/// };}
///
<CODE ENDS>

<CODE BEGINS>
///
/// enum open_args_share_access_want4 {
///    OPEN_ARGS_SHARE_ACCESS_WANT_ANY_DELEG           = 3,
///    OPEN_ARGS_SHARE_ACCESS_WANT_NO_DELEG            = 4,
///    OPEN_ARGS_SHARE_ACCESS_WANT_CANCEL              = 5,
///    OPEN_ARGS_SHARE_ACCESS_WANT_SIGNAL_DELEG_WHEN_RESRC_AVAIL
///        = 17,
///    OPEN_ARGS_SHARE_ACCESS_WANT_PUSH_DELEG_WHEN_UNCONTENDED
///        = 18,
///    OPEN_ARGS_SHARE_ACCESS_WANT_DELEG_TIMESTAMPS     = 20,
///    OPEN_ARGS_SHARE_ACCESS_WANT_OPEN_XOR_DELEGATION = 21
/// };}
///
<CODE ENDS>
<CODE BEGINS>
///
/// enum open_args_open_claim4 {
///    OPEN_ARGS_OPEN_CLAIM_NULL          = 0,
///    OPEN_ARGS_OPEN_CLAIM_PREVIOUS      = 1,
///    OPEN_ARGS_OPEN_CLAIM_DELEGATE_CUR = 2,
///    OPEN_ARGS_OPEN_CLAIM_DELEG_PREV   = 3,
///    OPEN_ARGS_OPEN_CLAIM_FH           = 4,
///    OPEN_ARGS_OPEN_CLAIM_DELEG_CUR_FH = 5,
///    OPEN_ARGS_OPEN_CLAIM_DELEG_PREV_FH = 6
/// };    
///
<CODE ENDS>

<CODE BEGINS>
///
/// enum open_args_createmode4 {
///    OPEN_ARGS_CREATEMODE_UNCHECKED4     = 0,
///    OPEN_ARGS_CREATE_MODE_GUARDED       = 1,
///    OPEN_ARGS_CREATEMODE_EXCLUSIVE4     = 2,
///    OPEN_ARGS_CREATE_MODE_EXCLUSIVE4_1  = 3
/// };    
///
<CODE ENDS>

<CODE BEGINS>
///
/// typedef open_arguments4 fattr4_open_arguments;
///
<CODE ENDS>

<CODE BEGINS>
///
/// %/*
/// % * Determine what OPEN supports.
/// % */
/// const FATTR4_OPEN_ARGUMENTS     = 86;
///
<CODE ENDS>

<CODE BEGINS>
///
/// const OPEN4_SHARE_ACCESS_WANT_OPEN_XOR_DELEGATION = 0x200000;
///
<CODE ENDS>
4. OPEN grants only one of Open or Delegation Stateid

The OPEN (See Section 18.16 of [RFC8881]) procedure returns an open stateid to the client to reference the state of the file. The client could also request a delegation stateid in the OPEN arguments. The file can be considered open for the client as long as the count of open and delegated stateids is greater than 0. Either type of stateid is sufficient to enable the server to treat the file as if it were open, which allows READ (See Section 18.25 of [RFC8881]), WRITE (See Section 18.38 of [RFC8881]), LOCK (See Section 18.12 of [RFC8881]), and LAYOUTGET (see Section 18.50 of [RFC8881]) operations to proceed. If the client gets both an open and a delegation stateid as part of the OPEN, then it has to return them both to the server. A further consideration is that during each operation, the client can send a costly GETATTR (See Section 18.7 of [RFC8881]).

If the client knows that the server supports the OPEN4_SHARE_ACCESS_WANT_OPEN_XOR_DELEGATION flag (as determined by an earlier GETATTR operation which queried for the FATTR4_OPEN_ARGUMENTS attribute), then the client can supply that flag during the OPEN and only get either an open or delegation stateid.

The client is already prepared to not get a delegation stateid even if requested. In order to not send an open stateid, the server can indicate that fact with the result flag of OPEN4_RESULT_NO_OPEN_STATEID. The open stateid field, OPEN4resok.stateid (see Section 18.16.2 of [RFC8881]), will also be set to the special all zero stateid.

4.1. Implementation Experience

The CLOSE operation (see Section 18.2 of [RFC8881]) neither explicitly nor implicitly releases any delegation stateids. This is not symmetrical with the OPEN operation, which can grant both an open and a delegation stateid. This draft could have tried to extend the CLOSE operation to release both stateids, but implementation experience shows that is more costly than the approach which has been proposed.

Consider a small workload of creating a file with content. That takes 3 synchronous and 1 asynchronous operations with existing implementations. The first synchronous one has to OPEN the file, the
second synchronous one performs the WRITE to the file, the third synchronous one has to CLOSE the file, and the fourth asynchronous one uses DELEGRETURN (see Section 18.6 of [RFC8881]) to return the delegation stateid.

With the proposed approach of setting the OPEN_ARGS_SHARE_ACCESS_WANT_OPEN_XOR_DELEGATION flag during the OPEN, the number of operations is always 3. The first two compounds are still synchronous, but the last is asynchronous. I.e., since the client no longer has to send a CLOSE operation, it can delay the DELEGRETURN until either the server requests it back via delegation recall or garbage collection causes the client to return the stateid.

This approach reduces the cost of synchronous operations by 33% and the total number of operations by 25%. Contrast that against the alternative proposal of having CLOSE return both stateids, which would not reduce the number of synchronous operations.

5. Proxying of Times

When a client is granted a write delegation on a file, it becomes the authority for the file contents and associated attributes. If the server queries the client as to the state of the file via a CB_GETATTR (see Section 20.1 of [RFC8881]), then, according to the unextended NFSv4 protocol, it can only determine the size of the file and the change attribute. In the case of the client holding the delegation, it has the current values of the access and modify times. There is no way that other clients can have access to these values. While the client could send a compound of the form: SEQ, PUTFH, SETATTR (time_modify | time_access), DELEGRETURN, to notify the server of the proxied values, that SETATTR operation would cause either or both of FATTR4_CHANGE or FATTR4_TIME_METADATA to be modified to the current time on the server. There is no current provision to obtain these values before delegation return using CB_GETATTR. As a result, it can not pass these times up to an application expecting POSIX compliance, as is often necessary for correct operation.

With the addition of the new flag: OPEN4_SHARE_ACCESS_WANT_DELEG_TIMESTAMPS, the client and server can negotiate that the client will be the authority for these values and upon return of the delegation stateid via a DELEGRETURN (see section 18.6 of [RFC8881]), the times will be passed back to the server. If the server is queried by another client for either the size or the times, it will need to use a CB_GETATTR to query the client which holds the delegation (see Section 20.1 of [RFC8881]).
If a server informs the client via the FATTR4.OPEN_ARGUMENTS attribute that it supports OPEN_ARGS_SHARE_ACCESS_WANT_DELEG_TIMESTAMPS and it returns a valid delegation stateid for an OPEN operation which sets the OPEN4_SHARE_ACCESS_WANT_DELEG_TIMESTAMPS flag, then it MUST query the client via a CB_GETATTR for the FATTR4.TIME.DELEG.ACCESS (see Section 5.2) attribute and FATTR4.TIME.DELEG.MODIFY attribute (see Section 5.2). (The change time can be derived from the modify time.) Further, when it gets a SETATTR (see Section 18.34 of [RFC8881]) in the same compound as the DELEGRETURN, then it MUST accept those FATTR4.TIME.DELEG.ACCESS attribute and FATTR4.TIME.DELEG.MODIFY attribute changes and derive the change time or reject the changes with NFS4ERR_DELAY (see Section 15.1.1.3 if [RFC8881]).

These new attributes are invalid to be used with GETATTR, VERIFY, and NVERIFY and can only be used with CB_GETATTR and SETATTR by a client holding an appropriate delegation. The SETATTR SHOULD either be in a separate compound before the one containing the DELEGRETURN or when in the same compound, as an operation before the DELEGRETURN. Failure to properly sequence the operations may lead to race cases.

A key prerequisite of this approach is that the server and client are in time synchronization with each other. Note that while the base NFSv4.2 does not require such synchronization, the use of RPCSEC_GSS typically makes such a requirement. When the client presents either FATTR4.TIME.DELEG.ACCESS or FATTR4.TIME.DELEG.MODIFY attributes to the server, the server MUST decide for both of them whether the time presented is before the corresponding FATTR4.TIME.ACCESS or FATTR4.TIME.MODIFY attribute on the file or past the current server time. When the time presented is before the original time, then the update is ignored. When the time presented is in the future, the server can either clamp the new time to the current time, or it may return NFS4ERR_DELAY to the client, allowing it to retry. Note that if the clock skew is large, the delay approach would result in access to the file being denied until the clock skew is exceeded.

A change in the access time MUST NOT advance the change time, also known as the time_metadata attribute (see Section 5.8.2.42 of [RFC8881]), but a change in the modify time might advance the change time (and in turn the change attribute (See Section 5.8.1.4 of [RFC8881]). If the modify time is greater than the change time and before the current time, then the change time is adjusted to the modify time and not the current time (as is most likely done on most SETATTR calls that change the metadata). If the modify time is in the future, it will be clamped to the current time.
Note that each of the possible times, access, modify, and change, are compared to the current time. They should all be compared against the same time value for the current time. I.e., do not retrieve a different value of the current time for each calculation.

If the client sets the OPEN4_SHARE_ACCESS_WANT_DELEG_TIMESTAMPS flag in an OPEN operation, then it MUST support the FATTR4_TIME_DELEG_ACCESS and FATTR4_TIME_DELEG_MODIFY attributes both in the CB_GETATTR and SETATTR operations.

5.1. Use case

Consider a NFSv3 client which wants to access data on a server which only supports NFSv4.2. An implementation could add a NFSv3 server which is a NFSv4.2 client gateway the two incompatible systems. As NFSv3 is a stateless protocol, the state is not kept on the client, but rather the NFSv3 server. As the NFSv3 server is already managing the state, it can proxy file delegations to avoid spurious GETATTRs. I.e., as the client queries the NFSv3 server for the attributes, they can be served without the NFSv3 server sending a GETATTR to the NFSv4.2 server.

5.2. XDR for Proxying of Times

```c
<CODE BEGINS>
///
/// /* attributes for the delegation times being cached and served by the "client"
/// */
/// typedef nfstime4 fattr4_time_deleg_access;
/// typedef nfstime4 fattr4_time_deleg_modify;
///
<CODE ENDS>
```

```c
<CODE BEGINS>
///
/// %/*
/// % * New RECOMMENDED Attribute for delegation caching of times
/// % */
/// const FATTR4_TIME_DELEG_ACCESS = 84;
/// const FATTR4_TIME_DELEG_MODIFY = 85;
///
<CODE ENDS>
```
6. Extraction of XDR

This document contains the external data representation (XDR) [RFC4506] description of the new open flags for delegating the file to the client. The XDR description is embedded in this document in a way that makes it simple for the reader to extract into a ready-to-compile form. The reader can feed this document into the following shell script to produce the machine readable XDR description of the new flags:

```bash
#!/bin/sh
grep '^ *///' $* | sed 's?^ */// ??' | sed 's?^ *///$??'
```

That is, if the above script is stored in a file called "extract.sh", and this document is in a file called "spec.txt", then the reader can do:

```bash
sh extract.sh < spec.txt > delstid_prot.x
```

The effect of the script is to remove leading white space from each line, plus a sentinel sequence of "///". XDR descriptions with the sentinel sequence are embedded throughout the document.

Note that the XDR code contained in this document depends on types from the NFSv4.2 nfs4_prot.x file (generated from [RFC7863]). This includes both nfs types that end with a 4, such as offset4, length4, etc., as well as more generic types such as uint32_t and uint64_t.

While the XDR can be appended to that from [RFC7863], the various code snippets belong in their respective areas of the that XDR.

6.1. Code Components Licensing Notice

Both the XDR description and the scripts used for extracting the XDR description are Code Components as described in Section 4 of "Legal Provisions Relating to IETF Documents" [LEGAL]. These Code Components are licensed according to the terms of that document.
7. Security Considerations

While we are extending some capabilities for client delegation, there are no new security concerns. The client cannot be queried by other clients as to the cached attributes. The client could report false data for the cached attributes, but it already has this ability via a SETATTR operation (see Section 18.30 of [RFC8881]).

8. IANA Considerations

There are no IANA considerations.

9. References

9.1. Normative References


9.2. Informative References
Appendix A. Acknowledgments

Trond Myklebust and David Flynn all worked on the prototype at Hammerspace.

Dave Noveck, Chuck Lever, Rick Macklem, and Zaheduzzaman Sarker provided reviews of the document.

Authors’ Addresses

Thomas Haynes
Hammerspace
Email: loghyr@hammerspace.com

Trond Myklebust
Hammerspace
Email: trondmy@hammerspace.com
Abstract

This document extends the specification of Network Time Protocol (NTP) version 4 in RFC 5905 with special modes called the NTP interleaved modes, that enable NTP servers to provide their clients and peers with more accurate transmit timestamps that are available only after transmitting NTP packets. More specifically, this document describes three modes: interleaved client/server, interleaved symmetric, and interleaved broadcast.

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 21 April 2022.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.
1. Introduction  

RFC 5905 [RFC5905] describes the operations of NTPv4 in a client/server, symmetric, and broadcast mode. The transmit and receive timestamps are two of the four timestamps included in every NTPv4 packet used for time synchronization.

For a highly accurate and stable synchronization, the transmit and receive timestamp should be captured close to the beginning of the actual transmission and the end of the reception respectively. An asymmetry in the timestamping causes the offset measured by NTP to have an error.

There are at least four options where a timestamp of an NTP packet may be captured with a software NTP implementation running on a general-purpose operating system:

1. User space (software)
2. Network device driver or kernel (software)
3. Data link layer (hardware - MAC chip)
4. Physical layer (hardware - PHY chip)

Software timestamps captured in user space in the NTP implementation itself are least accurate. They do not include system calls used for sending and receiving packets, processing and queuing delays in the system, network device drivers, and hardware. Hardware timestamps captured at the physical layer are most accurate.

A transmit timestamp captured in the driver or hardware is more accurate than the user-space timestamp, but it is available to the NTP implementation only after it sent the packet using a system call. The timestamp cannot be included in the packet itself unless the driver or hardware supports NTP and can modify the packet before or during the actual transmission.

The protocol described in RFC 5905 does not specify any mechanism for a server to provide its clients and peers with a more accurate transmit timestamp that is known only after the transmission. A packet that strictly follows RFC 5905, i.e. it contains a transmit timestamp corresponding to the packet itself, is said to be in basic mode.

Different mechanisms could be used to exchange timestamps known after the transmission. The server could respond to each request with two packets. The second packet would contain the transmit timestamp corresponding to the first packet. However, such a protocol would enable a traffic amplification attack, or it would use packets with an asymmetric length, which would cause an asymmetry in the network delay and an error in the measured offset.

This document describes an interleaved client/server, interleaved symmetric, and interleaved broadcast mode. In these modes, the server sends a packet which contains a transmit timestamp corresponding to the transmission of the previous packet that was sent to the client or peer. This transmit timestamp can be captured in any software or hardware component involved in the transmission of the packet. Both servers and clients/peers are required to keep some state specific to the interleaved mode.

An NTPv4 implementation that supports the client/server and broadcast interleaved modes interoperates with NTPv4 implementations without this capability. A peer using the symmetric interleaved mode does not fully interoperate with a peer which does not support it. The mode needs to be configured specifically for each symmetric association.
The interleaved modes do not change the NTP packet header format and do not use new extension fields. The negotiation is implicit. The protocol is extended with new values that can be assigned to the origin and transmit timestamp. Servers and peers check the origin timestamp to detect requests conforming to the interleaved mode. A response can be valid only in one mode. If a client or peer that does not support interleaved mode received a response conforming to the interleaved mode, it would be rejected as bogus.

An explicit negotiation would require a new extension field. RFC 5905 does not specify how servers should handle requests with an unknown extension field. The original use of extension fields was authentication with Autokey [RFC5906], which cannot be negotiated. Some existing implementations do not respond to requests with unknown extension fields. This behavior would prevent clients from reliably detecting support for the interleaved mode.

Requests and responses cannot always be formed in interleaved mode. It cannot be used exclusively. Servers, clients, and peers that support the interleaved mode need to support also the basic mode.

This document assumes familiarity with RFC 5905.

1.1. Requirements Language

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

2. Interleaved Client/server mode

The interleaved client/server mode is similar to the basic client/server mode. The difference between the two modes is in the values saved to the origin and transmit timestamp fields.

The origin timestamp is a cookie which is used to detect that a received packet is a response to the last packet sent in the other direction of the association. It is a copy of one of the timestamps from the packet to which it is responding, or zero if it is not a response. Servers following RFC 5905 ignore the origin timestamp in client requests. A server response which does not have a matching origin timestamp is called bogus.

A client request in the basic mode has an origin timestamp equal to the transmit timestamp from the last valid server response, or is zero (which indicates the first request of the association). A
server response in the basic mode has an origin timestamp equal to the transmit timestamp from the client request. The transmit timestamp in the response corresponds to the transmission of the response in which the timestamp is contained.

A client request in the interleaved mode has an origin timestamp equal to the receive timestamp from the last valid server response. A server response in the interleaved mode has an origin timestamp equal to the receive timestamp from the client request. The transmit timestamp in the response corresponds to the transmission of the previous response which had the receive timestamp equal to the origin timestamp from the request.

A server which supports the interleaved mode needs to save pairs of local receive and transmit timestamps. The server SHOULD discard old timestamps to limit the amount of memory needed to support clients using the interleaved mode. The server MAY separate the timestamps by IP addresses, but it SHOULD NOT separate them by port numbers to support clients that change their port between requests, as recommended in RFC 9109 [RFC9109].

The server MAY restrict the interleaved mode to specific IP addresses and/or authenticated clients.

Both servers and clients that support the interleaved mode MUST NOT send a packet that has a transmit timestamp equal to the receive timestamp in order to reliably detect whether received packets conform to the interleaved mode. One way to ensure that is to increment the transmit timestamp by 1 unit (i.e. about 1/4 of a nanosecond) if the two timestamps are equal, or a new timestamp can be generated.

The transmit and receive timestamps in server responses need to be unique to prevent two different clients from sending requests with the same origin timestamp and the server responding in the interleaved mode with an incorrect transmit timestamp. If the timestamps are not guaranteed to be monotonically increasing, the server SHOULD check that the transmit and receive timestamps are not already saved as a receive timestamp of a previous request (from the same IP address if the server separates timestamps by addresses), and generate a new timestamp if necessary.

When the server receives a request from a client, it SHOULD respond in the interleaved mode if the following conditions are met:

1. The request does not have a receive timestamp equal to the transmit timestamp.
2. The origin timestamp from the request matches the local receive timestamp of a previous request that the server has saved (for the IP address if it separates timestamps by addresses).

A response in the interleaved mode MUST contain the transmit timestamp of the response which contained the receive timestamp matching the origin timestamp from the request. The server SHOULD drop the timestamps after sending the response. The receive timestamp MUST NOT be used again to detect a request conforming to the interleaved mode.

If the conditions are not met (i.e. the request is not detected to conform to the interleaved mode), the server MUST NOT respond in the interleaved mode. The server MAY always respond in the basic mode. In any case, the server SHOULD save the new receive and transmit timestamps.

The first request from a client is always in the basic mode and so is the server response. It has a zero origin timestamp and zero receive timestamp. Only when the client receives a valid response from the server, it will be able to send a request in the interleaved mode.

The protocol recovers from packet loss. When a client request or server response is lost, the client will use the same origin timestamp in the next request. The server can respond in the interleaved mode if it still has the timestamps corresponding to the origin timestamp. If the server already responded to the timestamp in the interleaved mode, or it had to drop the timestamps for other reasons, it will respond in the basic mode and save new timestamps, which will enable an interleaved response to the subsequent request. The client SHOULD limit the number of requests in the interleaved mode between server responses to prevent processing of very old timestamps in case a large number of consecutive requests is lost.

An example of packets in a client/server exchange using the interleaved mode is shown in Figure 1. The packets in the basic and interleaved mode are indicated with B and I respectively. The timestamps t1, t3 and t11 point to the same transmissions as t1̃, t3̃ and t11̃, but they may be less accurate. The first exchange is in the basic mode followed by a second exchange in the interleaved mode. For the third exchange, the client request is in the interleaved mode, but the server response is in the basic mode, because the server did not have the pair of timestamps t6 and t7 (e.g. they were dropped to save timestamps for other clients using the interleaved mode).
When the client receives a response from the server, it performs the tests described in RFC 5905. Two of the tests are modified for the interleaved mode:

1. The check for duplicate packets SHOULD compare both receive and transmit timestamps in order to not drop a valid response in the interleaved mode if it follows a response in the basic mode and they contain the same transmit timestamp.

2. The check for bogus packets SHOULD compare the origin timestamp with both transmit and receive timestamps from the request. If the origin timestamp is equal to the transmit timestamp, the response is in the basic mode. If the origin timestamp is equal to the receive timestamp, the response is in the interleaved mode.

The client SHOULD NOT update its NTP state when an invalid response is received, to not lose the timestamps which will be needed to complete a measurement when the subsequent response in the interleaved mode is received.

If the packet passed the tests and conforms to the interleaved mode, the client can compute the offset and delay using the formulas from RFC 5905 and one of two different sets of timestamps. The first set is RECOMMENDED for clients that filter measurements based on the delay. The corresponding timestamps from Figure 1 are written in parentheses.

- **T1** - local transmit timestamp of the previous request (t1)
- **T2** - remote receive timestamp from the previous response (t2)
T3 - remote transmit timestamp from the latest response (t3)

T4 - local receive timestamp of the previous response (t4)

The second set gives a more accurate measurement of the current offset, but the delay is much more sensitive to a frequency error between the server and client due to a much longer interval between T1 and T4.

T1 - local transmit timestamp of the latest request (t5)

T2 - remote receive timestamp from the latest response (t6)

T3 - remote transmit timestamp from the latest response (t3)

T4 - local receive timestamp of the previous response (t4)

Clients MAY filter measurements based on the mode. The maximum number of dropped measurements in the basic mode SHOULD be limited in case the server does not support or is not able to respond in the interleaved mode. Clients that filter measurements based on the delay will implicitly prefer measurements in the interleaved mode over the basic mode, because they have a shorter delay due to a more accurate transmit timestamp (T3).

The server MAY limit saving of the receive and transmit timestamps to requests which have an origin timestamp specific to the interleaved mode in order to not waste resources on clients using the basic mode. Such an optimization will delay the first interleaved response of the server to a client by one exchange.

A check for a non-zero origin timestamp works with SNTP clients that always set the timestamp to zero and clients that implement NTP data minimization [I-D.ietf-ntp-data-minimization]. From the server’s point of view, such clients start a new association with each request.

To avoid searching the saved receive timestamps for non-zero origin timestamps from requests conforming to the basic mode, the server can encode in low-order bits of the receive and transmit timestamps below precision of the clock a flag indicating whether the timestamp is a receive timestamp. If the server receives a request with a non-zero origin timestamp which does not indicate it is a receive timestamp of the server, the request does not conform to the interleaved mode and it is not necessary to perform the search and/or save the new receive and transmit timestamp.
3. Interleaved Symmetric mode

The interleaved symmetric mode uses the same principles as the interleaved client/server mode. A packet in the interleaved symmetric mode has a transmit timestamp which corresponds to the transmission of the previous packet sent to the peer and an origin timestamp equal to the receive timestamp from the last packet received from the peer.

To enable synchronization in both directions of a symmetric association, both peers need to support the interleaved mode. For this reason, it SHOULD be disabled by default and enabled with an option in the configuration of the active side of the association.

In order to prevent the peer from matching the transmit timestamp with an incorrect packet when the peers' transmissions do not alternate (e.g. they use different polling intervals) and a previous packet was lost, the use of the interleaved mode in symmetric associations requires additional restrictions.

Peers which have an association need to count valid packets received between their transmissions to determine in which mode a packet should be formed. A valid packet in this context is a packet which passed all NTP tests for duplicate, replayed, bogus, and unauthenticated packets. Other received packets may update the NTP state to allow the (re)initialization of the association, but they do not change the selection of the mode.

A peer A SHOULD send a peer B a packet in the interleaved mode only when all of the following conditions are met:

1. The peer A has an active association with the peer B which was specified with the option enabling the interleaved mode, OR the peer A received at least one valid packet in the interleaved mode from the peer B.

2. The peer A did not send a packet to the peer B since it received the last valid packet from the peer B.

3. The previous packet that the peer A sent to the peer B was the only response to a packet received from the peer B.

The first condition is needed for compatibility with implementations that do not support or are not configured for the interleaved mode. The other conditions prevent a missing response from causing a mismatch between the remote transmit (T2) and local receive timestamp (T3), which would cause a large error in the measured offset and delay.
An example of packets exchanged in a symmetric association is shown in Figure 2. The minimum polling interval of the peer A is twice as long as the maximum polling interval of the peer B. The first packets sent by the peers are in the basic mode. The second and third packet sent by the peer A is in the interleaved mode. The second packet sent by the peer B is in the interleaved mode, but the following packets sent by the peer B are in the basic mode, because multiple responses are sent per request.

![Diagram of packet timestamps in interleaved symmetric mode](image)

If the peer A has no association with the peer B and it responds with symmetric passive packets, it does not need to count the packets in order to meet the restrictions, because each request has at most one response. The peer SHOULD process the requests in the same way as a server which supports the interleaved client/server mode. It MUST NOT respond in the interleaved mode if the request was not in the interleaved mode.

The peers SHOULD compute the offset and delay using one of the two sets of timestamps specified in the client/server section. They MAY switch between them to minimize the interval between T1 and T4 in order to reduce the error in the measured delay.

4. Interleaved Broadcast mode

A packet in the interleaved broadcast mode contains two transmit timestamps. One corresponds to the packet itself and is saved in the transmit timestamp field. The other corresponds to the previous packet and is saved in the origin timestamp field. The packet is compatible with the basic mode, which uses a zero origin timestamp.

An example of packets sent in the broadcast mode is shown in Figure 3.
A client which does not support the interleaved mode ignores the origin timestamp and processes all packets as if they were in the basic mode.

A client which supports the interleaved mode SHOULD check if the origin timestamp is not zero to detect packets in the interleaved mode. The client SHOULD also compare the origin timestamp with the transmit timestamp from the previous packet to detect lost packets. If the difference is larger than a specified maximum (e.g. 1 second), the packet SHOULD NOT be used for synchronization in the interleaved mode.

The client SHOULD compute the offset using the origin timestamp from the received packet and the local receive timestamp of the previous packet. If the client needs to measure the network delay, it SHOULD use the interleaved client/server mode.

5. Protocol Failures

An incorrect client implementation of the basic mode (RFC 5905) can work reliably with servers that implement only the basic mode, but the protocol can fail intermittently with servers that implement the interleaved mode.

If the client sets the origin timestamp to other values than the transmit timestamp from the last valid server response, or zero, the origin timestamp can match a receive timestamp of a previous server response (possibly to a different client), causing an unexpected interleaved response. The client is expected to drop the response as bogus. If it did not check for bogus packets, it would be vulnerable to off-path attacks.
If the client set the origin timestamp to a constant non-zero value, this mismatch would be expected to happen once per the NTP era (about 136 years) if the NTP server was responding at the maximum rate needed to go through all timestamp values (about 2 billion responses per second). With lower rates of requests the chance of hitting a server timestamp decreases proportionally.

The worst case of this failure would be a client that specifically sets the origin timestamp to the server’s receive timestamp, i.e. the client accidentally implemented the interleaved mode, but it does not accept interleaved responses. This client would still be able to synchronize its clock. It would drop interleaved responses as bogus and set the origin timestamp to the receive timestamp from the last valid response in the basic mode. As servers are required to not respond twice to the same origin timestamp in the interleaved mode, at least every other response would be in the basic mode and accepted by the client.

Intermittent protocol failures can be caused also by an incorrect server implementation of the interleaved mode. A server which does not ensure the receive and transmit timestamps in its responses are unique in a sufficiently long interval can misinterpret requests formed correctly in the basic mode as interleaved and respond in the interleaved mode. The response would be dropped by the client as bogus.

A duplicated server receive timestamp can cause an expected interleaved response to contain a transmit timestamp which does not correspond to the transmission of the previous response from which the client copied the receive timestamp to the origin timestamp in the request, but a different response which contained the same receive timestamp. The response would be accepted by the client with a small error in the transmit timestamp equal to the difference between the transmit timestamps of the two different responses. As the two requests to which the responses responded were received at the same time (according to the server’s clock), the two transmissions would be expected to be close to each other and the difference between them would be comparable to the error a basic response normally has in its transmit timestamp.

One reason for a duplicated server timestamp can be a large backward step of the server’s clock. If the timestamps the server has saved do not fully cover the second pass of the clock over the repeated interval, two requests received in different passes of the clock can get the same receive timestamp. The client which made the first request can get the transmit timestamp corresponding to the transmission of the second response. From the server’s point of view, the error of the transmit timestamp would be still small, but
from the client’s point of view the server already failed when it made the step as it was serving wrong time before or after the step with a much larger error than the error caused by the protocol failure.

6. Security Considerations

The security considerations of time protocols in general are discussed in RFC 7384 [RFC7384], and specifically the security considerations of NTP are discussed in RFC 5905.

Security issues that apply to the basic modes apply also to the interleaved modes. They are described in The Security of NTP’s Datagram Protocol [SECNTP].

Clients and peers SHOULD NOT leak the receive timestamp in packets sent to other peers or clients (e.g. as a reference timestamp) to prevent off-path attackers from easily getting the origin timestamp needed to make a valid response in the interleaved mode.

Clients using the interleaved mode SHOULD randomize all bits of both receive and transmit timestamps, as recommended for the transmit timestamp in the NTP client data minimization [I-D.ietf-ntp-data-minimization], to make it more difficult for off-path attackers to guess the origin timestamp in the server response.

The client data minimization cannot be fully implemented in the interleaved mode. The origin timestamp cannot be zeroed out, which makes the clients more vulnerable to tracking as they move between networks.

Attackers can force the server to drop its timestamps in order to prevent clients from getting an interleaved response. They can send a large number of requests, send requests with a spoofed source address, or replay an authenticated request if the interleaved mode is enabled only for authenticated clients. Clients SHOULD NOT rely on servers to be able to respond in the interleaved mode.
Protecting symmetric associations in the interleaved mode against replay attacks is even more difficult than in the basic mode. In both modes, the NTP state needs to be protected between the reception of the last non-replayed response and transmission of the next request in order for the request to contain the origin timestamp expected by the peer. The difference is in the timestamps needed to complete a measurement. In the basic mode only one valid response is needed at a time and it is used as soon as it is received, but the interleaved mode needs two consecutive valid responses. The NTP state needs to be protected all the time to not lose the timestamps which are needed to complete the measurement when the second response is received.

7. IANA Considerations

This memo includes no request to IANA.

8. Acknowledgements

The interleaved modes described in this document are based on the implementation written by David Mills in the NTP project (http://www.ntp.org). The specification of the broadcast mode is based purely on this implementation. The specification of the symmetric mode has some modifications. The client/server mode is specified as a new mode compatible with the symmetric mode, similarly to the basic symmetric and client/server modes.

The authors would like to thank Theresa Enghardt, Daniel Franke, Benjamin Kaduk, Erik Kline, Tal Mizrahi, Steven Sommars, Harlan Stenn, and Kristof Teichel for their useful comments.

9. References

9.1. Normative References

[I-D.ietf-ntp-data-minimization]


Lichvar & Malhotra Expires 21 April 2022
9.2. Informative References


Authors’ Addresses

Miroslav Lichvar
Red Hat
Purkynova 115
612 00 Brno
Czech Republic

Email: mlichvar@redhat.com

Aanchal Malhotra
Boston University
111 Cummington St
Boston, 02215
United States of America

Email: aanchal4@bu.edu
Abstract

This document defines the Path Computation Element Communication Protocol (PCEP) extension for Central Control Dynamic Routing (CCDR) based applications in Native IP networks. It describes the key information that is transferred between the Path Computation Element (PCE) and the Path Computation Clients (PCC) to accomplish the End-to-End (E2E) traffic assurance in the Native IP network under PCE as a central controller (PCECC).
This document is subject to BCP 78 and the IETF Trust’s Legal
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 Revised BSD License text as
described in Section 4.e of the Trust Legal Provisions and are
provided without warranty as described in the Revised BSD License.

Table of Contents
1. Introduction ........................................... 3
2. Conventions used in this document ....................... 3
   2.1. Use of RBNF ..................................... 4
3. Terminology ........................................... 4
4. Capability Advertisement ................................ 4
   4.1. Open Message ................................... 4
5. PCEP Messages ......................................... 6
   5.1. The PCInitiate Message ........................... 6
   5.2. The PCRpt Message ............................... 7
6. PCECC Native IP TE Procedures .......................... 8
   6.1. BGP Session Establishment Procedures ............. 9
   6.2. Explicit Route Establishment Procedures .......... 12
   6.3. BGP Prefix Advertisement Procedures ............... 15
   6.4. Selection of Raw Mode and Tunnel Mode Forwarding
        Strategy .......................................... 17
   6.5. Clean Up ......................................... 17
   6.6. Other Procedures .................................. 18
7. New PCEP Objects ....................................... 18
   7.1. CCI Object ....................................... 18
   7.2. BGP Peer Info Object ............................. 19
   7.3. Explicit Peer Route Object ...................... 21
   7.4. Peer Prefix Advertisement Object ................. 23
8. New Error-Types and Error-Values Defined ................ 26
9. BGP Considerations ..................................... 28
10. Deployment Considerations ............................... 28
11. Manageability Considerations ........................... 29
   11.1. Control of Function and Policy .................. 29
   11.2. Information and Data Models ..................... 29
   11.3. Liveness Detection and Monitoring ............... 29
   11.4. Verify Correct Operations ....................... 29
   11.5. Requirements on Other Protocols ................. 30
   11.6. Impact on Network Operations ..................... 30
12. Implementation Status .................................. 30
   12.1. Proof of Concept based on ODL ................... 30
   12.2. ZTE ............................................ 31
13. Security Considerations ............................... 31
14. IANA Considerations ................................... 31
1. Introduction

Generally, Multiprotocol Label Switching Traffic Engineering (MPLS-TE) requires the corresponding network devices to support Resource ReSerVation Protocol (RSVP)/Label Distribution Protocol (LDP) protocols to ensure the End-to-End (E2E) traffic performance. But in native IP network scenarios described in [RFC8735], there will be no such signaling protocol to synchronize the actions among different network devices. It is feasible to use the central control mode described in [RFC8283] to correlate the forwarding behavior among different network devices. [RFC8821] describes the architecture and solution philosophy for the E2E traffic assurance in the Native IP network via multiple Border Gateway Protocol (BGP) sessions-based solution. It requires only the PCE to send the instructions to the PCCs, to build multiple BGP sessions, distribute different prefixes on the established BGP sessions and assign the different paths to the BGP next hops.

This document describes the corresponding Path Computation Element Communication Protocol (PCEP) extensions to transfer the key information about BGP peer, peer prefix advertisement, and the explicit peer route on on-path routers.

2. Conventions used in this document

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

The message formats in this document are illustrated using Routing Backus-Naur Form (RBNF) encoding, as specified in [RFC5511]. The use of RBNF is illustrative only and may elide certain important details; the normative specification of messages is found in the prose description. If there is any divergence between the RBNF and the prose, the prose is considered authoritative.

3. Terminology

This document uses the following terms defined in [RFC5440]: PCC, PCE, PCEP.

The following terminology is used in this document:

- BPI: BGP Peer Info
- CCDR: Central Control Dynamic Routing
- CCI: Central Controller Instructions, defined in [RFC9050]
- E2E: End-to-End
- EPR: Explicit Peer Route
- PCECC: PCE as a Central Controller, defined in [RFC8283]
- PPA: Peer Prefix Advertisement
- PST: Path Setup Type, defined in [RFC8408]
- SRP: Stateful PCE Request Parameters, defined in [RFC8231]
- RR: Route Reflector

4. Capability Advertisement

4.1. Open Message

During the PCEP Initialization Phase, PCEP Speakers (PCE or PCC) advertise their support of Native IP extensions.

This document defines a new Path Setup Type (PST) [RFC8408] for Native-IP, as follows:

- PST = 4: Path is a Native IP TE path as per [RFC8821].
A PCEP speaker MUST indicate its support of the function described in this document by sending a PATH-SETUP-TYPE-CAPABILITY TLV in the OPEN object with this new PST included in the PST list.

[RFC9050] defined the PCECC-CAPABILITY sub-TLV to exchange information about their PCECC capability. A new flag is defined in PCECC-CAPABILITY sub-TLV for Native IP:

N (NATIVE-IP-TE-CAPABILITY - 1 bit - 30): If set to 1 by a PCEP speaker, it indicates that the PCEP speaker is capable of TE in a Native IP network as specified in this document. The flag MUST be set by both the PCC and PCE to support this extension.

If a PCEP speaker receives the PATH-SETUP-TYPE-CAPABILITY TLV with the newly defined path setup type, but without the N bit set in PCECC-CAPABILITY sub-TLV, it MUST:

* send a PCErr message with Error-Type=10 (Reception of an invalid object) and Error-Value=39 (PCECC NATIVE-IP-TE-CAPABILITY bit is not set).

* terminate the PCEP session

If a PCEP speaker receives the PATH-SETUP-TYPE-CAPABILITY TLV with the newly defined path setup type, but without the PCECC-CAPABILITY sub-TLV, it MUST:

* send a PCErr message with Error-Type=10 (Reception of an invalid object) and Error-Value=33 (Missing PCECC Capability sub-TLV).

* terminate the PCEP session

If one or both speakers (PCE and PCC) have not indicated support and willingness to use the PCEP extensions for Native-IP, the PCEP extensions for the Native-IP MUST NOT be used. If a Native-IP operation is attempted when both speakers have not agreed on the OPEN messages, the receiver of the message MUST:

* send a PCErr message with Error-Type=19 (Invalid Operation) and Error-value=TBD1 (Attempted Native-IP operations when the capability was not advertised) and

* terminate the PCEP session.
5. PCEP Messages

PCECC Native IP TE solution uses the existing PCE Label Switched Path (LSP) Initiate Request message (PCInitiate) [RFC8281], and PCE Report message (PCRpt) [RFC8231] to accomplish the multiple BGP sessions establishment, E2E Native-IP TE path deployment, and route prefixes advertisement among different BGP sessions. A new PST for Native-IP is used to indicate the path setup based on TE in Native IP networks.

The extended PCInitiate message described in [RFC9050] is used to download or remove the central controller’s instructions (CCIs). [RFC9050] specifies an object called CCI for the encoding of the central controller’s instructions. This document specifies a new CCI Object-Type for Native IP. The PCEP messages are extended in this document to handle the PCECC operations for Native IP. Three new PCEP Objects (BGP Peer Info (BPI) Object, Explicit Peer Route (EPR) Object, and Peer Prefix Advertisement (PPA) Object) are defined in this document. Refer to Section 7 for detailed object definitions. All PCEP procedures specified in [RFC9050] continue to apply unless specified otherwise.

5.1. The PCInitiate Message

The PCInitiate Message defined in [RFC8281] and extended in [RFC9050] is further extended to support Native-IP CCI.

The format of the extended PCInitiate message is as follows:

```
<PCInitiate Message> ::= <Common Header>
    <PCE-initiated-lsp-list>

Where:
<Common Header> is defined in [RFC5440]

<PCE-initiated-lsp-list> ::= <PCE-initiated-lsp-request> [<PCE-initiated-lsp-list>]

<PCE-initiated-lsp-request> ::= (<PCE-initiated-lsp-instantiation>|
    <PCE-initiated-lsp-deletion>|
    <PCE-initiated-lsp-central-control>)

<PCE-initiated-lsp-central-control> ::= <SRP>
    <LSP>
    <cci-list>

<cci-list> ::= <CCI> [<BPI>|<EPR>|<PPA>] [<cci-list>]
```

Where:

<PCE-initiated-lsp-instantiation> and <PCE-initiated-lsp-deletion> are as per [RFC8281].

The LSP and SRP objects are defined in [RFC8231].

When the PCInitiate message is used for Native IP instructions, i.e. When the CCI Object-Type is 2, the SRP, LSP and CCI objects MUST be present. The error handling for missing SRP, LSP or CCI objects is as per [RFC9050]. Furthermore, one, and only one, object among BPI, EPR or PPA objects MUST be present. The PLSP-ID and Symbolic Path Name TLVs are set as per the existing rules in [RFC8231], [RFC8281], and [RFC9050]. The Symbolic Path Name is used by the PCE/PCC to uniquely identify the E2E native IP TE path. The related Native-IP instructions with BPI, EPR or PPA objects are identified by the same Symbolic Path Name.

If none of the BPI, EPR or PPA objects are present, the receiving PCC MUST send a PCErr message with Error-type=6 (Mandatory Object missing) and Error-value=19 (Native IP object missing). If there is more than one instance of BPI, EPR or PPA object present, the receiving PCC MUST send a PCErr message with Error-type=19 (Invalid Operation) and Error-value=22 (Only one BPI, EPR or PPA object can be included in this message).

When the PCInitiate message is not used for Native IP instructions, i.e. When CCI Object-Type is not equal to 2, the BPI, EPR and PPA objects SHOULD NOT be present. If present, they MUST be ignored by the receiver.

To clean up the existing Native IP instructions, the SRP object MUST set the R (remove) bit.

5.2. The PCRpt Message

The PCRpt message is used to acknowledge the Native-IP instructions received from the central controller (PCE) as well as during the State Synchronization phase.

The format of the PCRpt message is as follows:
<PCRpt Message> ::= <Common Header>
    <state-report-list>

Where:

<state-report-list> ::= <state-report>[<state-report-list>]

<state-report> ::= (<lsp-state-report>|
    <central-control-report>)

<lsp-state-report> ::= [<SRP>]
    <LSP>
    <path>

<central-control-report> ::= [<SRP>]
    <LSP>
    <cci-list>

<cci-list> ::= <CCI>
    [<BPI>|<EPR>|<PPA>]
    [<cci-list>]

Where: <path> is as per [RFC8231] and the LSP and SRP objects are also defined in [RFC8231].

The error handling for missing CCI objects is as per [RFC9050]. Furthermore, one, and only one, object among BPI, EPR or PPA object MUST be present.

If none of the BPI, EPR or PPA objects are present, the receiving PCE MUST send a PCErr message with Error-type=6 (Mandatory Object missing) and Error-value=19 (Native IP object missing). If there are more than one instance of BPI, EPR or PPA objects present, the receiving PCE MUST send a PCErr message with Error-type=19 (Invalid Operation) and Error-value=22 (Only one BPI, EPR or PPA object can be included in this message).

When the PCInitiate message is not used for Native IP instructions, i.e. When CCI Object-Type is not equal to 2, the BPI, EPR and PPA objects SHOULD NOT be present. If present, they MUST be ignored by the receiver.

6. PCECC Native IP TE Procedures

The detailed procedures for the TE in the native IP environment are described in the following sections.
6.1. BGP Session Establishment Procedures

The PCInitiate and PCRpt message pair is used to exchange the configuration parameters for a BGP peer session. This pair of PCEP messages are exchanged between a PCE and each BGP peer (acting as PCC) which needs to establish a BGP session. After the BGP peer session has been initiated via this pair of PCEP messages, the BGP session establishes and operates in a normal fashion. The BGP peers can be used for External BGP (EBGP) peers or Internal BGP (IBGP) peers. For IBGP connection topologies, the Route Reflector (RR) is required.

The PCInitiate message should be sent to PCC which is acting as BGP router and/or RR.

The RR topology for a single Autonomous System (AS) is shown in Figure 1. The BGP routers R1, R3, and R7 are within a single AS. R1 and R7 are BGP RR clients, and R3 is a RR. The PCInitiate message should be sent to the BGP routers R1, R3 and R7 that need to establish a BGP session.

PCInitiate message creates an auto-configuration function for these BGP peers by providing the indicated Peer AS and the Local/Peer IP Address.

When the PCC receives the BPI and CCI object (with the R bit set to 0 in the SRP object) in the PCInitiate message, the PCC should try to establish the BGP session with the indicated Peer as per AS and Local/Peer IP address.

During the establishment procedure, the PCC should report to the PCE the status of the BGP session via the PCRpt message, with the status field in the BPI object set to the appropriate value and the corresponding SRP and CCI objects included.

When the PCC receives this message with the R bit set to 1 in the SRP object in the PCInitiate message, the PCC should clear the BGP configuration and tear down the BGP session that is indicated by the BPI object.

When the PCC clears successfully the specified BGP session configuration, it should report the result via the PCRpt message, with the BPI object included, and the corresponding SRP and CCI objects.
Figure 1: BGP Session Establishment Procedures (R3 act as RR)

The message peers, message type, message key parameters and procedures in the above figures are shown below:
The Local/Peer IP address MUST be dedicated to the usage of the native IP TE solution, and MUST NOT be used by other BGP sessions that are established manually or in other ways. If the Local IP Address or Peer IP Address within the BPI object is used in other existing BGP sessions, the PCC MUST report such an error situation via a PCErr message with:

- **Error-type=33 (Native IP TE failure)** and **Error-value=1 (Local IP is in use)**, or

- **Error-type=33 (Native IP TE failure)** and **Error-value=2 (Remote IP is in use)**.

The detailed Error-Types and Error-Values are defined in Section 8.
If the established BGP session is broken, the PCC MUST report such information via PCRpt message with the status field set to "BGP session down" in the associated BPI Object. The error code field within the BPI object should indicate the reason that leads to the BGP session being down. In the future, when the BGP session is up again, the PCC MUST report that as well via the PCRpt message with the status field set to "BGP Session Established".

6.2. Explicit Route Establishment Procedures

The explicit route establishment procedures can be used by PCE to install a route on the PCC, using the PCInitiate and PCRpt message pair. Such explicit routes operate the same as static routes installed by network management protocols (Network Configuration Protocol (NETCONF)/YANG). The procedures of such explicit route addition and removal must be controlled by the PCE in a specific order so that the pathways are established without loops.

The PCInitiate message should be sent to every router on the path. In the example, for the explicit route from R1 to R7, the PCInitiate message should be sent to R1, R2 and R4, as shown in Figure 3. For the explicit route from R7 to R1, the PCInitiate message should be sent to R7, R4 and R2, as shown in Figure 5.

When the PCC receives the EPR and the CCI object (with the R bit set to 0 in the SRP object) in the PCInitiate message, the PCC should install the explicit route to the peer in the RIB/FIB.

When the PCC installs successfully the explicit route to the peer, it should report the result via the PCRpt messages, with the EPR object and the corresponding SRP and CCI objects included.

When the PCC receives the EPR and the CCI object with the R bit set to 1 in the SRP object in the PCInitiate message, the PCC MUST remove the explicit route to the peer that is indicated by the EPR object.

When the PCC has removed the explicit route that is indicated by this object, it should report the result via the PCRpt message, with the EPR object included, and the corresponding SRP and CCI object.
Figure 3: Explicit Route Establish Procedures (From R1 to R7)

The message peers, message type, message key parameters and procedures in the above figures are shown below:

Figure 4: Message Information and Procedures
Figure 5: Explicit Route Establish Procedures (From R7 to R1)

The message peers, message type, message key parameters and procedures in the above figures are shown below:

Figure 6: Explicit Route Establish Procedures (From R7 to R1)
To avoid the transient loop while deploying the explicit peer route, the EPR object should be sent to the PCCs in the reverse order of the E2E path. To remove the explicit peer route, the EPR object should be sent to the PCCs in the same order as the E2E path.

To accomplish ECMP effects, the PCE can send multiple EPR/CCI objects to the same node, with the same route priority and peer address value but a different next-hop address.

The PCC should verify that the next hop address is reachable. In case of failure, the PCC MUST send the corresponding error via PCErr message, with the error information: Error-type=33 (Native IP TE failure), Error-value=3 (Explicit Peer Route Error).

When the peer info is not the same as the peer info that is indicated in the BPI object in PCC for the same path that is identified by Symbolic Path Name TLV, a PCErr message MUST be reported, with the error information: Error-type=33 (Native IP TE failure), Error-value=4, EPR/BPI Peer Info Mismatch. Note that the same error can be used in case no BPI is received at the PCC.

If the PCE needs to update the path, it should first instruct the new CCI with updated EPR corresponding to the new next hop to use and then instruct the removal of the older CCI.

### 6.3. BGP Prefix Advertisement Procedures

The detailed procedures for BGP prefix advertisement are shown below, using the PCInitiate and PCRpt message pair.

The PCInitiate message should be sent to PCC that acts as a BGP peer edge router only. In the example, it should be sent to R1 and R7 respectively.

When the PCC receives the PPA and the CCI object (with the R bit set to 0 in the SRP object) in the PCInitiate message, the PCC should send the prefixes indicated in this object to the identified BGP peer via the corresponding BGP session [RFC4271].

When the PCC has successfully sent the prefixes to the appointed BGP peer, it should report the result via the PCRpt messages, with the PPA object and the corresponding SRP and CCI objects included.

When the PCC receives the PPA and the CCI object with the R bit set to 1 in the SRP object in the PCInitiate message, the PCC MUST withdraw the prefixes advertisement to the peer indicated by this object.
When the PCC withdraws successfully the prefixes that are indicated by this object, it should report the result via the PCRpt message, with the PPA object included, and the corresponding SRP and CCI objects.

```
+------------------+
|                  |
| R3               |
+------------------+
|                  |
+------------------+
| PCE              |
+------------------+

PCInitiate/PCRpt +---+ PCInitiate/PCRpt

+---+ +---+ +---+ +---+
| R1| +---+ +---+ +---+

(BGP Router)

+---+ +---+ +---+
| R5| +---+ +---+

(BGP Router)

+---+ +---+
| R6| +---+

Figure 7: BGP Prefix Advertisement Procedures

The message peers, message type, message key parameters and procedures in the above figures are shown below:

```
+-------+                                      +-------+
| PCC    |                                      |  PCE  |
| R1     |                                      +-------+
+------|       |                                           |
| PCC  |       |                                           |
| R7    |       | (Instruct R1 to advertise Prefix 1_A to R7) |
|       | -----| PPA Object(Peer IP=R7_A, Prefix=1_A)        |
|       |     |----PCRpt,CC-ID=X,Symbolic Path Name=Class A-->
|       |     |              PPA Object(Peer IP=R1_A, Prefix=7_A)   |
|       |     |----PCRpt,CC-ID=Z,Symbolic Path Name=Class A-----|
|       |     |              PPA Object(Peer IP=R1_A, Prefix=7_A)   |
|       |     |              PPA Object(Peer IP=R1_A, Prefix=7_A)   |

(Instruct R7 to advertise Prefix 7_A to R1 )

<--PCInitiate,CC-ID=Z,Symbolic Path Name=Class A-----

PFA Object(Peer IP=R1_A, Prefix=7_A)

Figure 8: Message Information and Procedures
The AFI/SAFI for the corresponding BGP session should match the Peer Prefix Advertisement Object-Type, AFI/SAFI should be 1/1 for the IPv4 prefix and 2/1 for the IPv6 prefix. In case of mismatch, an error: Error-type=33 (Native IP TE failure), Error-value=5 (BPI/PPA address family mismatch) SHOULD be reported via PCErr message.

When the peer info is not the same as the peer info that is indicated in the BPI object in PCC for the same path that is identified by Symbolic Path Name TLV, an error: Error-type=33 (Native IP TE failure), Error-value=6 (PPA/BPI peer info mismatch) SHOULD be reported via the PCErr message. Note that the same error can be used in case no BPI is received at the PCC.

6.4. Selection of Raw Mode and Tunnel Mode Forwarding Strategy

Normally, when the above procedures are finished, the user traffic will be forwarded via the appointed path, but the forwarding will be based solely on the destination of user traffic. If there is traffic from different attached points to the same destination coming into the network, they could share the priority path which may not be the initial desire. For example, as illustrated in Figure 1, the initial aim is to ensure traffic that enters the network via R1 and exits the network at R7 via R5-R6-R7. If some traffic enters the network via the R2 router, passes through R5 and exits at R7, they may share the priority path among R5-R6-R7, which may not be the desired effect.

The above normal traffic forwarding behavior is clarified as a Raw mode forwarding strategy. Such a mode can achieve only the moderate traffic path control effect. To achieve the strict traffic path control effect, the entry point should tunnel the user traffic from the entry point of the network to the exit point of the network, which is also between the BGP peer established via Section 6.1. Such forwarding behavior is called the Tunnel mode forwarding strategy. For simplicity, the IPinIP tunnel type is used between the BGP peers by default.

The selection of Raw mode and Tunnel mode forwarding strategies are controlled via the "T" bit in BPI Object that is defined in Section 7.2

6.5. Clean Up

To remove the Native-IP state from the PCC, the PCE MUST send explicit CCI cleanup instructions for PPA, EPR and BPI objects respectively with the R flag set in the SRP object. If the PCC receives a PCInitiate message but does not recognize the Native-IP information in the CCI, the PCC MUST generate a PCErr message with Error-Type=19 (Invalid operation) and Error-value=TBD2 (Unknown...
Native-IP Info) and MUST include the SRP object to specify the error is for the corresponding cleanup (via a PCInitiate message).

6.6. Other Procedures

The handling of the state synchronization, redundant PCEs, re-delegation and clean up is the same as other CCIs as specified in [RFC9050].

7. New PCEP Objects

One new CCI Object type and three new PCEP objects are defined in this document. All new PCEP objects are as per [RFC5440].

7.1. CCI Object

The Central Control Instructions (CCI) Object (defined in [RFC9050]) is used by the PCE to specify the forwarding instructions. This document defines another object type for Native-IP procedures.

CCI Object-Type is 2 for Native-IP as below:

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            CC-ID                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Reserved             |             Flags             |
|---------------------------------------------------------------|
|                                                               |
|                                                               |
|                                                        //                        Optional TLV                         // |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 9: CCI Object for Native IP

The field CC-ID is as described in [RFC9050]. The following fields are defined for CCI Object-Type 2

Reserved: is set to zero while sending and ignored on receipt.

Flags: is used to carry any additional information about the Native-IP CCI. Currently, no flag bits are defined. Unassigned flags are set to zero while sending and ignored on receipt.

The Symbolic Path Name TLV [RFC8231] MUST be included in the CCI Object-Type 2 to identify the E2E TE path in the Native IP environment.
7.2. BGP Peer Info Object

The BGP Peer Info object is used to specify the information about the peer with which the PCC should establish the BGP session. This object should only be included and sent to the source and destination router of the E2E path in case there is no Route Reflection (RR) involved. If the RR is used between the source and destination routers, then such information should be sent to the source router, RR and destination router respectively.

By default, the Local/Peer IP address MUST be dedicated to the usage of the native IP TE solution, and MUST NOT be used by other BGP sessions that are established by manual or other configuration mechanisms.

BGP Peer Info Object-Class is 46

BGP Peer Info Object-Type is 1 for IPv4 and 2 for IPv6

The format of the BGP Peer Info object body for IPv4 (Object-Type=1) is as follows:

```
 0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Peer AS Number                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   ETTL        |     Status    |   Error Code  |    Flag     |T|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Local IP Address                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Peer IP Address                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
//                    Optional TLVs                            //
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 10: BGP Peer Info Object Body Format for IPv4

The format of the BGP Peer Info object body for IPv6 (Object-Type=2) is as follows:

```
Peer AS Number: 4 Bytes, to indicate the AS number of Remote Peer. Note that if 2-byte AS numbers are in use, the low-order bits (16 through 31) MUST be used, and the high-order bits (0 through 15) MUST be set to zero.

ETTL: 1 Byte, EBGP Time To Live, to indicate the multi-hop count for the EBGP session. It should be 0 and ignored when Local AS and Peer AS are the same.

Status: 1 Byte, Indicate BGP session status between the peers. Its values are defined below:

- 0: Reserved
- 1: BGP Session Established
- 2: BGP Session Establishment In Progress
- 3: BGP Session Down
- 4-255: Reserved

Error Code: 1 Byte, Indicate the reason that the BGP session can’t be established.

- 0: Unspecific
1: ASes do not match, BGP Session Failure
2: Peer IP can’t be reached, BGP Session Failure
3-255: Reserved

Flag: 1 Byte.

Currently, only bit 7 (T bit) is defined. When the T bit is set, the traffic should be sent in the IPinIP tunnel (Tunnel source is Local IP Address, tunnel destination is Peer IP Address). When the T bit is cleared, the traffic is sent via its original source and destination address. The Tunnel mode (T bit is set) is used when the operator wants to ensure only the traffic from the specified (entry, exit) pair, and the Raw mode (T bit is clear) is used when the operator wants to ensure traffic from any entry to the specified destination. Unassigned flags are set to zero while sending and ignored on receipt.

Local IP Address (4/16 Bytes): IP address of the local router, used to peer with another end router. When Object-Type is 1, the length is 4 bytes; when Object-Type is 2, the length is 16 bytes.

Peer IP Address (4/16 Bytes): IP address of the peer router, used to peer with the local router. When Object-Type is 1, the length is 4 bytes; when Object-Type is 2, the length is 16 bytes;

Optional TLVs: TLVs that are associated with this object, can be used to convey other necessary information for dynamic BGP session establishment. No TLVs are currently defined.

When the PCC receives a BPI object, with Object-Type=1, it should try to establish a BGP session with the peer in AFI/Safi=1/1.

When the PCC receives a BPI object with Object-Type=2, it should try to establish a BGP session with the peer in AFI/Safi=2/1.

7.3. Explicit Peer Route Object

The Explicit Peer Route object is defined to specify the explicit peer route to the corresponding peer address on each device that is on the E2E Native-IP TE path. This Object should be sent to all the devices on the path that is calculated by the PCE. Although the object is named as Explicit Peer Route, it can be seen that the routes it installs are simply host routes. The use of this object to install host routes for any purpose other than reaching the corresponding peer address on each device that is on the E2E Native-
IP TE path is outside the scope of this specification.

It is RECOMMENDED that the path established by this object should have higher priority than the other paths calculated by dynamic IGP protocol, but should have lower priority than the static route configured by manual or NETCONF or any other static means.

Explicit Peer Route Object-Class is 47.

Explicit Peer Route Object-Type is 1 for IPv4 and 2 for IPv6

The format of the Explicit Peer Route object body for IPv4 (Object-Type=1) is as follows:

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       Route Priority        |          Reserved               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Peer IPv4 Address                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Next Hop IPv4 Address to the Peer               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
//                    Optional TLVs                            //
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 12: Explicit Peer Route Object Body Format for IPv4

The format of the Explicit Peer Route object body for IPv6 (Object-Type=2) is as follows:
Route Priority: 2 Bytes; the priority of this explicit route. The higher priority should be preferred by the device. This field is used to indicate the preferred path at each hop.

Reserved: is set to zero while sending, ignored on receipt.

Peer (IPv4/IPv6) Address: Peer Address for the BGP session (4/16 Bytes).

Next Hop (IPv4/IPv6) Address to the Peer: To indicate the next hop address (4/16 Bytes) to the corresponding peer address.

Optional TLVs: TLVs that are associated with this object, can be used to convey other necessary information for explicit peer path establishment. No TLVs are currently defined.

7.4. Peer Prefix Advertisement Object

The Peer Prefix Advertisement object is defined to specify the IP prefixes that should be advertised to the corresponding peer. This object should only be included and sent to the source/destination router of the E2E path.

The prefix information included in this object MUST only be advertised to the indicated peer, and MUST NOT be advertised to other BGP peers.

Peer Prefix Advertisement Object-Class is 48
Peer Prefix Advertisement Object-Type is 1 for IPv4 and 2 for IPv6

The format of the Peer Prefix Advertisement object body is as follows:

```
+-----------------------------------------------+  
| Peer IPv4 Address                           |  
+-----------------------------------------------+  
| No. of Prefix | Reserved                     |  
+-----------------------------------------------+  
| IPv4 Prefix #1                                   |  
| Prefix #1 Len | Reserved                     |  
+-----------------------------------------------+  
| IPv4 Prefix #n                                   |  
| Prefix #n Len | Reserved                     |  
+-----------------------------------------------+  
// Optional TLVs                                   
```

Figure 14: Peer Prefix Advertisement Object Body Format for IPv4
Peer IPv4 Address: 4 Bytes. Identifies the peer IPv4 address that the associated prefixes will be sent to.

No. of Prefix: 1 Byte. Identifies the number of prefixes that are advertised to the peer in the PPA object.

Reserved: 3 Bytes. MUST be set to zero while sending and MUST be ignored on receipt.

IPv4 Prefix: 4 Bytes. Identifies the prefix that will be sent to the peer identified by Peer IPv4 Address.

Prefix Len: 1 Byte. Identifies the length of the prefix.

Optional TLVs: TLVs that are associated with this object, can be used to convey other necessary information for prefix advertisement. No TLVs are currently defined.
For IPv6:

Peer IPv6 Address: 16 Bytes. Identifies the peer IPv6 address that the associated prefixes will be sent to.

IPv6 Prefix: Identifies the prefix that will be sent to the peer identified by Peer IPv6 Address.

If in the future, a requirement is identified to advertise IPv4 prefixes toward an IPv6 peering address, or IPv6 prefixes towards an IPv4 peering address, then a new Peer Prefix Advertisement Object-Types can be defined for these purposes.

8. New Error-Types and Error-Values Defined

A PCEP-ERROR object is used to report a PCEP error and is characterized by an Error-Type that specifies that type of error and an Error-value that provides additional information about the error. An additional Error-Type and several Error-values are defined to represent the errors related to the newly defined objects that are related to Native IP TE procedures.
<table>
<thead>
<tr>
<th>Error-Type</th>
<th>Meaning</th>
<th>Error-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>33</td>
<td>Native IP TE failure</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0:Unassigned</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1:Local IP is in use</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2:Remote IP is in use</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3:Explicit Peer Route Error</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4:EPR/BPI Peer Info mismatch</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5:BPI/PPA Address Family mismatch</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6:PPA/BPI Peer Info mismatch</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Mandatory Object missing</td>
<td></td>
</tr>
<tr>
<td></td>
<td>19:Native IP object missing</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Reception of an invalid object</td>
<td></td>
</tr>
<tr>
<td></td>
<td>39:PCECC NATIVE-IP-TE-CAPABILITY bit is not set</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>Invalid Operation</td>
<td></td>
</tr>
<tr>
<td></td>
<td>22:Only one BPI, EPR or PPA object can be included in this message</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TBD1:At tempted Native-IP operations when the capability was not advertised</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TBD2:Unknown Native-IP Info</td>
<td></td>
</tr>
</tbody>
</table>

Figure 16: Newly defined Error-Type and Error-Value
9. BGP Considerations

This document defines the procedures and objects to create the BGP sessions and advertise the associated prefixes dynamically. Only the key information, for example, peer IP addresses, and peer AS numbers are exchanged via the PCEP protocol. Other parameters that are needed for the BGP session setup should be derived from their default values.

When the PCE sends out the PCInitiate message with the BPI object embedded to establish the BGP session between the PCC peers, the PCC should report the BGP session status. For instance, the PCC could respond with "BGP Session Establishment In Progress" initially and on session establishment send another PCRpt message with the state updated to "BGP Session Established". If there is any error during the BGP session establishment, the PCC should indicate the reason with the appropriate status value set in the BPI object.

Upon receiving such key information, the BGP module on the PCC should try to accomplish the task appointed by the PCEP protocol and report the successful status to the PCEP modules after the session is set up.

There is no influence on the current implementation of BGP Finite State Machine (FSM). The PCEP focuses only on the success and failure status of the BGP session and acts upon such information accordingly.

The error-handling procedures related to incorrect BGP parameters are specified in Section 6.1, Section 6.2, and Section 6.3.

10. Deployment Considerations

The information transferred in this document is mainly used for the BGP session setup, explicit route deployment and the prefix distribution. The planning, allocation and distribution of the peer addresses within IGP should be accomplished in advance and they are out of the scope of this document.

The communication of PCE and PCC described in this document SHOULD follow the state synchronization procedures described in [RFC8232], treat the three newly defined objects (BPI, EPR and PPA) associated with the same symbolic path name as the attribute of the same path in the LSP-DB (LSP State Database).
When PCE detects one or some of the PCCs are out of its control, it should recompute and redeploy the traffic engineering path for native IP on the currently active PCCs. The PCE should ensure the avoidance of the possible transient loop in such node failure when it deploys the explicit peer route on the PCCs.

In case of a PCE failure, a new PCE can gain control over the central controller instructions as described in [RFC9050].

As per the PCEP procedures in [RFC8281], the State Timeout Interval timer is used to ensure that a PCE failure does not result in automatic and immediate disruption for the services. Similarly, as per [RFC9050], the central controller instructions are not removed immediately upon PCE failure. Instead, they could be re-delegated to the new PCE before the expiration of this timer, or be cleaned up on the expiration of this timer. This allows for network clean up without manual intervention. The PCC supports the removal of CCI as one of the behaviors applied on the expiration of the State Timeout Interval timer.

11. Manageability Considerations

11.1. Control of Function and Policy

A PCE or PCC implementation SHOULD allow the PCECC Native-IP capability to be enabled/disabled as part of the global configuration.

11.2. Information and Data Models

[ RFC7420 ] describes the PCEP MIB; this MIB could be extended to get the PCECC Native-IP capability status. The PCEP YANG [ I-D.ietf-pce-pcep-yang ] module could be extended to enable/disable the PCECC Native-IP capability.

11.3. Liveness Detection and Monitoring

Mechanisms defined in this document do not imply any new liveness detection and monitoring requirements in addition to those already listed in [ RFC5440 ]. The operator relies on existing IP liveness detection and monitoring.

11.4. Verify Correct Operations

Verification of the mechanisms defined in this document can be built on those already listed in [ RFC5440 ], [ RFC8231 ] and [ RFC9050 ]. Further, the operator needs to be able to verify the status of BGP sessions and prefix advertisements.
11.5. Requirements on Other Protocols

Mechanisms defined in this document require the interaction with BGP. Section 9 describes in detail the considerations regarding the BGP. During the BGP session establishment, the Local/Peer IP address MUST be dedicated to the usage of the native IP TE solution, and MUST NOT be used by other BGP sessions that are established manually or in other ways.

11.6. Impact on Network Operations

[RFC8821] describes the various deployment considerations in CCDR architecture and their impact on network operations.

12. Implementation Status

[Note to the RFC Editor - remove this section before publication, as well as remove the reference to RFC 7942.]

This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft, and is based on a proposal described in [RFC7942]. The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs. Please note that the listing of any individual implementation here does not imply endorsement by the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

According to [RFC7942], "This will allow reviewers and working groups to assign due consideration to documents that have the benefit of running code, which may serve as evidence of valuable experimentation and feedback that has made the implemented protocols more mature. It is up to the individual working groups to use this information as they see fit".

12.1. Proof of Concept based on ODL

At the time of posting the -26 version of this document, there are no known implementations of this mechanism. A proof of concept for the overall design has been verified using another SBI protocol on the Open Daylight (ODL) controller.
12.2. ZTE

ZTE is preparing an implementation of this document at the time of posting the -29 version of this document.

13. Security Considerations

In this setup, the BGP sessions, prefix advertisement, and explicit peer route establishment are all controlled by the PCE. See [RFC4271] for security consideration of classical BGP implementation, and [RFC4272] for classical BGP vulnerabilities analysis. Security considerations in [RFC5440] for basic PCEP protocol, [RFC8231] for PCEP extension for stateful PCE and [RFC8281] for PCE-Initiated LSP setup should be considered. To prevent a bogus PCE from sending harmful messages to the network nodes, the network devices should authenticate the validity of the PCE and ensure a secure communication channel between them. Thus, the mechanisms described in [RFC8253] for the usage of TLS for PCEP and [RFC9050] for malicious PCE should be used.

If suitable default values as discussed in Section 9 aren’t enough and securing the BGP transport is required (for example, the TCP-AO [RFC5925]), it can be provided through the addition of optional TLVs to the BGP Peer Info object that conveys the necessary additional information (for example, a key chain [RFC8177] name).

14. IANA Considerations

14.1. Path Setup Type Registry

[RFC8408] created a sub-registry within the "Path Computation Element Protocol (PCEP) Numbers" registry called "PCEP Path Setup Types". IANA is requested to allocate a new code point within this sub-registry, as follows:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>Native IP TE Path</td>
<td>This document</td>
</tr>
</tbody>
</table>

14.2. PCECC-CAPABILITY sub-TLV’s Flag field

[RFC9050] created a sub-registry within the "Path Computation Element Protocol (PCEP) Numbers" registry to manage the value of the PCECC-CAPABILITY sub-TLV’s 32-bit Flag field. IANA is requested to allocate a new bit position within this registry, as follows:

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>30</td>
<td>NATIVE IP</td>
<td>This document</td>
</tr>
</tbody>
</table>
14.3. PCEP Object

IANA is requested to allocate new codepoints in the "PCEP Objects" sub-registry as follows:

<table>
<thead>
<tr>
<th>Object-Class Value</th>
<th>Name</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>44</td>
<td>CCI Object</td>
<td>This document</td>
</tr>
<tr>
<td></td>
<td>Object-Type</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2: Native IP</td>
<td></td>
</tr>
<tr>
<td>46</td>
<td>BGP Peer Info</td>
<td>This document</td>
</tr>
<tr>
<td></td>
<td>Object-Type</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1: IPv4 address</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2: IPv6 address</td>
<td></td>
</tr>
<tr>
<td>47</td>
<td>Explicit Peer Route</td>
<td>This document</td>
</tr>
<tr>
<td></td>
<td>Object-Type</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1: IPv4 address</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2: IPv6 address</td>
<td></td>
</tr>
<tr>
<td>48</td>
<td>Peer Prefix Advertisement</td>
<td>This document</td>
</tr>
<tr>
<td></td>
<td>Object-Type</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1: IPv4 address</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2: IPv6 address</td>
<td></td>
</tr>
</tbody>
</table>

14.4. PCEP-Error Object

IANA is requested to allocate new error types and error values within the "PCEP-ERROR Object Error Types and Values" sub-registry of the PCEP Numbers registry for the following errors:
Error-Type  Meaning                             Error-value
6    Mandatory Object missing                           19:Native IP object missing
10   Reception of an invalid object                   39:PCECC NATIVE-IP-TE-CAPABILITY bit is not set
19   Invalid Operation                                 22:Only one BPI, EPR or PPA object can be included in this message
                                                TBD1:Attempted Native-IP operations when the capability was not advertised
                                                TBD2:Unknown Native-IP Info
33   Native IP TE failure                              1:Local IP is in use
                                            2:Remote IP is in use
                                            3:Explicit Peer Route Error
                                            4:EPR/BPI Peer Info mismatch
                                            5:BPI/PPA Address Family mismatch
                                            6:PPA/BPI Peer Info mismatch

The reference for the new Error-type/value should be set to this document.

14.5. CCI Object Flag Field

IANA is requested to create a new sub-registry to manage the Flag field of the new CCI Object called "CCI Object Flag Field for Native-IP". New values are to be assigned by IETF review [RFC8126]. Each bit should be tracked with the following qualities:

- bit number (counting from bit 0 as the most significant bit)
- capability description
- defining RFC

Currently, no flags are assigned.

14.6. BPI Object Status Code

IANA is requested to create a new sub-registry "BPI Object Status Code Field" within the "Path Computation Element Protocol (PCEP) Numbers". New values are assigned by IETF review [RFC8126]. Each value should be tracked with the following qualities: value, meaning, and defining RFC. The following values are defined in this document:
### BPI Object Error Code

IANA is requested to create a new sub-registry "BPI Object Error Code Field" within the "Path Computation Element Protocol (PCEP) Numbers". New values are assigned by IETF review [RFC8126]. Each value should be tracked with the following qualities: value, meaning, and defining RFC. The following values are defined in this document:

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Reserved</td>
<td>This document</td>
</tr>
<tr>
<td>1</td>
<td>ASes does not match, BGP Session Failure</td>
<td>This document</td>
</tr>
<tr>
<td>2</td>
<td>Peer IP can’t be reached, BGP Session Failure</td>
<td>This document</td>
</tr>
<tr>
<td>3-255</td>
<td>Unassigned</td>
<td>This document</td>
</tr>
</tbody>
</table>

### BPI Object Flag Field

IANA is requested to create a new sub-registry "BPI Object Flag Field" within the "Path Computation Element Protocol (PCEP) Numbers". New values are to be assigned by IETF review [RFC8126]. Each bit should be tracked with the following qualities:

- bit number (counting from bit 0 as the most significant bit)
- capability description
- defining RFC

The following values are defined in this document:

<table>
<thead>
<tr>
<th>Bit</th>
<th>Meaning</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-6</td>
<td>Unassigned</td>
<td>This document</td>
</tr>
<tr>
<td>7</td>
<td>T (IPnIP) bit</td>
<td>This document</td>
</tr>
</tbody>
</table>

## Acknowledgement

Thanks Mike Koldychev, Susan Hares, Siva Sivabalan and Adam Simpson for their valuable suggestions and comments.
17. References

17.1. Normative References


17.2. Informative References

[I-D.ietf-pce-pcep-yang]
Dhody, D., Beeram, V. P., Hardwick, J., and J. Tantsura,
"A YANG Data Model for Path Computation Element Communications Protocol (PCEP)",
Work in Progress, Internet-Draft, draft-ietf-pce-pcep-yang-25, 21 May 2024,

[RFC4272]
Murphy, S., "BGP Security Vulnerabilities Analysis",
RFC 4272, DOI 10.17487/RFC4272, January 2006,

[RFC7942]
Sheffer, Y. and A. Farrel, "Improving Awareness of Running Code: The Implementation Status Section",
BCP 205, RFC 7942, DOI 10.17487/RFC7942, July 2016,


Authors' Addresses

Aijun Wang
China Telecom
Beiqijia Town, Changping District
Beijing
Beijing, 102209
China
Email: wangaijun@tsinghua.org.cn

Boris Khasanov
MTS Web Services (MWS)
Andropova av., 18/9 115432
Moscow
Email: bhassanov@yahoo.com

Sheng Fang
Huawei Technologies
Huawei Bld., No.156 Beiqing Rd.
Beijing
China
Email: fsheng@huawei.com

Ren Tan  
Huawei Technologies  
Huawei Bld., No.156 Beiqing Rd.  
Beijing  
China  
Email: tanren@huawei.com

Chun Zhu  
ZTE Corporation  
50 Software Avenue, Yuhua District  
Nanjing  
Jiangsu, 210012  
China  
Email: zhu.chun1@zte.com.cn

Wang, et al.  
Expires 7 February 2025  
[Page 38]
Applicability of Abstraction and Control of Traffic Engineered Networks (ACTN) to Network Slicing
draft-ietf-teas-applicability-actn-slicing-07

Abstract

Network abstraction is a technique that can be applied to a network domain to obtain a view of potential connectivity across the network by utilizing a set of policies to select network resources.

Network slicing is an approach to network operations that builds on the concept of network abstraction to provide programmability, flexibility, and modularity. It may use techniques such as Software Defined Networking (SDN) and Network Function Virtualization (NFV) to create multiple logical or virtual networks, each tailored for a set of services that share the same set of requirements.

Abstraction and Control of Traffic Engineered Networks (ACTN) is described in RFC 8453. It defines an SDN-based architecture that relies on the concept of network and service abstraction to detach network and service control from the underlying data plane.

This document outlines the applicability of ACTN to network slicing in a Traffic Engineered (TE) network that utilizes IETF technologies. It also identifies the features of network slicing not currently within the scope of ACTN, and indicates where ACTN might be extended.

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 26 January 2025.

Copyright Notice

Copyright (c) 2024 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 Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction ............................................. 3
   1.1. Terminology ......................................... 4
2. Overview of Key Requirements for Network Slicing ............ 5
   2.1. Resource Partitioning ................................ 5
   2.2. Network Topology Customization and Virtualization .... 6
   2.3. Service Isolation .................................... 6
   2.4. Control and Orchestration ............................ 7
3. Abstraction and Control of Traffic Engineered (TE) Networks (ACTN): Overview of Key Components ............... 7
   3.1. ACTN Virtual Network as a Network Slice ............ 8
   3.2. ACTN Virtual Network and Scaling Network Slices ...... 9
   3.3. Management Components for ACTN and Network Slicing ... 9
   3.4. Examples of ACTN Delivering Network Slice Services ... 10
      3.4.1. ACTN Used for Virtual Private Line ............ 11
      3.4.2. ACTN Used for VPN Delivery Model ............. 13
      3.4.3. ACTN Used to Deliver a Virtual Customer Network ... 15
4. YANG Models .............................................. 16
   4.1. Network Slice Service Mapping from TE to ACTN VN Models ............................................. 16
   4.2. Interfaces and YANG Models .......................... 18
   4.3. ACTN VN Telemetry ................................... 19
5. IANA Considerations ...................................... 20
6. Security Considerations .................................. 20
7. Acknowledgements ...................................... 20
8. Contributors ........................................... 21
1. Introduction

The principles of network resource separation are not new. For years, the concepts of separated overlay and logical (virtual) networking have existed, allowing multiple services to be deployed over a single physical network comprised of a single or multiple layers. However, several key aspects differentiate overlay and virtual networking from network slicing.

A network slice is a virtual (that is, logical) network with its own network topology and a set of network resources that are used to provide connectivity that conforms to specific Service Level Agreements (SLAs) or a set of Service Level Objectives (SLOs). The network resources used to realize a network slice belong to the network that is sliced. The resources may be assigned and dedicated to an individual slice, or they may be shared with other slices enabling different degrees of service guarantee and providing different levels of isolation between the traffic in each slice.

[RFC9543] provides a definition for network slicing in the context of IETF network technologies. In particular, that document defines the term "IETF Network Slice" as the generic network slice concept applied to a network that uses IETF technologies. An IETF Network Slice could span multiple technologies (such as IP, MPLS, or optical) and multiple administrative domains. The logical network that is an IETF Network Slice may be kept separate from other concurrent logical networks with independent control and management: each can be created or modified on demand. Since this document is focused entirely on IETF technologies, it uses the term "network slice" as a more concise expression. Further discussion on the topic of IETF Network Slices and details of how an IETF Network Slice service may be requested and realized as an IETF Network Slice can be found in [RFC9543].

Within this document, the terms "network slice", "network slice service", and "network slice controller" refer to network slicing of networks built using IETF technologies as described in [RFC9543].

At one end of the spectrum, a Virtual Private Wire (VPW) or a Virtual Private Network (VPN) may be used to build a network slice. In these cases, the network slices do not require the service provider to isolate network resources for the provision of the service - the service is "virtual".
At the other end of the spectrum, there may be a detailed description of a complex network service that will meet the needs of a set of applications with connectivity and service function requirements that may include compute resources, storage capabilities, and access to content. Such a service may be requested dynamically (that is, instantiated when an application needs it, and released when the application no longer needs it), and modified as the needs of the application change. An example of such a type of service can be provided using an enhanced VPN described in [I-D.ietf-teas-enhanced-vpn]. It is often based on Traffic Engineering (TE) constructs in the underlay network.

Abstraction and Control of TE Networks (ACTN) [RFC8453] is a framework that facilitates the abstraction of underlying network resources to higher-layer applications and that allows network operators to create and supply virtual networks for their customers through the abstraction of the operators' network resources.

ACTN is a toolset capable of delivering network slice functionality. This document outlines the application of ACTN and associated enabling technologies to provide network slicing in a network that utilizes IETF TE-based technologies. It describes how the ACTN functional components can be used to support model-driven partitioning of resources into variable-sized bandwidth units to facilitate network sharing and virtualization. Furthermore, the use of model-based interfaces to dynamically request the instantiation of virtual networks can be extended to encompass requesting and instantiation of specific service functions (which may be both physical or virtual), and to partition network resources such as compute resources, storage capability, and access to content. In Section 3, the document highlights how the ACTN approach might be extended to address the requirements of network slicing where the underlying network is TE-capable.

1.1. Terminology

This document re-uses terminology from [RFC8453], [RFC9543] and [I-D.ietf-teas-enhanced-vpn].

Service Provider: See "Provider" in [RFC9543].

Consumer: See [RFC9543].

Service Functions (SFs): Components that provide specific functions within a network. SFs are often combined in a specific sequence called a service function chain to deliver services [RFC7665].

Resource: Any feature, including connectivity, buffers, compute,
storage, and content delivery that forms part of or can be accessed through a network. Resources may be shared between users, applications, and clients, or they may be dedicated for use by a unique customer.

Infrastructure Resources: The hardware and software for hosting and connecting SFs. These resources may include computing hardware, storage capacity, network resources (e.g., links and switching/routing devices enabling network connectivity), and physical assets for radio access.

Service Level Agreement (SLA): See [RFC9543].

Service Level Expectation (SLE): See [RFC9543].

Service Level Objective (SLO): See [RFC9543].

IETF Network Slice Service: See [RFC9543].

2. Overview of Key Requirements for Network Slicing

According to Section 6.2 of [RFC9543] "Expressing Connectivity Intents", the customer expresses requirements for a particular network slice by specifying what is required rather than how the requirement is to be fulfilled. That is, the customer’s view of a network slice is an abstract one expressed as a network slice service request.

The concept of network slicing is a key capability to serve a customer with a wide variety of different service needs expressed as SLOs/SLEs in terms of, e.g., latency, reliability, capacity, and service function-specific capabilities.

This section outlines the key capabilities required to realize network slicing in a TE-enabled IETF technology network.

2.1. Resource Partitioning

Network resources need to be allocated and dedicated for use by a specific network slice service, or they may be shared among multiple slice services. This allows a flexible approach that can deliver a range of services by partitioning (that is, slicing) the available network resources to make them available to meet the customer’s SLA.
2.2. Network Topology Customization and Virtualization

Network virtualization enables the creation of multiple virtual networks that are operationally decoupled from the underlying physical network, and are run on top of it. Slicing enables the creation of virtual networks as customer services.

2.3. Service Isolation

A customer may request, through their SLA, that changes to the other services delivered by the service provider do not have any negative impact on the delivery of the service. This quality is referred to as "isolation" in (Section 8 of [RFC9543]).

Delivery of service isolation may be achieved in the underlying network by various forms of resource partitioning ranging from dedicated allocation of resources for a specific slice, to sharing or resources with safeguards.

Although multiple network slices may utilize resources from a single underlying network, isolation should be understood in terms of the following three categorizations.

* Performance isolation requires that service delivery for one network slice does not adversely impact congestion, or performance levels perceived by the users of other slices.

* Security isolation means that attacks or faults occurring in one slice do not impact on other slices. Moreover, the security functions supporting each slice must operate independently so that an attack or misconfiguration of security in one slice will not prevent proper security function in the other slices. Further, privacy concerns require that traffic from one slice is not delivered to an endpoint in another slice, and that it should not be possible to determine the nature or characteristics of a slice from any external point.

* Management isolation means that each slice must be independently viewed, utilized, and managed as a separate network. Furthermore, it should be possible to prevent the operator of one slice from being able to control, view, or detect any aspect of any other network slice.
2.4. Control and Orchestration

An orchestrator is used to coordinate disparate processes and resources for creating, managing, and deploying the network slicing service in a network. The following aspects of orchestration should be considered:

* Multi-domain Orchestration: Managing connectivity to set up a network slice across multiple administrative domains.

* End-to-end Orchestration: Combining resources for an end-to-end service (e.g., underlay connectivity with firewalling, and guaranteed bandwidth with minimum delay).

3. Abstraction and Control of Traffic Engineered (TE) Networks (ACTN): Overview of Key Components

ACTN is designed to facilitate end-to-end connectivity and provides virtual connectivity services (such as virtual links and virtual networks) to the user. The ACTN framework [RFC8453] introduces three functional components and two interfaces:

* Customer Network Controller (CNC)

* Multi-domain Service Coordinator (MDSC)

* Provisioning Network Controller (PNC)

* CNC-MDSC Interface (CMI)

* MDSC-PNC Interface (MPI)

RFC 8453 also highlights how:

* Abstraction of the underlying network resources is provided to higher-layer applications and customers.

* Virtualization is achieved by selecting resources according to criteria derived from the details and requirements of the customer, application, or service.

* Creation of a virtualized environment is performed to allow operators to view and control multi-domain networks as a single virtualized network.

* A network is presented to a customer as a single virtual network via open and programmable interfaces.
The ACTN-managed infrastructure consists of traffic engineered network resources. The concept of traffic engineering is broad: it describes the planning and operation of networks using a method of reserving and partitioning of network resources in order to facilitate traffic delivery across a network (see [RFC9522] for more details). In the context of ACTN, traffic engineered network resources may include:

* Statistical packet bandwidth.
* Physical forwarding plane resources, such as wavelengths and time slots.
* Forwarding and cross-connect capabilities.

Therefore, an ACTN network may be "sliced" with each customer being given a different partial and abstracted topology view of the physical underlay network.

3.1. ACTN Virtual Network as a Network Slice

To support multiple customers, each with its own view and control of a virtual network constructed using an underlay network, a service provider needs to partition the network resources to create network slices assigned to each customer.

An ACTN Virtual Network (VN) is a customer view of a slice of the ACTN-managed infrastructure. It is a network slice that is presented to the customer by the ACTN provider as a set of abstracted resources. See [I-D.ietf-teas-actn-vn-yang] for a detailed description of ACTN VNs and an overview of how various different types of YANG models are applicable to the ACTN framework.

Depending on the agreement between a customer and a provider, various VN operations are possible:

* Network Slice Creation: A VN could be pre-configured and created through static configuration or through a dynamic request and negotiation between a customer and service provider. The VN must meet the network slice requirements specified in the SLA to satisfy the customers objectives.

* Network Slice Operations: The VN may be modified or deleted based on direct customer requests. Also, the way that the VN is engineered can be adjusted by the operator to continuously ensure that the delivered service complies with the requested SLA. The customer can further act upon the VN to manage their traffic flows across the network slice.
Network Slice View: A VN topology is viewed from the customer’s perspective. This may be the entire VN topology, or a collection of tunnels that are expressed as customer endpoints, access links, intra-domain paths and inter-domain links.

Section 3, "Virtual Network Primitives", in [RFC8454] describes a set of functional primitives that support these different ACTN VN operations.

3.2. ACTN Virtual Network and Scaling Network Slices

If the service provider must manage and maintain state in the core of the network for every network slice, then this will quickly limit the number of customer services that can be supported.

The importance of scalability for network slices is discussed in [I-D.ietf-teas-enhanced-vpn] and further in [I-D.ietf-teas-nrp-scalability]. That work notes the importance of collecting network slices or their composite connectivity constructs into groups that require similar treatment in the network before realizing those groups in the network.

The same consideration applies to ACTN VNs. But fortunately, ACTN VNs may be arranged hierarchically by recursing the MDSCs so that one VN is realized over another VN. This allows the VNs presented to the customer to be aggregated before they are instantiated in the physical network.

3.3. Management Components for ACTN and Network Slicing

The ACTN management components (CNC, MDSC, and PNC) and interfaces (CMI and MPI) are introduced in Section 3 and described in detail in [RFC8453]. The management components for network slicing are described in [RFC9543] and are known as the customer orchestration system, the IETF Network Slice Controller (NSC), and the network controller. The network slicing management components are separated by the Network Slice Service Interface and the Network Configuration Interface, modeling the architecture described in [RFC8309].

The mapping between network slicing management components and ACTN management components is presented visually in Figure 1 and provides a reference for understanding the material in Section 3.4 and Section 4.
Figure 1: Mapping Between IETF Network Slice and ACTN Management Components

Note 1 - The Service Orchestrator may also contain some MDSC service-related functions, as described in section 4.2 of [RFC8453].

Note 2 - The Service Orchestrator-to-MDSC Interface (XMI) is an interface between two MDSC functional elements encompassing different MDSC service-related functions which is not defined in [RFC8453].

3.4. Examples of ACTN Delivering Network Slice Services

The following examples build on the ACTN framework to provide control, management, and orchestration for the network slice life-cycle. These network slices utilize common physical infrastructure, and meet specific service-level requirements.

Three examples are shown. Each uses ACTN to achieve a different network slicing scenario. All three scenarios can be scaled up in capacity or be subject to topology changes as well as changes in customer requirements.
3.4.1. ACTN Used for Virtual Private Line

In the example shown in Figure 2, ACTN provides virtual connections between multiple customer locations (sites accessed through Customer Edge nodes - CEs). The service is requested by the customer (via CNC-A) and delivered as a Virtual Private Line (VPL) service. The characteristics of this model include the following benefits.

* Programmable: The service setup and operation is managed by the network provider via APIs.

* Virtual: The private line connectivity is provided from Site A to Site C (VPL1) and from Site B to Site C (VPL2) across the ACTN-managed physical network.

* Flexible: On-demand adjustments to the connectivity and bandwidth are available according to the customer’s requests, which may be automated.

In terms of the network slicing concept defined in [RFC9543], in this example the customer requests a single network slice with two pairs of point-to-point connectivity constructs between the service demarcation points CE1 and CE3, and CE2 and CE3 with each pair comprising one connectivity construct in each direction.
Boundary:
Between: Customer & Network Operator: CMI

Boundary:
Between: Consumer & Network Provider: XMI

Key: ... ACTN control connectivity
    === Physical connectivity
    --- Logical connectivity

Figure 2: Virtual Private Line Model
3.4.2. ACTN Used for VPN Delivery Model

In the example shown in Figure 3, ACTN provides VPN connectivity between two sites across three physical networks. The users of the two sites express the requirements for the VPN. The request is directed to the CNC, and the CNC interacts with the network provider’s MDSC. The main characteristics of this model are as follows.

* Provides edge-to-edge VPN multi-access connectivity.

* Most of the function is managed by the network provider, with some flexibility delegated to the customer-managed CNC.

In terms of the network slicing concept defined in [RFC9543], in this example, the customer requests a single network slice with a pair of point-to-point connectivity constructs (one in each direction) between the service demarcation points at site A and site B. The customer is unaware that the service is delivered over multiple physical networks.
Figure 3: VPN Model

Key: ... ACTN control connectivity
    === Physical connectivity
    --- Logical connectivity
3.4.3. ACTN Used to Deliver a Virtual Customer Network

In the example shown in Figure 4, ACTN provides a virtual network to the customer. This virtual network is managed by the customer. The figure shows two virtual networks (Network Slice 1 and Network Slice 2) each created for different customers under the care of different CNCs. There are two physical networks controlled by separate PNCs. Network Slice 2 is built using resources from just one physical network, while Network Slice 1 is constructed using resources from both physical networks.

The characteristics of this model include the following.

* The MDSC provides the topology to the customer so that the customer can control their network slice to fit their needs.
* Customers may interact with their assigned network slices directly. The customer may implement their own network control methods and traffic classification, mapping, prioritization, and manage their own addressing schemes.
* Customers may further slice their virtual networks so that this becomes a recursive model.
* Service isolation can be provided through selection of physical networking resources through a combination of efforts of the MSDC and PNC.
* The network slice may include nodes with specific capabilities. These can be delivered as Physical Network Functions (PNFs) or Virtual Network Functions (VNFs).
### 4. YANG Models

#### 4.1. Network Slice Service Mapping from TE to ACTN VN Models

The TE-service mapping model [I-D.ietf-teas-te-service-mapping-yang] creates a binding relationship across a L3VPN Service Model (L3SM) [RFC8299], L2VPN Service Model (L2SM) [RFC8466], and TE Tunnel model [I-D.ietf-teas-yang-te], via the generic ACTN Virtual Network (VN) model [I-D.ietf-teas-actn-vn-yang].

---

**Figure 4: Network Slicing**

Key: --- ACTN control connection

... Virtualization/abstraction through slicing
When necessary, it must be possible to map between a slice service request and an ACTN VN model. The ACTN VN model is a generic virtual network service model that allows customers to specify a VN that meets the customer’s service objectives with various constraints, which could be included in the initial request, and how the service is delivered. Therefore, a request for a network slice service may be mapped directly to a request for a VN.

The TE-service mapping model [I-D.ietf-teas-te-service-mapping-yang] binds the L3SM with TE-specific parameters. This binding facilitates seamless service operation and enables visibility of the underlay TE network. The TE-service model developed in that document can also be extended to support other services, including L2SM, and the Layer 1 Connectivity Service Model (L1CSM) [I-D.ietf-ccamp-l1csm-yang] L1CSM network service models.

Figure 5 shows the relationship between the YANG models discussed above.

```
| ------------------------ | <= | ------------------------ |
| L3SM                     |   | Augmented Service Model  |
| ------------------------ | <= | References              |
| L2SM                     |   | References              |
| ------------------------ | <= | References              |
| L1CSM                    |   | References              |
| ------------------------ | <= | References              |
| TE & Service             | <= | References              |
| Mapping Types            | import | References |
```

Figure 5: Relationships Between YANG Models

Work is still needed to define YANG models to help map network slice services to Traffic Engineering (TE) models. For example, [I-D.dhody-teas-ietf-network-slice-mapping] shows how the Virtual Network (VN) model and the TE Tunnel model can support network slice services.
4.2. Interfaces and YANG Models

Figure 6 shows the two ACTN components (MDSC and PNC) and one ACTN interface (MPI), as listed in Section 3. The figure also shows the Device Configuration Interface between the PNC and the devices in the physical network. That interface might be used to install state on every device in the network, or might instruct a "head-end" node when a control plane is used within the physical network. In the context of [RFC8309], the Device Configuration Interface uses one or more device configuration models.

Figure 6 also shows the Network Slice Service Interface. This interface allows a customer to make requests for delivery of the service, and it facilitates the customer modifying and monitoring the service. In the context of [RFC8309], this is a customer service interface and uses a service model.

When an ACTN system is used to manage the delivery of network slices, a network slice resource model is needed. This model will be used for instantiation, operation, and monitoring of network and function resource slices. The YANG model defined in [I-D.ietf-teas-ietf-network-slice-nbi-yang] provides a suitable basis for requesting, controlling, and deletion, of a Network Slice Service.
Figure 6: The YANG Interfaces in Context

4.3. ACTN VN Telemetry

The ACTN VN telemetry model [I-D.ietf-teas-actn-pm-telemetry-autonomics] provides a way for a customer to define performance monitoring relevant for the VN/network slice via the NETCONF subscription mechanisms [RFC8639], [RFC8640], or using the equivalent mechanisms in RESTCONF [RFC8641], [RFC8650].

Key characteristics of [I-D.ietf-teas-actn-pm-telemetry-autonomics] include the following:

* An ability to provide scalable VN-level telemetry aggregation based on a customer subscription model for key performance parameters defined by the customer.

* An ability to facilitate proactive re-optimization and reconfiguration of VNs/network slices based on autonomic network traffic engineering scaling configuration mechanisms.
5. IANA Considerations

This document makes no requests for action by IANA.

6. Security Considerations

Network slicing involves the control of network resources in order to meet the service requirements of customers. In some deployment models using ACTN, the customer may directly request a modification in the behaviour of resources owned and operated by a service provider. Such changes could significantly affect the service providers ability to provide services to other customers. Furthermore, the resources allocated for or consumed by a customer will typically be billable by the service provider.

Therefore, it is crucial that the mechanisms used in any network slicing system allow for authentication of requests, security of those requests, and tracking of resource allocations.

It should also be noted that while the partitioning or slicing of resources is virtual, as mentioned in Section 2.3 the customers expect and require that there is no risk data leakage from one slice to another, no transfer of knowledge of the structure or even existence of other slices. Further, in some service requests, there is an expectation that changes to one slice (under the control of one customer) should not have detrimental effects on the operation of other slices (whether under control of different or the same customers) even within limits allowed within the SLA. Thus, slices are assumed to be private and to provide the appearance of genuine physical connectivity.

Some service providers may offer secure network slices as a service. Such services may claim to include edge-to-edge encryption for the customer’s traffic. However, a customer should take full responsibility for the privacy and integrity of their traffic and should carefully consider using their own edge-to-edge encryption.

ACTN operates using the NETCONF [RFC6241] or RESTCONF [RFC8040] protocols and assumes the security characteristics of those protocols. Deployment models for ACTN should fully explore the authentication and other security aspects before networks start to carry live traffic.

7. Acknowledgements

Thanks to Italo Busi, Qin Wu, Andy Jones, Ramon Casellas, Gert Grammel, and Kiran Makhijani for their insight and useful discussions about network slicing.
This work is partially supported by the European Commission under Horizon 2020 grant agreement number 101015857 Secured autonomic traffic management for a Tera of SDN flows (Teraflow).

8. Contributors

The following people contributed text to this document.

Young Lee  
Email: younghlee.tx@gmail.com

Mohamed Boucadair  
Email: mohamed.boucadair@orange.com

Sergio Belotti  
Email: sergio.belotti@nokia.com

Daniele Ceccarelli  
Email: dceccare@cisco.com

9. Normative References


10. Informative References


Internet-Draft          ACTN and Network Slicing               July 2024

Mapping YANG Data Model", Work in Progress, Internet-
Draft, draft-ietf-teas-te-service-mapping-yang-15, 16
March 2024, <https://datatracker.ietf.org/doc/html/draft-

[I-D.ietf-teas-yang-te]
Saad, T., Gandhi, R., Liu, X., Beeram, V. P., and I.
Bryskin, "A YANG Data Model for Traffic Engineering
Tunnels, Label Switched Paths and Interfaces", Work in
Progress, Internet-Draft, draft-ietf-teas-yang-te-36, 2
draft-ietf-teas-yang-te-36>.

and A. Bierman, Ed., "Network Configuration Protocol
(NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011,

Chaining (SFC) Architecture", RFC 7665,
DOI 10.17487/RFC7665, October 2015,

Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017,

"YANG Data Model for L3VPN Service Delivery", RFC 8299,
DOI 10.17487/RFC8299, January 2018,

Explained", RFC 8309, DOI 10.17487/RFC8309, January 2018,

[RFC8454]  Lee, Y., Belotti, S., Dhody, D., Ceccarelli, D., and B.
Yoon, "Information Model for Abstraction and Control of TE
Networks (ACTN)", RFC 8454, DOI 10.17487/RFC8454,

Data Model for Layer 2 Virtual Private Network (L2VPN)
Service Delivery", RFC 8466, DOI 10.17487/RFC8466, October
Internet-Draft ACTN and Network Slicing July 2024


Authors’ Addresses

Daniel King
Old Dog Consulting
Email: daniel@olddog.co.uk

John Drake
Independent
Email: je_drake@yahoo.com

Haomian Zheng
Huawei Technologies
Email: zhenghaomian@huawei.com

Adrian Farrel
Old Dog Consulting
Email: adrian@olddog.co.uk

A Routing Architecture for Satellite Networks
draft-li-arch-sat-07

Abstract

Satellite networks present some interesting challenges for packet networking. The entire topology is continually in motion, with links far less reliable than what is common in terrestrial networks. Some changes to link connectivity can be anticipated due to orbital mechanics.

This document proposes a scalable routing architecture for satellite networks based on existing routing protocols and mechanisms, enhanced with scheduled link connectivity change information. This document proposes no protocol changes.

This document presents the author’s view and is neither the product of the IETF nor a consensus view of the community.

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 4 January 2025.

Copyright Notice

Copyright (c) 2024 IETF Trust and the persons identified as the document authors. All rights reserved.
1. Introduction

Satellite networks present some interesting challenges for packet networking. The entire topology is continually in motion, with links far less reliable than what is common in terrestrial networks. Some changes to link connectivity can be anticipated due to orbital mechanics.
This document proposes a scalable routing architecture for satellite networks based on existing routing protocols and mechanisms, enhanced with scheduled link connectivity change information. This document proposes no protocol changes.

Large-scale satellite networks are being deployed, presenting an unforeseen application for conventional routing protocols. The high rate of intentional topological change and the extreme scale are unprecedented in terrestrial networking. Links between satellites can utilize free-space optics technology that allows liberal connectivity. Still, there are limitations due to the range of the links and conjunction with the sun, resulting in links that are far less reliable than network designers are used to. In addition, links can change their endpoints dynamically, resulting in structural changes to the topology.

Current satellite networks are proprietary and little information is generally available for analysis and discussion. This document is based on what is currently accessible.

This document proposes one approach to provide a routing architecture for such networks utilizing current standards-based routing technology and providing a solution for the scalability of the network while incorporating the rapid rate of topological change. This document intends to provide some initial guidance for satellite network operators, but without specific details, this document can only provide the basis for a more complete analysis and design.

This document presents the author’s view and is neither the product of the IETF nor a consensus view of the community.

1.1. Related Work

A survey of related work can be found in [Westphal]. Link state routing for satellite networks has been considered in [Cao] and [Zhang].

1.2. Terms and Abbreviations

* Constellation: A set of satellites.

* Downlink: The half of a ground link leading from a satellite to an Earth station.

* Gateway: An Earth station that participates in the network and acts as the interconnect between satellite constellations and the planetary network. Gateways have a much higher bandwidth than user stations, have ample computing capabilities, and perform
traffic engineering duties, subsuming the functionality of a network controller or Path Computation Element (PCE). [RFC4655] Multiple gateways are assumed to exist, each serving a portion of the network.

* GEO: Geostationary Earth Orbit. A satellite in GEO has an orbit that is synchronized to planetary rotation, so it effectively sits over one spot on the planet.

* Ground link: A link between a satellite and an Earth station.

* Earth station: A node in the network that is on or close to the planetary surface and has a link to a satellite. This includes ships, aircraft, and other vehicles below LEO. [ITU]

* IGP: Interior Gateway Protocol. A routing protocol that is used within a single administrative domain. Note that ‘gateway’ in this context is semantically equivalent to ‘router’ and has no relationship to the ‘gateway’ used in the rest of this document.

* IS-IS: Intermediate System to Intermediate System routing protocol. An IGP that is commonly used by service providers. [ISO10589] [RFC1195]

* ISL: Inter-satellite link. Frequently implemented with free-space optics that allow signaling using photons without any intervening medium. [Bell]

* L1: IS-IS Level 1

* L1L2: IS-IS Level 1 and Level 2

* L2: IS-IS Level 2

* LEO: Low Earth Orbit. A satellite in LEO has an altitude of 2,000km or less.

* Local gateway: Each user station is associated with a single gateway in its region.

* LSP: IS-IS Link State Protocol Data Unit. An LSP is a set of packets that describe a node’s connectivity to other nodes.

* MEO: Medium Earth Orbit. A satellite in MEO is between LEO and GEO orbits and has an altitude between 2,000km and 35,786km.

* SID: Segment Identifier [RFC8402]
2. Overview

2.1. Topological Considerations

Satellites travel in specific orbits around their parent planet. Some of them have their orbital periods synchronized to planetary rotation, so they are effectively stationary over a single point. Other satellites have orbits that cause them to travel across regions of the planet gradually or quite rapidly. Respectively, these are typically known as Geostationary Earth Orbits (GEO), Medium Earth Orbit (MEO), or Low Earth Orbit (LEO), depending on altitude. This discussion is not Earth-specific; as we get to other planets, we can test this approach’s generality.

Satellites may have data interconnections with one another through Inter-Satellite Links (ISLs). Due to differences in orbits, ISLs may be connected temporarily, with periods of potential connectivity computed through orbital mechanics. Multiple satellites may be in the same orbit but separated in space, with a roughly constant separation. Satellites in the same orbit may have ISLs that have a higher duty cycle than ISLs between different orbits but are still not guaranteed to be always connected.

Earth stations can communicate with one or more satellites in their region. User stations are Earth stations with a limited capacity and communicate with only a single satellite at a time. Other Earth stations that may have richer connectivity and higher bandwidth are commonly called gateways and provide connectivity between the satellite network and conventional wired networks. Gateways serve user stations in their geographic proximity and are replicated globally as necessary to provide coverage and meet service density...
goals. User stations are associated with a single local gateway. Traffic from one Earth station to another may need to traverse a path across multiple satellites via ISLs.

2.2. Link Changes

Like conventional network links, ISLs and ground links can fail without warning. However, unlike terrestrial links, there are predictable times when ISLs and ground links can potentially connect and disconnect. These predictions can be computed and cataloged in a schedule that can be distributed to relevant network elements. Predictions of a link connecting are not guaranteed: a link may not connect for many reasons. Link disconnection predictions due to orbital mechanics are effectively guaranteed, as the underlying physics will not improve unexpectedly.

2.3. Scalability

Some proposed satellite networks are fairly large, with tens of thousands of proposed satellites. [CNN] A key concern is the ability to reach this scale and larger, as useful networks tend to grow.

As we know, the key to scalability is the ability to create hierarchical abstractions, so a key question of any routing architecture will be about the abstractions that can be created to contain topological information.

Normal routing protocols are architected to operate with a static but somewhat unreliable topology. Satellite networks lack the static organization of terrestrial networks, so normal architectural practices for scalability may not apply and alternative approaches may need consideration.

In a traditional deployment of a link-state routing protocol, current implementations can be deployed with a single area that spans a few thousand routers. A single area would also provide no isolation for topological changes, causing every link change to be propagated throughout the entire network. This would be insufficient for the needs of large satellite networks.

Multiple areas or multiple instances of an IGP can be used to improve scalability, but there are limitations to traditional approaches.

The IETF currently actively supports two link-state Interior Gateway Protocols (IGPs): OSPF [RFC2328][RFC5340] and IS-IS.
OSPF requires that the network operate around a backbone area, with subsidiary areas hanging off of the backbone. While this works well for traditional terrestrial networks, this does not seem appropriate for satellite networks, where there is no centralized portion of the topology.

IS-IS has a different hierarchical structure, where Level 1 (L1) areas are connected sets of nodes, and then Level 2 (L2) is a connected subset of the topology that intersects all of the L1 areas. Individual nodes can be L1, L2, or both (L1L2). Traditional IS-IS designs require that any node or link that is to be used as transit between L2 areas must appear as part of the L2 topology. In a satellite network, any satellite could end up being used for L2 transit, and so every satellite and link would be part of L2, negating any scalability benefits from IS-IS’s hierarchical structure.

We elaborate on IS-IS-specific considerations in Section 4.

2.4. Assumptions

In this section, we discuss some of the assumptions that are the basis for this architectural proposal.

The data payload is IP packets.

Satellites are active participants in the control and data plane for the network, participating in protocols, and forwarding packets.

There may be a terrestrial network behind each gateway that may interconnect to the broader Internet. The architecture of the terrestrial network is assumed to be a typical IS-IS and BGP [RFC4271] deployment and is not discussed further.

The satellite network interconnects user stations and gateways. Interconnection between the satellite network and the satellite networks of other network operators is outside of the scope of this document.

2.4.1. Traffic Patterns

We assume that the primary use of the satellite network is to provide access from a wide range of geographic locations. We also assume that providing high-bandwidth bulk transit between peer networks is not a goal. It has been noted that satellite networks can provide lower latencies than terrestrial fiber networks [Handley]. This proposal does not preclude such applications but does not articulate the mechanisms necessary for user stations to perform the appropriate
traffic engineering computations. Low-latency, multicast, and anycast applications are not discussed further.

As with most access networks, we assume that there will be bidirectional traffic between the user station and the gateway, but that the bulk of the traffic will be from the gateway to the user station. We expect that the uplink from the gateway to the satellite network to be the bandwidth bottleneck, and that gateways will need to be replicated to scale the uplink bandwidth, as the satellite capacity reachable from a gateway will be limited.

We assume that it is not essential to provide optimal routing for traffic from user station to user station. If this traffic is sent to a gateway first and then back into the satellite network, this might be acceptable to some operators as long as the traffic volume remains very low. This type of routing is not discussed further.

We assume that traffic for a user station should enter the satellite network through a gateway that is in some close geographic proximity to the user station. This is to reduce the number of ISLs used by the path to the user station. Similarly, we assume that user station traffic should exit the satellite network through the gateway that is in the closest geographic proximity to the user station. Jurisdictional requirements for landing traffic in certain regions may alter these assumptions, but such situations are outside of the scope of this document.

This architecture does not preclude gateway-to-gateway traffic across the satellite constellations, but it does not seek to optimize it.

2.4.2. User Station Constraints

The user station is an entity whose operation is conceptually shared between the satellite constellation operator and the operator of the cluster of end stations it serves. For example, the user station is trusted to attach MPLS label stacks to end-user packets. It gets the information to do so from some combination of its direct satellite and its local gateway, via protocols outside the scope of this document. Equally, it bootstraps communication via an exchange with the current local satellite so it can find and communicate with its local gateway, again with the details of how that is done being outside the scope of this document.

User stations that can concurrently access multiple satellites are not precluded by this proposal, but are not discussed in detail.
2.4.3. Stochastic Connectivity

We assume that links in general will be available when scheduled. As with any network, there will be failures, and the schedule is not a guarantee, but we also expect that the schedule is mostly accurate. We assume that at any given instant, there are enough working links and aggregate bandwidth to run the network and support the traffic demand. If this assumption does not hold, no routing architecture can magically make the network more capable.

Satellites that are in the same orbit may be connected by ISLs. These are called intra-orbit ISLs. Satellites that are in different orbits may also be connected by ISLs. These are called inter-orbit ISLs. We assume that, in general, intra-orbit ISLs have higher reliability and persistence than inter-orbit ISLs.

We assume that the satellite network is connected (in the graph theory sense) almost always, even if some links are down. This implies that there is almost always some path to the destination. In the extreme case with no such path, we assume that it is acceptable to drop the payload packets. We do not require buffering of traffic when a link is down. Instead, traffic should be rerouted.

2.5. Problem Statement

The goal of the routing architecture is to provide an organizational structure to protocols running on the satellite network such that topology information is conveyed through relevant portions of the network, that paths are computed across the network, and that data can be delivered along those paths, and the structure can scale without any changes to the organizational structure.

3. Forwarding Plane

The end goal of a network is to deliver traffic. In a satellite network where the topology is in a continual state of flux and the user stations frequently change their association with the satellites, having a highly flexible and adaptive forwarding plane is essential. Toward this end, we propose to use MPLS as the fundamental forwarding plane architecture [RFC3031]. Specifically, we propose to use a Segment Routing (SR) [RFC8402] based approach with an MPLS data plane [RFC8660], where each satellite is assigned a node Segment Identifier (SID). This allows the architecture to support both IPv4 and IPv6 concurrently. A path through the network can then be expressed as a label stack of node SIDs. IP forwarding is not used within the internals of the satellite network, although each satellite may be assigned an IP address for management purposes. Existing techniques may be used to limit the size of the SR label.
stack so that it only contains the significant waypoints along the path [Giorgetti]. The label stack operates as a loose source route through the network. If there is an unexpected topology change in the network, the IGP will compute a new path to the next waypoint, allowing packet delivery despite ISL failures. While the IGP is converging, there may be micro-loops in the topology. These can be avoided by using TI-LFA alternate paths [I-D.ietf-rtgwg-segment-routing-ti-lfa], or traffic will loop until discarded based on its TTL.

We assume that there is a link-layer mechanism for a user station to associate with a satellite. User stations will have an IP address assigned from a prefix managed by its local gateway. The mechanisms for this assignment and its communication to the end station are not discussed herein but might be similar to DHCP [RFC2131]. User station IP addresses change infrequently and do not reflect their association with their first-hop satellite. Gateways and their supporting terrestrial networks advertise prefixes covering all its local user stations into the global Internet.

User stations may be assigned a node SID, in which case MPLS forwarding can be used for all hops to the user station. Alternatively, if the user station does not have a node SID, then the last hop from the satellite to the end station can be performed based on the destination IP address of the packet. This does not require a full longest-prefix-match lookup as the IP address is merely a unique identifier at this point.

Similarly, gateways may be assigned a node SID. A possible optimization is that a single SID value be assigned as a global constant to always direct traffic to the topologically closest gateway. If traffic engineering is required for traffic that is flowing to a gateway, a specific path may be encoded in a label stack that is attached to the packet by the user station or by the first-hop satellite.

Gateways can also perform traffic engineering using different paths and label stacks for separate traffic flows. Routing a single traffic flow across multiple paths has proven to cause performance issues with transport protocols, so that approach is not recommended. Traffic engineering is discussed further in Section 6.
4. IGP Suitability and Scalability

As discussed in Section 2.3, IS-IS is architecturally the best fit for satellite networks, but does require some novel approaches to achieve the scalability goals for a satellite network. In particular, we propose that all nodes in the network be L1L2 so that local routing is done based on L1 information and then global routing is done based on L2 information.

IS-IS has the interesting property that it does not require interface addresses. This feature is commonly known as ‘unnumbered interfaces’. This is particularly helpful in satellite topologies because it implies that ISLs may be used flexibly. Sometimes an interface might be used as an L1 link to another satellite and a few orbits later it might be used as an L1L2 link to a completely different satellite without any reconfiguration or renumbering.

Scalability for IS-IS can be achieved through a proposal known as Area Proxy [I-D.ietf-lsr-isis-area-proxy]. With this proposal, all nodes in an L1 area combine their information into a single L2 Link State Protocol Data Unit (LSP). This implies that the size of the L1 Link State Database (LSDB) scales as the number of nodes in the L1 area and the size of the L2 LSDB scales with the number of L1 areas.

With Area Proxy, topological changes within an L1 area will not be visible to other areas, so the overhead of link state changes will be greatly reduced.

The Area Proxy proposal also includes the concept of an Area SID. This is useful because it allows traffic engineering to construct a path that traverses areas with a minimal number of label stack entries.

Suppose, for example, that a network has 1,000 L1 areas, each with 1,000 satellites. This would then mean that the network supports 1,000,000 satellites, but only requires 1,000 entries in its L1 LSDB and 1,000 entries in its L2 LSDB; numbers that are easily achievable today. The resulting MPLS label table would contain 1,000 node SIDs from the L1 (and L2) LSDB and 1,000 area SIDs from the L2 LSDB. If each satellite advertises an IP address for management purposes, then the IP routing table would have 1,000 entries for the L1 management addresses and 1,000 area proxy addresses from L2.

In this proposal, IS-IS does not carry IP routes other than those in the satellite topology. In particular, there are no IP routes for user stations or the remainder of the Internet.
5. Stripes and Areas

A significant problem with any link state routing protocol is that of area partition. While there have been many proposals for automatic partition repair, none has seen notable production deployment. It seems best to avoid this issue and ensure areas have an extremely low probability of partitioning.

As discussed above, intra-orbit ISLs are assumed to have higher reliability and persistence than inter-orbit ISLs. However, even intra-orbit ISLs are not sufficiently reliable to avoid partition issues. Therefore, we propose to group a small number of adjacent orbits as an IS-IS L1 area, called a stripe. We assume that for any given reliability requirement, there is a small number of orbits that can be used to form a stripe that satisfies the reliability requirement.

Stripes are connected to other adjacent stripes using the same ISL mechanism, forming the L2 topology of the network. Each stripe should have multiple L2 connections and never become partitioned from the remainder of the network.

By using a stripe as an L1 area, in conjunction with Area Proxy, the overhead of the architecture is greatly reduced. Each stripe contributes a single LSP to the L2 LSDB, completely hiding all the details about the satellites within the stripe. The resulting architecture scales proportionately to the number of stripes required, not the number of satellites.

Groups of MEO and GEO satellites with interconnecting ISLs can also form an IS-IS L1L2 area. Satellites that lack intra-constellation ISLs are better as independent L2 nodes.

6. Traffic Forwarding and Traffic Engineering

Forwarding in this architecture is straightforward. A path from a gateway to a user station on the same stripe only requires a single node SID for the satellite that provides the downlink to the user station.
Similarly, a user station returning a packet to a gateway need only provide a gateway node SID.

For off-stripe forwarding, the situation is a bit more complex. A gateway would need to provide the area SID of the final stripe on the path plus the node SID of the downlink satellite. For return traffic, user stations or first-hop satellites would want to provide the area SID of the stripe that contains the satellite that provides access to the gateway as well as the gateway SID.

As an example, consider a packet from an Internet source S to a user station D. A local gateway L has injected a prefix covering D into BGP and advertised it globally. The packet is forwarded to L using IP forwarding. When L receives the packet, it performs a lookup in a
pre-computed forwarding table. This contains a SID list for the user station that has already been converted into a label stack. Suppose the user station is currently associated with a different stripe so that the label stack will contain an area label A and a label U for the satellite associated with the user station, resulting in a label stack (A, U).

The local gateway forwards this into the satellite network. The first-hop satellite now forwards based on the area label A at the top of the stack. All area labels are propagated as part of the L2 topology. This forwarding continues until the packet reaches a satellite adjacent to the destination area. That satellite pops label A, removing that label and forwarding the packet into the destination area.

The packet is now forwarded based on the remaining label U, which was propagated as part of the L1 topology. The last satellite forwards the packet based on the destination address D and forwards the packet to the user station.

The return case is similar. The label stack, in this case, consists of a label for the local gateway’s stripe/area, A’, and the label for the local gateway, L, resulting in the stack (A’, L). The forwarding mechanisms are similar to the previous case.

Very frequently, access networks congest due to oversubscription and the economics of access. Network operators can use traffic engineering to ensure that they get higher efficiency out of their networks by utilizing all available paths and capacity near any congestion points. In this particular case, the gateway will have information about all of the traffic it is generating and can use all of the possible paths through the network in its topological neighborhood. Since we’re already using SR, this is easily done just by adding more explicit SIDs to the label stack. These can be additional area SIDs, node SIDs, or adjacency SIDs. Path computation can be performed by Path Computation Elements (PCE). [RFC4655]

Each gateway or its PCE will need topological information from the areas it will route through. It can do this by participating in the IGP directly, via a tunnel, or another delivery mechanism such as BGP-LS [RFC9552]. User stations do not participate in the IGP.
Traffic engineering for packets flowing into a gateway can also be provided by an explicit SR path. This can help ensure that ISLs near the gateway do not congest with traffic for the gateway. These paths can be computed by the gateway or PCE and then distributed to the first-hop satellite or user station, which would apply them to traffic. The delivery mechanism is outside of the scope of this document.

7. Scheduling

The most significant difference between terrestrial and satellite networks from a routing perspective is that some of the topological changes that will happen to the network can be anticipated and computed. Both link and node changes will affect the topology and the network should react smoothly and predictably.

The management plane is responsible for providing information about scheduled topological changes. The exact details of how the information is disseminated are outside of the scope of this document but could be done through a YANG model [I-D.ietf-tvr-schedule-yang]. Scheduling information needs to be accessible to all of the nodes that will make routing decisions based on the topological changes in the schedule, so data about an L1 topological change will need to be circulated to all nodes in the L1 area and information about L2 changes will need to propagate to all L2 nodes, plus the gateways and PCEs that carry the related topological information.

There is very little that the network should do in response to a topological addition. A link coming up or a node joining the topology should not have any functional change until the change is proven to be fully operational based on the usual IS-IS liveness mechanisms. Nodes may pre-compute their routing table changes but should not install them before all relevant adjacencies are received. The benefits of this pre-computation appear to be very small. Gateways and PCEs may also choose to pre-compute paths based on these changes, but should not install paths using the new parts of the topology until they are confirmed to be operational. If some path pre-installation is performed, gateways and PCEs must be prepared for the situation where the topology fails to become operational and may need to take alternate steps instead, such as reverting any related pre-installed paths.

The network may choose not to pre-install or pre-compute routes in reaction to topological additions, at a small cost of some operational efficiency.
Topological deletions are an entirely different matter. If a link or node is to be removed from the topology, the network should act before the anticipated change to route traffic around the expected topological loss. Specifically, at some point before the topology change, the affected links should be set to a high metric to direct traffic to alternate paths. This is a common operational procedure in existing networks when links are taken out of service, such as when proactive maintenance needs to be performed. This type of change does require some time to propagate through the network, so the metric change should be initiated far enough in advance that the network converges before the actual topological change. Gateways and PCEs should also update paths around the topology change and install these changes before the topology change occurs. The time necessary for both IGP and path changes will vary depending on the exact network and configuration.

Strictly speaking, changing to a high metric should not be necessary. It should be possible for each router to exclude the link and recompute paths. However, it seems safer to change the metric and use the IGP methods for indicating a topology change, as this can help avoid issues with incomplete information dissemination and synchronization.

8. Future Work

This architecture needs to be validated by satellite operators, both via simulation and operational deployment. Meaningful simulation hinges on the exact statistics of ISL connectivity, and that information is not publicly available currently.

Current available information about ISLs indicates that links are mechanically steered and will need to track the opposite end of the link continually. The angles and distances that can be practically supported are unknown, as are any limitations about the rate of change.

It is expected that intra-orbit and inter-orbit ISL links will have very different properties. Intra-orbit links should be much more stable, but still far less stable than terrestrial links. Inter-orbit links will be less stable. Links between satellites that are roughly parallel should be possible, but will likely have a limited duration. Two orbits may be roughly orthogonal, resulting in a limited potential for connectivity. Finally, in some topologies there may be parallel orbits where the satellites move in opposite directions, giving a relative speed between satellites around 34,000mph. Links in this situation may not be possible or may be so short-lived as to be impractical.
The key question to address is whether the parameters of a given network can yield a stripe assignment that produces stable, connected areas that work within the scaling bounds of the IGP. If links are very stable, a stripe could be just a few parallel orbits, with only a few hundred satellites. However, if links are unstable, a stripe might have to encompass dozens of orbits and thousands of satellites, which might be beyond the scaling limitations of a given IGP’s implementation.

9. Deployment Considerations

The network behind a gateway is expected to be a normal terrestrial network. Conventional routing architectural principles apply. An obvious approach would be to extend IS-IS to the terrestrial network, applying L1 areas as necessary for scalability.

The terrestrial network may have one or more BGP connections to the broader Internet. Prefixes for user stations should be advertised to the Internet near the associated gateway. If gateways are not interconnected by the terrestrial network, then it may be advisable to use one autonomous system per gateway as it might simplify the external perception of the network and subsequent policy considerations. Otherwise, one autonomous system may be used for the entire terrestrial network.

10. Security Considerations

This document discusses one possible routing architecture for satellite networks. It proposes no new protocols or mechanisms and thus has no new security impact. Security for IS-IS is provided by [RFC5304] and [RFC5310].

User stations will interact directly with satellites, potentially using proprietary mechanisms, and under the control of the satellite operator who is responsible for the security of the user station.

11. Acknowledgements

The author would like to thank Dino Farinacci, Olivier De jonckere, Eliot Lear, Adrian Farrel, Acee Lindem, Erik Kline, Sue Hares, Joel Halpern, and Dean Bogdanovic for their comments.

12. IANA Considerations

This document makes no requests for IANA.

13. References
13.1. Normative References

[I-D.ietf-lsr-isis-area-proxy]


13.2. Informative References

Li
Expires 4 January 2025


[RFC1195] Callon, R., "Use of OSI IS-IS for routing in TCP/IP and 
dual environments", RFC 1195, DOI 10.17487/RFC1195, 

RFC 2131, DOI 10.17487/RFC2131, March 1997, 

[RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, 
DOI 10.17487/RFC2328, April 1998, 

Border Gateway Protocol 4 (BGP-4)", RFC 4271, 
DOI 10.17487/RFC4271, January 2006, 

Computation Element (PCE)-Based Architecture", RFC 4655, 
DOI 10.17487/RFC4655, August 2006, 
<https://www.rfc-editor.org/info/rfc4655>.

[ RFC5340 ] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF 

[ RFC9552 ] Talaulikar, K., Ed., "Distribution of Link-State and 
Traffic Engineering Information Using BGP", RFC 9552, 
DOI 10.17487/RFC9552, December 2023, 

[Westphal] Westphal, C., Han, L., and R. Li, "LEO Satellite 
Networking Relaunched: Survey and Current Research 
Challenges", October 2023, 

[Zhang] Zhang, X., Yang, Y., Xu, M., and J. Luo, "ASER: Scalable 
Distributed Routing Protocol for LEO Satellite Networks", 
IEEE 46th Conference on Local Computer Networks (LCN), 
Edmonton, AB, Canada, 2021, 
DOI 10.1109/LCN52139.2021.9524989, 2021, 
<https://doi.org/10.1109/LCN52139.2021.9524989>.

Author’s Address

Tony Li
Juniper Networks
Email: tony.li@tony.li