

DETNET
Internet-Draft
Intended status: Standards Track
Expires: 28 April 2022

T. Eckert
Futurewei Technologies USA
S. Bryant
University of Surrey ICS
A.G. Malis
Malis Consulting
25 October 2021

Deterministic Networking (DetNet) Data Plane - MPLS TC Tagging for
Cyclic Queuing and Forwarding (MPLS-TC TCQF)
draft-eckert-detnet-mpls-tc-tcqf-01

Abstract

This memo defines the use of the MPLS TC field of MPLS Label Stack Entries (LSE) to support cycle tagging of packets for Multiple Buffer Cyclic Queuing and Forwarding (TCQF). TCQF is a mechanism to support bounded latency forwarding in DetNet network.

Target benefits of TCQF include low end-to-end jitter, ease of high-speed hardware implementation, optional ability to support large number of flow in large networks via DiffServ style aggregation by applying TCQF to the DetNet aggregate instead of each DetNet flow individually, and support of wide-area DetNet networks with arbitrary link latencies and latency variations.

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 28 April 2022.

Copyright Notice

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

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

Table of Contents

1. Introduction (informative)	2
2. Using TCWF in the DetNet Architecture and MPLS forwarding plane (informative)	3
3. MPLS T-CWF forwarding (normative)	6
3.1. Configuration Data model and tag processing for MPLS TC TCWF	6
3.2. Packet processing	6
3.3. TCWF with label stack operations	7
3.4. Ingress operations	8
4. TCWF Pseudocode (normative)	8
5. Operational considerations (informative)	9
5.1. Controller plane computation of cycle mappings	10
6. Security Considerations	11
7. IANA Considerations	11
8. Changelog	11
9. References	12
9.1. Normative References	12
9.2. Informative References	12
Authors' Addresses	13

1. Introduction (informative)

Cyclic Queuing and Forwarding [CWF], is an IEEE standardized queuing mechanism in support of deterministic bounded latency. See also [I-D.ietf-detnet-bounded-latency], Section 6.6.

CWF benefits for Deterministic QoS include the tightly bounded jitter it provides as well as the per-flow stateless operation, minimizing the complexity of high-speed hardware implementations and allowing to support on transit hops arbitrary number of DetNet flow in the forwarding plane because of the absence of per-hop, per-flow QoS processing. In the terms of the IETF QoS architecture, CWF can be called DiffServ QoS technology, operating only on a traffic aggregate.

CQFs is limited to only limited-scale wide-area network deployments because it cannot take the propagation latency of links into account, nor potential variations thereof. It also requires very high precision clock synchronization, which is uncommon in wide-area network equipment beyond mobile network fronthaul. See [I-D.eckert-detnet-bounded-latency-problems] for more details.

This specification introduces and utilizes an enhanced form of CQF where packets are tagged with a cycle identifier, and a limited number of cycles, e.g.: 3...7 are used to overcome these distance and clock synchronization limitations. Because this memo defines how to use the TC field of MPLS LSE as the tag to carry the cycle identifier, it calls this scheme TC Tagged multiple buffer CQF (TC TCQF). See [I-D.qiang-DetNet-large-scale-DetNet] and [I-D.dang-queuing-with-multiple-cyclic-buffers] for more details of the theory of operations of TCQF. Note that TCQF is not necessarily limited to deterministic operations but could also be used in conjunction with congestion controlled traffic, but those considerations are outside the scope of this memo.

TCQF is likely especially beneficial when MPLS networks are designed to avoid per-hop, per-flow state even for traffic steering, which is the case for networks using SR-MPLS [RFC8402] for traffic steering of MPLS unicast traffic and/or BIER-TE [I-D.ietf-bier-te-arch] for tree engineering of MPLS multicast traffic. In these networks, it is specifically undesirable to require per-flow signaling to P-LSR solely for DetNet QoS because such per-flow state is unnecessary for traffic steering and would only be required for the bounded latency QoS mechanism and require likely even more complex hardware and manageability support than what was previously required for per-hop steering state (e.g. In RSVP-TE). Note that the DetNet architecture [RFC8655] does not include full support for this DiffServ model, which is why this memo describes how to use MPLS TC TCQF with the DetNet architecture per-hop, per-flow processing as well as without it.

2. Using TCQF in the DetNet Architecture and MPLS forwarding plane (informative)

This section gives an overview of how the operations of T-CQF relates to the DetNet architecture. We first revisit QoS with DetNet in the absence of T-CQF.

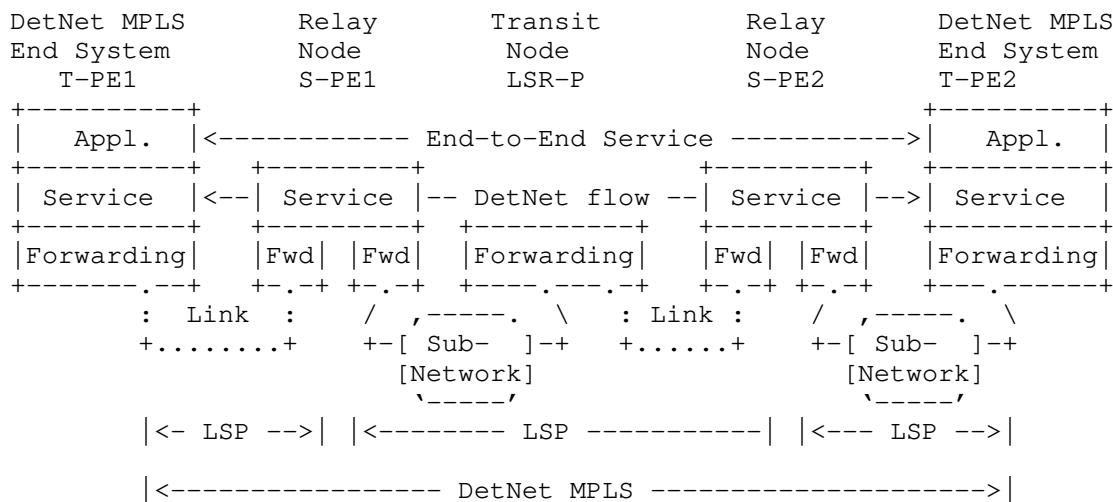


Figure 1: A DetNet MPLS Network

The above Figure 1, is copied from [RFC8964], Figure 2, and only enhanced by numbering the nodes to be able to better refer to them in the following text.

Assume a DetNet flow is sent from T-PE1 to T-PE2 across S-PE1, LSR, S-PE2. In general, bounded latency QoS processing is then required on the outgoing interface of T-PE1 towards S-PE1, and any further outgoing interface along the path. When T-PE1 and S-PE2 know that their next-hop is a service LSR, their DetNet flow label stack may simply have the DetNet flows Service Label (S-Label) as its Top of Stack (ToS) LSE, explicitly indicating one DetNet flow.

On S-PE1, the next-hop LSR is not DetNet aware, which is why S-PE1 would need to send a label stack where the S-Label is followed by a Forwarding Label (F-Label), and LSR-P would need to perform bounded latency based QoS on that F-Label.

For bounded latency QoS mechanisms relying on per-flow regulator state, such as in [TSN-ATS], this requires the use of a per-detnet flow F-Label across the network from S-PE1 to S-PE2, for example through RSVP-TE [RFC3209] enhanced as necessary with QoS parameters matching the underlying bounded latency mechanism (such as [TSN-ATS]).

With TC TCWF, a sequence of LSR and DetNet service node implements TC TCWF, ideally from T-PE1 (ingress) to T-PE2 (egress). The ingress node needs to perform per-DetNet-flow per-packet "shaping" to assign each packet of a flow to a particular TCWF cycle. This ingress-edge-function is currently out of scope of this document (TBD), but would be based on the same type of edge function as used in CWF.

All LSR/Service node after the ingress node only have to map a received TCWF tagged DetNet packet to the configured cycle on the output interface, not requiring any per-DetNet-flow QoS state. These LSR/Service nodes do therefore also not require per-flow interactions with the controller plane for the purpose of bounded latency.

Per-flow state therefore is therefore only required on nodes that are DetNet service nodes, or when explicit, per-DetNet flow steering state is desired, instead of ingress steering through e.g.: SR-MPLS.

Operating TCWF per-flow stateless across a service node, such as S-PE1, S-PE2 in the picture is only an option. It is of course equally feasible to Have one TCWF domain from T-PE1 to S-PE2, start a new TCWF domain there, running for example up to S-PE2 and start another one to T-PE2.

A service node must act as an egress/ingress edge of a TCWF domain if it needs to perform operations that do change the timing of packets other than the type of latency that can be considered in configuration of TCWF (see Section 5.1).

For example, if T-PE1 is ingress for a TCWF domain, and T-PE2 is the egress, S-PE1 could perform the DetNet Packet Replication Function (PRF) without having to be a TCWF edge node as long as it does not introduce latencies not included in the TCWF setup and the controller plane reserves resources for the multitude of flows created by the replication taking the allocation of resources in the TCWF cycles into account.

Likewise, S-PE2 could perform the Packet Elimination Function without being a TCWF edge node as this most likely does not introduce any non-TCWF acceptable latency - and the controller plane accordingly reserves only for one flow the resources on the S-PE2->T-PE2 leg.

If on the other hand, S-PE2 was to perform the Packet Reordering Function (PRF), this could create large peaks of packets when out-of-order packets are released together. A PRF would either have to take care of shaping out those bursts for the traffic of a flow to again conform to the admitted CIR/PIR, or else the service node would have to be a TCWF egress/ingress, performing that shaping itself as an ingress function.

3. MPLS T-CWF forwarding (normative)

3.1. Configuration Data model and tag processing for MPLS TC TCWF

```

tcwf
+-- uint16 cycles
+-- uint16 cycle_time
+-- uint32 cycle_clock_offset
+-- if_config[oif] # Outgoing InterFace
    +-- uint32 cycle_clock_offset
    +-- cycle_map[iif] # Incoming InterFace
        +--uint8 oif_cycle[iif_cycle]

tcwf_tc[oif]
+--uint8 tc[oif_cycle]
```

Figure 2: TCWF Configuration Data Model

3.2. Packet processing

This section explains the MPLS T-CWF packet processing and through it, introduces the semantic of the objects in Figure 2

tcwf contains the router/LSR wide configuration of TCWF parameters, independent of the specific tagging mechanism on any interface. Any interface can have a different tagging method.

The model represents a single TCWF domain, which is a set of interfaces acting both as ingress (iif) and egress (oif) interfaces, capable to forward TCWF packets amongst each other. A router/LSR may have multiple TCWF domains each with a set of interfaces disjoint from those of any other TCWF domain.

tcwf.cycles is the number of cycles used across all interfaces in the TCWF domain. router/LSR MUST support 3 and 4 cycles. To support interfaces with MPLS TC tagging, 7 or less cycles MUST be used across all interfaces in the CWF domain.

The unit of tcwf.cycle_time is micro-seconds. router/LSR MUST support configuration of cycle-times of 20,50,100,200,500,1000,2000 usec.

Cycles start at an offset of tcwf.cycle_clock_offset in units of nsec as follows. Let clock1 be a timestamp of the local reference clock for TCWF, at which cycle 1 starts, then:

```

tcwf.cycle_clock_offset = (clock1 mod (tcwf.cycle_time * tcwf.cycles)
)
```

The local reference clock of the LSR/router is expected to be synchronized with the neighboring LSR/router in TCWF domain. `tcwf.cycle_clock_offset` can be configurable by the operator, or it can be read-only. In either case will the operator be able to configure working TCWF forwarding through appropriately calculated cycle mapping.

`tcwf.if_config[oif]` is optional per-interface configuration of TCWF parameters. `tcwf.if_config[oif].cycle_clock_offset` may be different from `tcwf.cycle_clock_offset`, for example, when interfaces are on line cards with independently synchronized clocks, or when non-uniform ingress-to-egress propagation latency over a complex router/LSR fabric makes it beneficial to allow per-egress interface or line card configuration of `cycle_clock_offset`. It may be configurable or read-only.

The value of -1 for `tcwf.if_config[oif].cycle_clock_offset` is used to indicate that the domain wide `tcwf.cycle_clock_offset` is to be used for `oif`. This is the only permitted negative number for this parameter.

When a packet is received from `iif` with a cycle value of `iif_cycle` and the packet is routed towards `oif`, then the cycle value (and buffer) to use on `oif` is `tcwf.if_config[oif].cycle_map[iif].oif_cycle[iif_cycle]`. This is called the cycle mapping and is must be configurable. This cycle mapping always happens when the packet is received with a cycle tag on an interface in a TCWF domain and forwarded to another interface in the same TCWF domain.

`tcwf_tc[oif].tc[oif_cycle]` defines how to map from the internal cycle number `oif_cycle` to an MPLS TC value on interface `oif`. When `tcwf_tc[oif]` is configured, `oif` will use MPLS TC tagging for TCWF. This mapping not only used to map from internal cycle number to MPLS TC tag when sending packets, but also to map from MPLS TC tag to the internal cycle number when receiving packets.

3.3. TCWF with label stack operations

In the terminology of [RFC3270], TCWF QoS as defined here, is TC-Inferred-PSC LSP (E-LSP) behavior: Packets are determined to belong to the TCWF PSC solely based on the TC of the received packet.

The internal cycle number SHOULD be assigned from the Top of Stack (ToS) MPLS label TC bits before any other label stack operations happens. On the egress side, the TC value of the ToS MPLS label SHOULD be assigned from the internal cycle number after any label stack processing.

With this order of processing, TCWF can support forwarding of packets with any label stack operations such as label swap in the case of LDP or RSVP-TE created LSP, or no label changes from SID hop-by-hop forwarding and/or SID/label pop as in the case of SR-MPLS traffic steering.

3.4. Ingress operations

The ingress LSR of a TCWF domain has to mark packets with an internal cycle number and ensure that any such marked traffic complies with the traffic envelope admitted by the controller plane.

The algorithms to map packets of traffic flows into cycles are outside the scope of this specification, because there can be multiple ones of varying complexity. In a most simple admission model, a particular flow is allocated a maximum number of bytes in every cycle. This can easily be mapped into an appropriate policing gate.

For the purpose of this specification, such ingress operations is simply represented as an (internal/virtual) interface from which the packet is received, complete with a correctly assigned internal cycle number.

4. TCWF Pseudocode (normative)

The following pseudocode restates the forwarding behavior of Section 3 in an algorithmic fashion as pseudocode. It uses the objects of the TCWF configuration data model defined in Section 3.1.

```
void receive(pak) {
    // Receive side TCWF - remember cycle in
    // packet internal header
    iif = pak.context.iif
    if (tcwf.if_config[iif]) { // TCWF enabled on iif
        if (tcwf_tc[iif]) { // MPLS TCWF enabled on iif
            tc = pak.mpls_header.lse[ tos ].tc
            pak.context.tcwf_cycle = map_tc2cycle( tc, tcwf_tc[iif] )
        } else // other future encap/tagging options for TCWF
        {
        }
    }
    forward(pak);
}

void inject_tcwf_pak(pak, cycle) {
    pak.context.iif = INTERNAL
    pak.context.tcwf_cycle = cycle
    forward(pak);
}
```



```

void forward(pak) {
    // Forwarding including any LSE operations
    oif = pak.context.oif = forward_process(pak)

    // ... optional DetNet PREOF functions here
    // ... if router is DetNet service node

    if(pak.context.tcwf_cycle && // non TCWF packets cycle is 0
        tcwf.if_config[oif]) { // TCWF enabled
        // Map tcwf_cycle iif to oif
        cycle = pak.context.tcwf_cycle
            = map_cycle(cycle,
                tcwf.if_config[oif].cycle_map[[iif])

        if(tcwf.mpls_tc_tag[iif]) { // TC-TCWF
            pak.mpls_header.lse[tos].tc =
                map_cycle2tc(cycle, tcwf_tc[oif])
        } else // other future encap/tagging options for TCWF

            tcwf_enqueue(pak, oif.cycleq[cycle])
    }
}

// Started when TCWF is enabled on an interface
// dequeues packets from oif.cycleq
void send_tcwf(oif) {
    cycle = 1
    cc = tcwf.cycle_time *
        tcwf.cycle_time
    o = tcwf.cycle_clock_offset
    nextcyclestart = floor(tnow / cc) * cc + cc + o

    while(1) {
        while(tnow < nextcyclestart) { }
        while(pak = dequeue(oif.cycleq(cycle))) {
            send(pak)
        }
        cycle = (cycle + 1) mod tcwf.cycles + 1
        nextcyclestart += tcwf.cycle_time
    }
}

```

Figure 3: TCWF Pseudocode

5. Operational considerations (informative)

5.1. Controller plane computation of cycle mappings

The cycle mapping is computed by the controller plane by taking at minimum the link, interface serialization and node internal forwarding latencies as well as the `cycle_clock_offsets` into account.

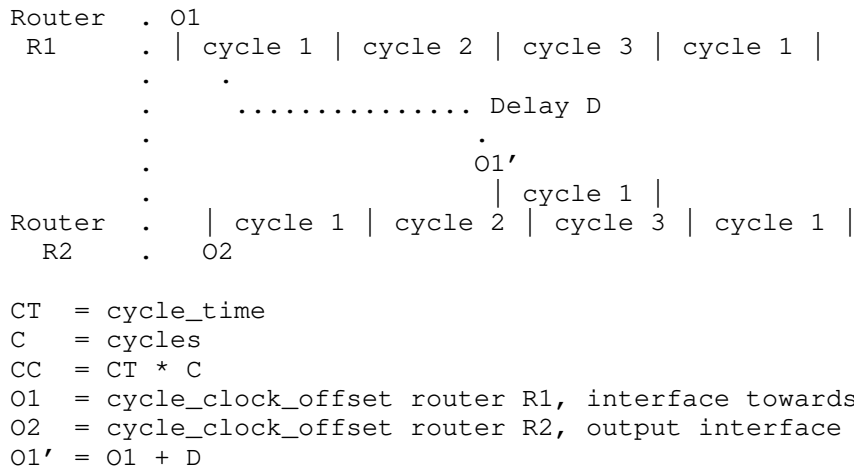


Figure 4: Calculation reference

Consider in {#Calc1} that Router R1 sends packets via $C = 3$ cycles with a `cycle_clock_offset` of $O1$ towards Router R2. These packets arrive at R2 with a `cycle_clock_offset` of $O1'$ which includes through D all latencies incurred between releasing a packet on R1 from the cycle buffer until it can be put into a cycle buffer on R2: serialization delay on R1, link delay, non_CQF delays in R1 and R2, especially forwarding in R2, potentially across an internal fabric to the output interface with the sending cycle buffers.

$$A = (\text{ceil}((O1' - O2) / CT) + C + 1) \bmod CC$$

$$\text{map}(i) = (i - 1 + A) \bmod C + 1$$

Figure 5: Calculating cycle mapping

{#Calc2} shows a formula to calculate the cycle mapping between R1 and R2, using the first available cycle on R2. In the example of {#Calc1} with $CT = 1$, $(O1' - O2) \approx 1.8$, A will be 0, resulting in $\text{map}(1)$ to be 1, $\text{map}(2)$ to be 2 and $\text{map}(3)$ to be 3.

The offset " C " for the calculation of A is included so that a negative $(O1 - O2)$ will still lead to a positive A .

In general, D will be variable [$D_{min}...D_{max}$], for example because of differences in serialization latency between min and max size packets, variable link latency because of temperature based length variations, link-layer variability (radio links) or in-router processing variability. In addition, D also needs to account for the drift between the synchronized clocks for $R1$ and $R2$. This is called the Maximum Time Interval Error (MTIE).

Let $A(d)$ be A where $O1'$ is calculated with $D = d$. To account for the variability of latency and clock synchronization, $map(i)$ has to be calculated with $A(D_{max})$, and the controller plane needs to ensure that that $A(D_{min})...A(D_{max})$ does cover at most $(C - 1)$ cycles.

If it does cover C cycles, then C and/or CT are chosen too small, and the controller plane needs to use larger numbers for either.

This $(C - 1)$ limitation is based on the understanding that there is only one buffer for each cycle, so a cycle cannot receive packets when it is sending packets. While this could be changed by using double buffers, this would create additional implementation complexity and not solve the limitation for all cases, because the number of cycles to cover [$D_{min}...D_{max}$] could also be $(C + 1)$ or larger, in which case a tag of $1...C$ would not suffice.

6. Security Considerations

TBD.

7. IANA Considerations

This document has no IANA considerations.

8. Changelog

00

Initial version

01

Added new co-author.

Changed Data Model to "Configuration Data Model",

and changed syntax from YANG tree to a non-YANG tree, removed empty section targeted for YANG model. Reason: the configuration parameters that we need to specify the forwarding behavior is only a subset of what likely would be a good YANG model, and any work to

define such a YANG model not necessary to specify the algorithm would be scope creep for this specification. Better done in a separate YANG document. Example additional YANG aspects for such a document are how to map parameters to configuration/operational space, what additional operational/monitoring parameter to support and how to map the YANG objects required into various pre-existing YANG trees.

Improved text in forwarding section, simplified sentences, used simplified configuration data model.

9. References

9.1. Normative References

- [RFC3270] Le Faucheur, F., Wu, L., Davie, B., Davari, S., Vaananen, P., Krishnan, R., Cheval, P., and J. Heinanen, "Multi-Protocol Label Switching (MPLS) Support of Differentiated Services", RFC 3270, DOI 10.17487/RFC3270, May 2002, <<https://www.rfc-editor.org/info/rfc3270>>.
- [RFC8655] Finn, N., Thubert, P., Varga, B., and J. Farkas, "Deterministic Networking Architecture", RFC 8655, DOI 10.17487/RFC8655, October 2019, <<https://www.rfc-editor.org/info/rfc8655>>.
- [RFC8964] Varga, B., Ed., Farkas, J., Berger, L., Malis, A., Bryant, S., and J. Korhonen, "Deterministic Networking (DetNet) Data Plane: MPLS", RFC 8964, DOI 10.17487/RFC8964, January 2021, <<https://www.rfc-editor.org/info/rfc8964>>.

9.2. Informative References

- [CQF] IEEE Time-Sensitive Networking (TSN) Task Group., "IEEE Std 802.1Qch-2017: IEEE Standard for Local and Metropolitan Area Networks - Bridges and Bridged Networks - Amendment 29: Cyclic Queuing and Forwarding", 2017.
- [I-D.dang-queuing-with-multiple-cyclic-buffers] Liu, B. and J. Dang, "A Queuing Mechanism with Multiple Cyclic Buffers", Work in Progress, Internet-Draft, draft-dang-queuing-with-multiple-cyclic-buffers-00, 22 February 2021, <<https://www.ietf.org/archive/id/draft-dang-queuing-with-multiple-cyclic-buffers-00.txt>>.
- [I-D.eckert-detnet-bounded-latency-problems] Eckert, T. and S. Bryant, "Problems with existing DetNet bounded latency queuing mechanisms", Work in Progress, Internet-Draft, draft-eckert-detnet-bounded-latency-

problems-00, 12 July 2021,
<<https://www.ietf.org/archive/id/draft-eckert-detnet-bounded-latency-problems-00.txt>>.

[I-D.ietf-bier-te-arch]

Eckert, T., Cauchie, G., and M. Menth, "Tree Engineering for Bit Index Explicit Replication (BIER-TE)", Work in Progress, Internet-Draft, draft-ietf-bier-te-arch-10, 9 July 2021, <<https://www.ietf.org/archive/id/draft-ietf-bier-te-arch-10.txt>>.

[I-D.ietf-detnet-bounded-latency]

Finn, N., Boudec, J. L., Mohammadpour, E., Zhang, J., Varga, B., and J. Farkas, "DetNet Bounded Latency", Work in Progress, Internet-Draft, draft-ietf-detnet-bounded-latency-07, 1 September 2021, <<https://www.ietf.org/archive/id/draft-ietf-detnet-bounded-latency-07.txt>>.

[I-D.qiang-DetNet-large-scale-DetNet]

Qiang, L., Geng, X., Liu, B., Eckert, T., Geng, L., and G. Li, "Large-Scale Deterministic IP Network", Work in Progress, Internet-Draft, draft-qiang-DetNet-large-scale-DetNet-05, 2 September 2019, <<https://www.ietf.org/archive/id/draft-qiang-DetNet-large-scale-DetNet-05.txt>>.

[RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, DOI 10.17487/RFC3209, December 2001, <<https://www.rfc-editor.org/info/rfc3209>>.

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

[TSN-ATS] Specht, J., "P802.1Qcr - Bridges and Bridged Networks Amendment: Asynchronous Traffic Shaping", IEEE , 9 July 2020, <<https://1.ieee802.org/tsn/802-1qcr/>>.

Authors' Addresses

Toerless Eckert
Futurewei Technologies USA
2220 Central Expressway
Santa Clara, CA 95050
United States of America

Email: tte@cs.fau.de

Stewart Bryant
University of Surrey ICS

Email: s.bryant@surrey.ac.uk

Andrew G. Malis
Malis Consulting

Email: agmalis@gmail.com

MPLS Working Group
Internet-Draft
Intended status: Standards Track
Expires: 18 July 2022

R. Gandhi
P. Brissette
Cisco Systems, Inc.
E. Leyton
Verizon Wireless
14 January 2022

Encapsulation of Simple TWAMP (STAMP) for Pseudowires in MPLS Networks
draft-gandhi-mpls-stamp-pw-01

Abstract

Pseudowires (PWs) are used in MPLS networks for various services including carrying layer 2 and layer 3 data packets. This document describes the procedure for encapsulation of the Simple Two-Way Active Measurement Protocol (STAMP) defined in RFC 8762 and its optional extensions defined in RFC 8972 for PWs in MPLS networks. The procedure uses PW Generic Associated Channel (G-ACH) to encapsulate the STAMP test packets with or without an IP/UDP header.

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 18 July 2022.

Copyright Notice

Copyright (c) 2022 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	2
1.1. Requirements	3
2. Conventions Used in This Document	3
2.1. Requirements Language	4
2.2. Abbreviations	4
2.3. Reference Topology	4
3. Overview	5
4. Session-Sender Test Packet	6
4.1. Session-Sender Test Packet with IP/UDP Header	6
4.2. Session-Sender Test Packet without IP/UDP Header	8
5. Session-Reflector Test Packet	9
5.1. Session-Reflector Test Packet with IP/UDP Header	9
5.2. Session-Reflector Test Packet without IP/UDP Header	11
6. Security Considerations	12
7. IANA Considerations	12
8. References	12
8.1. Normative References	12
8.2. Informative References	13
Acknowledgments	14
Authors' Addresses	14

1. Introduction

The Simple Two-way Active Measurement Protocol (STAMP) provides capabilities for the measurement of various metrics in IP networks [RFC8762] without the use of a control channel to pre-signal session parameters. [RFC8972] defines optional extensions for STAMP.

Pseudowires (PWs) are used in MPLS networks for various services including carrying layer 2 and layer 3 data packets [RFC6658]. The PWs are bidirectional in nature. The PWs can be point-to-point or point-to-multipoint. A PW Generic Associated Channel (G-ACh) [RFC5586] provides a mechanism to transport Operations, Administration, and Maintenance (OAM) and other control messages over MPLS data plane. The G-ACh channel types identify the various OAM messages being transported over the channel.

This document describes the procedure for encapsulation of the STAMP defined in [RFC8762] and its optional extensions defined in [RFC8972] for point-to-point PWs in MPLS networks. The procedure uses PW Generic Associated Channel (G-ACh) to encapsulate the STAMP test packets with or without an IP/UDP header. The procedure for point-to-multipoint PWs will be added in future.

1.1. Requirements

The STAMP test packets need to be transmitted with the same MPLS label stack that is used by the PW traffic to ensure proper validation of underlay path taken by the actual PW traffic. Also, the test packets need to follow the same ECMP path taken by the PW traffic. The STAMP test packets may be encapsulated over the PW associated channel with or without an IP/UDP header.

In case of MPLS Transport Profile (MPLS TP), the STAMP test packets need to be transmitted on the Generic Associated Channel without using an IP header to have the same forwarding behavior as the data traffic.

The requirements for the encapsulation of the STAMP test packets for the PWs in MPLS networks can be summarized as follows:

- o The PW associated channel MUST support STAMP test packets with IP/UDP header.
- o The PW associated channel MUST support STAMP test packets without IP/UDP header.
- o The Session-Sender test packets MUST follow the same underlay path taken by the traffic for the associated PW channel.
- o The Session-Sender test packets MUST follow the same ECMP underlay path taken by the traffic for the associated PW channel.
- o The Session-Reflector test packets MAY follow the same reverse underlay path taken by Session-Sender test packets.
- o The Session-Reflector test packets MAY follow the same reverse ECMP underlay path taken by Session-Sender test packets.

2. Conventions Used in This Document

2.1. Requirements Language

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

2.2. Abbreviations

ECMP: Equal Cost Multi-Path.

G-ACh: Generic Associated Channel.

GAL: G-ACh Label.

HMAC: Hashed Message Authentication Code.

MPLS: Multiprotocol Label Switching.

OAM: Operations, Administration, and Maintenance.

PLE: Private Line Emulation.

PW: Pseudowires.

SHA: Secure Hash Algorithm.

STAMP: Simple Two-way Active Measurement Protocol.

TC: Traffic Class.

2.3. Reference Topology

In the Reference Topology shown in Figure 1, there exists a packet pseudowire to transport data between LSRs S1 and R1. The STAMP Session-Sender on LSR S1 initiates a Session-Sender test packet and the STAMP Session-Reflector on LSR R1 transmits a reply test packet. The reply test packet is transmitted to the STAMP Session-Sender on the same path (same set of links and nodes) in the reverse direction of the path taken towards the Session-Reflector.

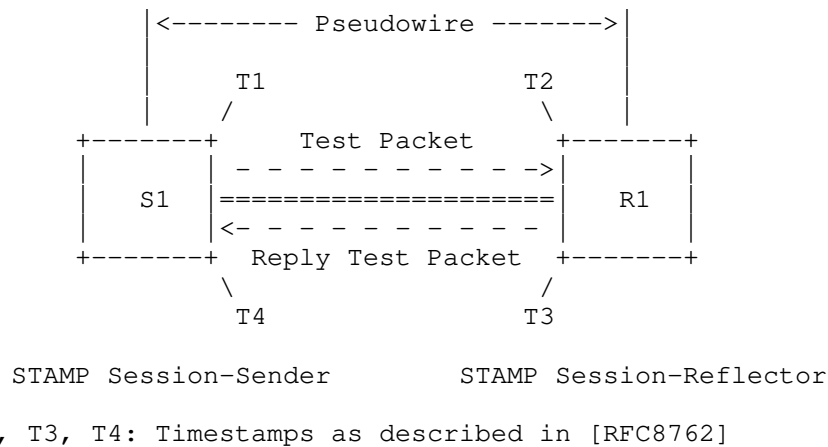


Figure 1: Reference Topology

3. Overview

The STAMP Session-Sender and Session-Reflector test packets defined in [RFC8972] are transmitted over the PWs in MPLS networks. The base STAMP test packets can be encapsulated using IP/UDP header and may use Destination UDP port 862 [RFC8762].

The STAMP test packets are encapsulated with MPLS header using the same label stack as the PW traffic and the PW G-ACh header. The encapsulation allows the STAMP test packets to follow the same path as the PW traffic, and provide the same ECMP path selection on the intermediate nodes.

There are two ways in which STAMP test packets may be encapsulated over a PW associated channel, either using an IP/UDP header or without using an IP/UDP header.

For encapsulating the STAMP test packets over a PW associated channel with an IP/UDP header, IPv4 and IPv6 G-ACh types [RFC4385] are used for both Session-Sender and Session-Reflector test packets. The destination UDP port numbers in the Session-Sender and Session-Reflector test packets discriminate the test packets. The IP version (IPv4 or IPv6) MUST match the IP version used for signaling for dynamically established PWs or MUST be configured for statically provisioned PWs.

For encapsulating the STAMP test packets over a PW associated channel without an IP/UDP header, two new G-ACh types are defined in this document, one for the Session-Sender test packets and one for the Session-Reflector test packets. The different G-ACh types are required for the Session-Sender and Session-Reflector test packets as the STAMP test packet formats do not have a way to discriminate them.

The Time to Live (TTL)/Hop Limit (HL) and Generalized TTL Security Mechanism (GTSM) procedures from [RFC5082] apply to this encapsulation, and hence the TTL/HL is set to 255.

The G-ACh label (GAL) [RFC5586] is not added in the MPLS label stack.

4. Session-Sender Test Packet

4.1. Session-Sender Test Packet with IP/UDP Header

The content of an example STAMP Session-Sender test packet encapsulated over a PW associated channel using an IP/UDP header is shown in Figure 2. The STAMP G-ACh header [RFC5586] with G-ACh MUST immediately follow the bottom of the MPLS label stack. The payload contains the STAMP Session-Sender test packet defined in [RFC8972].

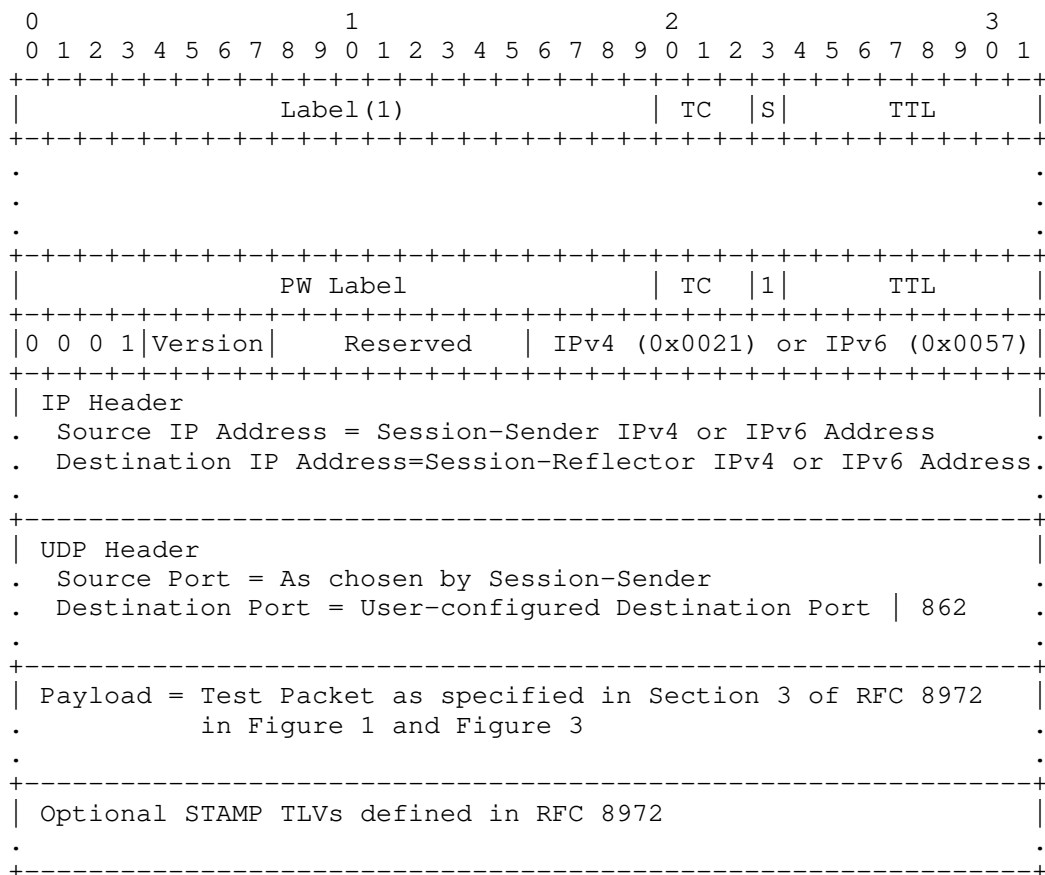


Figure 2: Example Session-Sender Test Packet with IP/UDP Header

The STAMP Session-Sender test packet G-ACh header contains following fields:

Version: The Version field is set to 0, as defined in [RFC4385].

Reserved: Reserved Bits MUST be set to zero upon transmission and ignored upon receipt.

Channel Type: G-ACh channel type for IPv4 header (0x0021) or IPv6 header (0x0057) [RFC4385].

4.2. Session-Sender Test Packet without IP/UDP Header

The content of an example STAMP Session-Sender test packet encapsulated over a PW associated channel without using an IP/UDP header is shown in Figure 3. The STAMP G-ACh header [RFC5586] with new STAMP Session-Sender G-ACh type (value TBD1) MUST immediately follow the bottom of the MPLS label stack. The payload contains the STAMP Session-Sender test packet defined in [RFC8972].

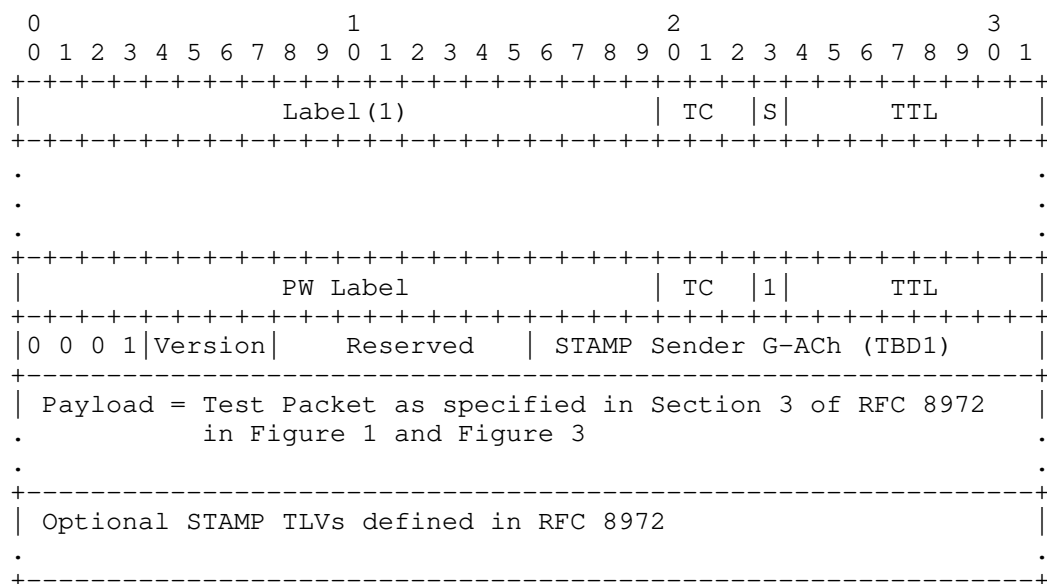


Figure 3: Example Session-Sender Test Packet without IP/UDP Header

The STAMP Session-Sender test packet G-ACh header contains following fields:

Version: The Version field is set to 0, as defined in [RFC4385].

Reserved: Reserved Bits MUST be set to zero upon transmission and ignored upon receipt.

Channel Type: G-ACh channel type for STAMP Session-Sender packet (TBD1).

5. Session-Reflector Test Packet

The STAMP Session-Reflector reply test packet is sent on the same path in the reverse direction of a bidirectional PW. The STAMP test packet can be sent using an MPLS header with or without IP/UDP header. The Session-Reflector test packet is sent with an IP/UDP header if the Session-Sender test packet is received with an IP/UDP header, otherwise, it is sent without an IP/UDP header.

5.1. Session-Reflector Test Packet with IP/UDP Header

The content of an example STAMP Session-Reflector test packet encapsulated over a PW associated channel using an IP/UDP header is shown in Figure 4. The STAMP G-ACh header [RFC5586] with G-ACh MUST immediately follow the bottom of the MPLS label stack. The payload contains the STAMP Session-Reflector test packet defined in [RFC8972].

The STAMP Session-Reflector reply test packet MUST use the IP/UDP information from the received test packet when an IP/UDP header is present in the received test packet.

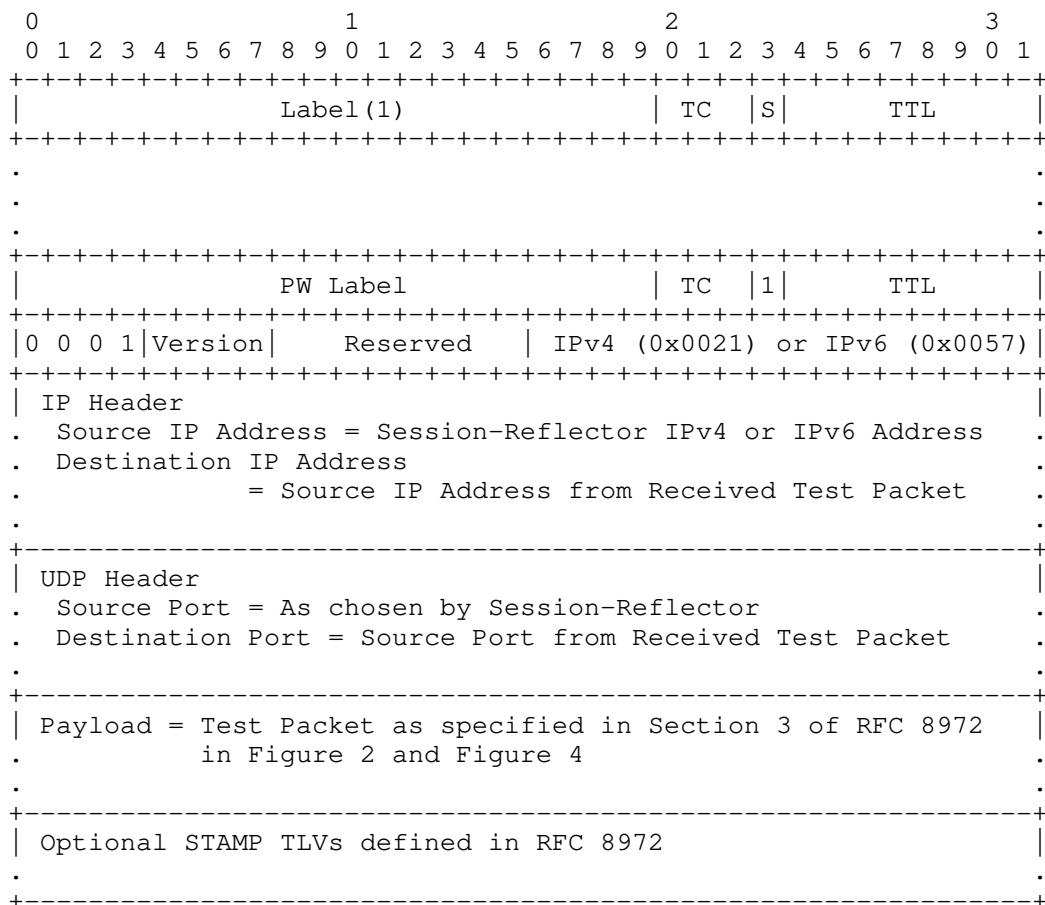


Figure 4: Example Session-Reflector Test Packet with IP/UDP Header

The STAMP Session-Reflector test packet G-ACh header contains following fields:

Version: The Version field is set to 0, as defined in [RFC4385].

Reserved: Reserved Bits MUST be set to zero upon transmission and ignored upon receipt.

Channel Type: G-ACh channel type for IPv4 header (0x0021) or IPv6 header (0x0057) [RFC4385].

5.2. Session-Reflector Test Packet without IP/UDP Header

The content of an example STAMP Session-Reflector test packet encapsulated over a PW associated channel without using an IP/UDP header is shown in Figure 5. The STAMP G-ACh header [RFC5586] with new STAMP Session-Reflector G-ACh type (value TBD2) MUST immediately follow the bottom of the MPLS label stack. The payload contains the STAMP Session-Reflector test packet defined in [RFC8972].

The STAMP Session-Reflector reflects the test packet back to the Session-Sender using the same channel of the reverse direction of the PW on which it was received. The Session-Reflector has enough information to reflect the test packet received by it to the Session-Sender using the PW context.

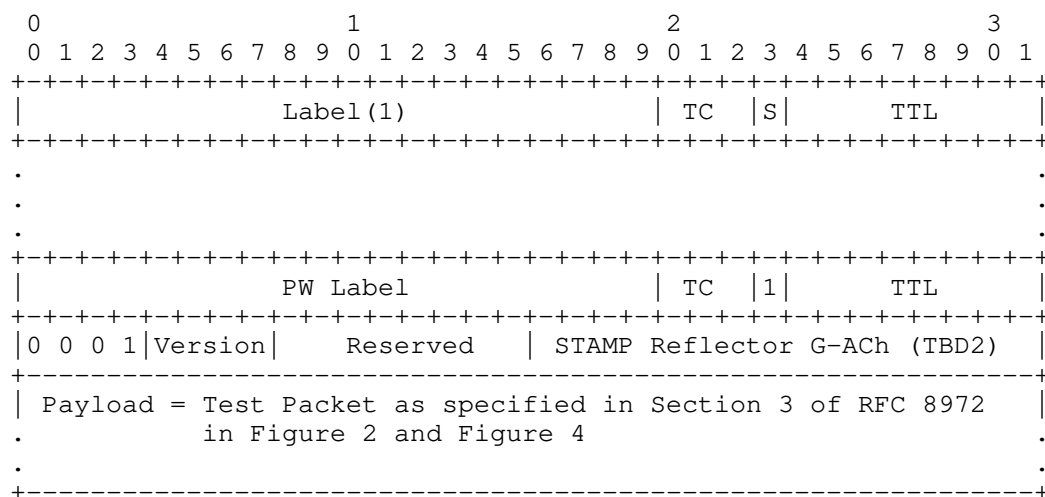


Figure 5: Example Session-Reflector Test Packet without IP/UDP Header

The STAMP Session-Reflector test packet G-ACh header contains following fields:

Version: The Version field is set to 0, as defined in [RFC4385].

Reserved: Reserved Bits MUST be set to zero upon transmission and ignored upon receipt.

Channel Type: G-ACh channel type for STAMP Session-Reflector packet (TBD2).

6. Security Considerations

The usage of STAMP protocol is intended for deployment in limited domains [RFC8799]. As such, it assumes that a node involved in STAMP protocol operation has previously verified the integrity of the path and the identity of the far-end STAMP Session-Reflector.

If desired, attacks can be mitigated by performing basic validation and sanity checks, at the STAMP Session-Sender, of the counter or timestamp fields in received reply test packets. The minimal state associated with these protocols also limits the extent of disruption that can be caused by a corrupt or invalid packet to a single test cycle.

Use of HMAC-SHA-256 in the authenticated mode protects the data integrity of the test packets. Cryptographic measures may be enhanced by the correct configuration of access-control lists and firewalls.

The security considerations specified in [RFC8762] and [RFC8972] also apply to the procedure described in this document. Specifically, the message integrity protection using HMAC, as defined in [RFC8762] Section 4.4, also apply to the procedure described in this document.

Routers that support G-ACh are subject to the same security considerations as defined in [RFC4385] and [RFC5586].

7. IANA Considerations

IANA maintains G-ACh Type Registry (see <https://www.iana.org/assignments/g-ach-parameters/g-ach-parameters.xhtml>). IANA is requested to allocate values for the STAMP G-ACh Types from "MPLS Generalized Associated Channel (G-ACh) Types (including Pseudowire Associated Channel Types)" registry.

Value	Description	Reference
TBD1	STAMP Session-Sender G-ACh Type	This document
TBD2	STAMP Session-Reflector G-ACh Type	This document

Table 1: STAMP G-ACh Type

8. References

8.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC4385] Bryant, S., Swallow, G., Martini, L., and D. McPherson, "Pseudowire Emulation Edge-to-Edge (PWE3) Control Word for Use over an MPLS PSN", RFC 4385, DOI 10.17487/RFC4385, February 2006, <<https://www.rfc-editor.org/info/rfc4385>>.
- [RFC5586] Bocci, M., Ed., Vigoureux, M., Ed., and S. Bryant, Ed., "MPLS Generic Associated Channel", RFC 5586, DOI 10.17487/RFC5586, June 2009, <<https://www.rfc-editor.org/info/rfc5586>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8762] Mirsky, G., Jun, G., Nydell, H., and R. Foote, "Simple Two-Way Active Measurement Protocol", RFC 8762, DOI 10.17487/RFC8762, March 2020, <<https://www.rfc-editor.org/info/rfc8762>>.
- [RFC8972] Mirsky, G., Min, X., Nydell, H., Foote, R., Masputra, A., and E. Ruffini, "Simple Two-Way Active Measurement Protocol Optional Extensions", RFC 8972, DOI 10.17487/RFC8972, January 2021, <<https://www.rfc-editor.org/info/rfc8972>>.

8.2. Informative References

- [RFC5082] Gill, V., Heasley, J., Meyer, D., Savola, P., Ed., and C. Pignataro, "The Generalized TTL Security Mechanism (GTSM)", RFC 5082, DOI 10.17487/RFC5082, October 2007, <<https://www.rfc-editor.org/info/rfc5082>>.
- [RFC6658] Bryant, S., Ed., Martini, L., Swallow, G., and A. Malis, "Packet Pseudowire Encapsulation over an MPLS PSN", RFC 6658, DOI 10.17487/RFC6658, July 2012, <<https://www.rfc-editor.org/info/rfc6658>>.
- [RFC8799] Carpenter, B. and B. Liu, "Limited Domains and Internet Protocols", RFC 8799, DOI 10.17487/RFC8799, July 2020, <<https://www.rfc-editor.org/info/rfc8799>>.

[I-D.schmutzer-bess-ple]

Gringeri, S., Whittaker, J., Leymann, N., Schmutzer, C.,
Chiesa, L. D., Nainar, N. K., Pignataro, C., Smallegange,
G., Brown, C., and F. Dada, "Private Line Emulation over
Packet Switched Networks", Work in Progress, Internet-
Draft, draft-schmutzer-bess-ple-03, 17 August 2021,
<<https://www.ietf.org/archive/id/draft-schmutzer-bess-ple-03.txt>>.

Acknowledgments

TBA.

Authors' Addresses

Rakesh Gandhi
Cisco Systems, Inc.
Canada

Email: rgandhi@cisco.com

Patrice Brissette
Cisco Systems, Inc.
Canada

Email: pbrisset@cisco.com

Edward Leyton
Verizon Wireless

Email: edward.leyton@verizonwireless.com

Internet
Internet-Draft
Intended status: Standards Track
Expires: 6 August 2022

Y. Qu
Futurewei
A. Lindem
S. Litkowski
Cisco Systems
J. Tantsura
Juniper
2 February 2022

A YANG Model for MPLS MSD
draft-qu-mpls-mpls-msd-yang-03

Abstract

This document defines a YANG data module augmenting the IETF MPLS YANG model to provide support for MPLS Maximum SID Depths (MSDs) as defined in RFC 8476 and RFC 8491.

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 6 August 2022.

Copyright Notice

Copyright (c) 2022 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. Overview	2
1.1. Requirements Language	2
2. YANG Module for MPLS MSD	3
3. Security Considerations	6
4. IANA Considerations	7
5. Acknowledgements	7
6. References	7
6.1. Normative References	7
6.2. Informative References	9
Authors' Addresses	9

1. Overview

YANG [RFC7950] is a data definition language used to define the contents of a conceptual data store that allows networked devices to be managed using NETCONF [RFC6241] or RESTCONF [RFC8040].

This document defines a YANG data model augmenting the IETF MPLS YANG model [RFC8960], which itself augments [RFC8349], to provide operational state for various MSDs[RFC8662].

The augmentation defined in this document requires support for the MPLS base model[RFC8960] which defines basic MPLS configuration and state.

The YANG module in this document conforms to the Network Management Datastore Architecture (NMDA) [RFC8342].

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. YANG Module for MPLS MSD

This document defines a YANG module for MSD extensions [RFC8476][RFC8491] to MPLS base model as defined in [RFC8960].

```
module: ietf-mpls-msd
  augment /rt:routing/mpls:mpls:
    +--ro node-msd
      +--ro node-msds* []
        +--ro msd-type?    identityref
        +--ro msd-value?   uint8
  augment /rt:routing/mpls:mpls/mpls:interfaces/mpls:interface:
    +--ro link-msd
      +--ro link-msds* []
        +--ro msd-type?    identityref
        +--ro msd-value?   uint8
```

```
<CODE BEGINS> file "ietf-mpls-msd@2021-08-02.yang"
module ietf-mpls-msd {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-mpls-msd";
  prefix mpls-msd;

  import ietf-routing {
    prefix rt;
    reference
      "RFC 8349: A YANG Data Model for Routing
        Management (NMDA Version)";
  }

  import ietf-mpls {
    prefix mpls;
    reference "RFC 8960: A YANG Data Model for MPLS Base";
  }

  organization
    "IETF MPLS - MPLS Working Group";
  contact
    "WG Web:  <https://tools.ietf.org/wg/mpls/>
    WG List:  <mailto:mpls@ietf.org>

    Author:   Yingzhen Qu
              <mailto:yingzhen.qu@futurewei.com>
    Author:   Acee Lindem
              <mailto:acee@cisco.com>
    Author:   Stephane Litkowski
```

Author: <mailto:slitkows.ietf@gmail.com>
Jeff Tantsura
<mailto:jefftant.ietf@gmail.com>

```
";
description
  "The YANG module augments the base MPLS model, and it is to
  manage different types of Maximum SID Depth (MSD).

  This YANG model conforms to the Network Management
  Datastore Architecture (NMDA) as described in RFC 8342.

  Copyright (c) 2021 IETF Trust and the persons identified as
  authors of the code. All rights reserved.

  Redistribution and use in source and binary forms, with or
  without modification, is permitted pursuant to, and subject
  to the license terms contained in, the Simplified BSD License
  set forth in Section 4.c of the IETF Trust's Legal Provisions
  Relating to IETF Documents
  (https://trustee.ietf.org/license-info).

  This version of this YANG module is part of RFC XXXX;
  see the RFC itself for full legal notices.

  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 (RFC 2119) (RFC 8174) when, and only when,
  they appear in all capitals, as shown here.";
```

```
reference "RFC XXXX: YANG Data Model for Segment Routing.";

revision 2021-08-02 {
  description
    "Initial Version";
  reference "RFC XXXX: YANG Data Model for Segment Routing.";
}

identity msd-base-type {
  description
    "Base identity for Maximum SID Depth (MSD) Type";
}

identity msd-mpls {
  base msd-base-type;
  description
```



```
    "Base MPLS Imposition MSD.";
  reference
    "RFC 8491: Singling Maximum SID Depth (MSD) using IS-IS.";
}

identity msd-erld {
  base msd-base-type;
  description
    "msd-erld is defined to advertise the Entropy Readable
     Label Depth (ERLD).";
  reference
    "RFC 8662: Entropy Label for Source Packet Routing in
     Networking (SPRING) Tunnels";
}

augment "/rt:routing/mpls:mpls" {
  description
    "This module augments MPLS data model (RFC 8960)
     with node MSD.";
  container node-msd {
    config false;
    description
      "Maximum SID Depth (MSD) operational state.";
    list node-msds {
      leaf msd-type {
        type identityref {
          base msd-base-type;
        }
        description
          "MSD types";
      }
      leaf msd-value {
        type uint8;
        description
          "MSD value, in the range of 0-255.";
      }
    }
    description
      "Node MSD is the smallest link MSD supported by
       the node.";
  }
}

augment "/rt:routing/mpls:mpls/mpls:interfaces/mpls:interface" {
  description
    "This module augments MPLS data model (RFC 8960)
     with link MSD.";
  container link-msd {
```

```
    config false;
    description
      "Maximum SID Depth (MSD) interface operational state.";
    list link-msds {
      leaf msd-type {
        type identityref {
          base msd-base-type;
        }
        description
          "MSD type";
      }
      leaf msd-value {
        type uint8;
        description
          "MSD value, in the range of 0-255.";
      }
      description
        "List of link MSDs";
    }
  }
}
}
<CODE ENDS>
```

3. Security Considerations

The YANG modules specified in this document define a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC8446].

The NETCONF Access Control Model (NACM) [RFC8341] provides the means to restrict access for particular NETCONF or RESTCONF users to a pre-configured subset of all available NETCONF or RESTCONF protocol operations and content.

There are a number of data nodes defined in the modules that are writable/creatable/deletable (i.e., config true, which is the default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g., edit-config) to these data nodes without proper protection can have a negative effect on network operations.

Some of the readable data nodes in the modules may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get, get-config, or notification) to these data nodes. These are the subtrees and data nodes and their sensitivity/vulnerability:

```
/rt:routing/mpls:mpls/msd/node-msds
```

```
/rt:routing/mpls:mpls/msd/link-msds
```

Exposure of the node's maximum SID depth may be useful in mounting a Denial-of-Service (DoS) attack by sending packets to the node that the router can't process.

4. IANA Considerations

This document registers URIs in the IETF XML registry [RFC3688]. Following the format in [RFC3688], the following registrations is requested to be made:

```
URI: urn:ietf:params:xml:ns:yang:ietf-mpls-msd
Registrant Contact: The IESG.
XML: N/A, the requested URI is an XML namespace.
```

This document registers the YANG modules in the YANG Module Names registry [RFC6020].

```
name: ietf-mpls-msd
namespace: urn:ietf:params:xml:ns:yang:ietf-mpls-msd
prefix: mpls-msd
reference: RFC XXXX
```

5. Acknowledgements

This document was produced using Marshall Rose's xml2rfc tool.

The YANG model was developed using the suite of YANG tools written and maintained by numerous authors.

6. References

6.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <<https://www.rfc-editor.org/info/rfc3688>>.
- [RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <<https://www.rfc-editor.org/info/rfc6020>>.
- [RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <<https://www.rfc-editor.org/info/rfc6241>>.
- [RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <<https://www.rfc-editor.org/info/rfc6242>>.
- [RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <<https://www.rfc-editor.org/info/rfc7950>>.
- [RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <<https://www.rfc-editor.org/info/rfc8040>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", STD 91, RFC 8341, DOI 10.17487/RFC8341, March 2018, <<https://www.rfc-editor.org/info/rfc8341>>.
- [RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture (NMDA)", RFC 8342, DOI 10.17487/RFC8342, March 2018, <<https://www.rfc-editor.org/info/rfc8342>>.

- [RFC8349] Lhotka, L., Lindem, A., and Y. Qu, "A YANG Data Model for Routing Management (NMDA Version)", RFC 8349, DOI 10.17487/RFC8349, March 2018, <<https://www.rfc-editor.org/info/rfc8349>>.
- [RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <<https://www.rfc-editor.org/info/rfc8446>>.
- [RFC8476] Tantsura, J., Chunduri, U., Aldrin, S., and P. Psenak, "Signaling Maximum SID Depth (MSD) Using OSPF", RFC 8476, DOI 10.17487/RFC8476, December 2018, <<https://www.rfc-editor.org/info/rfc8476>>.
- [RFC8491] Tantsura, J., Chunduri, U., Aldrin, S., and L. Ginsberg, "Signaling Maximum SID Depth (MSD) Using IS-IS", RFC 8491, DOI 10.17487/RFC8491, November 2018, <<https://www.rfc-editor.org/info/rfc8491>>.
- [RFC8960] Saad, T., Raza, K., Gandhi, R., Liu, X., and V. Beeram, "A YANG Data Model for MPLS Base", RFC 8960, DOI 10.17487/RFC8960, December 2020, <<https://www.rfc-editor.org/info/rfc8960>>.

6.2. Informative References

- [RFC8662] Kini, S., Kompella, K., Sivabalan, S., Litkowski, S., Shakir, R., and J. Tantsura, "Entropy Label for Source Packet Routing in Networking (SPRING) Tunnels", RFC 8662, DOI 10.17487/RFC8662, December 2019, <<https://www.rfc-editor.org/info/rfc8662>>.

Authors' Addresses

Yingzhen Qu
Futurewei
2330 Central Expressway
Santa Clara, CA 95050
United States of America

Email: yingzhen.qu@futurewei.com

Acee Lindem
Cisco Systems
301 Midenhall Way
Cary, NC 27513

Email: acee@cisco.com

Stephane Litkowski
Cisco Systems

Email: slitkows.ietf@gmail.com

Jeff Tantsura
Juniper

Email: jefftant.ietf@gmail.com