Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: May 7, 2020

X. Geng
M. Chen
Huawei Technologies
Y. Ryoo
ETRI
Z. Li
China Mobile
R. Rahman
Cisco Systems

November 04, 2019

Deterministic Networking (DetNet) Configuration YANG Model
draft-ietf-detnet-yang-04

Abstract

This document contains the specification for Deterministic Networking flow configuration YANG Model. The model allows for provisioning of end-to-end DetNet service along the path without dependency on any signaling protocol.

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

Requirements Language

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

Status of This Memo

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

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

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

This Internet-Draft will expire on May 7, 2020.

Table of Contents

1. Introduction

DetNet (Deterministic Networking) provides a capability to carry specified unicast or multicast data flows for real-time applications with extremely low packet loss rates and assured maximum end-to-end delivery latency. A description of the general background and concepts of DetNet can be found in [I-D.ietf-detnet-architecture].

This document defines a YANG model for DetNet based on YANG data types and modeling language defined in [RFC6991] and [RFC7950], which includes DetNet service module and DetNet configuration module, and YANG model for topology discovery is defined in [I-D.ietf-detnet-topology-yang]. DetNet service module is designed for describe characteristics of services being provided for application flows over a network, while the DetNet configuration module is designed for DetNet flow path establishment, flow status reporting, and DetNet functions configuration in order to achieve end-to-end bounded latency and zero congestion loss.

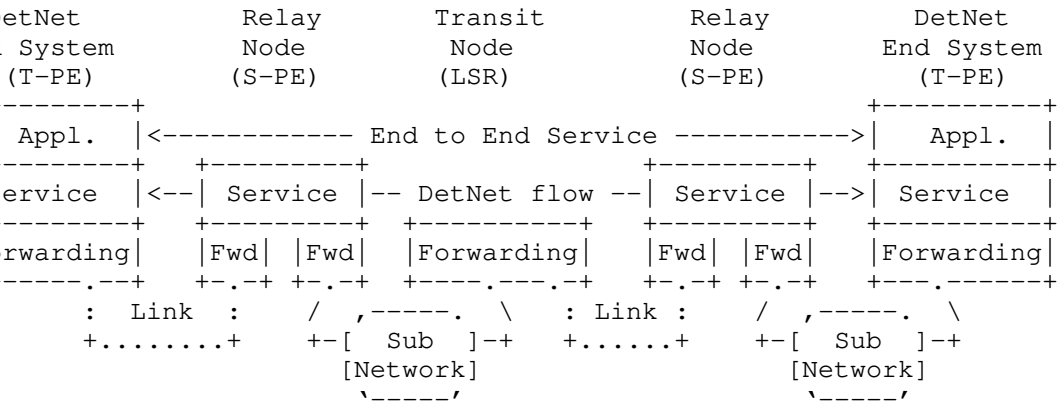


Figure 1: An End-to-end DetNet-Enabled Network

As showed in the picture, in an end-to-end DetNet-enabled network, application flow is carried over a DetNet service and the DetNet service is instantiated as different configuration parameters in different network device along the path. DetNet service is an abstract concept for service provider, and DetNet configuration needs device specific attributes. YANG Models for DetNet service and DetNet configuration are defined in detail respectively in section 3 and section 4.

Editor's notes:

Detnet YANG model and DetNet information model are supposed to keep the same structure and describes the same attributes by different methods. But the design of these two models are still under discussion. The divergence will be settled in the following versions.

2. Terminologies

This documents uses the terminologies defined in [I-D.ietf-detnet-architecture].

3. DetNet Service Module

DetNet Service Module includes service quality attributes, service endpoints attributes and service encapsulation type attributes, which are defined in Section 3.1, 3.2, 3.3 respectively.

3.1. Service Quality

DetNet service quality includes the following attributes:

- Maximum Latency: MaxLatency is the maximum latency from Ingress to Egress(es) for a single packet of the DetNet flow. MaxLatency is specified as an integer number of nanoseconds.

- Maximum Latency Variation: MaxLatencyVariation is the difference between the minimum and the maximum end-to-end one-way latency. MaxLatencyVariation is specified as an integer number of nanoseconds.

- Maximum Loss: MaxLoss defines the maximum Packet Loss Ratio (PLR) parameter for the DetNet service between the Ingress and Egress(es) of the DetNet domain.

- Maximum Consecutive Loss: Maximum Consecutive Loss defines the consecutive packet loss number.

- Maximum Misordering: MaxMisordering describes the maximum number of packets that can be received out of order.

3.2. Service Endpoints

Endpoints attribute defines the starting and termination reference points of the DetNet flow by pointing to the ingress interface/node and egress interface(s)/node(s).

3.3. Service Encapsulation

Service Encapsulation attribute defines the data plane type of the DetNet service in a DetNet domain, e.g., MPLS, IP.

Editor's notes: DetNet service module is just defined in the document and the yang model is still under work.

4. DetNet Configuration Module

DetNet configuration module includes DetNet App-flow configuration, DetNet Service Sub-layer configuration, and DetNet Forwarding Sub-layer configuration and DetNet sub-network. The corresponding attributes used in different sub-layers are defined in Section 3.1, 3.2, 3.3, 3.4 respectively.

4.1. DetNet Appliction Flow Configuration Attributes

DetNet application flow is responsible for mapping between application flows and DetNet flows at the edge node(egress/ingress node). Where the application flows can be either layer 2 or layer 3 flows. To identify a flow at the User Network Interface (UNI), as defined in [I-D.ietf-detnet-flow-information-model], the following flow attributes are introduced:

- DetNet L3 Flow Identification, refers to Section 7.1.1 of [I-D.ietf-detnet-flow-information-model]

- DetNet L2 Flow Identification, refers to Section 7.1.2 of [I-D.ietf-detnet-flow-information-model]

Application flow can also do flow filtering and policing at the ingress to prevent the misbehaviored flows from going into the network, which needs:

- Traffic Specification, refers to Section 7.2 of [I-D.ietf-detnet-flow-information-model]

4.2. DetNet Service Sub-layer Configuration Attributes

DetNet service functions, e.g., DetNet tunnel initialization/ termination and service protection, are provided in DetNet service sub-layer. To support these functions, the following service attributes need to be configured:

- DetNet flow identification, refers to Section 8.1.3 of [I-D.ietf-detnet-flow-information-model].

- Service function indication, indicates which service function will be invoked at a DetNet edge, relay node or end station. (DetNet tunnel initialization or termination are default functions in DetNet service layer, so there is no need for explicit indication.)

- Flow Rank, refers to Section 8.3 of [I-D.ietf-detnet-flow-information-model].

- Service Rank, refers to Section 16 of [I-D.ietf-detnet-flow-information-model].

- Service Sub-layer, refers to Section 4.5 and Section 4.6 of [I-D.ietf-detnet-mpls]

- Forwarding Sub-layer, refers to Section 4.3 of [I-D.ietf-detnet-ip] and Section 4.5 and Section 4.6 of [I-D.ietf-detnet-mpls]

4.3. DetNet Forwarding Sub-layer Configuration Attributes

As defined in [I-D.ietf-detnet-architecture], DetNet forwarding sub-layer optionally provides congestion protection for DetNet flows over paths provided by the underlying network. Explicit route is another mechanism that is used by DetNet to avoid temporary interruptions caused by the convergence of routing or bridging protocols, and it is also implemented at the DetNet forwarding sub-layer.

To support congestion protection and explicit route, the following transport layer related attributes are necessary:

- Traffic Specification, refers to Section 7.2 of [I-D.ietf-detnet-flow-information-model]. It may used for bandwidth reservation, flow shaping, filtering and policing.

- Explicit path, existing explicit route mechanisms can be reused. For example, if Segment Routing (SR) tunnel is used as the transport tunnel, the configuration is mainly at the ingress node of the transport layer; if the static MPLS tunnel is used as the transport tunnel, the configurations need to be at every transit node along the path; for pure IP based transport tunnel, it's similar to the static MPLS case.

4.4. DetNet Sub-network Configurations Attributes

TBD

5. Overview of DetNet YANG

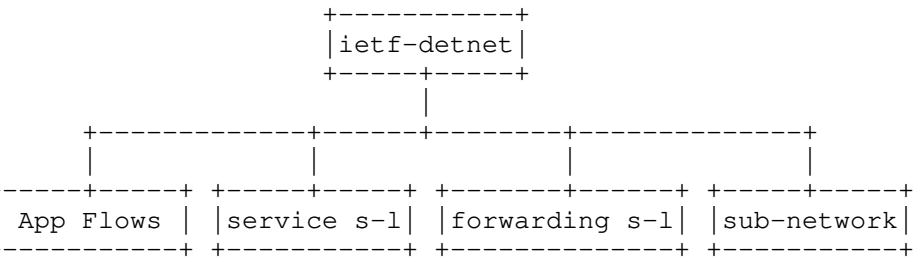
5.1. DetNet YANG Considerations

5.1.1. DetNet Service YANG Considerations

TBD

5.1.2. DetNet Configuration YANG Considerations

The picture shows that the general structure of the DetNet YANG Model:



There are four instances in DetNet YANG Model: App-flow instance, service sub-layer instance, forwarding sub-layer instance and sub-network instance, respectively corresponding to four parts of DetNet functions defined in section 3. In each instance, there are four elements: name, in-segments, out-segments and operations, which means:

- Name: indicates the key value of the instance identification.
- In-segments: indicates the key value of identification, e.g., Layer 2 App flow identification, Layer 3 App flow identification and DetNet flow identification.
- Out-segments: indicates the information of DetNet processing(e.g., DetNet forwarding, DetNet header Encapsulation) and the mapping relationship to the lower sub-layer/sub-network.
- Operations: indicates DetNet functions, e.g., DetNet forwarding functions, DetNet Service functions, DetNet Resource Reservation.

These elements are different when the technologies used for the specific instance is different. So this structure is abstract, which allows for different technology specifics as defined in different data plane drafts.

5.2. DetNet YANG Structures

5.2.1. DetNet Service YANG Structure

TBD

5.2.2. DetNet Configuration YANG Structure

```
         +--rw app-flow
         |  +--rw operations
         |  |  +--rw sequence-number
         |  |     +--rw sequence-number-generation-type?   sequence-number-gener
ation-type
         |  |     +--rw sequence-number-length?            uint8
         |  +--rw in-segments
         |  |  +--rw app-flow-type?              flow-type-ref
         |  |  +--rw source-mac-address?         yang:mac-address
         |  |  +--rw destination-mac-address?    yang:mac-address
         |  |  +--rw ethertype?                  eth:ethertype
         |  |  +--rw vlan-id?                    uint16
         |  |  +--rw pcp?                        uint8
         |  |  +--rw src-ipv4-prefix             inet:ipv4-prefix
         |  |  +--rw dest-ipv4-prefix            inet:ipv4-prefix
         |  |  +--rw protocol                    uint8
         |  |  +--rw dscp?                       uint8
         |  |  +--rw dscp-bitmask?               uint8
         |  |  +--rw src-ipv6-prefix             inet:ipv6-prefix
         |  |  +--rw dest-ipv6-prefix            inet:ipv6-prefix
         |  |  +--rw next-header                 uint8
         |  |  +--rw traffic-class?              uint8
         |  |  +--rw traffic-class-bitmask?      uint8
         |  |  +--rw flow-label?                 inet:ipv6-flow-label
         |  |  +--rw flow-label-flag?            boolean
         |  |  +--rw lower-source-port?          inet:port-number
         |  |  +--rw upper-source-port?          inet:port-number
         |  |  +--rw lower-destination-port?     inet:port-number
         |  |  +--rw upper-destination-port?     inet:port-number
         |  +--rw out-segments
         |     +--rw detnet-service-sub-layer?   lower-layer-ref
         +--rw service-sub-layer
         |  +--rw operations
         |  |  +--rw service-operation
         |  |  |  +--rw service-operation-type?   service-operation-ref
         |  |  +--rw service-protection
         |  |     +--rw service-protection-type?   service-protection-type
         |  +--rw in-segments
         |  |  +--rw detnet-service-type?   flow-type-ref
         |  |  +--rw detnet-service-list* [detnet-service-index]
         |  |     +--rw detnet-service-index        uint8
```

```
          |  |     +--rw src-ipv4-prefix            inet:ipv4-prefix
          |  |     +--rw dest-ipv4-prefix           inet:ipv4-prefix
          |  |     +--rw protocol                   uint8
          |  |     +--rw dscp?                      uint8
          |  |     +--rw dscp-bitmask?              uint8
          |  |     +--rw src-ipv6-prefix            inet:ipv6-prefix
          |  |     +--rw dest-ipv6-prefix           inet:ipv6-prefix
          |  |     +--rw next-header                uint8
          |  |     +--rw traffic-class?             uint8
          |  |     +--rw traffic-class-bitmask?     uint8
          |  |     +--rw flow-label?                inet:ipv6-flow-label
          |  |     +--rw flow-label-flag?           boolean
          |  |     +--rw mpls-flow-identification
          |  |        +--rw platform-label-flag?       boolean
          |  |        +--rw non-platform-label-space
          |  |        |  +--rw incoming-interface?        if:interface-ref
          |  |        |  +--rw non-platform-label-stack* [index]
          |  |        |     +--rw index    uint8
          |  |        |     +--rw label?   rt-type:mpls-label
          |  |        |     +--rw tc?      uint8
          |  |        +--rw platform-label-space
          |  |           +--rw label?   rt-type:mpls-label
          |  |           +--rw tc?      uint8
          |  +--rw out-segments
          |     +--rw detnet-service-processing-type?   flow-type-ref
          |     +--rw detnet-service-encapsulation
          |        +--rw detnet-service-processing-list* [detnet-service-processi
ng-index]
          |           +--rw detnet-service-processing-index     uint32
          |           +--rw ip-flow
          |           |  +--rw ipv4-flow
          |           |  |  +--rw src-ipv4-address     inet:ipv4-address
          |           |  |  +--rw dest-ipv4-address    inet:ipv4-address
          |           |  |  +--rw protocol             uint8
          |           |  |  +--rw dscp?                uint8
          |           |  +--rw ipv6-flow
          |           |  |  +--rw src-ipv6-address     inet:ipv6-address
          |           |  |  +--rw dest-ipv6-address    inet:ipv6-address
          |           |  |  +--rw next-header          uint8
          |           |  |  +--rw traffic-class?       uint8
          |           |  |  +--rw flow-label?          inet:ipv6-flow-label
          |           |  +--rw l4-port-header
          |           |     +--rw source-port?        inet:port-number
          |           |     +--rw destination-port?   inet:port-number
          |           +--rw mpls-flow
          |           |  +--rw detnet-mpls-label-stack* [index]
          |           |     +--rw index                    uint8
          |           |     +--rw label?                   rt-type:mpls-label
          |           |     +--rw tc?                      uint8
```

```
          |            |     +--rw s-bit?                    boolean
          |            |     +--rw d-cw-encapsulate-flag?    boolean
          |            +--rw detnet-forwarding-sub-layer-info
          |               +--rw detnet-forwarding-sub-layer?   lower-layer-ref
          +--rw forwarding-sub-layer
          |  +--rw operations
          |  |  +--rw forwarding-operation
          |  |  |  +--rw forwarding-operation-type?   forwarding-operation-ref
          |  |  +--rw resource-allocate
          |  |  |  +--rw interval?                        uint32
          |  |  |  +--rw max-packets-per-interval?        uint32
          |  |  |  +--rw max-payload-size?                uint32
          |  |  |  +--rw average-packets-per-interval?    uint32
          |  |  |  +--rw average-payload-size?            uint32
          |  |  +--rw qos
          |  +--rw in-segments
          |  |  +--rw detnet-forwarding-type?     flow-type-ref
          |  |  +--rw src-ipv4-prefix             inet:ipv4-prefix
          |  |  +--rw dest-ipv4-prefix            inet:ipv4-prefix
          |  |  +--rw protocol                    uint8
          |  |  +--rw dscp?                       uint8
          |  |  +--rw dscp-bitmask?               uint8
          |  |  +--rw src-ipv6-prefix             inet:ipv6-prefix
          |  |  +--rw dest-ipv6-prefix            inet:ipv6-prefix
          |  |  +--rw next-header                 uint8
          |  |  +--rw traffic-class?              uint8
          |  |  +--rw traffic-class-bitmask?      uint8
          |  |  +--rw flow-label?                 inet:ipv6-flow-label
          |  |  +--rw flow-label-flag?            boolean
          |  |  +--rw mpls-flow-identification
          |  |     +--rw platform-label-flag?        boolean
          |  |     +--rw non-platform-label-space
          |  |     |  +--rw incoming-interface?         if:interface-ref
          |  |     |  +--rw non-platform-label-stack* [index]
          |  |     |     +--rw index    uint8
          |  |     |     +--rw label?   rt-type:mpls-label
          |  |     |     +--rw tc?      uint8
          |  |     +--rw platform-label-space
          |  |        +--rw label?   rt-type:mpls-label
          |  |        +--rw tc?      uint8
          |  +--rw out-segments
          |     +--rw detnet-forwarding-processing-type?   flow-type-ref
          |     +--rw natively-detnet-forwarding
          |     |  +--rw ipv4-flow
          |     |  |  +--rw ipv4-next-hop-address?   inet:ipv4-address
          |     |  +--rw ipv6-flow
          |     |     +--rw ipv6-next-hop-address?   inet:ipv6-address
          |     +--rw detnet-forwarding-encapsulation
```

```
          |         +--rw ip-flow
          |         |  +--rw ipv4-flow
          |         |  |  +--rw src-ipv4-address     inet:ipv4-address
          |         |  |  +--rw dest-ipv4-address    inet:ipv4-address
          |         |  |  +--rw protocol             uint8
          |         |  |  +--rw dscp?                uint8
          |         |  +--rw ipv6-flow
          |         |  |  +--rw src-ipv6-address     inet:ipv6-address
          |         |  |  +--rw dest-ipv6-address    inet:ipv6-address
          |         |  |  +--rw next-header          uint8
          |         |  |  +--rw traffic-class?       uint8
          |         |  |  +--rw flow-label?          inet:ipv6-flow-label
          |         |  +--rw l4-port-header
          |         |     +--rw source-port?        inet:port-number
          |         |     +--rw destination-port?   inet:port-number
          |         +--rw mpls-flow
          |         |  +--rw detnet-mpls-label-stack* [index]
          |         |     +--rw index                    uint8
          |         |     +--rw label?                   rt-type:mpls-label
          |         |     +--rw tc?                      uint8
          |         |     +--rw s-bit?                   boolean
          |         |     +--rw d-cw-encapsulate-flag?   boolean
          |         +--rw lower-layer-info
          |            +--rw lower-layer-type?   flow-type-ref
          |            +--rw interface
          |            |  +--rw outgoing-interface?   if:interface-ref
          |            +--rw sub-layer
          |               +--rw sub-layer?   lower-layer-ref
```

6. DetNet Service YANG Model

TBD

7. DetNet Configuration YANG Model

```
<CODE BEGINS> file ietf-detnet-config@20190324.yang
module ietf-detnet-config {
  namespace "urn:ietf:params:xml:ns:yang:ietf-detnet-config";
  prefix "ietf-detnet";

  import ietf-yang-types {
    prefix "yang";
  }

  import ietf-inet-types{
    prefix "inet";
  }
```

```
  import ietf-ethertypes {
    prefix "eth";
  }

  import ietf-routing-types {
    prefix "rt-type";
  }

  import ietf-interfaces {
    prefix "if";
  }

  organization "IETF DetNet Working Group";

  contact
    "WG Web:   <http://tools.ietf.org/wg/detnet/>
     WG List:  <mailto: detnet@ietf.org>
     WG Chair: Lou Berger
                <mailto:lberger@labn.net>

                Janos Farkas
                <mailto:janos.farkas@ericsson.com>

     Editor:   Xuesong Geng
                <mailto:gengxuesong@huawei.com>

     Editor:   Mach Chen
                <mailto:mach.chen@huawei.com>

     Editor:   Zhenqiang Li
                <mailto:lizhenqiang@chinamobile.com>

     Editor:   Reshad Rahman
                <mailto:rrahman@cisco.com>

     Editor:   Yeoncheol Ryoo
                <mailto:dbduscjf@etri.re.kr>";

  description
    "This YANG module describes the parameters needed
     for DetNet flow configuration and flow status reporting";

  revision 2019-03-24 {
    description "initial revision";
    reference "RFC XXXX: draft-ietf-detnet-yang-02";
  }

  identity ttl-action {
```

```
    description
      "Base identity from which all TTL
       actions are derived";
  }

  identity no-action {
    base "ttl-action";
    description
      "Do nothing regarding the TTL";
  }

  identity copy-to-inner {
    base "ttl-action";
    description
      "Copy the TTL of the outer header
       to the inner header";
  }

  identity decrease-and-copy-to-inner {
    base "ttl-action";
    description
      "Decrease TTL by one and copy the TTL
       to the inner header";
  }

  identity config-type {
    description
      "Base identity from which all configuration instances are derived";
  }

  identity App-flow {
    base "config-type";
    description
      "App-flow configuration";
  }

  identity service-sub-layer {
    base "config-type";
    description
      "A DetNet MPLS or IP service sub-layer configuration";
  }

  identity forwarding-sub-layer {
    base "config-type";
    description
      "A DetNet MPLS or IP forwarding sub-layer configuration";
  }
```

```
  identity tsn-sub-network {
    base "config-type";
    description
      "A TSN sub-net configuration";
  }

  identity flow-type {
    description
      "Base identity from which all flow type are derived";
  }

  identity ipv4 {
    base "flow-type";
    description
      "An IPv4 flow";
  }

  identity ipv6 {
    base "flow-type";
    description
      "An IPv6 flow";
  }

  identity mpls {
    base "flow-type";
    description
      "An MPLS flow";
  }

  identity l2 {
    base "flow-type";
    description
      "An MPLS flow";
  }

  identity tsn {
    base "flow-type";
    description
      "An MPLS flow";
  }

  identity service-operation {
    description
      "Base identity from which all service operation are derived";
  }

  identity service-initiation {
    base "service-operation";
```

```
    description
      "A DetNet service encapsulates";
  }

  identity service-termination {
    base "service-operation";
    description
      "A DetNet service decapsulates";
  }

  identity service-relay {
    base "service-operation";
    description
      "A DetNet service swap";
  }

  identity forwarding-operation {
    description
      "Base identity from which all data plane operation are derived";
  }

  identity natively-forward {
    base "forwarding-operation";
    description
      "A packet natively forward to lower-layer";
  }

  identity impose-and-forward {
    base "forwarding-operation";
    description
      "Impose a header(MPLS/IP) and forward to lower-layer";
  }

  identity pop-and-forward {
    base "forwarding-operation";
    description
      "Pop an identified packet header and forward to lower-layer";
  }

  identity pop-impose-and-forward {
    base "forwarding-operation";
    description
      "Pop an identified packet header, impose a one or more outgoing
       header and forward to lower-layer ";
  }

  identity swap-and-forward {
    base "forwarding-operation";
```

```
    description
      "Swap an identified packet header with outgoing header and forward
       to lower-layer ";
  }

  identity pop-and-lookup {
    base "forwarding-operation";
    description
      "Pop an identified packet header and perform a lookup";
  }
  identity label-space {
    description
      "Base identity from which all label space are derived";
  }

  identity platform-label {
    base "label-space";
    description
      "label allocated from the platform label space";
  }

  identity non-platform-label {
    base "label-space";
    description
      "label allocated from the non-platform label space";
  }

  typedef ttl-action-definition {
    type identityref {
      base "ttl-action";
    }
    description
      "TTL action definition";
  }

  typedef config-type-ref {
    type identityref {
      base "config-type";
    }
    description
      "config-type-ref";
  }

  typedef flow-type-ref {
    type identityref {
      base "flow-type";
    }
    description
```

```
      "flow-type-ref";
  }

  typedef service-operation-ref{
    type identityref {
      base "service-operation";
  }
  description
    "service-operation-ref";
  }

  typedef forwarding-operation-ref {
    type identityref {
      base "forwarding-operation";
    }
    description
      "forwarding-operation-ref";
  }

  typedef label-space-ref {
    type identityref {
      base "label-space";
    }
    description
      "label-space-ref";
  }

  typedef lower-layer-ref {
    type leafref {
      path "/ietf-detnet:detnet-config/ietf-detnet:detnet-config-list"
      + "/ietf-detnet:name";
    }
    description
     "lower-layer-ref";
  }


  typedef service-protection-type {
    type enumeration {
      enum none {
        description
          "no service protection provide";
    }
    enum replication {
      description
        "A Packet Replication Function (PRF) replicates
         DetNet flow packets and forwards them to one or
         more next hops in the DetNet domain.  The number
```

```
          of packet copies sent to each next hop is a
          DetNet flow specific parameter at the node doing
          the replication.  PRF can be implemented by an
          edge node, a relay node, or an end system";
        }
        enum elimination {
          description
            "A Packet Elimination Function (PEF) eliminates
             duplicate copies of packets to prevent excess
             packets flooding the network or duplicate
             packets being sent out of the DetNet domain.
             PEF can be implemented by an edge node, a relay
             node, or an end system.";
        }
        enum ordering {
          description
            "A Packet Ordering Function (POF) re-orders
             packets within a DetNet flow that are received
             out of order.  This function can be implemented
             by an edge node, a relay node, or an end system.";
        }
        enum elimination-ordering {
          description
            "A combination of PEF and POF that can be
             implemented by an edge node, a relay node, or
             an end system.";
        }
        enum elimination-replication {
          description
            "A combination of PEF and PRF that can be
             implemented by an edge node, a relay node, or
             an end system";
        }
        enum elimination-ordering-replicaiton {
          description
            "A combination of PEF, POF and PRF that can be
             implemented by an edge node, a relay node, or
             an end system";
        }
      }
      description
        "service-protection-type";
    }

    typedef sequence-number-generation-type {
      type enumeration {
       enum none {
         description
```

```
          "No sequence number generation function provide";
    }
    enum copy-from-app-flow {
      description
        "Copy the app-flow sequence number to the DetNet-flow";
    }
    enum generate-by-detnet-flow {
      description
        "Generate the sequence number by DetNet flow";
    }
  }
  description
    "sequence-number-generation-type";
  }

  grouping l4-port-header {
    description
      "The TCP/UDP port(source/destination) information";
    leaf source-port {
      type inet:port-number;
      description
        "The source port number";
    }
    leaf destination-port {
      type inet:port-number;
      description
        "The destination port number";
    }
  }

  grouping ipv4-header {
    description
      "The IPv4 packet header information";
    leaf src-ipv4-address {
      type inet:ipv4-address;
      mandatory true;
      description
        "The source IP address of the header";
    }
    leaf dest-ipv4-address {
      type inet:ipv4-address;
      mandatory true;
      description
        "The destination IP address of the header";
    }
    leaf protocol {
      type uint8;
      mandatory true;
```

```
      description
        "The protocol of the header";
    }
    leaf dscp {
      type uint8;
      description
        "The DSCP field of the header";
    }
  }

  grouping ipv6-header {
    description
      "The IPv6 packet header information";
    leaf src-ipv6-address {
      type inet:ipv6-address;
      mandatory true;
      description
        "The source IP address of the header";
    }
    leaf dest-ipv6-address {
      type inet:ipv6-address;
      mandatory true;
      description
        "The destination IP address of the header";
    }
    leaf next-header {
      type uint8;
      mandatory true;
      description
        "The next header of the IPv6 header";
    }
    leaf traffic-class {
      type uint8;
      description
        "The traffic class value of the header";
    }
    leaf flow-label {
      type inet:ipv6-flow-label;
      description
        "The flow label value of the header";
    }
  }

  grouping mpls-header {
    description
      "The MPLS packet header information";
    leaf label {
      type rt-type:mpls-label;
```

```
      description
        "The label value of the MPLS header";
    }
    leaf tc {
      type uint8;
      description
        "The traffic class value of the MPLS header";
    }
    leaf s-bit {
      type boolean;
      description
        "The s-bit value of the MPLS header,
         which indicates the bottom of the label shack";
    }
    leaf d-cw-encapsulate-flag {
      type boolean;
      description
        "the indication of whether D-CW  is encapsulated or not,
         when the D-CW is encapsulated, the sequence number is
         determined by sequence generation type";
    }
  }

  grouping l2-header {
    description
      "The Ethernet or TSN packet header information";
    leaf source-mac-address {
      type yang:mac-address;
      description
        "The source MAC address value of the ethernet header";
    }
    leaf destination-mac-address {
      type yang:mac-address;
      description
        "The destination MAC address value of the ethernet header";
    }
    leaf ethertype {
      type eth:ethertype;
      description
        "The ethernet packet type value of the ethernet header";
    }
    leaf vlan-id {
      type uint16;
      description
        "The Vlan value of the ethernet header";
    }
    leaf pcp {
      type uint8;
```

```
      description
        "The priority value of the ethernet header";
    }
  }

  grouping l4-port-identification {
    description
      "The TCP/UDP port(source/destination) identification information";
    leaf lower-source-port {
      type inet:port-number;
      description
        "The lower source port number of the source port range";
    }
    leaf upper-source-port {
      type inet:port-number;
      description
        "The upper source port number of the source port range";
    }
    leaf lower-destination-port {
      type inet:port-number;
      description
        "The lower destination port number or the destination port range";
    }
    leaf upper-destination-port {
      type inet:port-number;
      description
        "The upper destination port number of the destination port range";
    }
  }

  grouping ipv4-flow-identification {
    description
      "The IPv4 packet header identification information";
    leaf src-ipv4-prefix {
      type inet:ipv4-prefix;
      mandatory true;
      description
        "The source IP address of the header";
    }
    leaf dest-ipv4-prefix {
      type inet:ipv4-prefix;
      mandatory true;
      description
        "The destination IP address of the header";
    }
    leaf protocol {
      type uint8;
      mandatory true;
```

```
      description
        "The protocol of the header";
    }
    leaf dscp {
      type uint8;
      description
        "The DSCP field of the header";
    }
    leaf dscp-bitmask {
      type uint8;
      description
        "The bitmask value that determines whether to use
         the DSCP(IPv4) value for flow identification or not";
     }
  }

  grouping ipv6-flow-identification {
    description
      "The IPv6 packet header identification information";
    leaf src-ipv6-prefix {
      type inet:ipv6-prefix;
      mandatory true;
      description
        "The source IP address of the header";
    }
    leaf dest-ipv6-prefix {
      type inet:ipv6-prefix;
      mandatory true;
      description
        "The destination IP address of the header";
    }
    leaf next-header {
      type uint8;
      mandatory true;
      description
        "The next header of the IPv6 header";
    }
    leaf traffic-class {
      type uint8;
      description
        "The traffic class value of the header";
    }
    leaf traffic-class-bitmask {
      type uint8;
      description
        "The bitmask value that determines whether to use
         the Traffic class(IPv6) value for flow identification or not";
    }
```

```
    leaf flow-label {
      type inet:ipv6-flow-label;
      description
        "The flow label value of the header";
    }
    leaf flow-label-flag {
      type boolean;
      description
        "The flag that determines whether to use
         the Flow Label value for flow identification or not";
    }
  }

  grouping mpls-flow-identification {
    description
      "The MPLS packet header identification information";
    leaf label {
      type rt-type:mpls-label;
      description
        "The label value of the MPLS header";
    }
    leaf tc {
      type uint8;
      description
        "The traffic class value of the MPLS header";
    }
  }

  grouping l2-flow-identification {
    description
      "The Ethernet or TSN packet header identification information";
    leaf source-mac-address {
      type yang:mac-address;
      description
        "The source MAC address value of the ethernet header";
    }
    leaf destination-mac-address {
      type yang:mac-address;
      description
        "The destination MAC address value of the ethernet header";
    }
    leaf ethertype {
      type eth:ethertype;
      description
        "The ethernet packet type value of the ethernet header";
    }
    leaf vlan-id {
      type uint16;
```

```
      description
        "The Vlan value of the ethernet header";
    }
    leaf pcp {
      type uint8;
      description
        "The priority value of the ethernet header";
    }
  }

  grouping traffic-specification {
    description
      "traffic-specification specifies how the Source
       transmits packets for the flow.  This is the
       promise/request of the Source to the network.
       The network uses this traffic specification
       to allocate resources and adjust queue
       parameters in network nodes.";
    reference
      "draft-ietf-detnet-flow-information-model";
    leaf interval {
      type uint32;
        description
          "The period of time in which the traffic
           specification cannot be exceeded";
    }
    leaf max-packets-per-interval{
      type uint32;
      description
        "The maximum number of packets that the
         source will transmit in one Interval.";
    }
    leaf max-payload-size{
      type uint32;
      description
        "The maximum payload size that the source
         will transmit.";
    }
    leaf average-packets-per-interval {
      type uint32;
      description
        "The average number of packets that the
         source will transmit in one Interval";
    }
    leaf average-payload-size {
      type uint32;
      description
        "The average payload size that the
```

```
          source will transmit.";
      }
    }

    container detnet-config {
      description
        "DetNet configurations";
      leaf node-id {
        type yang:dotted-quad;
        description
          "A 32-bit number in the form of a dotted quad that is used by
           identifying a DetNet node";
      }
      list detnet-config-list {
        key "name";
        description
          "list of the DetNet configurations";
        leaf name {
          type string;
          description
            "The name to identify the DetNet configuration";
        }
        leaf config-type {
          type config-type-ref;
          description
            "The DetNet configuration type such as a App-flow, service
             sub-layer, forwarding sub-layer, and TSN sub-network";
        }
        container App-flow {
          when "../config-type = 'ietf-detnet:App-flow'";
          description
            "The DetNet App-flow configuration";
          container operations {
            description "operations";
            container sequence-number {
              description "The DetNet sequence number operations grouping";
              leaf sequence-number-generation-type {
                type sequence-number-generation-type;
                description "The DetNet sequence number generation type";
              }
              leaf sequence-number-length {
                type uint8;
                description
                  "The DetNet sequence number length";
              }
            }
          }
          container in-segments {
```

```
          description "The App-flow identification information";
          leaf app-flow-type {
            type flow-type-ref;
            description
              "The App-flow type such as a L2, IPv4, and IPv6";
          }
          uses l2-flow-identification {
            when "app-flow-type = 'ietf-detnet:tsn' or 'ietf-detnet:l2'";
          }
          uses ipv4-flow-identification {
            when "app-flow-type = 'ietf-detnet:ipv4'";
          }
          uses ipv6-flow-identification {
            when "app-flow-type = 'ietf-detnet:ipv6'";
          }
          uses l4-port-identification {
when "app-flow-type = 'ietf-detnet:ipv6' or 'ietf-detnet:ipv4'";
            or 'ietf-detnet:ipv4'";
          }
        }
        container out-segments {
          description
            "The DetNet service information associated with this App-flow";
          leaf detnet-service-sub-layer {
            type  lower-layer-ref;
            description "Specify associated service sub-layer";
          }
        }
      }
      container service-sub-layer {
        when "../config-type = 'ietf-detnet:service-sub-layer'";
        description "The DetNet service sub-layer configuration";
        container operations {
          description
            "The DetNet service sub-layer operations grouping";
          container service-operation {
            description "The DetNet service operations grouping";
            leaf service-operation-type {
              type service-operation-ref;
              description
              "The DetNet service operations type such as DetNet
               service initiation, termination, and relay";
            }
          }
          container service-protection {
            description
              "The DetNet service protection operations grouping";
            leaf service-protection-type {
```

```
              type service-protection-type;
              description
                "The DetNet service protection type such as PRF, PEF, PEOF,
                 PERF, and PEORF";
            }
          }
        }
        container in-segments {
          when "../operations/service-operation"
          + "/service-operation-type != 'service-initation'";
          description
            "DetNet service identification information";
          leaf detnet-service-type {
            type flow-type-ref;
          description
            "incoming DetNet service flow type";
          }
          list detnet-service-list {
            key "detnet-service-index";
            description
              "Incoming DetNet member flows or a compound flow";
            leaf detnet-service-index {
              type uint8;
              description
                "Incoming DetNet service index";
            }
            uses ipv4-flow-identification {
      when "../detnet-service-type = 'ietf-detnet:ipv4'";
            }
            uses ipv6-flow-identification {
      when "../detnet-service-type = 'ietf-detnet:ipv6'";
            }
            container mpls-flow-identification {
              when "../../detnet-service-type = 'ietf-detnet:mpls'";
              description
                "MPLS type DetNet service identification";
      leaf label-space {
        type label-space-ref;
                description
                  "Indicate the incoming MPLS label is associated with
                   platform label space or not";
              }
              container non-platform-label-space {
when "../label-space = 'ietf-detnet:non-platform-label'";
                description
                  "MPLS label is associated with non-platform label space,
                   all of the F-labels and incoming interface information was
                   used for identification";
```

```
              leaf incoming-interface {
                type if:interface-ref;
                description
                  "DetNet service incoming interface information";
              }
              list non-platform-label-stack {
                key "index";
                description
                  "All of the label information from the outer label
                   to the current label";
                leaf index {
                  type uint8;
                  description
                    "Index of the labels stack";
                }
                uses mpls-flow-identification;
              }
            }
            container platform-label-space {
        when "../label-space = 'ietf-detnet:platform-label'";
              description
                "MPLS label is associated with platform label space, only
                 the F-label is used for identification";
              uses mpls-flow-identification;
            }
          }
        }
        container out-segments {
          when "../operations/service-operation"
          + "/service-operation-type != 'service-termination'";
          description
            "DetNet Service outgoing processing grouping";
          leaf detnet-service-processing-type {
            type flow-type-ref;
          description
            "Outgoing DetNet service flow type";
          }
          container detnet-service-encapsulation {
            description
              "DetNet service encapsulation information";
            list detnet-service-processing-list {
              key "detnet-service-processing-index";
              description
                "The list of single or multiple outgoing DetNet service(s)";
              leaf detnet-service-processing-index {
                type uint32;
                description "Outgoing segment entry";
```

```
            }
            container ip-flow {
              when "../../../detnet-service-processing-type ="
              + "'ietf-detnet:ipv4' or 'ietf-detnet:ipv6'";
              description
                "IP type DetNet flow(s) encapsulation information";
              container ipv4-flow {
                when "../../../../detnet-service-processing-type ="
                + "'ietf-detnet:ipv4'";
                description
                  "IPv4 packet header encapsulation information";
                uses ipv4-header;
              }
              container ipv6-flow {
                when "../../../../detnet-service-processing-type ="
                + "'ietf-detnet:ipv6'";
                  description
                    "IPv6 packet header encapsulation information";
                  uses ipv6-header;
              }
              container l4-port-header {
                description
                  "TCP/UDP source or destination port number";
                uses l4-port-header;
              }
            }
            container mpls-flow {
              when "../../../detnet-service-processing-type ="
              + "'ietf-detnet:mpls'";
              description
                "MPLS type DetNet flow(s) encapsulation information";
                list detnet-mpls-label-stack {
                  key "index";
                  description
                    "The list of MPLS labels stack for swap or encapsulation";
                  leaf index {
                    type uint8;
                    description "Index of the labels stack";
                  }
                  uses mpls-header;
                }
            }
            container detnet-forwarding-sub-layer-info {
              description
                "The forwarding sub-layer information that associated with
                 this DetNet service sub-layer";
              leaf detnet-forwarding-sub-layer {
                type lower-layer-ref;
```

```
                  description
                    "Specify associated forwarding sub-layer";
                }
              }
            }
          }
        }
        container forwarding-sub-layer {
          when "../config-type = 'ietf-detnet:forwarding-sub-layer'";
          description
                    "The DetNet forwarding sub-layer configuration";
          container operations {
            description
                      "The DetNet forwarding sub-layer operations grouping";
            container forwarding-operation {
              description
                "DetNet forwarding function operations grouping";
              leaf forwarding-operation-type {
                type forwarding-operation-ref;
                description
                  "DetNet forwarding operation type such as
                   natively forward, impose and forward, pop and forward,
                   pop and impose and forward, swap and forward,
                   and pop and lookup";
              }
            }
            container resource-allocate {
              description
                "resource-allocation function operations grouping";
              uses traffic-specification;
            }
            container qos {
              description
                "QoS function operations grouping";
            }
          }
          container in-segments {
            description
              "DetNet forwarding sub-layer packet identification information";
            leaf detnet-forwarding-type {
              type flow-type-ref;
              description
                "incoming DetNet forwarding packet type";
            }
            uses ipv4-flow-identification {
      when "detnet-forwardinge-type = 'ietf-detnet:ipv4'";
            }
```

```
          uses ipv6-flow-identification {
    when "detnet-forwarding-type = 'ietf-detnet:ipv6'";
          }
          container mpls-flow-identification {
            when "../detnet-forwarding-type = 'ietf-detnet:mpls'";
            description
              "MPLS type identification information";
    leaf label-space {
      type label-space-ref;
              description
                "Indicate the incoming MPLS label is associated with platform
                 label space or not";
            }
            container non-platform-label-space {
when "../label-space = 'ietf-detnet:non-platform-label'";
              description
                "MPLS label is associated with non-platform label space,
                 all of the F-labels and incoming interface information was
                 used for identification";
              leaf incoming-interface {
                type if:interface-ref;
                description
                  "The information of DetNet forwarding packet incoming
                   interface";
              }
              list non-platform-label-stack {
                key "index";
                description
                  "All of the label information from the outer label to
                   the current label";
                leaf index {
                  type uint8;
                  description
                    "index number 0 indicate last inner label";
                }
                uses mpls-flow-identification;
              }
            }
            container platform-label-space {
             when "../label-space = 'ietf-detnet:platform-label'";
              description
                "MPLS label is associated with platform label space, only
                 the F-label is used for identification";
              uses mpls-flow-identification;
            }
          }
        }
        container out-segments {
```

```
          description
            "DetNet forwarding sub-layer packet processing information";
          leaf detnet-forwarding-processing-type {
            type flow-type-ref;
            description
              "outgoing DetNet forwarding packet type";
          }
          container natively-detnet-forwarding {
            when "../../operations/forwarding-operation"
            + " /forwarding-operation-type = 'natively-forwarding'";
            description
            "Packet forwarding processing information";
            container ipv4-flow {
              when "../../detnet-forwarding-processing-type ="
              + "'ietf-detnet:ipv4'";
              description
                "IPv4 type packet forwarding information";
              leaf ipv4-next-hop-address {
                type inet:ipv4-address;
                description
                  "IPv4 type Next hop IP address";
              }
            }
            container ipv6-flow {
              when "../../detnet-forwarding-processing-type ="
              + "'ietf-detnet:ipv6'";
              description
                "IPv6 type packet forwarding information";
              leaf ipv6-next-hop-address {
                type inet:ipv6-address;
                description
                  "IPv6 type Next hop IP address";
              }
            }
          }
          container detnet-forwarding-encapsulation {
            when "../../operations/forwarding-operation"
            + "/forwarding-operation-type != 'natively-forward'";
            description
              "Packet encapsulation information";
            container ip-flow {
              when "../../detnet-forwarding-processing-type = "
              + "'ietf-detnet:ipv4' or 'ietf-detnet:ipv6'";
              description
                "The IP type DetNet flow(s) encapsulation information";
              container ipv4-flow {
                when "../../../detnet-forwarding-processing-type = "
                + "'ietf-detnet:ipv4'";
```

```
            description
              "IPv4 packet header encapsulation information";
            uses ipv4-header;
          }
          container ipv6-flow {
            when "../../../detnet-forwarding-processing-type = "
            + "'ietf-detnet:ipv6'";
            description
              "IPv6 packet header encapsulation information";
            uses ipv6-header;
          }
          container l4-port-header {
            description
               "TCP/UDP source or destination port number";
            uses l4-port-header;
          }
        }
        container mpls-flow {
          when "../../detnet-forwarding-processing-type = "
          + "'ietf-detnet:mpls'";
          description
            "MPLS label encapsulation information";
          list detnet-mpls-label-stack {
            key "index";
              description
                "The list of MPLS labels stack for swap or encapsulation";
            leaf index {
              type uint8;
              description
               "Index of the labels stack";
            }
            uses mpls-header;
          }
        }
        container lower-layer-info {
          description
            "The lower-layer information associated with
             this forwarding sub-layer";
          leaf lower-layer-type {
            type flow-type-ref;
            description
              "indicate lower-layer type";
          }
          container interface {
            when "../lower-layer-type = 'ietf-detnet:l2'";
            description
              "indicate the lower-layer is the outgoing interface";
            leaf outgoing-interface {
```

```
                  type if:interface-ref;
                  description
                   "Outgoing interface";
                }
              }
              container sub-layer {
                when "../lower-layer-type != 'ietf-detnet:l2'";
                description
                  "indicate the lower-layer is some of the DetNet sub-layer
                   or TSN sub-network";
                leaf sub-layer {
                  type lower-layer-ref;
                  description
                   "Specify associated DetNet sub-layer or TSN sub-network";
                }
              }
            }
          }
        }
      }
      container sub-network {
        when "../config-type = 'ietf-detnet:tsn-sub-network'";
       description
         "sub-network";
      }
    }
  }
}
```

<CODE ENDS>

8. Open Issues

There are some open issues that are still under discussion:

- The Relationship with 802.1 TSN YANG models is TBD. TSN YANG models include: P802.1Qcw, which defines TSN YANG for Qbv, Qbu, and Qci, and P802.1CBcv, which defines YANG for 802.1CB. The possible problem here is how to avoid possible overlap among yang models defined in IETF and IEEE. A common YANG model may be defined in the future to shared by both TSN and DetNet. More discussion are needed here.

- How to support DetNet OAM is TBD.

These issues will be resolved in the following versions of the draft.

9. IANA Considerations

This document makes no request of IANA.

Note to RFC Editor: this section may be removed on publication as an RFC.

10. Security Considerations

<TBD>

11. Acknowledgements

12. References

12.1. Normative References

[I-D.finn-detnet-bounded-latency]
Finn, N., Boudec, J., Mohammadpour, E., Zhang, J., Varga, B., and J. Farkas, "DetNet Bounded Latency", draft-finn-detnet-bounded-latency-04 (work in progress), June 2019.

[I-D.ietf-detnet-architecture]
Finn, N., Thubert, P., Varga, B., and J. Farkas, "Deterministic Networking Architecture", draft-ietf-detnet-architecture-13 (work in progress), May 2019.

[I-D.ietf-detnet-flow-information-model]
Farkas, J., Varga, B., Cummings, R., Jiang, Y., and D. Fedyk, "DetNet Flow Information Model", draft-ietf-detnet-flow-information-model-06 (work in progress), October 2019.

[I-D.ietf-detnet-ip]
Varga, B., Farkas, J., Berger, L., Fedyk, D., Malis, A., Bryant, S., and J. Korhonen, "DetNet Data Plane: IP", draft-ietf-detnet-ip-03 (work in progress), October 2019.

[I-D.ietf-detnet-mpls]
Varga, B., Farkas, J., Berger, L., Fedyk, D., Malis, A., Bryant, S., and J. Korhonen, "DetNet Data Plane: MPLS", draft-ietf-detnet-mpls-03 (work in progress), October 2019.

[I-D.ietf-detnet-topology-yang]
Geng, X., Chen, M., Li, Z., and R. Rahman, "Deterministic Networking (DetNet) Topology YANG Model", draft-ietf-detnet-topology-yang-00 (work in progress), January 2019.

[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>.

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <https://www.rfc-editor.org/info/rfc6991>.

[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>.

12.2. Informative References

[I-D.geng-detnet-info-distribution]
Geng, X., Chen, M., Li, Z., Qin, F., and L. Qiang, "IGP-TE Extensions for DetNet Information Distribution", draft-geng-detnet-info-distribution-04 (work in progress), July 2019.

[I-D.ietf-detnet-use-cases]
Grossman, E., "Deterministic Networking Use Cases", draft-ietf-detnet-use-cases-20 (work in progress), December 2018.

[I-D.ietf-teas-yang-te]
Saad, T., Gandhi, R., Liu, X., Beeram, V., and I. Bryskin, "A YANG Data Model for Traffic Engineering Tunnels and Interfaces", draft-ietf-teas-yang-te-21 (work in progress), April 2019.

[I-D.ietf-teas-yang-te-topo]
Liu, X., Bryskin, I., Beeram, V., Saad, T., Shah, H., and O. Dios, "YANG Data Model for Traffic Engineering (TE) Topologies", draft-ietf-teas-yang-te-topo-22 (work in progress), June 2019.

[I-D.thubert-tsvwg-detnet-transport]
Thubert, P., "A Transport Layer for Deterministic Networks", draft-thubert-tsvwg-detnet-transport-01 (work in progress), October 2017.

[I-D.varga-detnet-service-model]
Varga, B. and J. Farkas, "DetNet Service Model", draft-varga-detnet-service-model-02 (work in progress), May 2017.

[IEEE802.1CB]
IEEE, "IEEE, "Frame Replication and Elimination for Reliability (IEEE Draft P802.1CB)", 2017, <http://www.ieee802.org/1/files/private/cb-drafts/>.", 2016.

[IEEE802.1Q-2014]
"IEEE, "IEEE Std 802.1Q Bridges and Bridged Networks", 2014, <http://ieeexplore.ieee.org/document/6991462/>.", 2014.

[IEEE802.1Qbu]
IEEE, "IEEE, "IEEE Std 802.1Qbu Bridges and Bridged Networks - Amendment 26: Frame Preemption", 2016, <http://ieeexplore.ieee.org/document/7553415/>.", 2016.

[IEEE802.1Qbv]
"IEEE, "IEEE Std 802.1Qbu Bridges and Bridged Networks - Amendment 25: Enhancements for Scheduled Traffic", 2015, <http://ieeexplore.ieee.org/document/7572858/>.", 2016.

[IEEE802.1Qcc]
IEEE, "IEEE, "Stream Reservation Protocol (SRP) Enhancements and Performance Improvements (IEEE Draft P802.1Qcc)", 2017, <http://www.ieee802.org/1/files/private/cc-drafts/>.".

[IEEE802.1Qch]
IEEE, "IEEE, "Cyclic Queuing and Forwarding (IEEE Draft P802.1Qch)", 2017, <http://www.ieee802.org/1/files/private/ch-drafts/>.", 2016.

[IEEE802.1Qci]
IEEE, "IEEE, "Per-Stream Filtering and Policing (IEEE Draft P802.1Qci)", 2016, <http://www.ieee802.org/1/files/private/ci-drafts/>.", 2016.

[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>.

[RFC4875] Aggarwal, R., Ed., Papadimitriou, D., Ed., and S. Yasukawa, Ed., "Extensions to Resource Reservation Protocol - Traffic Engineering (RSVP-TE) for Point-to-Multipoint TE Label Switched Paths (LSPs)", RFC 4875, DOI 10.17487/RFC4875, May 2007, <https://www.rfc-editor.org/info/rfc4875>.

[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>.

Authors' Addresses

Xuesong Geng
Huawei Technologies

Email: gengxuesong@huawei.com

Mach(Guoyi) Chen
Huawei Technologies

Email: mach.chen@huawei.com

Yeoncheol Ryoo
ETRI

Email: dbduscjf@etri.re.kr

Zhenqiang Li
China Mobile

Email: lizhenqiang@chinamobile.com

Reshad Rahman
Cisco Systems

Email: rrahman@cisco.com