

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 13, 2020

X. Geng
M. Chen
Huawei Technologies
Y. Ryoo
ETRI
Z. Li
China Mobile
R. Rahman
Cisco Systems
D. Fedyk
LabN Consulting, L.L.C.
June 11, 2020

Deterministic Networking (DetNet) Configuration YANG Model draft-ietf-detnet-yang-06

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 December 13, 2020.

Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1.	Introduction	3
2.	Terminologies	3
3.	DetNet Configuration Module	3
3.1.	DetNet Application Flow Configuration Attributes	3
3.2.	DetNet Service Sub-layer Configuration Attributes	3
3.3.	DetNet Forwarding Sub-layer Configuration Attributes	4
4.	DetNet YANG Structure Considerations	4
5.	DetNet Configuration YANG Structures	5
6.	DetNet Configuration YANG Model	16
7.	An alternative DetNet YANG model	34
7.1.	Conventions	35
7.1.1.	Aggregation	35
7.2.	Alternative DetNet Configuration YANG	36
7.3.	Alternative DetNet Configuration YANG Model	50
7.4.	Test Configuration	67
7.4.1.	DetNet Test Configuration YANG Model	67
7.4.2.	DetNet Test Configuration YANG Model	69
8.	Open Issues	70
9.	IANA Considerations	71
10.	Security Considerations	71
11.	Acknowledgements	71
12.	References	71
12.1.	Normative References	71
12.2.	Informative References	72
	Authors' Addresses	74

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 [[RFC8655](#)].

This document defines a YANG model for DetNet based on YANG data types and modeling language defined in [[RFC6991](#)] and [[RFC7950](#)]. DetNet service, which is designed for describing the characteristics of services being provided for application flows over a network, and DetNet configuration, which 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, are both included in this document.

2. Terminologies

This documents uses the terminologies defined in [[RFC8655](#)].

3. DetNet Configuration Module

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

3.1. DetNet Application 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 map a flow at the User Network Interface (UNI), the corresponding attributes are defined in [[I-D.ietf-detnet-flow-information-model](#)].

3.2. 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:

- o DetNet flow identification
- o 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). The corresponding arguments for service functions also needs to be defined.

3.3. DetNet Forwarding Sub-layer Configuration Attributes

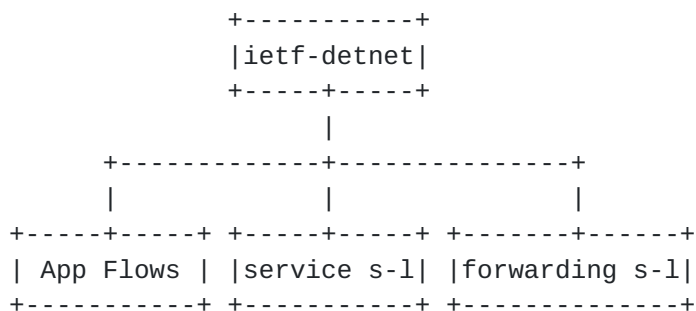
As defined in [[RFC8655](#)], DetNet forwarding sub-layer optionally provides congestion protection for DetNet flows over paths provided by the underlying network. Explicit route is another mechanism that is used by DetNet to avoid temporary interruptions caused by the convergence of routing or bridging 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:

- o Traffic Specification, refers to Section 7.2 of [[I-D.ietf-detnet-flow-information-model](#)]. It may used for resource reservation, flow shaping, filtering and policing.
- o 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. DetNet YANG Structure Considerations

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



There are three instances in DetNet YANG Model: App-flow instance, service sub-layer instance and forwarding sub-layer 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:

- o Name: indicates the key value of the instance identification.
- o In-segments: indicates the key value of identification, e.g., Layer 2 App flow identification, Layer 3 App flow identification and DetNet flow identification.
- o 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.
- o 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. DetNet Configuration YANG Structures

```

module: ietf-detnet-config-modify
  +--rw app-flows
  |   +--rw app-flow* [name]
  |   |   +--rw name                string
  |   |   +--ro app-id?             uint16
  |   |   +--rw app-flow-bidir-congruent? boolean
  |   |   +--rw traffic-requirements
  |   |   |   +--rw min-bandwidth?    uint64
  |   |   |   +--rw max-latency?      uint32
  |   |   |   +--rw max-latency-variation? uint32
  |   |   |   +--rw max-loss?         uint8
  |   |   |   +--rw max-consecutive-loss-tolerance? uint32
  |   |   |   +--rw max-misordering?  uint32
  |   |   +--rw traffic-specification
  |   |   |   +--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 in-segment
  |   |   |   +--rw interface?        if:interface-ref
  |   |   |   +--rw (data-flow-type)?
  |   |   |   |   +--:(tsn-app-flow)
  |   |   |   |   |   +--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
|         | +---:(ip-app-flow)
|         |         | +--rw src-ip-prefix?      inet:ip-prefix
|         |         | +--rw dest-ip-prefix?     inet:ip-prefix
|         |         | +--rw next-header?        uint8
|         |         | +--rw traffic-class?       uint8
|         |         | +--rw flow-label?         inet:ipv6-flow-label
|         |         | +--rw source-port
|         |         | | +--rw (port-range-or-operator)?
|         |         | | | +---:(range)
|         |         | | | | +--rw lower-port    inet:port-number
|         |         | | | | +--rw upper-port    inet:port-number
|         |         | | | +---:(operator)
|         |         | | | | +--rw operator?     operator
|         |         | | | | +--rw port          inet:port-number
|         |         | +--rw destination-port
|         |         | | +--rw (port-range-or-operator)?
|         |         | | | +---:(range)
|         |         | | | | +--rw lower-port    inet:port-number
|         |         | | | | +--rw upper-port    inet:port-number
|         |         | | | +---:(operator)
|         |         | | | | +--rw operator?     operator
|         |         | | | | +--rw port          inet:port-number
|         |         | +--rw ipsec-spi?          ipsec-spi
|         | +---:(mpls-app-flow)
|         |         | +--rw (label-space)?
|         |         | | +---:(context-label-space)
|         |         | | | +--rw mpls-label-stack
|         |         | | | | +--rw entry* [id]
|         |         | | | | | +--rw id          uint8
|         |         | | | | | +--rw label?       rt-types:mpls-label
|         |         | | | | | +--rw ttl?         uint8
|         |         | | | | | +--rw traffic-class? uint8
|         |         | +---:(platform-label-space)
|         |         | | +--rw label?             rt-types:mpls-label
| +--rw out-segment
|   +--rw (outgoing-options)
|     +---:(ingress-proxy)
|       | +--rw service-sub-layer?  service-sub-layer-ref
|     +---:(egress)
|       +--rw (next-hop-options)
|         +---:(simple-next-hop)
|           | +--rw mpls-header
|           | | +--rw mpls-label-stack
|           | | | +--rw entry* [id]
|           | | | | +--rw id          uint8

```



```

|           | |           +--rw label?          rt-types:mpls-label
|           | |           +--rw ttl?           uint8
|           | |           +--rw traffic-class? uint8
|           | +--rw outgoing-interface? if:interface-ref
|       +--:(next-hop-list)
|           +--rw next-hop-list
|               +--rw next-hop* [hop-index]
|                   +--rw hop-index            uint8
|                   +--rw mpls-header
|                       +--rw mpls-label-stack
|                           +--rw entry* [id]
|                               +--rw id          uint8
|                               +--rw label?      rt-types:mpls-label
|                               +--rw ttl?        uint8
|                               +--rw traffic-class? uint8
|                               +--rw outgoing-interface? if:interface-ref
+--rw service-sub-layer
|   +--rw service-sub-layer-list* [name]
|       +--rw name                string
|       +--ro service-id?         uint16
|       +--rw service-rank?       uint8
|       +--rw traffic-requirements
|           | +--rw min-bandwidth?          uint64
|           | +--rw max-latency?            uint32
|           | +--rw max-latency-variation?  uint32
|           | +--rw max-loss?               uint8
|           | +--rw max-consecutive-loss-tolerance? uint32
|           | +--rw max-misordering?        uint32
|       +--rw traffic-specification
|           | +--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 service-protection
|           | +--rw service-protection-type? service-protection-type
|           | +--rw sequence-number-length?  sequence-number-field
|       +--rw service-operation-type? service-operation-type
|       +--rw in-segment
|           | +--rw (incoming-options)
|           |   +--:(ingress-proxy)
|           |       | +--rw app-flow*          app-flow-ref
|           |       +--:(detnet-service-identification)
|           |           +--rw (detnet-flow-type)?
|           |               +--:(ip-detnet-flow)
|           |                   | +--rw src-ip-prefix?  inet:ip-prefix
|           |                   | +--rw dest-ip-prefix? inet:ip-prefix
|           |                   | +--rw next-header?    uint8

```



```

|         |         | +--rw traffic-class?      uint8
|         |         | +--rw flow-label?        inet:ipv6-flow-label
|         |         | +--rw source-port
|         |         | | +--rw (port-range-or-operator)?
|         |         | |   +--:(range)
|         |         | |   | +--rw lower-port  inet:port-number
|         |         | |   | +--rw upper-port  inet:port-number
|         |         | |   +--:(operator)
|         |         | |     +--rw operator?   operator
|         |         | |     +--rw port        inet:port-number
|         |         | +--rw destination-port
|         |         | | +--rw (port-range-or-operator)?
|         |         | |   +--:(range)
|         |         | |   | +--rw lower-port  inet:port-number
|         |         | |   | +--rw upper-port  inet:port-number
|         |         | |   +--:(operator)
|         |         | |     +--rw operator?   operator
|         |         | |     +--rw port        inet:port-number
|         |         | +--rw ipsec-spi?        ipsec-spi
|         |         +--:(mpls-detnet-flow)
|         |         +--rw (label-space)?
|         |         | +--:(context-label-space)
|         |         | | +--rw mpls-label-stack
|         |         | |   +--rw entry* [id]
|         |         | |     +--rw id          uint8
|         |         | |     +--rw label?      rt-types:mpls-label
|         |         | |     +--rw ttl?        uint8
|         |         | |     +--rw traffic-class? uint8
|         |         | +--:(platform-label-space)
|         |         |   +--rw label?          rt-types:mpls-label
|         +--rw out-segment
|         | +--rw (outgoing-options)
|         | | +--:(egress-proxy)
|         | | | +--rw app-flow*          app-flow-ref
|         | | +--:(detnet-service-output)
|         | | +--rw service-output-list* [service-output-index]
|         | |   +--rw service-output-index  uint8
|         | |   +--rw (header-type)?
|         | |   | +--:(detnet-mpls-header)
|         | |   | | +--rw mpls-label-stack
|         | |   | |   +--rw entry* [id]
|         | |   | |     +--rw id          uint8
|         | |   | |     +--rw label?      rt-types:mpls-label
|         | |   | |     +--rw ttl?        uint8
|         | |   | |     +--rw traffic-class? uint8
|         | |   +--:(detnet-ip-header)
|         | |     +--rw src-ip-address?    inet:ip-address
|         | |     +--rw dest-ip-address?   inet:ip-address

```



```

|         |         +--rw next-header?          uint8
|         |         +--rw traffic-class?         uint8
|         |         +--rw flow-label?           inet:ipv6-flow-label
|         |         +--rw source-port?          inet:port-number
|         |         +--rw destination-port?     inet:port-number
|         +--rw next-layer* [index]
|         |         +--rw index                  uint8
|         |         +--rw forwarding-sub-layer? forwarding-sub-layer-ref
+--rw forwarding-sub-layer
|   +--rw forwarding-sub-layer-list* [name]
|     +--rw name                                string
|     +--ro forwarding-id?                      uint16
|     +--rw traffic-requirements
|       |   +--rw min-bandwidth?                uint64
|       |   +--rw max-latency?                  uint32
|       |   +--rw max-latency-variation?        uint32
|       |   +--rw max-loss?                     uint8
|       |   +--rw max-consecutive-loss-tolerance? uint32
|       |   +--rw max-misordering?              uint32
|     +--rw traffic-specification
|       |   +--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 forwarding-operation-type? forwarding-operations-type
|     +--rw in-segment
|       |   +--rw (incoming-options)
|       |     +--:(detnet-service-forwarding)
|       |       |   +--rw service-sub-layer* service-sub-layer-ref
|       |       +--:(detnet-forwarding-identification)
|       |         +--rw interface?              if:interface-ref
|       |         +--rw (detnet-flow-type)?
|       |           +--:(ip-detnet-flow)
|       |             |   +--rw src-ip-prefix?   inet:ip-prefix
|       |             |   +--rw dest-ip-prefix? inet:ip-prefix
|       |             |   +--rw next-header?    uint8
|       |             |   +--rw traffic-class?  uint8
|       |             |   +--rw flow-label?     inet:ipv6-flow-label
|       |             |   +--rw source-port
|       |             |     +--rw (port-range-or-operator)?
|       |             |       +--:(range)
|       |             |         |   +--rw lower-port  inet:port-number
|       |             |         |   +--rw upper-port  inet:port-number
|       |             |         +--:(operator)
|       |             |           +--rw operator?    operator
|       |             |           +--rw port         inet:port-number
|       |             +--rw destination-port

```



```

| | | | +--rw (port-range-or-operator)?
| | | |   +--:(range)
| | | |     | +--rw lower-port  inet:port-number
| | | |     | +--rw upper-port  inet:port-number
| | | |     +--:(operator)
| | | |       +--rw operator?   operator
| | | |       +--rw port        inet:port-number
| | | +--rw ipsec-spi?          ipsec-spi
| | +--:(mpls-detnet-flow)
| |   +--rw (label-space)?
| |     +--:(context-label-space)
| |       | +--rw mpls-label-stack
| |       |   +--rw entry* [id]
| |       |     +--rw id          uint8
| |       |     +--rw label?      rt-types:mpls-label
| |       |     +--rw ttl?        uint8
| |       |     +--rw traffic-class? uint8
| |       +--:(platform-label-space)
| |         +--rw label?          rt-types:mpls-label
| +--rw out-segment
|   +--rw (outgoing-options)
|     +--:(service-operation)
|       | +--rw service-sub-layer*  service-sub-layer-ref
|     +--:(detnet-forwarding-output)
|       +--rw (next-hop-options)
|         +--:(simple-next-hop)
|           | +--rw (header-type)?
|           | | +--:(detnet-mpls-header)
|           | | | +--rw mpls-label-stack
|           | | |   +--rw entry* [id]
|           | | |     +--rw id          uint8
|           | | |     +--rw label?      rt-types:mpls-label
|           | | |     +--rw ttl?        uint8
|           | | |     +--rw traffic-class? uint8
|           | | +--:(detnet-ip-header)
|           | |   +--rw src-ip-address?  inet:ip-address
|           | |   +--rw dest-ip-address?  inet:ip-address
|           | |   +--rw next-header?      uint8
|           | |   +--rw traffic-class?    uint8
|           | |   +--rw flow-label?       inet:ipv6-flow-label
|           | |   +--rw source-port?      inet:port-number
|           | |   +--rw destination-port?  inet:port-number
|           | +--rw outgoing-interface?  if:interface-ref
|         +--:(next-hop-list)
|           +--rw next-hop-list
|             +--rw next-hop* [hop-index]
|               +--rw hop-index          uint8
|               +--rw (header-type)?

```



```
| | +---:(detnet-mpls-header)
| |   +--rw mpls-label-stack
| |     +--rw entry* [id]
| |       +--rw id          uint8
| |       +--rw label?      rt-types:mpls-
label
| |         +--rw ttl?      uint8
| |         +--rw traffic-class? uint8
| |   +---:(detnet-ip-header)
| |     +--rw src-ip-address?  inet:ip-address
| |     +--rw dest-ip-address? inet:ip-address
| |     +--rw next-header?    uint8
| |     +--rw traffic-class?  uint8
| |     +--rw flow-label?     inet:ipv6-flow-
label
| |       +--rw source-port?   inet:port-number
| |       +--rw destination-port? inet:port-number
| |   +--rw outgoing-interface? if:interface-ref
+--rw sub-network
  +--rw sub-network-list* [name]
    +--rw name string
```

```
grouping ip-header:
```

```

+--rw src-ip-address?    inet:ip-address
+--rw dest-ip-address?   inet:ip-address
+--rw next-header?       uint8
+--rw traffic-class?     uint8
+--rw flow-label?        inet:ipv6-flow-label
+--rw source-port?       inet:port-number
+--rw destination-port?  inet:port-number

```

grouping 12-header:

```
+--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
```

```
grouping destination-ip-port-identification:
```

```

+--rw destination-port
  +--rw (port-range-or-operator)?
    +--:(range)
      | +--rw lower-port   inet:port-number
      | +--rw upper-port   inet:port-number
    +--:(operator)
      +--rw operator?      operator
      +--rw port            inet:port-number

```

grouping source-ip-port-identification:

```

+--rw source-port
+--rw (port-range-or-operator)?

```

```
+-:-(range)
| +-rw lower-port inet:port-number
```

```

        | +--rw upper-port  inet:port-number
        +--:(operator)
          +--rw operator?   operator
          +--rw port        inet:port-number
grouping ip-flow-identification:
  +--rw src-ip-prefix?     inet:ip-prefix
  +--rw dest-ip-prefix?    inet:ip-prefix
  +--rw next-header?       uint8
  +--rw traffic-class?     uint8
  +--rw flow-label?        inet:ipv6-flow-label
  +--rw source-port
    | +--rw (port-range-or-operator)?
    |   +--:(range)
    |     | +--rw lower-port  inet:port-number
    |     | +--rw upper-port  inet:port-number
    |     +--:(operator)
    |       +--rw operator?   operator
    |       +--rw port        inet:port-number
  +--rw destination-port
    | +--rw (port-range-or-operator)?
    |   +--:(range)
    |     | +--rw lower-port  inet:port-number
    |     | +--rw upper-port  inet:port-number
    |     +--:(operator)
    |       +--rw operator?   operator
    |       +--rw port        inet:port-number
  +--rw ipsec-spi?         ipsec-spi
grouping mpls-flow-identification:
  +--rw (label-space)?
    +--:(context-label-space)
      | +--rw mpls-label-stack
      |   +--rw entry* [id]
      |     +--rw id          uint8
      |     +--rw label?      rt-types:mpls-label
      |     +--rw ttl?        uint8
      |     +--rw traffic-class? uint8
    +--:(platform-label-space)
      +--rw label?          rt-types:mpls-label
grouping traffic-specification:
  +--rw traffic-specification
    +--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
grouping traffic-requirements:
  +--rw traffic-requirements
    +--rw min-bandwidth?      uint64

```



```

    +--rw max-latency?                uint32
    +--rw max-latency-variation?      uint32
    +--rw max-loss?                   uint8
    +--rw max-consecutive-loss-tolerance? uint32
    +--rw max-misordering?            uint32
grouping data-flow-spec:
  +--rw (data-flow-type)?
    +--:(tsn-app-flow)
    | +--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
    +--:(ip-app-flow)
    | +--rw src-ip-prefix?            inet:ip-prefix
    | +--rw dest-ip-prefix?           inet:ip-prefix
    | +--rw next-header?              uint8
    | +--rw traffic-class?            uint8
    | +--rw flow-label?               inet:ipv6-flow-label
    | +--rw source-port
    | | +--rw (port-range-or-operator)?
    | | | +--:(range)
    | | | | +--rw lower-port  inet:port-number
    | | | | +--rw upper-port  inet:port-number
    | | | +--:(operator)
    | | | | +--rw operator?   operator
    | | | | +--rw port        inet:port-number
    | +--rw destination-port
    | | +--rw (port-range-or-operator)?
    | | | +--:(range)
    | | | | +--rw lower-port  inet:port-number
    | | | | +--rw upper-port  inet:port-number
    | | | +--:(operator)
    | | | | +--rw operator?   operator
    | | | | +--rw port        inet:port-number
    | +--rw ipsec-spi?               ipsec-spi
    +--:(mpls-app-flow)
    +--rw (label-space)?
    +--:(context-label-space)
    | +--rw mpls-label-stack
    | | +--rw entry* [id]
    | | | +--rw id                uint8
    | | | +--rw label?            rt-types:mpls-label
    | | | +--rw ttl?              uint8
    | | | +--rw traffic-class?    uint8
    +--:(platform-label-space)
    +--rw label?                    rt-types:mpls-label
grouping detnet-flow-spec:

```



```

+--rw (detnet-flow-type)?
  +--:(ip-detnet-flow)
    | +--rw src-ip-prefix?    inet:ip-prefix
    | +--rw dest-ip-prefix?   inet:ip-prefix
    | +--rw next-header?      uint8
    | +--rw traffic-class?    uint8
    | +--rw flow-label?       inet:ipv6-flow-label
    | +--rw source-port
    | | +--rw (port-range-or-operator)?
    | |   +--:(range)
    | |     | +--rw lower-port  inet:port-number
    | |     | +--rw upper-port  inet:port-number
    | |     +--:(operator)
    | |       +--rw operator?   operator
    | |       +--rw port        inet:port-number
    | +--rw destination-port
    | | +--rw (port-range-or-operator)?
    | |   +--:(range)
    | |     | +--rw lower-port  inet:port-number
    | |     | +--rw upper-port  inet:port-number
    | |     +--:(operator)
    | |       +--rw operator?   operator
    | |       +--rw port        inet:port-number
    | +--rw ipsec-spi?        ipsec-spi
  +--:(mpls-detnet-flow)
    +--rw (label-space)?
      +--:(context-label-space)
        | +--rw mpls-label-stack
        |   +--rw entry* [id]
        |     +--rw id          uint8
        |     +--rw label?      rt-types:mpls-label
        |     +--rw ttl?        uint8
        |     +--rw traffic-class? uint8
      +--:(platform-label-space)
        +--rw label?          rt-types:mpls-label

grouping app-flows-ref:
  +--rw app-flow* app-flow-ref

grouping service-sub-layer-ref:
  +--rw service-sub-layer* service-sub-layer-ref

grouping forwarding-sub-layer-ref:
  +--rw forwarding-sub-layer* forwarding-sub-layer-ref

grouping detnet-header:
  +--rw (header-type)?
    +--:(detnet-mpls-header)
      | +--rw mpls-label-stack
      |   +--rw entry* [id]
      |     +--rw id          uint8
      |     +--rw label?      rt-types:mpls-label

```



```

|         +--rw ttl?                uint8
|         +--rw traffic-class?      uint8
+--:(detnet-ip-header)
  +--rw src-ip-address?             inet:ip-address
  +--rw dest-ip-address?            inet:ip-address
  +--rw next-header?                 uint8
  +--rw traffic-class?               uint8
  +--rw flow-label?                  inet:ipv6-flow-label
  +--rw source-port?                 inet:port-number
  +--rw destination-port?            inet:port-number
grouping detnet-app-next-hop-content:
  +--rw (next-hop-options)
    +--:(simple-next-hop)
      | +--rw mpls-header
      | | +--rw mpls-label-stack
      | |   +--rw entry* [id]
      | |     +--rw id                uint8
      | |     +--rw label?             rt-types:mpls-label
      | |     +--rw ttl?               uint8
      | |     +--rw traffic-class?    uint8
      | +--rw outgoing-interface?     if:interface-ref
    +--:(next-hop-list)
      +--rw next-hop-list
        +--rw next-hop* [hop-index]
          +--rw hop-index              uint8
          +--rw mpls-header
            | +--rw mpls-label-stack
            |   +--rw entry* [id]
            |     +--rw id              uint8
            |     +--rw label?           rt-types:mpls-label
            |     +--rw ttl?             uint8
            |     +--rw traffic-class?  uint8
            +--rw outgoing-interface?   if:interface-ref
grouping detnet-forwarding-next-hop-content:
  +--rw (next-hop-options)
    +--:(simple-next-hop)
      | +--rw (header-type)?
      | | +--:(detnet-mpls-header)
      | | | +--rw mpls-label-stack
      | | |   +--rw entry* [id]
      | | |     +--rw id                uint8
      | | |     +--rw label?             rt-types:mpls-label
      | | |     +--rw ttl?               uint8
      | | |     +--rw traffic-class?    uint8
      | | +--:(detnet-ip-header)
      | |   +--rw src-ip-address?        inet:ip-address
      | |   +--rw dest-ip-address?       inet:ip-address
      | |   +--rw next-header?           uint8

```



```

| |      +--rw traffic-class?      uint8
| |      +--rw flow-label?         inet:ipv6-flow-label
| |      +--rw source-port?        inet:port-number
| |      +--rw destination-port?   inet:port-number
| +--rw outgoing-interface? if:interface-ref
+--:(next-hop-list)
  +--rw next-hop-list
    +--rw next-hop* [hop-index]
      +--rw hop-index              uint8
      +--rw (header-type)?
        | +--:(detnet-mpls-header)
        | | +--rw mpls-label-stack
        | |   +--rw entry* [id]
        | |     +--rw id            uint8
        | |     +--rw label?        rt-types:mpls-label
        | |     +--rw ttl?          uint8
        | |     +--rw traffic-class? uint8
        | +--:(detnet-ip-header)
        |   +--rw src-ip-address?   inet:ip-address
        |   +--rw dest-ip-address?  inet:ip-address
        |   +--rw next-header?      uint8
        |   +--rw traffic-class?    uint8
        |   +--rw flow-label?       inet:ipv6-flow-label
        |   +--rw source-port?      inet:port-number
        |   +--rw destination-port? inet:port-number
        +--rw outgoing-interface? if:interface-ref

```

6. DetNet Configuration YANG Model

<CODE BEGINS>

```

module ietf-detnet-config-modify {
  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-types";
  }

```



```
}

import ietf-routing {
  prefix "rt";
}

import ietf-packet-fields {
  prefix "packet-fields";
}
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 2020-03-04 {
  description "initial revision";
  reference "RFC XXXX: draft-ietf-detnet-yang-02";
```



```
}

typedef app-flow-ref {
  type leafref {
    path "/ietf-detnet:app-flows"
      + "/ietf-detnet:app-flow"
      + "/ietf-detnet:name";
  }
}

typedef service-sub-layer-ref {
  type leafref {
    path "/ietf-detnet:service-sub-layer"
      + "/ietf-detnet:service-sub-layer-list"
      + "/ietf-detnet:name";
  }
}

typedef forwarding-sub-layer-ref {
  type leafref {
    path "/ietf-detnet:forwarding-sub-layer"
      + "/ietf-detnet:forwarding-sub-layer-list"
      + "/ietf-detnet:name";
  }
}

typedef sub-network-ref {
  type leafref {
    path "/ietf-detnet:sub-network"
      + "/ietf-detnet:sub-network-list"
      + "/ietf-detnet:name";
  }
}

typedef ipsec-spi {
  type uint32 {
    range "1..max";
  }
  description
    "SPI";
}

typedef service-operation-type {
  type enumeration {
    enum service-initiation {
      description
        "Operation for DetNet service sub-layer encapsulation";
    }
  }
}
```



```
enum service-termination {
  description
    "Operation for DetNet service sub-layer decapsulation";
}
enum service-relay {
  description
    "Operation for DetNet service sub-layer swap";
}
enum non-detnet {
  description
    "No operation for DetNet service sub-layer";
}
}

typedef forwarding-operations-type {
  type enumeration {
    enum forward {
      description
        "Operation forward to next-hop";
    }
    enum impose-and-forward {
      description
        "Operation impose outgoing label(s) and forward to
        next-hop";
    }
    enum pop-and-forward {
      description
        "Operation pop incoming label and forward to next-hop";
    }
    enum pop-impose-and-forward {
      description
        "Operation pop incoming label, impose one or more
        outgoing label(s) and forward to next-hop";
    }
    enum swap-and-forward {
      description
        "Operation swap incoming label, with outgoing label and
        forward to next-hop";
    }
    enum pop-and-lookup {
      description
        "Operation pop incoming label and perform a lookup";
    }
  }
  description "MPLS operations types";
}
```



```
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";
    }
}

typedef sequence-number-generation-type {
    type enumeration {
        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";
        }
    }
}

typedef sequence-number-field {
    type enumeration {
        enum zero-sn {
            description "there is no DetNet sequence number field.";
        }
        enum short-sn {
            description "there is 16bit DetNet sequence number field";
            value 16;
        }
        enum long-sn {
            description "there is 28bit DetNet sequence number field";
            value 28;
        }
    }
}

grouping ip-header {
    description
        "The IPv4/IPv6 packet header information";
    leaf src-ip-address {
        type inet:ip-address;
        description
            "The source IP address of the header";
    }
    leaf dest-ip-address {
        type inet:ip-address;
        description
            "The destination IP address of the header";
    }
    leaf next-header {
```



```
    type uint8;
    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";
}
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 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 pcps {
        type uint8;
        description
            "The priority value of the ethernet header";
    }
}
grouping destination-ip-port-identification {
    description
        "The TCP/UDP port(source/destination) identification information";
    container destination-port {
        uses packet-fields:port-range-or-operator;
    }
}

grouping source-ip-port-identification {
    description
        "The TCP/UDP port(source/destination) identification information";
    container source-port {
        uses packet-fields:port-range-or-operator;
    }
}

grouping ip-flow-identification {
    description
        "The IPv4/IPv6 packet header identification information";
    leaf src-ip-prefix {
        type inet:ip-prefix;
        description
            "The source IP address of the header";
    }
    leaf dest-ip-prefix {
        type inet:ip-prefix;
        description
            "The destination IP address of the header";
    }
    leaf next-header {
        type uint8;
        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";
  }
  uses source-ip-port-identification;
  uses destination-ip-port-identification;
  leaf ipsec-spi {
    type ipsec-spi;
    description "Security parameter index of SA entry";
  }
}

grouping mpls-flow-identification {
  description
  "The MPLS packet header identification information";
  choice label-space {
    description "";
    case context-label-space {
      uses rt-types:mpls-label-stack;
    }
    case platform-label-space {
      leaf label {
        type rt-types:mpls-label;
      }
    }
  }
}

grouping traffic-specification {
  container 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.";
    }
}

grouping traffic-requirements {
    container traffic-requirements {
        description
            "FlowRequirements: defines the attributes of the App-flow
             regarding bandwidth, latency, latency variation, loss, and
             misordering tolerance.";
        leaf min-bandwidth {
            type uint64;
            description
                "MinBandwidth is the minimum bandwidth that has to be
                 guaranteed for the DetNet service. MinBandwidth is
                 specified in octets per second.";
        }
        leaf max-latency {
            type uint32;
            description
                "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";
        }
        leaf max-latency-variation {
            type uint32;
            description
                "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.";
    }
    leaf max-loss {
        type uint8;
        description
            "MaxLoss defines the maximum Packet Loss Ratio (PLR) parameter
            for the DetNet service between the Ingress and Egress(es) of
            the DetNet domain.";
    }
    leaf max-consecutive-loss-tolerance {
        type uint32;
        description
            "Some applications have special loss requirement, such as
            MaxConsecutiveLossTolerance. The maximum consecutive loss
            tolerance parameter describes the maximum number of
            consecutive packets whose loss can be tolerated. The maximum
            consecutive loss tolerance can be measured for example based
            on sequence number";
    }
    leaf max-misordering {
        type uint32;
        description
            "MaxMisordering describes the tolerable maximum number of
            packets that can be received out of order. The maximum
            allowed misordering can be measured for example based on
            sequence number. The value zero for the maximum allowed
            misordering indicates that in order delivery is required,
            misordering cannot be tolerated.";
    }
}

grouping data-flow-spec {
    description "app-flow identification";
    choice data-flow-type {
        case tsn-app-flow {
            uses l2-header;
        }
        case ip-app-flow {
            uses ip-flow-identification;
        }
        case mpls-app-flow {
            uses mpls-flow-identification;
        }
    }
}

grouping detnet-flow-spec {
    description "detnet-flow identificatioin";
```



```
choice detnet-flow-type {
  case ip-detnet-flow {
    uses ip-flow-identification;
  }
  case mpls-detnet-flow {
    uses mpls-flow-identification;
  }
}

grouping app-flows-ref {
  description "incoming or outgoing app-flow reference group";
  leaf-list app-flow {
    type app-flow-ref;
    description "List of ingress or egress app-flows";
  }
}

grouping service-sub-layer-ref {
  description "incoming or outgoing service sub-layer reference group";
  leaf-list service-sub-layer {
    type service-sub-layer-ref;
    description
      "List of incoming or outgoing service sub-layer
       that has to aggregate or disaggregate";
  }
}

grouping forwarding-sub-layer-ref {
  description "incoming or outgoing forwarding sub-layer reference group";
  leaf-list forwarding-sub-layer {
    type forwarding-sub-layer-ref;
    description
      "List of incoming or outgoing forwarding sub-layer
       that has to aggregate or disaggregate";
  }
}

grouping detnet-header {
  description "DetNet header info for DetNet encapsulation or swap";
  choice header-type {
    case detnet-mpls-header {
      description
        "MPLS label stack for DetNet MPLS encapsulation or forwarding";
      uses rt-types:mpls-label-stack;
    }
    case detnet-ip-header {
      description

```



```
    "IPv4/IPv6 packet header for DetNet IP encapsulation";
    uses ip-header;
  }
}
}

grouping detnet-app-next-hop-content {
  description
    "Generic parameters of DetNet next hops.";
  choice next-hop-options {
    mandatory true;
    description
      "Options for next hops.
      It is expected that further cases will be added through
      augments from other modules, e.g., for recursive
      next hops.";
    case simple-next-hop {
      description
        "This case represents a simple next hop consisting of the
        next-hop address and/or outgoing interface.
        Modules for address families MUST augment this case with a
        leaf containing a next-hop address of that address
        family.";
      container mpls-header {
        description
          "MPLS label stack for DetNet MPLS encapsulation or forwarding";
        uses rt-types:mpls-label-stack;
      }
      leaf outgoing-interface {
        type if:interface-ref;
      }
    }
  }
  case next-hop-list {
    container next-hop-list {
      description
        "Container for multiple next hops.";
      list next-hop {
        key "hop-index";
        description
          "An entry in a next-hop list.

          Modules for address families MUST augment this list
          with a leaf containing a next-hop address of that
          address family.";
        leaf hop-index {
          type uint8;
          description "";
        }
      }
    }
  }
}
```



```
        container mpls-header {
            description
                "MPLS label stack for DetNet MPLS encapsulation or forwarding";
            uses rt-types:mpls-label-stack;
        }
        leaf outgoing-interface {
            type if:interface-ref;
        }
    }
}
}
```

```
grouping detnet-forwarding-next-hop-content {
    description
        "Generic parameters of DetNet next hops.";
    choice next-hop-options {
        mandatory true;
        description
            "Options for next hops.
            It is expected that further cases will be added through
            augments from other modules, e.g., for recursive
            next hops.";
        case simple-next-hop {
            description
                "This case represents a simple next hop consisting of the
                next-hop address and/or outgoing interface.
                Modules for address families MUST augment this case with a
                leaf containing a next-hop address of that address
                family.";
            uses detnet-header;
            leaf outgoing-interface {
                type if:interface-ref;
            }
        }
        case next-hop-list {
            container next-hop-list {
                description
                    "Container for multiple next hops.";
                list next-hop {
                    key "hop-index";
                    description
                        "An entry in a next-hop list.

                        Modules for address families MUST augment this list
                        with a leaf containing a next-hop address of that
                        address family.";
                }
            }
        }
    }
}
```



```
        leaf hop-index {
            type uint8;
            description "";
        }
        uses detnet-header;
        leaf outgoing-interface {
            type if:interface-ref;
        }
    }
}
}
}

container app-flows {
    description
    "The DetNet app-flow configuration";
    list app-flow {
        key "name";
        description "";
        leaf name {
            type "string";
            description "The name to identify the DetNet app-flow";
        }
        leaf app-id {
            type uint16;
            config false;
            description
            "The DetNet Application ID";
        }
        leaf app-flow-bidir-congruent {
            type boolean;
            description
            "Defines the data path requirement of the App-flow whether
            it must share the same data path and physical path
            for both directions through the network,
            e.g., to provide congruent paths in the two directions.";
        }
        uses traffic-requirements;
        uses traffic-specification;
        container in-segment {
            description "app-flow identification";
            leaf interface {
                type if:interface-ref;
                description "";
            }
            uses data-flow-spec;
        }
    }
}
```



```
    container out-segment {
      description
        "ingress proxy that incoming app-flow map to detnet-flow
        and egress proxy that next-hop for outgoing app-flow";
      choice outgoing-options {
        mandatory true;
        description
          "";
        case ingress-proxy {
          leaf service-sub-layer {
            type service-sub-layer-ref;
          }
        }
        case egress {
          uses detnet-app-next-hop-content;
        }
      }
    }
  }
}

container service-sub-layer {
  description "The DetNet service sub-layer configuration";
  list service-sub-layer-list {
    key "name";
    description "";
    leaf name {
      type string;
      description "The name of the DetNet service sub-layer";
    }
    leaf service-id {
      type uint16;
      config false;
      description
        "The DetNet service ID";
    }
    leaf service-rank {
      type uint8;
      description
        "The DetNet rank for this service";
    }
  }
  uses traffic-requirements;
  uses traffic-specification;
  container service-protection {
    leaf service-protection-type {
      type service-protection-type;
      description
        "The DetNet service protection type
        such as PRF, PEF, PEOF, PERF, and PEORF";
    }
  }
}
```



```
    }
    leaf sequence-number-length {
      type sequence-number-field;
      description
        "sequence number field can choice 0 bit, 16bit, 28 bit field";
    }
  }
  leaf service-operation-type {
    type service-operation-type;
  }
  container in-segment {
    description
      "The DetNet service sub-layer inbound configuration.
      This should not be used if this service operation is initiation";
    choice incoming-options {
      mandatory true;
      description
        "";
      case ingress-proxy {
        uses app-flows-ref;
      }
      case detnet-service-identification {
        uses detnet-flow-spec;
      }
    }
  }
  container out-segment {
    description
      "The DetNet service sub-layer outbound configuration.
      This should not be used
      if this service operation is termination";
    choice outgoing-options {
      mandatory true;
      description
        "";
      case egress-proxy {
        uses app-flows-ref;
      }
      case detnet-service-output {
        //uses detnet-service-next-hop-content;
        list service-output-list {
          key "service-output-index";
          leaf service-output-index {
            type uint8;
          }
        }
        uses detnet-header;
        list next-layer{
          key "index";
```



```
leaf interface {
```

```

        type if:interface-ref;
        description "";
    }
    uses detnet-flow-spec;
}
}
}
container out-segment {
    description
        "The DetNet forwarding sub-layer outbound configuration. This should
not be used
        if this forwarding operation is pop-and-lookup";
    choice outgoing-options {
        mandatory true;
        description
            "";
        case service-operation {
            uses service-sub-layer-ref;
        }
        case detnet-forwarding-output {
            uses detnet-forwarding-next-hop-content;
        }
    }
}
}
}
}
container sub-network {
    description "";
    list sub-network-list {
        key "name";
        description "";
        leaf name {
            type string;
            description "";
        }
    }
}
}
}
<CODE ENDS>

```

7. An alternative DetNet YANG model

This is a model that is organized by the concepts in the flow model. In order to align with the principles in the flow document a ground up model was built. In order to illustrate this YANG model some example test configurations are used to validate the model. Building the DetNet YANG model is not that easy because the number of permutations of IP/MPLS data planes with and without aggregation at

each layer is hard to keep straight. This model tries to follow the

current DetNet data plane documents and supported data planes. The examples do not illustrate aggregation but the models are build for several levels of aggregation.

7.1. Conventions

This YANG model consists of three main sections: application, service sub-layer and forwarding sub-layer, that use a number of common building blocks. Many of the building blocks are also the same as the first model presented in this document. To minimize duplication, leaf-refs between the sections are utilized. With a single leaf-ref between an application and a service sub-layer the backend code can relate the two instances. The backend code can also populate a read only instance of a leaf-ref for the operational view so leaf-refs only need to be configured once for both objects. Early versions of the model used leaf-refs configure mainly outside the service sub-layer, however this turned out being awkward because the in the current model of the service sub-layer presented here it is binding a service id to the leaf-ref. Therefore the leaf-ref for an application and a forwarding sub-layer is more naturally configured from the service sub-layer.

Another point about the model is each component has two unidirectional parts. Applications would typically contain both directions and become a complete bidirectional entity containing both directions of a flow. Service sub-layers components can be configured as unidirectional with only one direction being configured. Forwarding sub-layer components are unidirectional. Leaf-refs are used for each direction even though there are cases where a single leaf-ref could work in a bidirectional model. DetNet Services require configuration of each unidirectional flow and have configuration options based on the roles they play and the types of traffic mappings.

DetNet Edges have application, service and forwarding configuration. DetNet Relays only use DetNet services sub-layer and forwarding sub-layer configuration. DetNet Transit nodes may only have forwarding sub-layer configuration. The behavior is DetNet service dependent, such that a a physical node may be an edge node for some flows, it may be a relay node for some flows and it may be a transit node for some flows. The full YANG model is defined and only the relevant aspects are configured at each stage.

7.1.1. Aggregation

Aggregation may be configured at each of the instances. Aggregation is data plane specific. DetNet Service Sub-layer MPLS aggregation for example, from application to service adds a per Application

Service sub-layer label. If a DetNet service is aggregated to another DetNet service there are a couple ways this can be accomplished.

A service sub-layer can play a role similar to an application to another service allowing aggregation of services. A service sub-layer can appear as relay service peering with that service. Relay functions are dependent in the traffic type. Aggregation can be achieved by encapsulation (MPLS) or by wild cards (IP) using the native IP header. IP headers that are used in place aggregate by allowing a broader address range (wild cards) or port range or DSCP filter. These aggregating flows also require the reverse operation to disaggregate the traffic at the edge of the DetNet service.

There are many ways to achieve aggregation and this can complicate the YANG design to some degree. The current models support aggregation but the configuration examples provided are illustrating no aggregation for this version of the draft.

7.2. Alternative DetNet Configuration YANG

module: ietf-detnet-config

```

+--rw detnet
  +--rw applications
    | +--rw app-list* [name]
    | | +--rw name string
    | | +--ro app-id? uint16
    | | +--rw app-flow-bidir-congruent? boolean
    | | +--ro service-outbound? service-ref
    | | +--ro service-inbound? service-ref
    | | +--rw traffic-requirements
    | | | +--rw min-bandwidth? uint64
    | | | +--rw max-latency? uint32
    | | | +--rw max-latency-variation? uint32
    | | | +--rw max-loss? percent
    | | | +--rw max-consecutive-loss-tolerance? uint32
    | | | +--rw max-misordering? uint32
    | | +--rw traffic-specification
    | | | +--rw interval? uint32
    | | | +--rw max-packets-per-interval? uint64
    | | | +--rw max-payload-size? uint32
    | +--rw app-ingress* [name]
    | | +--rw name string
    | | +--ro app-flow-status? identityref
    | | +--rw incoming-interface* if:interface-ref
    | | +--rw (detnet-forwarding-type)?
    | | | +--:(ethernet)
    | | | | +--rw ethernet

```



```

| | | +--rw placeholder? string
| | | +---:(ip)
| | | | +--rw ip
| | | | | +--rw source inet:ip-prefix
| | | | | +--rw destination inet:ip-prefix
| | | | | +--rw protocol-next uint8
| | | | | +--rw dscp* uint8
| | | | | +--rw ipv6-flow-label? inet:ipv6-flow-label
| | | | +--rw (l4)?
| | | | | +---:(tcp)
| | | | | | +--rw tcp
| | | | | | | +--rw source-port
| | | | | | | | +--rw (source-port)?
| | | | | | | | | +---:(range-or-operator)
| | | | | | | | | +--rw range-or-operator
| | | | | | | | | +--rw (port-range-or-operator)?
| | | | | | | | | +---:(range)
| | | | | | | | | | +--rw lower-port inet:
port-number
| | | | | | | | | | +--rw upper-port inet:
port-number
| | | | | | | | | +---:(operator)
| | | | | | | | | +--rw operator? opera
tor
| | | | | | | | | +--rw port inet:
port-number
| | | | | | | +--rw destination-port
| | | | | | | +--rw (destination-port)?
| | | | | | | +---:(range-or-operator)
| | | | | | | +--rw range-or-operator
| | | | | | | +--rw (port-range-or-operator)?
| | | | | | | +---:(range)
| | | | | | | | +--rw lower-port inet:
port-number
| | | | | | | | +--rw upper-port inet:
port-number
| | | | | | | +---:(operator)
| | | | | | | +--rw operator? opera
tor
| | | | | | | +--rw port inet:
port-number
| | | | | +---:(udp)
| | | | | | +--rw udp
| | | | | | | +--rw source-port
| | | | | | | | +--rw (source-port)?
| | | | | | | | +---:(range-or-operator)
| | | | | | | | +--rw range-or-operator
| | | | | | | | +--rw (port-range-or-operator)?

```


						+++:(range)		
port-number							+++rw lower-port	inet:
port-number							+++rw upper-port	inet:
						+++:(operator)		
tor							+++rw operator?	opera
port-number							+++rw port	inet:
						+++rw destination-port		
						+++rw (destination-port)?		
						+++:(range-or-operator)		
						+++rw range-or-operator		
						+++rw (port-range-or-operator)?		
						+++:(range)		
port-number							+++rw lower-port	inet:
port-number							+++rw upper-port	inet:
						+++:(operator)		
tor							+++rw operator?	opera
port-number							+++rw port	inet:
					+++:(ipsec)			
					+++rw ipsec			
					+++rw ipsec-spi?	ipsec-spi		
				+++:(mpls)				
				+++rw mpls				
				+++rw (label-space)?				
				+++:(interface)				
					+++rw interface			
					+++rw mpls-label-stack			
					+++rw entry* [id]			
					+++rw id		uint8	
label					+++rw label?		rt-types:mpls-	
					+++rw ttl?		uint8	
					+++rw traffic-class?		uint8	
				+++:(platform)				
				+++rw platform				
				+++rw mpls-label-stack				
				+++rw entry* [id]				
				+++rw id			uint8	
label				+++rw label?			rt-types:mpls-	
					+++rw ttl?		uint8	


```

| |                                     +--rw traffic-class?   uint8
| +--rw app-egress* [name]
|   +--rw name                string
|   +--ro app-flow-status?    identityref
|   +--rw next-hop
|     +--rw (application-type)?
|       +--:(ethernet)
|         | +--rw ethernet
|         |   +--rw ethernet-place-holder?  string
|       +--:(ip-mps)
|         +--rw ip-mps
|           +--rw (next-hop-options)
|             +--:(simple-next-hop)
|               | +--rw simple-next-hop
|               |   +--rw outgoing-interface?  if:interface
- ref
|               | +--rw mpls-label-stack
|               |   +--rw entry* [id]
|               |     +--rw id                uint8
|               |     +--rw label?           rt-types:mp
ls-label
|               |   +--rw ttl?                uint8
|               |   +--rw traffic-class?      uint8
|               +--:(special-next-hop)
|                 | +--rw special-next-hop
|                 |   +--rw special-next-hop?  enumeration
|               +--:(next-hop)
|                 +--rw next-hop* [hop-index]
|                   +--rw hop-index            uint8
|                   +--rw outgoing-interface?  if:interface
- ref
|                   +--rw index?                string
|                   +--rw backup-index?          string
|                   +--rw loadshare?             uint16
|                   +--rw role?                  nhlfe-role
|                   +--rw mpls-label-stack
|                     +--rw entry* [id]
|                       +--rw id                uint8
|                       +--rw label?           rt-types:mp
ls-label
|                       +--rw ttl?                uint8
|                       +--rw traffic-class?      uint8
+--rw services
| +--rw service-list* [name]
|   +--rw name                string
|   +--ro service-id?         uint16
|   +--rw service-rank?       uint8
|   +--rw outbound

```



```

|      | +--ro forward-sublayer-output* fwd-sub-layer-output-ref
|      | +--rw service-protection-type? identityref
|      | +--rw sequence_number_length? uint8
|      | +--rw application-input* [name]
|      | |   +--rw name application-ref
|      | |   +--rw (detnet-forwarding-type)?
|      | |   +--:(ip)
|      | |   |   +--rw ip
|      | |   |   +--rw source inet:ip-prefix
|      | |   |   +--rw destination inet:ip-prefix
|      | |   |   +--rw protocol-next uint8
|      | |   |   +--rw dscp* uint8
|      | |   |   +--rw ipv6-flow-label? inet:ipv6-flow-label
|      | |   |   +--rw (l4)?
|      | |   |   +--:(tcp)
|      | |   |   |   +--rw tcp
|      | |   |   |   +--rw source-port
|      | |   |   |   |   +--rw (source-port)?
|      | |   |   |   |   +--:(range-or-operator)
|      | |   |   |   |   +--rw range-or-operator
|      | |   |   |   |   +--rw (port-range-or-oper
ator)?
|      | |   |   |   |   +--:(range)
|      | |   |   |   |   |   +--rw lower-port
inet:port-number
|      | |   |   |   |   |   +--rw upper-port
inet:port-number
|      | |   |   |   |   |   +--:(operator)
|      | |   |   |   |   |   +--rw operator?
operator
|      | |   |   |   |   |   +--rw port
inet:port-number
|      | |   |   |   |   |   +--rw destination-port
|      | |   |   |   |   |   +--rw (destination-port)?
|      | |   |   |   |   |   +--:(range-or-operator)
|      | |   |   |   |   |   +--rw range-or-operator
|      | |   |   |   |   |   +--rw (port-range-or-oper
ator)?
|      | |   |   |   |   |   +--:(range)
|      | |   |   |   |   |   |   +--rw lower-port
inet:port-number
|      | |   |   |   |   |   |   +--rw upper-port
inet:port-number
|      | |   |   |   |   |   |   +--:(operator)
|      | |   |   |   |   |   |   +--rw operator?
operator
|      | |   |   |   |   |   |   +--rw port
inet:port-number

```



```

|         |         |         |         +---:(udp)
|         |         |         |         +---rw udp
|         |         |         |         +---rw source-port
|         |         |         |         |   +---rw (source-port)?
|         |         |         |         |   +---:(range-or-operator)
|         |         |         |         |   +---rw range-or-operator
|         |         |         |         |   +---rw (port-range-or-oper
ator)?
|         |         |         |         |   +---:(range)
|         |         |         |         |   |   +---rw lower-port
inet:port-number
|         |         |         |         |   |   +---rw upper-port
inet:port-number
|         |         |         |         |   +---:(operator)
|         |         |         |         |   +---rw operator?
operator
|         |         |         |         |   +---rw port
inet:port-number
|         |         |         |         +---rw destination-port
|         |         |         |         +---rw (destination-port)?
|         |         |         |         +---:(range-or-operator)
|         |         |         |         +---rw range-or-operator
|         |         |         |         +---rw (port-range-or-oper
ator)?
|         |         |         |         +---:(range)
|         |         |         |         |   +---rw lower-port
inet:port-number
|         |         |         |         |   +---rw upper-port
inet:port-number
|         |         |         |         +---:(operator)
|         |         |         |         +---rw operator?
operator
|         |         |         |         +---rw port
inet:port-number
|         |         |         |         +---:(ipsec)
|         |         |         |         +---rw ipsec
|         |         |         |         +---rw ipsec-spi?   ipsec-spi
|         |         |         +---:(mpls-service-label)
|         |         |         +---rw mpls-service-label?   rt-types:mpls-label-g
eneral-use
|         |   +---rw prev-relay-services* [name]
|         |   |   +---rw name                               service-ref
|         |   |   +---rw (detnet-forwarding-type)?
|         |   |   +---:(ip)
|         |   |   |   +---rw ip
|         |   |   |   +---rw source                       inet:ip-prefix
|         |   |   |   +---rw destination                 inet:ip-prefix
|         |   |   |   +---rw protocol-next               uint8

```


				+--rw dscp*	uint8
				+--rw ipv6-flow-label?	inet:ipv6-flow-label
				+--rw (l4)?	
				+--:(tcp)	
				+--rw tcp	
				+--rw source-port	
				+--rw (source-port)?	
				+--:(range-or-operator)	
				+--rw range-or-operator	
				+--rw (port-range-or-oper	
ator)?					
				+--:(range)	
				+--rw lower-port	
inet:port-number					
				+--rw upper-port	
inet:port-number					
				+--:(operator)	
				+--rw operator?	
operator					
				+--rw port	
inet:port-number					
				+--rw destination-port	
				+--rw (destination-port)?	
				+--:(range-or-operator)	
				+--rw range-or-operator	
				+--rw (port-range-or-oper	
ator)?					
				+--:(range)	
				+--rw lower-port	
inet:port-number					
				+--rw upper-port	
inet:port-number					
				+--:(operator)	
				+--rw operator?	
operator					
				+--rw port	
inet:port-number					
				+--:(udp)	
				+--rw udp	
				+--rw source-port	
				+--rw (source-port)?	
				+--:(range-or-operator)	
				+--rw range-or-operator	
				+--rw (port-range-or-oper	
ator)?					
				+--:(range)	
				+--rw lower-port	
inet:port-number					


```

| | | | | | | | +--rw upper-port
inet:port-number
| | | | | | | | +--:(operator)
| | | | | | | | +--rw operator?
operator
| | | | | | | | +--rw port
inet:port-number
| | | | | | | | +--rw destination-port
| | | | | | | | +--rw (destination-port)?
| | | | | | | | +--:(range-or-operator)
| | | | | | | | +--rw range-or-operator
| | | | | | | | +--rw (port-range-or-oper
ator)?
| | | | | | | | +--:(range)
| | | | | | | | | +--rw lower-port
inet:port-number
| | | | | | | | | +--rw upper-port
inet:port-number
| | | | | | | | +--:(operator)
| | | | | | | | +--rw operator?
operator
| | | | | | | | +--rw port
inet:port-number
| | | | | | | | +--:(ipsec)
| | | | | | | | +--rw ipsec
| | | | | | | | +--rw ipsec-spi? ipsec-spi
| | | | | | | | +--:(mpls-service-label)
| | | | | | | | +--rw mpls-service-label? rt-types:mpls-label-g
eneral-use
| | +--rw traffic-requirements
| | | +--rw min-bandwidth? uint64
| | | +--rw max-latency? uint32
| | | +--rw max-latency-variation? uint32
| | | +--rw max-loss? percent
| | | +--rw max-consecutive-loss-tolerance? uint32
| | | +--rw max-misordering? uint32
| | +--rw traffic-specification
| | | +--rw interval? uint32
| | | +--rw max-packets-per-interval? uint64
| | | +--rw max-payload-size? uint32
| | +--rw inbound
| | | +--rw forward-sublayer-input* fwd-sub-layer-input-ref
| | | +--rw service-protection-type? identityref
| | | +--rw sequence_number_length? uint8
| | | +--rw application-output* [name]
| | | | +--rw name application-ref
| | | | +--rw (detnet-forwarding-type)?
| | | | +--:(ip)

```


			+-rw ip	
			+-rw source	inet:ip-prefix
			+-rw destination	inet:ip-prefix
			+-rw protocol-next	uint8
			+-rw dscp*	uint8
			+-rw ipv6-flow-label?	inet:ipv6-flow-label
			+-rw (l4)?	
			+-:(tcp)	
			+-rw tcp	
			+-rw source-port	
			+-rw (source-port)?	
			+-:(range-or-operator)	
			+-rw range-or-operator	
			+-rw (port-range-or-oper	
ator)?				
			+-:(range)	
			+-rw lower-port	
inet:port-number				
			+-rw upper-port	
inet:port-number				
			+-:(operator)	
			+-rw operator?	
operator				
			+-rw port	
inet:port-number				
			+-rw destination-port	
			+-rw (destination-port)?	
			+-:(range-or-operator)	
			+-rw range-or-operator	
			+-rw (port-range-or-oper	
ator)?				
			+-:(range)	
			+-rw lower-port	
inet:port-number				
			+-rw upper-port	
inet:port-number				
			+-:(operator)	
			+-rw operator?	
operator				
			+-rw port	
inet:port-number				
			+-:(udp)	
			+-rw udp	
			+-rw source-port	
			+-rw (source-port)?	
			+-:(range-or-operator)	
			+-rw range-or-operator	
			+-rw (port-range-or-oper	


```

ator)?
    |          |          |          |          |          +---:(range)
    |          |          |          |          |          |  +---rw lower-port
inet:port-number
    |          |          |          |          |          |  +---rw upper-port
inet:port-number
    |          |          |          |          |          +---:(operator)
    |          |          |          |          |          +---rw operator?
operator
    |          |          |          |          |          +---rw port
inet:port-number
    |          |          |          |          |          +---rw destination-port
    |          |          |          |          |          +---rw (destination-port)?
    |          |          |          |          |          +---:(range-or-operator)
    |          |          |          |          |          +---rw range-or-operator
    |          |          |          |          |          +---rw (port-range-or-oper
ator)?
    |          |          |          |          |          +---:(range)
    |          |          |          |          |          |  +---rw lower-port
inet:port-number
    |          |          |          |          |          |  +---rw upper-port
inet:port-number
    |          |          |          |          |          +---:(operator)
    |          |          |          |          |          +---rw operator?
operator
    |          |          |          |          |          +---rw port
inet:port-number
    |          |          |          |          |          +---:(ipsec)
    |          |          |          |          |          +---rw ipsec
    |          |          |          |          |          +---rw ipsec-spi?    ipsec-spi
    |          |          |          |          |          +---:(mpls-service-label)
    |          |          |          |          |          +---rw mpls-service-label?  rt-types:mpls-label-g
eneral-use
    |          +---rw next-relay-services* [name]
    |          +---rw name                      service-ref
    |          +---rw (detnet-forwarding-type)?
    |          +---:(ip)
    |          |  +---rw ip
    |          |      +---rw source              inet:ip-prefix
    |          |      +---rw destination          inet:ip-prefix
    |          |      +---rw protocol-next        uint8
    |          |      +---rw dscp*                 uint8
    |          |      +---rw ipv6-flow-label?     inet:ipv6-flow-label
    |          |      +---rw (l4)?
    |          |          +---:(tcp)
    |          |          |  +---rw tcp
    |          |          |      +---rw source-port
    |          |          |          +---rw (source-port)?

```


				+++:(range-or-operator)
				+++rw range-or-operator
				+++rw (port-range-or-oper
ator)?				
				+++:(range)
				+++rw lower-port
inet:port-number				
				+++rw upper-port
inet:port-number				
				+++:(operator)
				+++rw operator?
operator				
				+++rw port
inet:port-number				
				+++rw destination-port
				+++rw (destination-port)?
				+++:(range-or-operator)
				+++rw range-or-operator
				+++rw (port-range-or-oper
ator)?				
				+++:(range)
				+++rw lower-port
inet:port-number				
				+++rw upper-port
inet:port-number				
				+++:(operator)
				+++rw operator?
operator				
				+++rw port
inet:port-number				
				+++:(udp)
				+++rw udp
				+++rw source-port
				+++rw (source-port)?
				+++:(range-or-operator)
				+++rw range-or-operator
				+++rw (port-range-or-oper
ator)?				
				+++:(range)
				+++rw lower-port
inet:port-number				
				+++rw upper-port
inet:port-number				
				+++:(operator)
				+++rw operator?
operator				
				+++rw port
inet:port-number				


```

| | | | +--rw destination-port
| | | | +---rw (destination-port)?
| | | | +---:(range-or-operator)
| | | | +---rw range-or-operator
| | | | +---rw (port-range-or-oper
ator)?
| | | | +---:(range)
| | | | | +---rw lower-port
inet:port-number
| | | | | +---rw upper-port
inet:port-number
| | | | +---:(operator)
| | | | +---rw operator?
operator
| | | | +---rw port
inet:port-number
| | | | +---:(ipsec)
| | | | +---rw ipsec
| | | | +---rw ipsec-spi?    ipsec-spi
| | | | +---:(mpls-service-label)
| | | | +---rw mpls-service-label?    rt-types:mpls-label-g
eneral-use
+--rw fwd-sub-layer-input* [name]
| +--rw name                string
| +--rw incoming-interface*  if:interface-ref
| +--rw (connectors)?
| | +---:(service-sub-layers)
| | | +--ro service-sub-layers*    service-ref
| | +---:(forwarding-sub-layers)
| | | +--ro forwarding-sub-layers*  fwd-sub-layer-output-ref
+--rw (detnet-forwarding-type)?
| +---:(ip)
| | +--rw ip
| | | +--rw source                inet:ip-prefix
| | | +--rw destination            inet:ip-prefix
| | | +--rw protocol-next          uint8
| | | +--rw dscp*                  uint8
| | | +--rw ipv6-flow-label?       inet:ipv6-flow-label
| | | +--rw (l4)?
| | | +---:(tcp)
| | | | +--rw tcp
| | | | +--rw source-port
| | | | +--rw (source-port)?
| | | | +---:(range-or-operator)
| | | | +--rw range-or-operator
| | | | +--rw (port-range-or-operator)?
| | | | +---:(range)
| | | | | +--rw lower-port        inet:por

```


t-number						+-rw upper-port	inet:por
t-number					+-:(operator)		
					+-rw operator?	operator	
					+-rw port	inet:por	
t-number					+-rw destination-port		
					+-rw (destination-port)?		
					+-:(range-or-operator)		
					+-rw range-or-operator		
					+-rw (port-range-or-operator)?		
					+-:(range)		
					+-rw lower-port	inet:por	
t-number					+-rw upper-port	inet:por	
t-number					+-:(operator)		
					+-rw operator?	operator	
					+-rw port	inet:por	
t-number					+-:(udp)		
					+-rw udp		
					+-rw source-port		
					+-rw (source-port)?		
					+-:(range-or-operator)		
					+-rw range-or-operator		
					+-rw (port-range-or-operator)?		
					+-:(range)		
					+-rw lower-port	inet:por	
t-number					+-rw upper-port	inet:por	
t-number					+-:(operator)		
					+-rw operator?	operator	
					+-rw port	inet:por	
t-number					+-rw destination-port		
					+-rw (destination-port)?		
					+-:(range-or-operator)		
					+-rw range-or-operator		
					+-rw (port-range-or-operator)?		
					+-:(range)		
					+-rw lower-port	inet:por	
t-number					+-rw upper-port	inet:por	
t-number					+-:(operator)		


```

|         |         |         +-rw operator?    operator
|         |         |         +-rw port        inet:por
t-number
|         |         |         +--:(ipsec)
|         |         |         +-rw ipsec
|         |         |         +-rw ipsec-spi?    ipsec-spi
|         +--:(mpls)
|         +-rw mpls
|         +-rw (label-space)?
|         +--:(interface)
|         |         +-rw interface
|         |         +-rw mpls-label-stack
|         |         +-rw entry* [id]
|         |         +-rw id                    uint8
|         |         +-rw label?                rt-types:mpls-lab
el
|         |         +-rw ttl?                  uint8
|         |         +-rw traffic-class?        uint8
|         +--:(platform)
|         +-rw platform
|         +-rw mpls-label-stack
|         +-rw entry* [id]
|         +-rw id                    uint8
|         +-rw label?                rt-types:mpls-lab
el
|         +-rw ttl?                  uint8
|         +-rw traffic-class?        uint8
+--rw fwd-sub-layer-output* [name]
  +-rw name                          string
  +-rw connectors
  | +-rw outbound-service*           service-ref
  | +-rw prev-forward-sub-layer*     fwd-sub-layer-output-ref
+--rw traffic-requirements
  | +-rw min-bandwidth?              uint64
  | +-rw max-latency?                uint32
  | +-rw max-latency-variation?      uint32
  | +-rw max-loss?                   percent
  | +-rw max-consecutive-loss-tolerance? uint32
  | +-rw max-misordering?            uint32
+--rw traffic-specification
  | +-rw interval?                   uint32
  | +-rw max-packets-per-interval?   uint64
  | +-rw max-payload-size?           uint32
+--rw next-hop
  +-rw (next-hop-options)
  +--:(simple-next-hop)
  | +-rw simple-next-hop
  | +-rw outgoing-interface?         if:interface-ref

```



```

|      +--rw mpls-label-stack
|      +--rw entry* [id]
|          +--rw id                uint8
|          +--rw label?            rt-types:mpls-label
|          +--rw ttl?              uint8
|          +--rw traffic-class?    uint8
+---:(special-next-hop)
|   +--rw special-next-hop
|   +--rw special-next-hop?    enumeration
+---:(next-hop)
    +--rw next-hop* [hop-index]
        +--rw hop-index                uint8
        +--rw outgoing-interface?      if:interface-ref
        +--rw index?                   string
        +--rw backup-index?            string
        +--rw loadshare?               uint16
        +--rw role?                   nhlfe-role
        +--rw mpls-label-stack
            +--rw entry* [id]
                +--rw id                uint8
                +--rw label?            rt-types:mpls-label
                +--rw ttl?              uint8
                +--rw traffic-class?    uint8

```

7.3. Alternative DetNet Configuration YANG Model

```

<CODE BEGINS> file "ietf-detnet-config@2020-06-02.yang"
module ietf-detnet-config {
  namespace "urn:ietf:params:xml:ns:yang:ietf-detnet-config";
  prefix "ietf-detnet";

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

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

  import ietf-interfaces {
    prefix "if";
  }

  import ietf-routing {
    prefix "rt";
  }

  import ietf-mpls {

```



```
    prefix "mpls";
}

import ietf-packet-fields {
    prefix "packet-fields";
}

organization "IETF DetNet Working Group";

contact
    "    Editor: Don Fedyk dfedyk@labn.net ";
description
    "This YANG module describes the parameters needed
    for DetNet flow configuration and flow status reporting";

revision 2020-06-02 {
    description "initial revision";
    reference
        "RFC XXXX: draft-ietf-detnet-flow-information-model-06";
}

identity status {
    description
        "Base identity from which all application-status
        actions are derived";
}

identity none {
    base "status";
    description
        "Application no ingress/egress";
    reference
        "draft-ietf-detnet-flow-information-model-06 Section 5.8";
}

identity ready {
    base "status";
    description
        "Application ingress/egress ready";
    reference
        "draft-ietf-detnet-flow-information-model-06 Section 5.8";
}

identity failed {
    base "status";
    description
        "Application ingres/egresss failed";
    reference
```



```
    "draft-ietf-detnet-flow-information-model-06 Section 5.8";
}

identity out-of-service {
    base "status";
    description
        "Application Administratively blocked";
    reference
        "draft-ietf-detnet-flow-information-model-06 Section 5.8";
}

identity partial-failed {
    base "status";
    description
        "Application One or more Egress ready, and one or more Egress
        failed. The DetNet flow can be used if the Ingress is
        Ready.";
    reference
        "draft-ietf-detnet-flow-information-model-06 Section 5.8";
}

typedef ipsec-spi {
    type uint32 {
        range "1..max";
    }
    description
        "SPI";
}

typedef application-ref {
    type leafref {
        path "/ietf-detnet:detnet/ietf-detnet:applications" +
            "/ietf-detnet:app-list/ietf-detnet:name";
    }
    description
        "This type is used by applications ingress and egress to
        reference application.";
}

typedef application-ingress-ref {
    type leafref {
        path "/ietf-detnet:detnet/ietf-detnet:applications" +
            "/ietf-detnet:app-ingress/ietf-detnet:name";
    }
    description
        "This type is used by applications to reference ingress
        interfaces.";
}
```



```
typedef application-egress-ref {
  type leafref {
    path "/ietf-detnet:detnet/ietf-detnet:applications" +
      "/ietf-detnet:app-egress/ietf-detnet:name";
  }
  description
    "This type is used by applications to reference egress
    interfaces.";
}

typedef service-ref {
  type leafref {
    path "/ietf-detnet:detnet/ietf-detnet:services" +
      "/ietf-detnet:service-list/ietf-detnet:name";
  }
  description
    "This type is used by applications and forwarding sub-layers
    data models that need to reference a configured
    service-sub-layer.";
}

typedef fwd-sub-layer-input-ref {
  type leafref {
    path "/ietf-detnet:detnet/ietf-detnet:fwd-sub-layer-input"
      + "/ietf-detnet:name";
  }
  description
    "This type is used by applications to reference input
    interfaces.";
}

typedef fwd-sub-layer-output-ref {
  type leafref {
    path "/ietf-detnet:detnet/ietf-detnet:fwd-sub-layer-output"
      + "/ietf-detnet:name";
  }
  description
    "This type is used by applications to reference output
    interfaces.";
}

// Service protection
identity service-protection-type {
  description
    "Base identity from which specific
    interface types are derived.";
}
```



```
identity no-protection {
  base "service-protection-type";
  description
    "no service protection provided";
}

identity replication {
  base "service-protection-type";
  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";
}

identity elimination {
  base "service-protection-type";
  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.";
}

identity ordering {
  base "service-protection-type";
  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.";
}

identity elimination-ordering {
  base "service-protection-type";
  description
    "A combination of PEF and POF that can be implemented by an
    edge node, a relay node, or an end system.";
}

identity elimination-replication {
  base "service-protection-type";
  description
    "A combination of PEF and PRF that can be implemented by an
    edge node, a relay node, or an end system";
}
```



```
}

identity elimination-ordering-replication {
  base "service-protection-type";
  description
    "A combination of PEF, POF and PRF that can be implemented by
    an edge node, a relay node, or an end system";
}

typedef percent {
  type uint8 {
    range "0 .. 100";
  }
  description "Percentage";
}

grouping ip-ports {
  container source-port {
    choice source-port {
      container range-or-operator {
        uses packet-fields:port-range-or-operator;
        description
          "Source port definition from range or operator.";
      }
      description
        "Choice of source port definition using range/operator or a
        choice to support future 'case' statements, such as one
        enabling a group of source ports to be referenced.";
    }
    description
      "Source port definition.";
  }
  container destination-port {
    choice destination-port {
      container range-or-operator {
        uses packet-fields:port-range-or-operator;
        description
          "Destination port definition from range or operator.";
      }
      description
        "Choice of destination port definition using range/operator
        or a choice to support future 'case' statements, such as
        one enabling a group of source ports to be referenced.";
    }
    description
      "Destination port definition.";
  }
}
```



```
grouping ip-incoming {
  description
    "The IPv4 packet header identification information";
  leaf source {
    type inet:ip-prefix;
    mandatory true;
    description "IP source address prefix";
  }
  leaf destination {
    type inet:ip-prefix;
    description "IP destination address prefix";
    mandatory true;
  }
}
leaf protocol-next {
  type uint8;
  mandatory true;
  description
    "Internet Protocol number. Refers to the protocol of the
    payload. In IPv6, this field is known as 'next-header', and
    if extension headers are present, the protocol is present in
    the 'upper-layer' header.";
  reference
    "RFC 791: Internet Protocol
    RFC 8200: Internet Protocol, Version 6 (IPv6) Specification.";
}
leaf-list dscp {
  type uint8 {
    range "0 .. 64";
  }
  description
    "The DSCP field of the header";
}
leaf ipv6-flow-label {
  type inet:ipv6-flow-label;
  description
    "The DSCP field of the header";
}
choice l4 {
  container tcp {
    uses ip-ports;
  }
  container udp {
    uses ip-ports;
  }
  container ipsec {
    leaf ipsec-spi {
      type ipsec-spi;
      description
```



```
        "Security parameter index of SA entry.";
    }
}
} // End of ip-incoming

grouping traffic-specification {
    container 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-06 Section 5.5";
        leaf interval {
            type uint32;
            description
                "The period of time in which the traffic specification
                cannot be exceeded";
        }
        leaf max-packets-per-interval{
            type uint64;
            units "bits";
            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 in bytes that the source will
                transmit.";
        }
    }
}

grouping traffic-requirements {
    container traffic-requirements {
        description
            "FlowRequirements: defines the attributes of the App-flow
            regarding bandwidth, latency, latency variation, loss, and
            misordering tolerance.";
        reference
            "draft-ietf-detnet-flow-information-model-06 Section 5.9";
        leaf min-bandwidth {
            type uint64;
            description
```



```
    "MinBandwidth is the minimum bandwidth that has to be
    guaranteed for the DetNet service. MinBandwidth is
    specified in octets per second.";
}
leaf max-latency {
    type uint32;
    description
        "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";
}
leaf max-latency-variation {
    type uint32;
    description
        "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.";
}
leaf max-loss {
    type percent;
    description
        "MaxLoss defines the maximum Packet Loss Ratio (PLR)
        parameter for the DetNet service between the Ingress and
        Egress(es) of the DetNet domain.";
}
leaf max-consecutive-loss-tolerance {
    type uint32;
    description
        "Some applications have special loss requirement, such as
        MaxConsecutiveLossTolerance. The maximum consecutive loss
        tolerance parameter describes the maximum number of
        consecutive packets whose loss can be tolerated. The
        maximum consecutive loss tolerance can be measured for
        example based on sequence number";
}
leaf max-misordering {
    type uint32;
    description
        "MaxMisordering describes the tolerable maximum number of
        packets that can be received out of order. The maximum
        allowed misordering can be measured for example based on
        sequence number. The value zero for the maximum allowed
        misordering indicates that in order delivery is required,
        misordering cannot be tolerated.";
}
}
```



```
}
```

```
// Copied from ietf-routing
grouping next-hop-state-content {
  description
    "Generic state parameters of next hops.";
  choice next-hop-options {
    mandatory true;
    description
      "Options for next hops.

      It is expected that further cases will be added through
      augments from other modules, e.g., for recursive next
      hops.";
    container simple-next-hop {
      description
        "This case represents a simple next hop consisting of the
        next-hop address and/or outgoing interface.

        Modules for address families MUST augment this case with
        a leaf containing a next-hop address of that address
        family.";
      leaf outgoing-interface {
        type if:interface-ref;
        description
          "Name of the outgoing interface.";
      }
      uses rt-types:mpls-label-stack;
    }
    container special-next-hop {
      uses rt:special-next-hop;
    }
  }
  list next-hop {
    key hop-index;
    description
      "An entry in a next-hop list.  Modules for address
      families MUST augment this list with a leaf containing a
      next-hop address of that address family.";
    leaf hop-index {
      type uint8;
      description "";
    }
    leaf outgoing-interface {
      type if:interface-ref;
      description
        "Name of the outgoing interface.";
    }
  }
}
```



```
        uses mpls:nhlfe-multiple-contents;
    }
}

grouping mpls-incoming {
    description "";
    choice label-space {
        description "";
        container interface{
            description
                "MPLS label is associated with non-platform label space, all
                of the F-labels and incoming interface information was used
                for identification";
            uses rt-types:mpls-label-stack;
        } //End of non-platform-label-space
        container platform {
            description
                "MPLS label is associated with platform label space, only
                the F-label is used for identification";
            uses rt-types:mpls-label-stack;
        }
    }
}

grouping ip-mpls-service {
    choice detnet-forwarding-type {
        container ip {
            uses ip-incoming;
        }
        leaf mpls-service-label {
            type rt-types:mpls-label-general-use;
            description
                "A service label for DetNet sent on egress, checked on
                ingress";
        } //End of Service-Label
    } //End of detnet-forwarding-type */
} //End of ip-mpls-service */

grouping ip-mpls-profile {
    choice detnet-forwarding-type {
        container ip {
            uses ip-incoming;
        }
        container mpls {
            uses mpls-incoming;
        } //End of mpls
    } //End of detnet-forwarding-type */
}
```



```
} // ip-mpls-profile */

grouping ip-mpls-incoming {
  leaf-list incoming-interface {
    type if:interface-ref;
    description
      "The name of the interface";
  }
  choice detnet-forwarding-type {
    description "Traffic type";
    container ip {
      description "IP encapsulation information";
      uses ip-incoming;
    }
    container mpls {
      uses mpls-incoming;
    } //End of mpls
  } //End of detnet-forwarding-type */
} // ip-mpls-incoming */

grouping eth-ip-mpls-incoming {
  leaf-list incoming-interface {
    type if:interface-ref;
    description
      "The name of the interface";
  }
  choice detnet-forwarding-type {
    description "Traffic type";
    container ethernet {
      leaf placeholder {
        type string;
        description "Place holder for matching ethernet";
      }
    }
    container ip {
      description "IP encapsulation information";
      uses ip-incoming;
    }
    container mpls {
      uses mpls-incoming;
    } //End of mpls
  } //End of detnet-forwarding-type */
} // ip-mpls-incoming */

grouping service-group {
  leaf service-protection-type {
    type identityref {
      base service-protection-type;
    }
  }
}
```



```
    }
    description
      "The DetNet service protection type such as PRF, PEF,
      PEOF, PERF, and PEORF";
  }
  leaf sequence_number_length {
    type uint8 {
      range "0 | 16 | 28";
    }
    default 0;
    description
      "When the sequence number field length is 16 or 28 bits for
      a flow, the sequence number MUST be incremented by one for
      each new app-flow packet sent. When the field length is 16
      bits, d-CW bits 4 to 15 MUST be set to zero (0).";
  }
}

container detnet {
  container applications {
    description
      "DetNet applications";
    list app-list {
      key "name";
      description
        "list of the DetNet configurations";
      leaf name {
        type string;
        description
          "The name to identify the DetNet configuration";
      }
      leaf app-id {
        type uint16;
        config false;
        description
          "The DetNet service ID";
        reference
          "draft-ietf-detnet-flow-information-model-06 Section 6.1";
      }
      leaf app-flow-bidir-congruent {
        type boolean;
        description
          "Defines the data path requirement of the App-flow
          whether it must share the same data path and physical
          path for both directions through the network, e.g., to
          provide congruent paths in the two directions.";
      }
      leaf service-outbound{
```



```
    type service-ref;
    description "Binding to this applications outgoing
    service";
}
leaf service-inbound{
    type service-ref;
    description "Binding to this applications incoming
    service";
}
uses traffic-requirements;
uses traffic-specification;
}
list app-ingress {
    key "name";
    description
    "Ingress DetNet application flows or a compound flow";
    leaf name {
        type string;
        description
        "Ingress DetNet application";
    }
    leaf app-flow-status {
        type identityref {
            base status;
        }
        config false;
        description "Status of ingress application flow";
    }
    uses eth-ip-mpls-incoming;
} //End of app-ingress
list app-egress {
    key "name";
    description
    "Ingress DetNet application flows or a compound flow";
    leaf name {
        type string;
        description
        "Ingress DetNet application";
    }
    leaf app-flow-status {
        type identityref {
            base status;
        }
        config false;
        description "Status of egress application flows";
    }
}
container next-hop {
    description
```



```
        "Route's next-hop attribute.";
    choice application-type {
        container ethernet {
            leaf ethernet-place-holder {
                type string;
                description "Place holder for matching ethernet";
            }
        }
        container ip-mpls {
            uses next-hop-state-content;
        }
    }
}
} // End of app-egress
} // End of applications
container services {
    description
        "DetNet services";
    list service-list {
        key "name";
        description
            "list of the DetNet configurations";
        leaf name {
            type string;
            description
                "The name to identify the DetNet configuration";
        }
        leaf service-id {
            type uint16;
            config false;
            description
                "The DetNet service ID";
        }
        leaf service-rank {
            type uint8;
            description
                "The DetNet rank for this service";
        }
    }
    container outbound{
        leaf-list forward-sublayer-output {
            type fwd-sub-layer-output-ref;
            config false;
        }
        uses service-group;
        list application-input {
            key "name";
            description
                "list of the DetNet applications";
        }
    }
}
```



```
    leaf name {
      type application-ref;
      description
        "List of input applications";
    }
    uses ip-mpls-service;
  }
  list prev-relay-services {
    key "name";
    description
      "list of the DetNet applications";
    leaf name {
      type service-ref;
      description
        "List of input applications";
    }
    uses ip-mpls-service;
  }
  uses traffic-requirements;
  uses traffic-specification;
}
container inbound{
  leaf-list forward-sublayer-input {
    type fwd-sub-layer-input-ref;
    config false;
    description
      "List of input forwarding sub-layer interfaces";
  }
  uses service-group;
  list application-output {
    key "name";
    description "";
    leaf name {
      type application-ref;
      //config false;
      description
        "List of input applications";
    }
    uses ip-mpls-service;
  }
  list next-relay-services {
    key "name";
    description "The Next service";
    leaf name {
      type service-ref;
      description "Binding to another service";
    }
    uses ip-mpls-service;
```



```
    }
  }
} // End of service-list
} // End of services

list fwd-sub-layer-input {
  key "name";
  description
    "Incoming DetNet member flows or a compound flow
    for receiving a packet from a port";
  leaf name {
    type string;
    description
      "Incoming DetNet sub-layer name";
  }
  leaf-list incoming-interface {
    type if:interface-ref;
    description
      "The name of the interface";
  }
  choice connectors {
    list services {
      key "name";
      description "Binding to this sub-layer services";
      leaf name {
        type service-ref;
        description "Binding to this sub-layer services";
      }
      uses ip-mpls-profile;
    }
    list forwarding-sub-layers {
      key "name";
      description "Binding to the next forwarding sub-layer ";
      leaf name {
        type fwd-sub-layer-output-ref;
        description "Binding to the next forwarding sub-layer ";
      }
      uses ip-mpls-profile;
    }
  }
} //End of sub-layer
list fwd-sub-layer-output {
  key "name";
  description
    "Outgoing DetNet member flows or a compound flow
    for sending a packet to a port";
  leaf name {
    type string;
```



```

        description
            "Outgoing DetNet sub-layer index";
    }
    container connectors {
        leaf-list outbound-service {
            type service-ref;
            description "Binding to this sub-layer services";
        }
        leaf-list prev-forward-sub-layer {
            type fwd-sub-layer-output-ref;
            description "Binding to the next forwarding sub-layer ";
        }
    }
    uses traffic-requirements;
    uses traffic-specification;
    container next-hop {
        description
            "IPv4 packet header encapsulation information";
        uses next-hop-state-content;
    }
    } // End of out-segments
} // End of detnet
} // End of ietf-detnet-flow
<CODE ENDS>

```

7.4. Test Configuration

The following XML test configuration shows some of the parameters for a DetNet application, service sub-layer and forwarding sub-layer.

7.4.1. DetNet Test Configuration YANG Model

The case illustrated is for an ingress IP application to an MPLS service sub-layer and MPLS forwarding sub-layer. In this example a unidirectional flow is configured. The application input traffic is IP. The Service links the application "app1", the service ID, a service label and the outgoing forwarding sublayer "fs1". The service label is from the remote end service for the unidirectional path.

```

<dn:detnet
  xmlns:dn="urn:ietf:params:xml:ns:yang:ietf-detnet-config"
  xmlns:in="urn:ietf:params:xml:ns:yang:ietf-inet-types"
  xmlns:rt="urn:ietf:params:xml:ns:yang:ietf-routing-types"
  xmlns:if="urn:ietf:params:xml:ns:yang:ietf-interfaces"
  xmlns:r="urn:ietf:params:xml:ns:yang:ietf-routing"
  xmlns:mpls="urn:ietf:params:xml:ns:yang:ietf-mpls"
  xmlns:pf="urn:ietf:params:xml:ns:yang:ietf-packet-fields">

```



```
<dn:applications>
  <dn:app-list>
    <dn:name>app1</dn:name>
    <dn:app-flow-bidir-congruent>false</dn:app-flow-bidir-congruent>
  </dn:app-list>
  <dn:app-ingress>
    <dn:name>port1</dn:name>
    <if:incoming-interface>eth0</if:incoming-interface>
    <dn:ip>
      <dn:source>1.1.1.1/32</dn:source>
      <dn:destination>2.2.2.2/32</dn:destination>
      <dn:protocol-next>6</dn:protocol-next>
      <dn:dscp>40</dn:dscp>
    </dn:ip>
  </dn:app-ingress>
</dn:applications>
<dn:services>
  <dn:service-list>
    <dn:name>ssl1</dn:name>
    <dn:service-rank>10</dn:service-rank>
    <dn:outbound>
      <dn:forward-sublayer-output>fsl1</dn:forward-sublayer-output>
      <dn:sequence_number_length>0</dn:sequence_number_length>
      <dn:application-input>
        <dn:name>app1</dn:name>
        <dn:mpls-service-label>55555</dn:mpls-service-label>
      </dn:application-input>
      <dn:traffic-requirements>
        <dn:min-bandwidth>100000000 </dn:min-bandwidth>
        <dn:max-latency>10000000</dn:max-latency>
        <dn:max-latency-variation>2000000</dn:max-latency-variation>
        <dn:max-loss>2</dn:max-loss>
        <dn:max-consecutive-loss-tolerance>5
        </dn:max-consecutive-loss-tolerance>
        <dn:max-misordering>0</dn:max-misordering>
      </dn:traffic-requirements>
      <dn:traffic-specification>
        <dn:interval>5</dn:interval>
        <dn:max-packets-per-interval>10</dn:max-packets-per-interval>
        <dn:max-payload-size>1500</dn:max-payload-size>
      </dn:traffic-specification>
    </dn:outbound>
  </dn:service-list>
</dn:services>
<dn:fwd-sub-layer-output>
  <dn:name>fsl1</dn:name>
  <dn:next-hop>
    <dn:next-hop>
```



```

    <dn:hop-index>1</dn:hop-index>
    <if:outgoing-interface>eth1</if:outgoing-interface>
    <dn:mpls-label-stack>
      <dn:entry>
        <dn:id>1</dn:id>
        <dn:label>42</dn:label>
        <dn:traffic-class>5</dn:traffic-class>
      </dn:entry>
    </dn:mpls-label-stack>
  </dn:next-hop>
</dn:next-hop>
</dn:fwd-sub-layer-output>
</dn:detnet>

```

7.4.2. DetNet Test Configuration YANG Model

The next case illustrated is for an egress IP application from an MPLS service sub-layer and MPLS forwarding sub-layer. In this example a unidirectional flow is configured. The application egress traffic is IP. The Service links the application "app2", the service ID, a service label and the outgoing forwarding sub-layer "fsl2". Again the service label is relevant to this egress service. The forwarding sub-layer has an mpls stack that it removes. The next label can be looked up because the forwarding sub-layer has a reference to any DetNet services and those services have registered the service label such that forwarding layer can forward to the respective service sub-layer. Since the service sub-layer label is unique to "app1" the service layer can forward the flow to the correct app1. While not illustrated a single service can associated multiple applications with a unique service label.

```

<dn:detnet
  xmlns:dn="urn:ietf:params:xml:ns:yang:ietf-detnet-config"
  xmlns:in="urn:ietf:params:xml:ns:yang:ietf-inet-types"
  xmlns:rt="urn:ietf:params:xml:ns:yang:ietf-routing-types"
  xmlns:if="urn:ietf:params:xml:ns:yang:ietf-interfaces"
  xmlns:r="urn:ietf:params:xml:ns:yang:ietf-routing"
  xmlns:mpls="urn:ietf:params:xml:ns:yang:ietf-mpls"
  xmlns:pf="urn:ietf:params:xml:ns:yang:ietf-packet-fields">
  <dn:applications>
    <dn:app-list>
      <dn:name>app2</dn:name>
      <dn:app-flow-bidir-congruent>false</dn:app-flow-bidir-congruent>
    </dn:app-list>
    <dn:app-egress>
      <dn:name>port1</dn:name>
      <dn:ip-mpls>
      <dn:next-hop>

```



```

        <dn:hop-index> </dn:hop-index>
        <if:outgoing-interface>eth0</if:outgoing-interface>
    </dn:next-hop>
</dn:ip-mps>
</dn:app-egress>
</dn:applications>
<dn:services>
  <dn:service-list>
    <dn:name>ssl2</dn:name>
    <dn:service-rank>10</dn:service-rank>
    <dn:inbound>
      <dn:forward-sublayer-input>fsl2</dn:forward-sublayer-input>
      <dn:sequence_number_length>0</dn:sequence_number_length>
      <dn:application-output>
        <dn:name>app1</dn:name>
        <dn:mps-service-label>55555</dn:mps-service-label>
      </dn:application-output>
    </dn:inbound>
  </dn:service-list>
</dn:services>
<dn:fwd-sub-layer-input>
  <dn:name>fsl2</dn:name>
  <if:incomming-interface>eth1</if:incomming-interface>
  <dn:service-sub-layer>
    <dn:name>ssl2</dn:name>
  </dn:service-sub-layer>
  <dn:mps>
    <dn:interface>
      <dn:mps-label-stack>
        <dn:entry>
          <dn:id>1</dn:id>
          <dn:label>42</dn:label>
          <dn:traffic-class>5</dn:traffic-class>
        </dn:entry>
      </dn:mps-label-stack>
    </dn:interface>
  </dn:mps>
</dn:fwd-sub-layer-input>
</dn:detnet>

```

8. Open Issues

There are some open issues that are still under discussion:

- o The name of 'in-segment' and 'out-segment' are still under discussion. These terms are used in YANG model for MPLS, but they may cause confusion in DetNet.

- o Whether application flow should be in service sub-layer is still under discussion.
- o Whether we are supposed to define a new YANG Model for DetNet as the current draft does, or we should augment the current YANG Model.

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-flow-information-model]

Varga, B., Farkas, J., Cummings, R., Jiang, Y., and D. Fedyk, "DetNet Flow Information Model", [draft-ietf-detnet-flow-information-model-10](#) (work in progress), May 2020.

[I-D.ietf-detnet-ip]

Varga, B., Farkas, J., Berger, L., Fedyk, D., and S. Bryant, "DetNet Data Plane: IP", [draft-ietf-detnet-ip-06](#) (work in progress), April 2020.

[I-D.ietf-detnet-mpls]

Varga, B., Farkas, J., Berger, L., Malis, A., Bryant, S., and J. Korhonen, "DetNet Data Plane: MPLS", [draft-ietf-detnet-mpls-06](#) (work in progress), April 2020.

- [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>>.
- [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>>.

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-23](#) (work in progress), March 2020.
- [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

Don Fedyk
LabN Consulting, L.L.C.

Email: dfedyk@labn.net