

none  
Internet-Draft  
Intended status: Informational  
Expires: July 30, 2018

L. Qiang  
Huawei  
A. Galis  
University College London  
L. Geng  
China Mobile  
K. Makhijani  
Huawei  
P. Martinez-Julia  
NICT  
H. Flinck  
Nokia  
X. de Foy  
InterDigital Inc.  
January 26, 2018

Technology Independent Information Model for Network Slicing  
draft-qi-ang-coms-netslicing-information-model-02

## Abstract

This document provides a technology independent information model for transport network slicing.

## 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 July 30, 2018.

## Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

Internet-Draft

Network slicing

January 2018

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

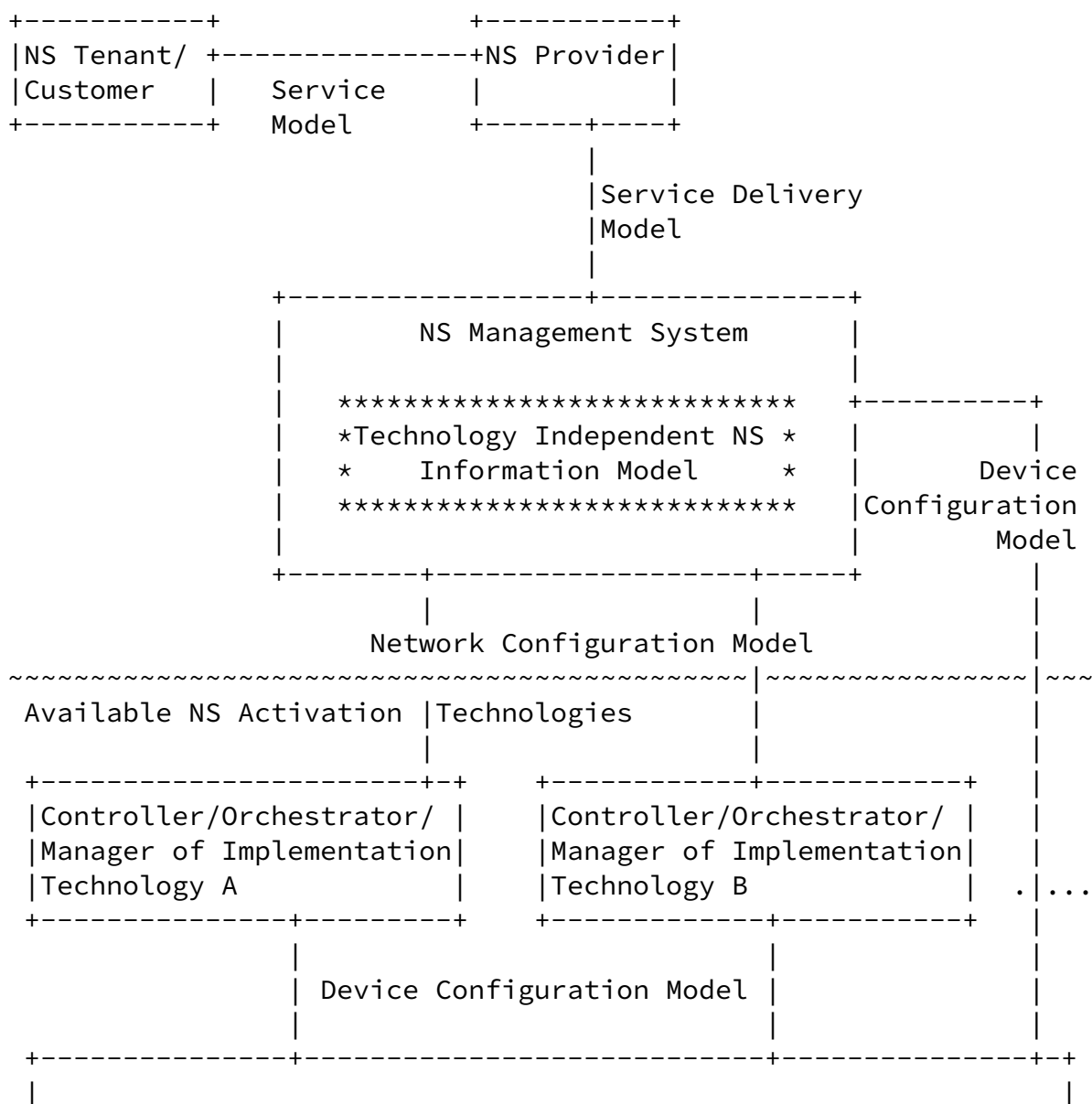
|                        |  |                    |
|------------------------|--|--------------------|
| <a href="#">1.</a>     | Introduction . . . . .                 | <a href="#">2</a>  |
| <a href="#">2.</a>     | Terminology . . . . .                  | <a href="#">4</a>  |
| <a href="#">3.</a>     | Network Slice Tree Structure . . . . . | <a href="#">4</a>  |
| <a href="#">3.1.</a>   | resources . . . . .                    | <a href="#">5</a>  |
| <a href="#">3.1.1.</a> | nodes . . . . .                        | <a href="#">6</a>  |
| <a href="#">3.1.2.</a> | links . . . . .                        | <a href="#">7</a>  |
| <a href="#">3.1.3.</a> | storage-units . . . . .                | <a href="#">8</a>  |
| <a href="#">3.1.4.</a> | compute-units . . . . .                | <a href="#">9</a>  |
| <a href="#">3.2.</a>   | generalized-function-block . . . . .   | <a href="#">9</a>  |
| <a href="#">3.3.</a>   | slice-level-attributes . . . . .       | <a href="#">10</a> |
| <a href="#">4.</a>     | Operations . . . . .                   | <a href="#">13</a> |
| <a href="#">5.</a>     | Yang Module . . . . .                  | <a href="#">14</a> |
| <a href="#">6.</a>     | Security Considerations . . . . .      | <a href="#">27</a> |
| <a href="#">7.</a>     | IANA Considerations . . . . .          | <a href="#">27</a> |
| <a href="#">8.</a>     | Acknowledgements . . . . .             | <a href="#">27</a> |
| <a href="#">9.</a>     | References . . . . .                   | <a href="#">27</a> |
| <a href="#">9.1.</a>   | Normative References . . . . .         | <a href="#">27</a> |
| <a href="#">9.2.</a>   | Informative References . . . . .       | <a href="#">27</a> |
|                        | Authors' Addresses . . . . .           | <a href="#">27</a> |

## [1.](#) Introduction

Network slicing is a tool to share network resources and to offer customized network architectures for diverse use cases that share the same underlying infrastructure [[NGMN-NS-Framework](#)]. Customers may not be familiar with underlying networking technologies, and therefore may prefer to interface with network slices in a technology-agnostic way. On the other hand, service providers may have multiple candidate technologies for supporting network slicing. As shown in Figure 1, there is a gap between technology-agnostic network slicing service requirements and specific implementation

technologies, that needs to be filled by a technology independent information model. Such a technology independent information model describes the entities that compose a network slice, their properties, attributes and operations, and the way they relate to each other of an end to end network slice that may span across

multiple technology domains. It is independent of any specific repository, software usage, protocol, or platform, hence supports common operations and management of network slices.



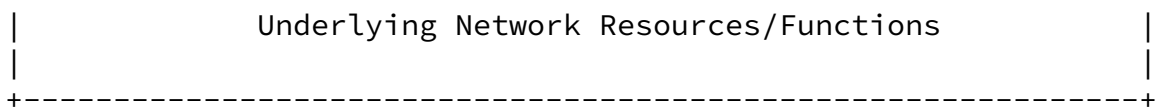


Figure 1: Technology Independent NS Information Model

mapping to specific technology is out of scope.

## 2. Terminology

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

Other network slicing related terminology used in this document are interpreted as description in [[COMS-PS](#)].

## 3. Network Slice Tree Structure

The YANG data modeling language [[RFC7950](#)] will be used to represent the transport network slicedata model. Moreover, the data model for network topologies developed in [[draft-ietf-i2rs-yang-network-topo](#)] will be used as a base.

The proposed NS information model includes the following elements: connectivity resources, storage resources, compute resources, service instance based on predefined function blocks, network slice level attributes, etc. It is presented as a tree structure of attributes. The Yang language is used to represent the network slice information model. The following tree shows an overview of the tree structure. New attributes proposed in this draft are in the "netslice:" namespace, while other attributes are defined in [[draft-ietf-i2rs-yang-network-topo](#)].

```
module: ietf-network
+--rw networks
  +--rw network* [network-id]
    +--rw network-id network-id
    +--rw network-types
    +--rw supporting-network* [network-ref]
      | +--rw network-ref
    +--rw node* [node-id]
      | +--...
      | +--rw netslice:compute-unit* [compute-unit-ref]
      | | +--rw netslice:compute-unit-ref compute-unit-ref
      | +--rw netslice:storage-unit* [storage-unit-ref]
      | | +--rw netslice:storage-unit-ref storage-unit-ref
      | +--rw netslice:service-instance* [service-instance-ref]
      | +--rw netslice:service-instance-ref service-instance-ref
    +--rw nt:link* [link-id]
      | +--...
      | +--rw netslice:link-qos
      | +--...
    +--rw netslice:compute-unit* [compute-unit-id]
      | +--...
    +--rw netslice:storage-unit* [storage-unit-id]
```

```

|   +--...
+--rw netslice:service-instance* [service-instance-id]
|   +--...
+--rw netslice:slice-level-attributes
    +--...

```

### [3.1.](#) resources

Basic resources are used to construct a network slice. Resources comprise: nodes, links, compute units and storage units.

Different resources can exist independently, they can also be bound together when necessary. For example, bind a storage unit to a connectivity node.

A whole network attribute can represent a network slice instance. The network slice instance "supporting network" list can include underlying networks which are used to implement the network slice.

In this model, nodes and links will represent virtual nodes and links exposed to the slice user.

The network-id attribute will represent a network slice instance ID.

#### [3.1.1.](#) nodes

```

+--rw node* [node-id]
|   +--rw node-id                                node-id
|   +--rw supporting-node* [network-ref node-ref]
|   |   +--rw network-ref
|   |   +--rw node-ref
|   +--rw nt:termination-point* [tp-id]
|   |   +--rw nt:tp-id                                tp-id
|   |   +--rw nt:supporting-termination-point*
|   |   |   [network-ref node-ref tp-ref]
|   |   |   +--rw nt:network-ref
|   |   |   +--rw nt:node-ref
|   |   |   +--rw nt:tp-ref
|   |   +--rw netslice:packet-rate?                int64

```

```

| | +--rw netslice:packet-loss-probability? int64
| | +--rw netslice:packet-loss-threshold? int64
| | +--rw netslice:received-packets? int64
| | +--rw netslice:sent-packets? int64
| +--rw netslice:compute-unit* [compute-unit-ref]
| | +--rw netslice:compute-unit-ref compute-unit-ref
| +--rw netslice:storage-unit* [storage-unit-ref]
| | +--rw netslice:storage-unit-ref storage-unit-ref
| +--rw netslice:service-instance* [service-instance-ref]
| | +--rw netslice:service-instance-ref service-instance-ref

```

Nodes are defined in [[draft-ietf-i2rs-yang-network-topo](#)].

Nodes are augmented with the following attributes, that used to represent requirements, configuration and statistics associated with a termination point:

**packet-rate:** the packet forwarding capability of a port for this node in the unit of pps (packet per second).

**packet-loss-probability:** a statistical value which reflects the probability of packet loss.

**packet-loss-threshold:** a threshold of the packet loss probability. If the value of packet-loss-probability is larger than packet-loss-threshold, should actively notify the management system.

**received-packets:** a statistical value which reflects the number of received packets in a period of time.

**sent-packets:** a statistical value which reflects the number of sent packets in a period of time.

### [3.1.2.](#) links

```

+--rw nt:link* [link-id]
| +--rw nt:link-id link-id
| +--rw nt:source
| | +--rw nt:source-node?
| | +--rw nt:source-tp?
| +--rw nt:destination

```

```

| | +--rw nt:dest-node?
| | +--rw nt:dest-tp?
| +--rw nt:supporting-link* [network-ref link-ref]
| | +--rw nt:network-ref
| | +--rw nt:link-ref
| +--rw netslice:link-qos
| | +--rw netslice:link-bandwidth-agreement? int64
| | +--rw netslice:link-throughput? int64
| | +--rw netslice:link-throughput-threshold? int64
| | +--rw netslice:link-latency-agreement? int64
| | +--rw netslice:link-latency? int64
| | +--rw netslice:link-jitter-agreement? int64
| | +--rw netslice:link-jitter? int64
| | +--rw netslice:link-jitter-threshold? int64
| | +--rw netslice:mandatory-node* [node-ref]
| | | +--rw netslice:node-ref node-ref
| | +--rw netslice:mandatory-link* [link-ref]
| | | +--rw netslice:link-ref link-ref
| | +--rw netslice:excluded-node* [node-ref]
| | | +--rw netslice:node-ref node-ref
| | +--rw netslice:excluded-link* [link-ref]
| | | +--rw netslice:link-ref link-ref

```

Links are defined in [[draft-ietf-i2rs-yang-network-topo](#)].

Links are associated with nodes through termination points placed under nodes. Links are augmented with QoS information as follows:

**link-bandwidth-agreement:** specify the bandwidth requirement for this link. If this parameter does not be set specifically, then the link will be constructed according to the default bandwidth value provided by management plane.

**link-throughput:** the current throughput of this link.

**link-throughput-threshold:** a threshold for link throughput. If the value of link-throughput is smaller than link-throughput-threshold, should actively notify the management system.

**link-latency-agreement:** specify the latency requirement for this



link. If this parameter does not be set specifically, then the link will be constructed according to the default latency agreement provided by management plane.

link-latency: the current latency of this link.

link-jitter-agreement: specify the jitter requirement for this link. If this parameter does not be set specifically, then the link will be constructed according to the default jitter agreement provided by management plane.

link-jitter: the current jitter of this link.

link-jitter-threshold: a threshold for link jitter. If the value of link-jitter is larger than link-jitter-threshold, should actively notify the management system.

mandatory-node/link: a list of underlying nodes/links that must be passed by the mapped physical path of this link.

exclusive-node/link: a list of underlying nodes/links that cannot be traversed by the mapped physical path of this link.

### [3.1.3.](#) storage-units

```

+--rw netslice:storage-unit* [storage-unit-id]
|   +--rw netslice:storage-unit-id          inet:uri
|   +--rw netslice:size?                     int64
|   +--rw netslice:access-rate               int32
|   +--rw netslice:access-mode?              access-qualifier
|   +--rw netslice:read-write-mode-type?     read-write-mode-type
|   +--rw netslice:redundancy-type?          redundancy-type
|   +--rw netslice:location?                 string

```

size: size of the storage unit in MB.

access-rate: the minimum rate to write/read 8KB files into/from the storage unit.

access-mode: there are two options include public or dedicated.

read-write-mode: there are two options include read only, and read & write.

redundancy-type: there are four options include best efforts (i.e, no redundancy), n+1 (n storage units with one extra backup), 2n (each

storage unit has one backup),  $2n+1$  ( $n$  storage units with  $n+1$  extra backup).

location: a string describing the location of the storage unit.

#### [3.1.4.](#) compute-units

```
+--rw netslice:compute-unit* [compute-unit-id]
|  +--rw netslice:compute-unit-id    inet:uri
|  +--rw netslice:num-cores?          int8
|  +--rw netslice:ram?                int64
|  +--rw netslice:access-mode?        access-mode-type
|  +--rw netslice:location?           string
|  +--rw netslice:unit-type            compute-unit-type
```

num-cores: the number of arithmetic logic unit.

ram: RAM in bytes.

access-mode: there are two options include shared or dedicated.

location: a string describing the location of the compute unit.

unit-type: two types of compute unit include GPU or CPU

#### [3.2.](#) generalized-function-block

Internet-Draft

Network slicing

January 2018

```

+--rw netslice:service-instance* [service-instance-id]
|  +--rw netslice:service-instance-id          inet:uri
|  +--rw netslice:domain-agent
|  |  +--rw netslice:agent-name?                string
|  |  +--rw netslice:sb-ip-address?             string
|  |  +--rw netslice:sb-port?                   string
|  |  +--rw netslice:nb-ip-address              string
|  |  +--rw netslice:nb-port?                   string
|  +--rw netslice:load-balancer [element-id]
|  |  +--rw element-id                          inet:uri
|  |  +--rw nt:termination-point* [tp-id]
|  |  |  +--rw nt:tp-id                          tp-id
|  |  |  +--rw nt:supporting-termination-point*
|  |  |  |  [network-ref node-ref tp-ref]
|  |  |  |  +--rw nt:network-ref
|  |  |  |  +--rw nt:node-ref
|  |  |  |  +--rw nt:tp-ref
|  |  |  +--rw netslice:packet-rate?             int64
|  |  |  +--rw netslice:packet-loss-probability? int64
|  |  |  +--rw netslice:packet-loss-threshold?   int64
|  |  |  +--rw netslice:received-packets?        int64
|  |  |  +--rw netslice:sent-packets?            int64
|  |  +--rw netslice:lb-name?                    string
|  |  +--rw netslice:ip-address?                  string
|  |  +--rw netslice:port?                        string

```

Some general features could be packaged into function blocks in advance, such as agent, firewall, load balancer, etc.

### [3.3.](#) slice-level-attributes

```

+--rw netslice:slice-level-attributes
|  +--rw netslice:service-time-start? yang:date-and-time
|  +--rw netslice:service-time-end?   yang:date-and-time
|  +--rw netslice:lifecycle-status?   lifecycle-status-type
|  +--rw netslice:access-control
|  |  +--rw netslice:match?            string
|  |  +--rw netslice:action?           string
|  |  +--rw netslice:priority?         string

```

```

|  +---rw netslice:counter?      int64
+---rw netslice:reliability-level?  reliability-level-type
+---rw netslice:resource-reservation-level?
                                resource-reservation-level-type
+---rw netslice:availability?      int64
+---rw netslice:availability-threshold?  string

```

The slice-level-attributes refers to a set of attributes applicable to a network slice. Some explanations are provided as follows for easy going:

service-time-start/end: specify the time during which the network slice service exists (e.g., three months, one year).

lifecycle-status: specify the status of the network slice, there are four enumeration values: construction, modification, activation and deletion.

access-control: illustrates each role can take what kind of operations on the network slice.

reliability-level: the ability of a network slice to be in a stable state. In this document, the main method to achieve reliability is "backup". If necessary, other methods also can be extended based on the current definition. The detailed definition of Reliability\_Level is provided in Table 1.

resource-reservation-level: classify different resource reservation levels of a network slice. This attribute is related to the slice isolation but is not strictly bound. The detailed definition is provided in Table 2.

availability: a statistical value which reflects the probability for a network slice instance to work with expected SLA in a period of time (e.g., 99.999% of time).

availability-threshold: a threshold of the availability. If the value of Availability is smaller than Availability\_Threshold, should actively notify the management system.

Internet-Draft

Network slicing

January 2018

| Value           | Explanation  | Note                          |
|-----------------|--|-------------------------------|
| none            | No specific reliability requirement  | The lowest reliability level  |
| path-backup     | Each path has a backup path  | Path reliability              |
| logical-backup  | Each node/link has a backup node/link  | Logical resource reliability  |
| physical-backup | Each node/link has a backup node/link, and the primary and backup nodes/links must be mapped to different physical devices/paths (the mapped two physical paths couldn't have any shared device) | Physical resource reliability |

=====+=====+

Table 1: Explanation of reliability-level

Qiang, et al.

Expires July 30, 2018

[Page 12]

Internet-Draft

Network slicing

January 2018

| Value             | Explanation  | Note   |
|-------------------|--|--|
| none              | No specific resource reservation requirement   | The lowest resource reservation level, the network slice instance will share and compete for resource with other network slice instances |
| shared-non-preemp | A certain of resource reservation, the free reserved resources could be used by other slice instances, and | Shared and non-preemptive  |

|                   |  |   |
|-------------------|--|---|
| itive             | uable to be retrieved if other slice instances are using them  |   |
| shared-preemptive | More stringent resource reservation, the free reserved resources could be used by other slice instances, and will be retrieved if the network slice needs them | Shared and preemptive                             |
| exclusive         | The reserved resources couldn't be used by other slice instances, even if these resources are free   | The highest resource reservation level, exclusive |

Table 2: Explanation of resource-reservation-level

#### 4. Operations

The defined information model should be able to support the following operations on network slices. Except for support the operations on a complete network slice, each element inside a network slice also should be able to be operated specifically.

- o construct: construct a network slice

- o delete: delete a network slice
- o modify: modify a constructed network slice
- o set\_element\_value: set the value of an indicated element in a network slice
- o get\_element\_value: get the value of an indicated element in a network slice
- o monitor: monitor the status of a network slice

- o enable\_report: enable the active report to the subscribes/management system when the monitored status changes beyond expectation

## 5. Yang Module

```
<CODE BEGINS> module ietf-coms-core {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-coms-core";
  prefix netslice;

  import ietf-yang-types { prefix "yang"; }
  import ietf-inet-types { prefix inet; }
  import ietf-network { prefix nd; }
  import ietf-network-topology { prefix lnk; }

  organization
    "IETF";

  contact
    "Editors:      X. de Foy, Cristina QIANG
      <mailto:>";

  description
    "This module contains a collection of YANG definitions for COMS.

    Copyright (c) 2016 IETF Trust and the persons identified as
    authors of the code.  All rights reserved.

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

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



```

revision "2018-01-26" {
    description
        "Initial revision of COMS topology.";
    reference
        "draft-qiang-coms-netslicing-information-model-02";
}

/*
    Types
*/

typedef read-write-mode-type {
    type enumeration {
        enum read-write {
            description "R/W";
        }
        enum read-only {
            description "R/O";
        }
    }
    description "Indicates if entity is read-write,
        read-only, etc.";
}

typedef access-mode {
    type enumeration {
        enum access-mode-public {
            description "Underlying storage can be
                shared with other instances";
        }
        enum access-mode-dedicated {
            description "Underlying storage is not
                shared with other instances";
        }
    }
    description "access-mode";
}

typedef compute-unit-type {
    type enumeration {
        enum compute-unit-cpu {
            description "Underlying compute unit is CPU based";
        }
        enum compute-unit-gpu {

```

```
        description "Underlying compute unit is GPU based";
    }
}
description "compute-unit-type";
}

typedef lifecycle-status-type {
    type enumeration {
        enum construction {
            description "construction";
        }
        enum modification {
            description "modification";
        }
        enum activation {
            description "activation";
        }
        enum deletion {
            description "deletion";
        }
    }
    description "Lifecycle status";
}

typedef resource-reservation-level-type {
    type enumeration {
        enum none {
            description "No specific reliability requirement";
        }
        enum shared-non-preemptive {
            description "Each path has a backup path";
        }
        enum shared-preemptive {
            description "Each node/link has a backup node/link";
        }
        enum exclusive {
            description "Each node/link has a backup node/link,
                mapped to different physical devices/paths";
        }
    }
    description "Resource reservation level";
}

typedef reliability-level-type {
    type enumeration {
        enum none {
            description "No specific reliability requirement";
```

}

Internet-Draft

Network slicing

January 2018

```
    enum path-backup {
        description "Each path has a backup path";
    }
    enum logical-backup {
        description "Each node/link has a backup node/link";
    }
    enum physical-backup {
        description "Each node/link has a backup node/link,
            mapped to different physical devices/paths";
    }
}
description "Reliability level";
}

typedef redundancy-type {
    type enumeration {
        enum none {
            description "no redundancy";
        }
        enum n+1 {
            description "n storage units with one extra backup";
        }
        enum 2n {
            description "each storage unit has one backup";
        }
        enum 2n+1 {
            description "n storage units with n+1 extra backup";
        }
    }
    description "Redundancy type";
}

typedef node-ref {
    type instance-identifier;
    description "A reference to a node";
}

typedef link-ref {
    type instance-identifier;
    description "A reference to a link";
}
```

```

}

typedef compute-unit-ref {
    type instance-identifier;
    description "A reference to a compute unit";
}

typedef storage-unit-ref {

```

```

    type instance-identifier;
    description "A reference to a storage unit";
}

typedef service-instance-ref {
    type instance-identifier;
    description "A reference to a service instance";
}

grouping rule {
    description "Access Control Rule";
    leaf match{
        type string;
        description "Match";
    }
    leaf action{
        type string;
        description "Action";
    }
    leaf priority{
        type string;
        description "Priority";
    }
    leaf counter{
        type int64;
        description "Counter";
    }
}

grouping port-config {
    description "Configuration of a port/connection point";
    leaf packet-rate {
        type int64;
    }
}

```

```

    description "Data rate in packets per seconds";
}
leaf packet-loss-probability {
    type int64;
    description "Packet loss probability (actual type is TBD)";
}
leaf packet-loss-threshold {
    type int64;
    description "Packet loss probability threshold to alert
management system (actual type is TBD)";
}
}

grouping port-stats {
    description "Statistics of a port/connection point";
}

```

```

leaf received-packets {
    type int64;
    description "Total number of packets received";
}
leaf sent-packets {
    type int64;
    description "Total number of packets sent";
}
}

grouping storage-unit-specs {
    description "Storage unit specs";
    leaf size {
        type int64;
        description "storage size in MB";
    }
    leaf access-rate {
        type int32;
        description "lower limit of storage access rate";
    }
    leaf access-mode {
        type access-mode;
        description "access-mode";
    }
    leaf read-write-mode-type {
        type read-write-mode-type;
    }
}

```

```

        description "Read and write mode";
    }
    leaf redundancy-type {
        type redundancy-type;
        description "Redundancy type";
    }
}

grouping storage-unit-desc {
    description "Storage unit description";
    leaf storage-unit-id {
        type inet:uri;
        description "storage-unit ID";
    }
    uses storage-unit-specs;
    leaf location {
        type string;
        description "Location hint";
    }
}

grouping compute-unit-specs {

```

```

    description "Compute unit specs";
    leaf num-cores {
        type int8;
        description "Number of CPU Cores";
    }
    leaf ram {
        type int64;
        description "RAM in bytes";
    }
    leaf access-mode {
        type access-mode-type;
        description "access mode";
    }
}

grouping compute-unit-desc {
    description "Compute unit description";
    leaf compute-unit-id {
        type inet:uri;
    }
}

```

```

    description "storage-unit ID";
  }
  uses compute-unit-specs;
  leaf location {
    type string;
    description "Location hint";
  }
  leaf unit-type {
    type compute-unit-type;
    description "specify the category of compute unit";
  }
}

grouping path-restrictions {
  description "Physical path restriction type: nodes and
  links of underlying networks
  that must or must not be traversed by a link";
  list mandatory-node {
    key "node-ref";
    description "List of mandatory nodes";
    leaf node-ref {
      type node-ref;
      description "Node";
    }
  }
  list mandatory-link {
    key "link-ref";
    description "List of mandatory links";
    leaf link-ref {

```

```

    type link-ref;
    description "Link";
  }
}
list excluded-node {
  key "node-ref";
  description "List of excluded nodes";
  leaf node-ref {
    type node-ref;
    description "Node";
  }
}

```

```

list excluded-link {
  key "link-ref";
  description "List of excluded links";
  leaf link-ref {
    type link-ref;
    description "Link";
  }
}

grouping link-qos-desc {
  description "QoS associated with a link";
  leaf link-bandwidth-agreement {
    type int64;
    description "Link bandwidth agreement";
  }
  leaf link-throughput {
    type int64;
    description "Link throughput";
  }
  leaf link-throughput-threshold {
    type int64;
    description "Link throughput threshold";
  }
  leaf link-latency-agreement {
    type int64;
    description "Link latency agreement";
  }
  leaf link-latency {
    type int64;
    description "Link latency";
  }
  leaf link-jitter-agreement {
    type int64;
    description "Link jitter agreement";
  }
}

```

```

leaf link-jitter {
  type int64;
  description "Link jitter";
}
leaf link-jitter-threshold {

```



```

    type int64;
    description "Link jitter threshold";
}
uses path-restrictions;
}

grouping slice-level-attributes {
    description "network slice level attributes";
    leaf service-time-start {
        type yang:date-and-time;
        description "Start of service";
    }
    leaf service-time-end {
        type yang:date-and-time;
        description "End of service";
    }
    leaf lifecycle-status {
        type lifecycle-status-type;
        description "Step in the slice lifecycle";
    }
    container access-control {
        uses rule;
        description "Control of access to operations per role";
    }
    leaf reliability-level {
        type reliability-level-type;
        description "Reliability level";
    }
    leaf resource-reservation-level {
        type resource-reservation-level-type;
        description "Resource reservation level";
    }
    leaf availability {
        type int64;
        description "Measure of probability to work with
        expected SLA (TBD: type should be expanded)";
    }
    leaf availability-threshold {
        type string;
        description "Availability threshold to actively
        notify the management system";
    }
}
}

```

```

grouping generalized-function-block {
    description "generalized function blocks that can be
    used to create an instance (more funcution blocks TBD)";

    container domain-agent {
        description "a network slice agent to receive manager request";
        leaf agent-name {
            type string;
            description "agent name";
        }
        leaf sb-ip-address {
            type string;
            description "IP Address of the server which for southbound protocols";
        }
        leaf sb-port {
            type string;
            description "Port of the server which for southbound protocols";
        }
        leaf nb-ip-address {
            type string;
            description "IP Address of the server which for northbound protocols";
        }
        leaf nb-port {
            type string;
            description "Port of the server which for northbound protocols";
        }
    }
}

container load-balancer {
    description "load balancer (type TBD)";
    leaf element-id {
        type inet:uri;
        description "load balancer element id";
    }
    list termination-point {
        use termination-point-desc;
    }

    leaf LB-name {
        type string;
        description "load balancer name";
    }
    leaf ip-address {
        type string;
        description "IP Address of the load balancer (type TBD)";
    }
    leaf port {
        type string;
    }
}

```

Internet-Draft

Network slicing

January 2018

```
        description "Port of the load balancer (type TBD)";
    }
}
}

grouping termination-point-desc {
    description "Augment network nodes termination points with
port information.";

    leaf tp-id {
        type tp-id;
        description
            "Termination point identifier.";
    }

    list supporting-termination-point {
        key "network-ref node-ref tp-ref";
        description
            "This list identifies any termination points that
the termination point is dependent on, or maps onto.
Those termination points will themselves be contained
in a supporting node.
This dependency information can be inferred from
the dependencies between links. For this reason,
this item is not separately configurable. Hence no
corresponding constraint needs to be articulated.
The corresponding information is simply provided by the
implementing system.";
        leaf network-ref {
            type leafref {
                path
                    "../..../nw:supporting-node/nw:network-ref";
            }
            description
                "This leaf identifies in which topology the
supporting termination point is present.";
        }

        leaf node-ref {
            type leafref {
                path
                    "../..../nw:supporting-node/nw:node-ref";
```

```

    }
    description
        "This leaf identifies in which node the supporting
        termination point is present.";
}

```

```

leaf tp-ref {
    type leafref {
        path
            "/nw:networks/nw:network[nw:network-id=current()/"
            + "../network-ref]/nw:node[nw:node-id=current()../"
            + "node-ref]/termination-point/tp-id";
    }
    description
        "Reference to the underlay node, must be in a
        different topology";
}
} // list supporting-termination-point
uses port-config;
uses port-stats;
}

grouping service-instance-desc {
    description "Service instance description. An instance
    is based on a predefined function block";
    leaf service-instance-id {
        type inet:uri;
        description "service instance ID";
    }
    uses generalized-function-block;
}

/*
Model
*/

augment "/nd:networks/nd:network" {
    description "Augment network nodes with slice information.";
    list compute-unit {
        key "compute-unit-id";
        description "Compute units";
    }
}

```

```

    uses compute-unit-desc;
}
list storage-unit {
    key "storage-unit-id";
    description "Storage units";
    uses storage-unit-desc;
}
list service-instance {
    key "service-instance-id";
    description "Service instance";
    uses service-instance-desc;
}
container slice-level-attributes {

```

```

    description "Attributes that apply to a whole network slice";
    uses slice-level-attributes;
}
}

augment "/nd:networks/nd:network/nd:node" {
    description "Augment network nodes with slice information.";
    list compute-unit {
        key "compute-unit-ref";
        description "List of compute units present in node";
        leaf compute-unit-ref {
            type compute-unit-ref;
            description "Compute unit present in node";
        }
    }
    list storage-unit {
        key "storage-unit-ref";
        description "List of storage units present in node";
        leaf storage-unit-ref {
            type storage-unit-ref;
            description "Storage unit present in node";
        }
    }
    list service-instance {
        key "service-instance-ref";
        description "an instance of a service provided by the node";
        leaf service-instance-ref {
            type service-instance-ref;

```

```

        description "Service instance present in node";
    }
}

augment "/nd:networks/nd:network/nd:node/lnk:termination-point" {
    description "Augment network nodes termination points with
    port information.";
    uses port-config;
    uses port-stats;
}

augment "/nd:networks/nd:network/lnk:link" {
    description "Augment network links with slice information.";
    container link-qos {
        description "QoS specifications for this link";
        uses link-qos-desc;
    }
}
}

```

<CODE ENDS>

## [6.](#) Security Considerations

Each component of the network slice has its own security requirements.

## [7.](#) IANA Considerations

There is no IANA action required by this document.

## [8.](#) Acknowledgements

Authors would like to acknowledge Guangpeng Li for help coding.

## [9.](#) References

### [9.1.](#) Normative References

[[draft-ietf-i2rs-yang-network-topo](#)]  
 "i2rs-yang-network-topo", <<https://www.ietf.org/id/>

[draft-ietf-i2rs-yang-network-topo-17.txt](#)>.

[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>>.

## 9.2. Informative References

[COMS-PS] "COMS Problem Statement", <<https://www.ietf.org/id/draft-geng-coms-problem-statement-00.txt>>.

[NGMN-NS-Framework]  
"NGMN Network Slicing Framework",  
<[https://www.ngmn.org/uploads/media/161010\\_NGMN\\_Network\\_Slicing\\_framework\\_v1.0.8.pdf](https://www.ngmn.org/uploads/media/161010_NGMN_Network_Slicing_framework_v1.0.8.pdf)>.

[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>>.

## Authors' Addresses

Li Qiang  
Huawei

Email: [qiangli3@huawei.com](mailto:qiangli3@huawei.com)

Qiang, et al.

Expires July 30, 2018

[Page 27]

---

Internet-Draft

Network slicing

January 2018

Alex Galis  
University College London

Email: [a.galis@ucl.ac.uk](mailto:a.galis@ucl.ac.uk)

Liang Geng  
China Mobile

Email: [gengliang@chinamobile.com](mailto:gengliang@chinamobile.com)

Kiran Makhijani  
Huawei

Email: Kiran.Makhijani@huawei.com

Pedro Martinez-Julia  
NICT

Email: pedro@nict.go.jp

Hannu Flinck  
Nokia

Email: hannu.flinck@nokia.com

Xavier de Foy  
InterDigital Inc.

Email: Xavier.DeFoy@InterDigital.com