

none
Internet-Draft
Intended status: Informational
Expires: May 3, 2018

X. de Foy
A. Rahman
InterDigital Inc.
A. Galis
University College London
K. Makhijani
L. Qiang
Huawei Technologies
October 30, 2017

Interconnecting (or Stitching) Network Slice Subnets
draft-defoy-coms-subnet-interconnection-01

Abstract

This document aims to define a network slice subnet as a general concept, and to augment a baseline network slice model with attributes that describe interconnections between network slice subnets.

Status of This Memo

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

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

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

This Internet-Draft will expire on May 3, 2018.

Copyright Notice

Copyright (c) 2017 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	2
1.1. Terminology	3
2. Interconnection Concepts	3
3. Information Model	5
3.1. Representing Interconnections	5
3.2. Relation to NS Information Model	5
4. Operations	6
5. Security Considerations	6
6. Next Steps	6
7. IANA Considerations	7
8. Informative References	7
Authors' Addresses	7

1. Introduction

Network Slicing enables deployment and management of services with diverse requirements on end-to-end partitioned virtual networks, including compute and storage resources, over the same infrastructure. [I-D.geng-coms-problem-statement] describes a problem statement for supervised heterogeneous network slicing, enabling users to deploy network slices including connectivity, computing and storage components.

Nevertheless, defining and managing a network slice (NS) end-to-end does not always have to be done directly: it may be convenient to define and manage subsets of NS components. The concept of network slice subnet is defined originally in [NGMN_Network_Slicing], though we only need to retain its definition in the most universal form: network slice subnet instances are similar to slice instances in most ways but cannot be operated in isolation as a complete network slice. They can be interconnected with other NS subnets to form a complete, end-to-end network slice (i.e. interconnection and/or stitching of NS subnets). To summarize: a NS subnet can be seen as a network slice with unconnected links. The term "network slice segment" has also occasionally been used to designate a NS subnet. Use cases for using NS subnets include managing multi-domain network slices, or even within one domain, isolate management and maintenance of different portions of a network slice. It includes also mapping services to an ordered chain of network slice subnets instances.

A model for network slicing is currently being defined in [I-D.qiang-coms-netslicing-information-model]. One question we would like to address is how to augment this base model to describe interconnections between NS subnets. The base model is not technology specific, and therefore the description of interconnections should not be either. Moreover, such an augmentation should both enable describing the intent for interconnection, as well as describing actual interconnections once NS subnets have been stitched together.

1.1. Terminology

Network slicing related terminology used in this document should be interpreted as described in [I-D.geng-coms-problem-statement].

"Network slice subnet" is a term defined in this draft. It is comprised of groups of connectivity, compute and storage resources, possibly together with network functions and network management entities, forming a complete instantiated logical/physical network in support of certain network and service characteristics but cannot be activated in isolation as an overall network slice.

2. Interconnection Concepts

The general goal of an interconnection between 2 NS subnets is to have links established between nodes from both subnets. A secondary goal is to keep NS subnet descriptions isolated from each other. This relative isolation will contribute to simplify and decentralize management, as well as enabling operations such as substituting a subnet with another, composing slice subnets of different domains, etc.

As described in Figure 1, we can represent a network slice subnet as a network slice that also has one or more logical nodes, which terminate (at logical termination points) links that need to be interconnected with external nodes (cross-subnet links).

During a stitching operation, logical termination points from both NS subnets can be paired together into an interconnection point. When implemented at the infrastructure layer, this interconnection point may be either implemented as a gateway, or abstracted away, in which case nodes from both NS subnets end up being directly interconnected between each other. In any cases, interconnected links will need to have compatible QoS attributes.

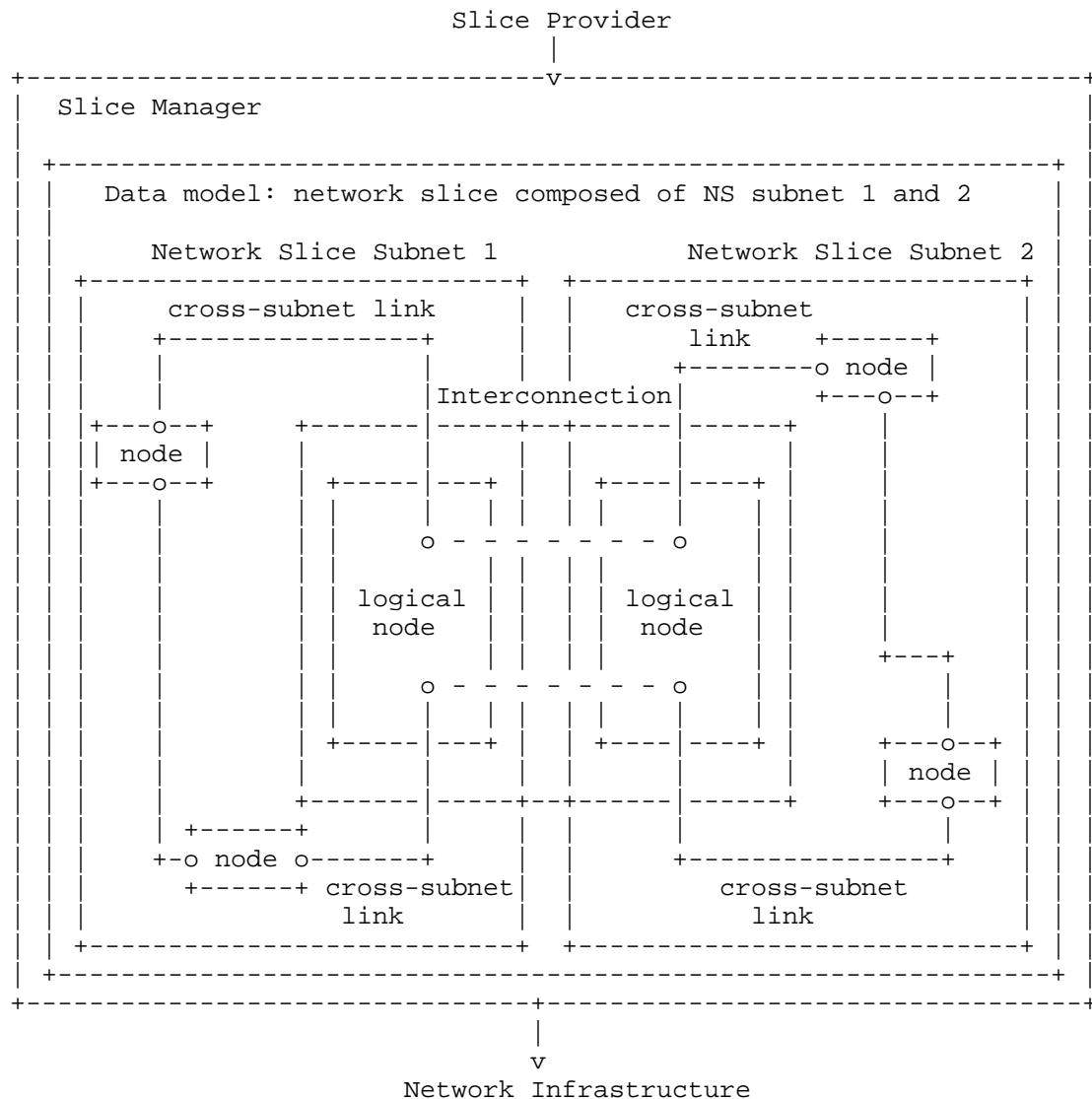


Figure 1: Network Slice Subnets Interconnection

Network slice interconnection information in data models can be used for different related purposes:

- o Anchoring interconnections at logical termination points: a NS subnet model should specify which link termination points are the "network slice subnet boundaries" that need to be interconnected.

- o Programming interconnections: a NS provider may set attributes in a NS subnet model to configure the interconnection with another NS subnet. For example, constraints on the interconnection (on throughput, latency, etc.) may be programmed to trigger an alarm that may lead the NS operator to disable NS subnets, replace NS subnets by others, etc. to maintain overall service performance.
- o Describing the state of interconnection (once NS subnets are interconnected).

3. Information Model

3.1. Representing Interconnections

A fairly minimal way to represent an interconnection is:

- o To represent an interconnection anchor in a subnet: a "logical termination point" in this subnet.
- o To program or represent an interconnection between subnets: a pair of logical termination points, one in each subnet.

Some form of grouping of logical termination points (for example, in logical nodes) may tell the NS manager to treat those termination points as a single unit for placement, implementation, etc.

Additional information may be useful to complement the description of an interconnections. Some attributes may be useful to describe an interconnection point anchor, while others may be useful to program or represent the state of an interconnection. For example, logical termination points may be associated with information that facilitates placement or stitching operations. Future work should determine what type of information would be useful to specify or represent a NS interconnection.

3.2. Relation to NS Information Model

The network slice information model defined in [I-D.qiang-coms-netslicing-information-model] will be used as a base. This model is itself based on the "ietf-network" model defined in [I-D.ietf-i2rs-yang-network-topo]

Individual network slice data model instances ("network" attributes of the "ietf-network" model) can represent network slice subnets. If there is a need to tie multiple subnets together, a parent network slice may be added to the model if necessary, but this is out of scope of the present draft. A "network" attribute may also represent

a full, end-to-end slice, in which case it does not need be interconnected using the mechanisms described in the present draft.

Interconnection anchors are logical termination points (TP) included in logical nodes. The base model can be augmented as described below. Those new logical node and logical TP attributes will typically be used only for nodes and termination points used as interconnection anchors. Logical nodes should be as simple as possible (e.g. should not include any compute or storage unit), so that they can be abstracted away in underlying networks during the interconnection operation, if needed.

```
module: ietf-network
+--rw networks
  +--rw network* [network-id]
    +--rw network-id
    +--rw network-types
    +--rw supporting-network* [network-ref]
      | +--rw network-ref
    +--rw node* [node-id]
      | +--... (augmented with new attributes of logical nodes)
      | +--rw nt:termination-point* [tp-id]
      | | ... (augmented with new attributes of logical TP)
    ...
```

4. Operations

Stitching may occur when network slice subnets are initially instantiated, or later after instantiation.

5. Security Considerations

Access control mechanisms for managing network slices can likely be reused for network slice subnets, since their models should be similar to each other.

Stitching 2 NS subnets together may be subject to some form of authorization by a NS tenant.

6. Next Steps

The present draft investigates one aspect of a non-technology specific representation of a network slice. It may therefore be part of the larger discussion on the need for such a representation.

Beyond this, next steps can include the following:

- o Discussing the definition and need for NS subnets. Is "NS with unconnected links" an adequate simple definition? Is there an agreement on the use cases? Should NS subnet interconnection be standardized?
- o Assuming there is some interest in this, further work is needed to better understand what attributes and operations are needed, and how to integrate them in a baseline NS model.
- o Additionally, we can also further study NS composition mechanisms, beyond the simple connect/disconnect mechanism in the present draft.

7. IANA Considerations

This document has no actions for IANA.

8. Informative References

- [I-D.geng-coms-problem-statement]
67, 4., Slawomir, S., Qiang, L., Matsushima, S., Galis, A., and L. Contreras, "Problem Statement of Supervised Heterogeneous Network Slicing", draft-geng-coms-problem-statement-00 (work in progress), September 2017.
- [I-D.ietf-i2rs-yang-network-topo]
Clemm, A., Medved, J., Varga, R., Bahadur, N., Ananthakrishnan, H., and X. Liu, "A Data Model for Network Topologies", draft-ietf-i2rs-yang-network-topo-17 (work in progress), October 2017.
- [I-D.qiang-coms-netslicing-information-model]
Qiang, L., Galis, A., 67, 4., kiran.makhijani@huawei.com, k., Martinez-Julia, P., Flinck, H., and X. Foy, "Technology Independent Information Model for Network Slicing", draft-qiang-coms-netslicing-information-model-01 (work in progress), October 2017.
- [NGMN_Network_Slicing]
NGMN, "Description of Network Slicing Concept", 10 2016, <https://www.ngmn.org/uploads/media/161010_NGMN_Network_Slicing_framework_v1.0.8.pdf>.

Authors' Addresses

Xavier de Foy
InterDigital Inc.
1000 Sherbrooke West
Montreal
Canada

Email: Xavier.Defoy@InterDigital.com

Akbar Rahman
InterDigital Inc.
1000 Sherbrooke West
Montreal
Canada

Email: Akbar.Rahman@InterDigital.com

Alex Galis
University College London

Email: a.galis@ucl.ac.uk

Kiran Makhijani
Huawei Technologies
2890 Central Expressway
Santa Clara CA 95050
USA

Email: kiran.makhijani@huawei.com

Li Qiang
Huawei Technologies
Huawei Campus, No. 156 Beiqing Rd.
Beijing 100095

Email: qiangli3@huawei.com

Opsawg Working Group
Internet-Draft
Intended status: Informational
Expires: May 3, 2018

X. Ding
W. Liu
Huawei
C. Li
China Telecom
October 30, 2017

Network Data Use Case for Wavelength Division Service
draft-ding-opsawg-wavelength-use-case-00

Abstract

This document describes use cases that demonstrate the applicability of network data to evaluate the performance of wavelength division service. The objective of this draft is not to cover the wavelength division service in detail. Rather, the intention is to illustrate the requirements of network data used to evaluate the performance of wavelength division service.

General characteristics of network data and two typical use cases are presented in this document to demonstrate the different application scenarios of network data in wavelength division service.

Status of This Memo

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

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

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

This Internet-Draft will expire on May 3, 2018.

Copyright Notice

Copyright (c) 2017 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	2
2. Conventions used in this document	3
3. Characteristics of network data	3
4. Use cases	4
4.1. Anomaly detection	4
4.2. Risk assessment	5
5. Data Issues	6
5.1. Merge data from different time periods	6
6. Security Considerations	6
7. Conclusions	7
8. Normative References	7
Authors' Addresses	7

1. Introduction

Wavelength-division multiplexing (WDM) is a method of combining multiple signals on laser beams at various infrared (IR) wavelengths for transmission along fiber optic media. A WDM system uses a multiplexer at the transmitter to join the several signals together, and a demultiplexer at the receiver to split them apart. During the wavelength division service running, network data is consistently generated from wavelength division devices and it can reflect the process of service running.

In the case of wavelength division service, customer is accustomed to handle the network failure after the service interruption. Such passive strategy is inefficient, and easily leads to long service interruption. Network data collected from device is real and reliable, and can help customer to predict the trend of wavelength division optical performance. Statistical characteristics of network data can help operator to judge the time point at which the service is abnormal or normal, or the service is risky or healthy .

This document attempts to describe the detailed use cases that lead to the requirements to support wavelength division performance evaluation. The objective of this draft is not to cover the wavelength division service in detail. Rather, the intention is to

illustrate the requirements of network data used to evaluate the performance of wavelength division service.

General characteristics of network data and two typical use cases are presented in this document to demonstrate the different application scenarios of network data in wavelength division service. Moreover, the question of how to integrate network data collected from different time periods is raised.

2. Conventions used in this document

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].

KPI: Key Performance Indicator. Network KPI represents the operational state of a network device, link or network protocol in the network. KPI data is usually represented to users as a set of time series

(e.g., $KPI = x_i, i=1..t$),

each time series is corresponding to one network KPI indicator value at different time point during specific time period.

3. Characteristics of network data

Network data describes the process that information collected from various data sources and transmitted to one or more receiving equipment for analysis tasks [I-D.ietf-wu-t2trg-network-telemetry]. Analysis tasks may include event correlation, anomaly detection, risk detection, performance monitoring, trend analysis, and other related processes.

Network data is a series of data points indexed in time order. It taken over time may have an internal structure (such as, trend, seasonal variation, or outliers). Trend means that, on average, the measurements tend to increase (or decrease) over time. Seasonality means that, there is a regularly repeating pattern of highs and lows related to calendar time such as seasons, quarters, months, days of the week, and so on. In regression, outliers are far away from the line. With time series data, outliers are far away from the other data.

Network time series data analysis comprises methods for analyzing time series data in order to extract meaningful statistics and other characteristics of the data.

Network data mainly consists several major characteristics:

- o Subject: The subject is the object to be measured, and it has multiple properties from different dimensions. An example of a wavelength division service performance monitoring scenario is that the subject of the measurement is the ' optical module ' whose attributes may include board name, device name, and so on.
- o Measured values: A subject may have one or more measured values, and each measurement corresponds to a specific indicator. Take the server status monitoring scenario example, the measured indicators may have FEC_bef (Forward Error Correction coding before error correction), FEC_aft (Forward Error Correction coding after error correction), input optical power, output optical power, etc.
- o Timestamp: Each report of the measured value will have a timestamp attribute to indicate its time.

4. Use cases

The following sections highlight some of the most common wavelength division use case scenarios and are in no way exhaustive.

4.1. Anomaly detection

In Data Analytics Engine, anomaly detection is the identification of items, events or observations which do not conform to an expected pattern or other items in data. Typically the anomalous items will translate to some kind of problem, such as optical layer problem.

For network equipment performance anomalies, multiple features are usually extracted from KPI data, such as time, value, frequency, etc., and used as the key factors for anomaly analysis.

Take wavelength division service as an example, collection information such as FEC_bef, input optical power, laser bias current and other key factors can be selected to keep track of wavelength division service over time and calculate the device statistics data in a specific time period such as average device downtime in the specified time window. These statistics data can be further used to detect wavelength division service anomaly or improve the accuracy rate for wavelength division KPI anomaly detection. In this scenario, we do not rely on the manual preconfigured threshold to trigger alarm, instead, we automatically detect KPI anomaly in advance and raise alarm, as seen in figure 1.

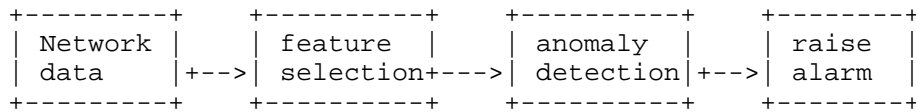


Figure 1: anomaly detection

4.2. Risk assessment

In Data Analytics Engine, risk assessment is a component aiming at providing an estimation of the overall network risk condition. Unlike the anomaly detection component that copes with network faults and failure that already happened, risk assessment module's goal is to anticipate network event, forecast short term change and risk in the network based on the trends of network data (e.g., fast growing, fast dropping, slowly increasing, and slowly decreasing of KPI data). This opens up a channel to reveal potential network problems or locate the need for network optimization and upgrade.

Network KPIs provide fine-grained understanding of network performance, which bring more value to network maintenance and operation, including identifying possible bottlenecks, dimensioning issues, and locating the need to perform network optimization. Based on the various monitor mechanisms, if any high risk is occurred in the network, administrators could be informed at a very early stage. The ability to handle large amount of noisy KPI data properly is vital to gain these desired insights.

Given hundreds of thousands of KPI data, it is a challenging issue to assess network risk. Good network risk assessment criteria should be indicative of local network-level problems, and hence be able to provide prompt warnings and help locate potential problems when trivial but persisting anomalies are observed. Meanwhile, it must also describe system performance in a global sense by aggregating multi-faceted information with large number of KPIs across the network infrastructure. There are two strategies to design such KPI network risk, as shown in figure 2:

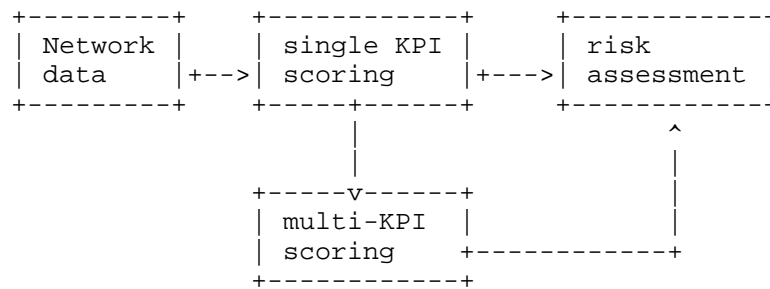


Figure 2: risk assessment

1) Single KPI scoring: The scoring strategy for single KPI. In this case, different dimensions of a KPI should be examined to score a KPI;

2) Multi-KPI scoring: The scoring strategy for assessing the network risk using values of many KPIs. If a device or a service is monitored by several key KPIs, the risk should be analyzed by the integration of these KPI scores.

5. Data Issues

5.1. Merge data from different time periods

In the process of data collection, the collection period of the same KPI may be different from each other. For example, for a multi-domain deployment service, there are many different collection periods for network devices, such as 30s, 5min, 15min, and so on.

KPI data collected from different domains is need to be analyzed for correlation. For example, anomaly detection of wavelength division service data from different domains is performed, and comparison is performed among different domains. So we need to merge data sets from different periods into a integrated data set using metrics in the period, such as mean value, peak value or media value. It then raises a question that how these data sets are stored and assessed with high efficiency.

6. Security Considerations

TBD.

7. Conclusions

TBD.

8. Normative References

- [I-D.ietf-wu-t2trg-network-telemetry]
Wu, Q., "Network Telemetry and Big Data Analysis", March 2016.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", March 1997.

Authors' Addresses

Xiaojian Ding
Huawei
101 Software Avenue, Yuhua District
Nanjing, Jiangsu 210012
China

Email: dingxiaojian1@huawei.com

Will(Shucheng) Liu
Huawei
Bantian, Longgang District
Shenzhen 518129
P.R. China

Email: liushucheng@huawei.com

Chen Li
China Telecom
No.118 Xizhimennei street, Xicheng District
Beijing 100035
P.R. China

Email: lichen@ctbri.com.cn

Network Working Group
Internet-Draft
Intended status: Informational
Expires: May 3, 2018

L. Geng
L. Wang
China Mobile
S. Kuklinski
Orange
L. Qiang
Huawei Technologies
S. Matsushima
Softbank
A. Galis
University College London
Luis. Contreras
Telefonica
October 30, 2017

Problem Statement of Supervised Heterogeneous Network Slicing
draft-geng-coms-problem-statement-01

Abstract

This document discusses the general requirements and problem statement of supervised heterogeneous network slicing. The purpose of this document is to identify the key network components that are used to create a network slice instance. Base on this information, a general network slice template can be visualized. Furthermore, the requirement of a common information model is identified and corresponding management consideration of heterogeneous network slice instance is also discussed.

Status of This Memo

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

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

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

This Internet-Draft will expire on May 3, 2018.

Copyright Notice

Copyright (c) 2017 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	2
1.1. Requirements Language	3
1.2. Terminology	3
2. The Concept of Supervised Heterogeneous Network Slicing . . .	4
2.1. Heterogeneity	7
2.2. The requirement of general supervision of network slicing	7
3. Network Resources for Supervised Heterogeneous Network Slice	8
3.1. Connectivity Resources	8
3.2. Computing Resources	9
3.3. Storage Resources	10
3.4. Generalized Function Blocks	10
3.5. Other Resources	10
4. The Requirement of Common Operation and Management for Supervised Heterogeneous Network Slice	11
4.1. Problem Scope	13
5. Management of Heterogeneous Network Slice	14
6. IANA Considerations	15
7. Security Considerations	15
8. Acknowledgements	15
9. Normative References	15
Authors' Addresses	15

1. Introduction

The concept of network slicing is not new but energized greatly under 5G work in 3GPP. It is expected that further 5G network should be capable of providing dedicated private network for different verticals according to their specific requirements, which are created by diversity of new services such as high definition (HD) video, virtual reality (VR) and V2X applications. Looking at the development of future network, no matter the service is connected via

5G cellular RAN, FTTx optical access network or other dedicated connections, this resource dedication has become a fundamental technology for services requiring extreme quality of user experience. The best effort transport is not good enough as both subscribers and application providers are looking for and willing to pay for certain level of quality dedication. Therefore it is inevitable for service providers (telecommunication infrastructure owners) to rethink the means of management and operation of their networks, which should support end-to-end slicing capabilities.

The requirements from different verticals may be extremely diversified. Typical examples includes high bandwidth, low latency, high level of isolation, specific security and encryption requirements and etc. These requirements may also change dynamically along time since the services of certain industry vertical changes very fast, and sometime spontaneously (i.e. burst bandwidth/latency requirement from on-line shopping provider on certain period). It is expected that the configuration of certain network slice instances are very dynamic in a case-by-case manner. Meanwhile, there are many technology options to fulfil particular requirements depending on considerations on many aspects including cost, TTM and etc. The diversity of both requirements and technology options makes network slices significantly heterogeneous.

In order to provide cost-effective and efficient network slice configuration, service provider needs to understand specifically the components it can make use to create a network slice instance and how these components map with the customer requirements. These components include both network resources and management entities.

1.1. Requirements Language

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

1.2. Terminology

Network Slicing - A management mechanism that Network Slice Provider can use to allocate dedicated network resources from shared network infrastructures to Network Slice Tenant.

Network Slice - A network slice is a managed group of dedicated network resources to meet certain network functionality and performance characteristics required by the network slice tenant(s). It is re-configurable and is supervised by the network slice provider.

Network Slice Provider - A network slice provider (NSP), typically a telecommunication service provider, is the owner or tenant of the network infrastructures from which network slices can be created. The network slice provider takes the responsibilities of managing, orchestrating and monitoring the corresponding resources to implement a network slice and provide the Network slice tenant certain level of management.

Network Slice Tenant - A network slice tenant (NST) is the user of specific network slice, in which customized services are hosted. Network slice tenants can make requests of the creation of new network slice through NSaaS platform. This request will be delivered to network slice controller for implementation purposes.

2. The Concept of Supervised Heterogeneous Network Slicing

Network slicing is a management mechanism that an NSP can use to allocate dedicated network resources from shared network infrastructures to an NST. This dedication may be performed in various forms on a diversity of resources depending on specific NSP's network availabilities. Typical examples include physical and logical isolation of network connectivity with certain QoS guarantees, bare metal and virtualized computing resources, dedicated storage and specific pre-define network functions such as NAT server, SDN controller and etc. Other network technologies such as ICN and CDN may also be part of the resource dedication. Network slicing gives the NSP full flexibility to either logically or physically lease a partition of their networks to the NST with required functionalities and performances.

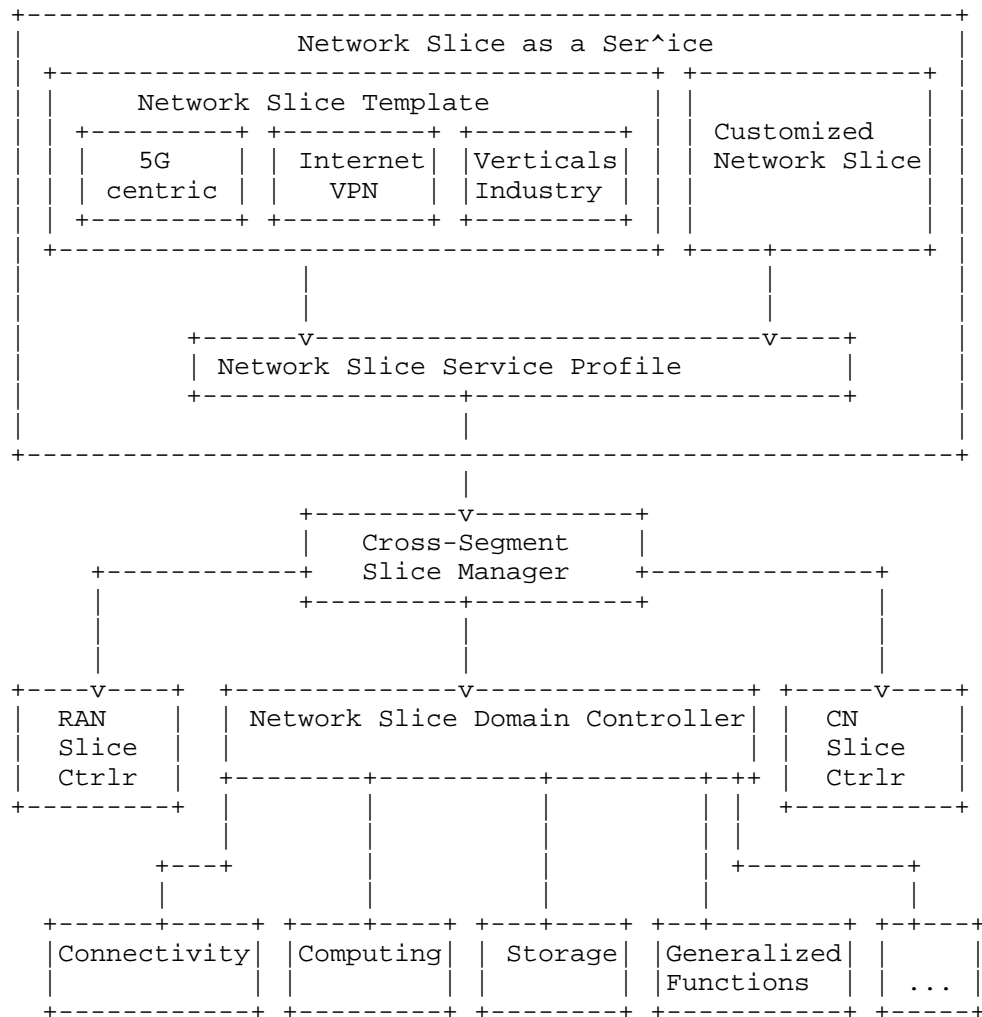


Figure 1: The concept of supervised heterogeneous network slicing

As network slicing is introduced to overall network management, it is anticipated that new business models may be created. With a more flexible, elastic and modularized network, the shared network infrastructures can be sliced and offered as a service to end users and verticals. For instance, a network slice with dedicated network resources with a customized topology can be provided as a service (NSaaS) to the NST.

Figure 1 illustrated the concept of how NSaaS is supervised within a heterogeneous network. In this concept, a network slice can either be implemented according to pre-defined network slice templates or customized requirements. On one hand, network slice template is designed by NSPs for the ease of mapping NST's request to a NSaaS with rather common resources and performance requirements. This includes but not limited to 5G-centric services such as eMBB and URLLC network slices, dedicated internet VPNs for game acceleration and video delivery, specific vertical industrial internet dedications such as smart factory and so on. On the other hand, implementing network slicing using customized approach provides NST with the full flexibility of composing its own network resource pools according to various requirements and constraints respectively. As the NST defines a preferred network slice service either from a pre-defined template or full customization, a network slice service profile is therefore generated.

The network slice service profile is sent to cross-segment slice manager, which coordinate resource from different network segment. The cross-segment network slice manager decomposes the network slice service profile and assigns corresponding segment network slice information to multiple network slice domain controllers. In this illustration, radio access network, transport network and core network are used as common examples of network segments with network slice domain controllers. However, it is worthy of mentioning that backhauling is only one of the use cases of transport segment network slicing.

In the slice controller within a specific network segment, NSaaS information is translated to resource-level description of a network slice for implementation purposes. In particular, it visualizes a network slice with specific resource components that comprise computing, storage, and generalized functions etc. The slice controller also coordinates heterogeneous domain controllers/managers for network slice implementation.

Although a general network slice may consist of resource components from various network segments, the supervised heterogeneous network slicing in this document refers only to IETF segment and sees the managed network slice a stand-alone one within IETF scope. It may be used by the cross-segment network slice manager to create a more comprehensive end-to-end network slice including other network segments that are out of scope of IETF.

2.1. Heterogeneity

Heterogeneity is the nature of network slicing since the requests of NSaaS from the NST are diversified. The slice controller has to supervise heterogeneous resources in various domains in response to NSaaS demands. The different types of resources a network slice controller needs to supervise includes connectivity, storage, computing, generalized functions and others. Furthermore, even for a single type of resources, an NSP may have difference management domains because of either technical or geographical variations. For example, the network slice controller is supposed to have the ability to coordinate multiple typical existing technology domains including optical transport network, IP routing network and layer-2 switched network. It also needs to integrate network domains with different management/control mechanism. For instance, the slice controller is necessary to be compatible with both SDN-enabled ACTN-aware networks and traditional EMS-managed networks. Another case is the management of virtualized resources using VIMs such as Openstack. In order to provide computing/storage resource support for network slicing, these domains are also inevitable required to be coordinated.

No matter it is a green field or brown field implementation, the network resources used to create a network slice are very likely to reside in different heterogeneous management domains. The supervised heterogeneous network slicing provides the capability of coordination and orchestrate the resources from different domains.

2.2. The requirement of general supervision of network slicing

Supervision is required by NSP, making use of OAM tools to maintenance the network slice. The slice controller needs to provide this capability to NSP to supervise the all the network slices that are implemented.

There are varieties of reasons why supervision is crucial in the case the network slicing. First of all, the network slice controller would not be able to deal with the lifecycle management of network slices without supervision abilities. Besides, given the characteristic of heterogeneous environment, the network slice controller must be crystal clear of the underlay resource information that is reported and synchronized by the domain controllers/managers. If this is seen as a network resource capability exposure approach, the network slice controller must supervise this approach to define how this information gathered and used. Furthermore, the network slice controller need to provide slice-level monitoring ability during the complete life circle of a network slice. This is essential for the claim benefits of quality guarantee by the implementation of network slicing. For example, the slice controller

need to supervise and monitor the jitter of a certain link in a network slice with deterministic property requirement by NST. This jitter performance should also be provided to NST for SLA references.

Last but not the least, it is common that some NST would like to participate in the management of its network slice. The slice controller should provide this capability by using the intra-slice management (discussed later in section 5). This management capability exposer must be supervised by NSP to avoid any resource/performance conflicts with other network slices.

3. Network Resources for Supervised Heterogeneous Network Slice

Fundamentally, network slices are created based on the shared network resources. There are many existing technologies which focus on the management of those network resources. For example, various type of domain SDN controllers supervise the connectivity resources within each technology or geographical domains, and MANO supervises the NFV infrastructures. As previously discussed, network slicing provides a management mechanism for NSP to create network slices from the underlay resources. It oversees all these resources and decides the placement of specific resources according to certain path and topology constraints.

Network slicing does not have any constraints on what type of resources NSPs may or may not use as part of the network slice creation. This is completely subjected to NSP's policy. However, for the ease of management and operation, it is worthy to have a guideline to at least categorize the common resources that NSP may offer to NST as a network slice service. The section endeavours to provide a prototype catalogue of the resource components for network slice creation. In general, the components that an NSP can use to create a network slice include connectivity, computing, storage and generalized functions. Other wide-scale network functionalities including ICN and CDN are also regarded as customized network resources.

3.1. Connectivity Resources

Connectivity is one of the essential components for a network slice. It can be as simple as a best effort point-to-point VPN or a physical link using a dedicated wavelength. It may also have more complex topology with other specific requirements including bandwidth, latency and etc. The characteristics of the connectivity component may include the following aspects.

- o Node - The description of a network node at network slice level abstraction. The abstraction level depends on the provided node

resource information from the south bound interface of the transport network slice controller.

- o Link - The description of a network link between two nodes in a network slice.
- o Topology - The description of connection topology of a network slice. It should explicitly describe the connectivity relationship between each access point of the network slice. An NSP should be able to understand the overall connectivity requirement of a network slice from this topology information.
- o Bandwidth - The description of bandwidth requirements of specific links within a network slice. The requirements includes exactly amounts of assured bandwidth, maximum bandwidth and other bandwidth QoS-specific requirements
- o Latency - The description of link latency requirements within a network slice. It should identify the exact amount of latency between a link defined in connection topology.
- o Determinism - The description of the determinism of a link latency. This should be defined in addition to the latency, which further specify the jitter of the latency for a given link.
- o Isolation level - The description of isolation level of a network slice. A NST may request logical isolation which can be mapped to tunnelling technologies. It may also request explicitly a dedicated lamda or even physical link for specific services.

3.2. Computing Resources

If an NST would like to host virtualized functions in a network slice, it may be interested in asking for specific computing resource including both bare metal servers and virtual machines. The computing resource can be specified considering the following characteristics.

- o CPU resources - The CPU specification including CPU model, frequency, quantity of physical/virtual CPU and etc.
- o RAM resources - The RAM size associated with the requested computing resources in a network slice.
- o Virtual resources - The pre-defined virtual resources including both virtual machines and containers associated with a specific network slice.

- o Other - This may include GPU requirements and other specific computing resources

3.3. Storage Resources

It is necessary for NSP to provide storage components in a network slice since NSTs may want to host contents on dedicated resources. Meanwhile, NSP may also prefer to use dedicated storage for specific service policies, authentication information and other management profiles.

- o General storage - The description of storage resource in a network slice. This may include the location, type, size and usage of the storage resource. The general storage requirements may closely related to the connectivity topology as well.
- o CDN service - If an NSP can provide a turn-key CDN solution for the NST. It can also include CDN service within a network slice.

3.4. Generalized Function Blocks

Many dedicated network functions, either physical or virtual, may requested by a NST. Typical example include common network functions as DHCP server, DNS, NAT, Firewall, SDN controller. Application-level functions may also exist in a network slice, such as session management, mobility management and etc. NSP should be able to provide such generalized function blocks according to NST's request.

- o Physical network function blocks- The description of dedicated physical network functions. Physical network functions are network equipments with dedicated software and hardware, which are strictly coupled for the purpose of a providing specific network function.
- o Virtual network function blocks- The description of virtualized network functions. VNFs are software entities which are normally hosted within pre-allocated virtual machines (or containers). The virtual resources which are required by the VNF should be also specified in terms of computing resources as described previously.

3.5. Other Resources

A category of customized resources is reserved for network slicing since NSPs may have unique capabilities that may be used as part of the network slicing to provide even greater innovative functionality. Resources such as ICN-aware subnet, CDN network and etc. are some potential attributes in this category.

4. The Requirement of Common Operation and Management for Supervised Heterogeneous Network Slice

Network slice can only be created with resource components that are available in its network. It is expected that different NSPs have various constraints and policies. The abstraction of resources is gathered from the network configuration models whose attributes are maintained by heterogeneous domain controllers/managers. Based on this information, a full set of resource capabilities is established. In order to provide network-slice-level OAM, it is extremely important for the heterogeneous transport network slice controller to visualize a network slice in terms of systematic association of these individual resources. This visualization model describes the network slice at a resource granularity and abstraction level that are provided by the underlay heterogeneous domain controllers and managers. It is a technology-unspecific model since how each partition of a network slice should map to specific resource capability is implementation-specific.

In addition, the heterogeneous transport network slice controller receives service delivery model from the cross-segment network slice manager with high level network slice requirements described in the network slice service profile. The service delivery model includes network slice service description in business view point. Before it can be mapped to resources capabilities, it has to be translated to a network-slice level abstraction in network resource view point, with the compatible granularity the is exposed from the underlay resource capabilities.

A common information model is required for the operation and management in the heterogeneous network slice controller to provide the above abilities. It acts as an intermediate guideline model, which provides comprehensive technology-unspecific description of a network slice. This model does not specify the allocation of certain partition of network slice to underlay technology domain or the mapping of certain protocol to network slice performance requirements. However, it is the fundamental model that specifies each node, link, function blocks and corresponding performance requirements in an established network slice.

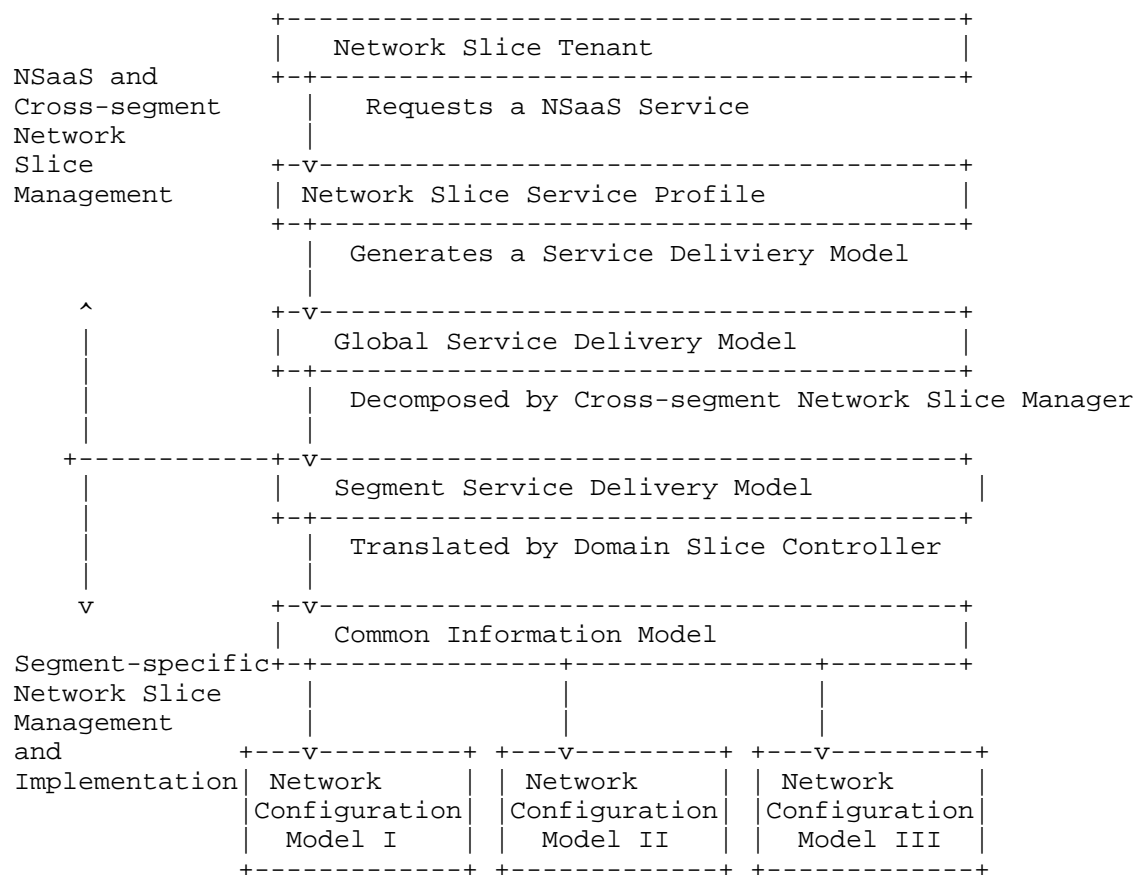


Figure 2: Common Information Model in Network Slicing

Figure 2 demonstrates the relationship between different models in the scope of supervised heterogeneous network slicing management. A service profile is created upon the request from a NST for a network slice service. This service profile is sent to cross-segment slice manager by global service delivery model. The global service delivery model is further decomposed to various segment service delivery model. In IETF's concern, the transport segment service delivery model is the starting point for a network-slice-aware service implementation.

for each network slice, and it comprehensively visualizes all the components topologies within the slice and corresponding functionality and performance parameters

In order to coordinate different technology domains to create the corresponding network slice, the common information model is mapped to network configuration models for different domains respectively.

The common operation and management is essential for network slicing since a network slice consist of resource components typically from diversity of management domains. There is yet a network-slice-aware approach that is able to orchestrate and coordinate these domains. The common operation and management of network slicing is designed for this purpose. At the same time, there are other requirements in terms of heterogeneous network slice management, which are enabled by this approach:

- o common resource negotiation and abstraction
- o exchanging information between multiple management domains
- o operations and monitoring across multiple management domains
- o network-slice operational control (i.e. life-cycle management) and management capability exposure to NST

4.1. Problem Scope

The common information model acts as the key element in common operation and management of network slicing. Foresee derivative including network slice monitoring, life cycle management, network slice interconnect, fault detection and protection mechanisms are also interesting and inevitable matters that supervised heterogeneous network slicing need to investigate.

More work is also needed to define a north interface for the slice controller, we can envision how the slice controller could interface with existing systems (e.g. using the NFV-MANO architecture as a guide), and by extension delimit the scope of the slice controller function.

In one scenario, the slice controller can interface with a virtual infrastructure manager (VIM), such as OpenStack. The slice controller can in this case be considered as a lower layer component of the VIM, or as a network management component controlled by the VIM. In another scenario, the slice controller can implement a VIM, and present an interface to an orchestrator.

5. Management of Heterogeneous Network Slice

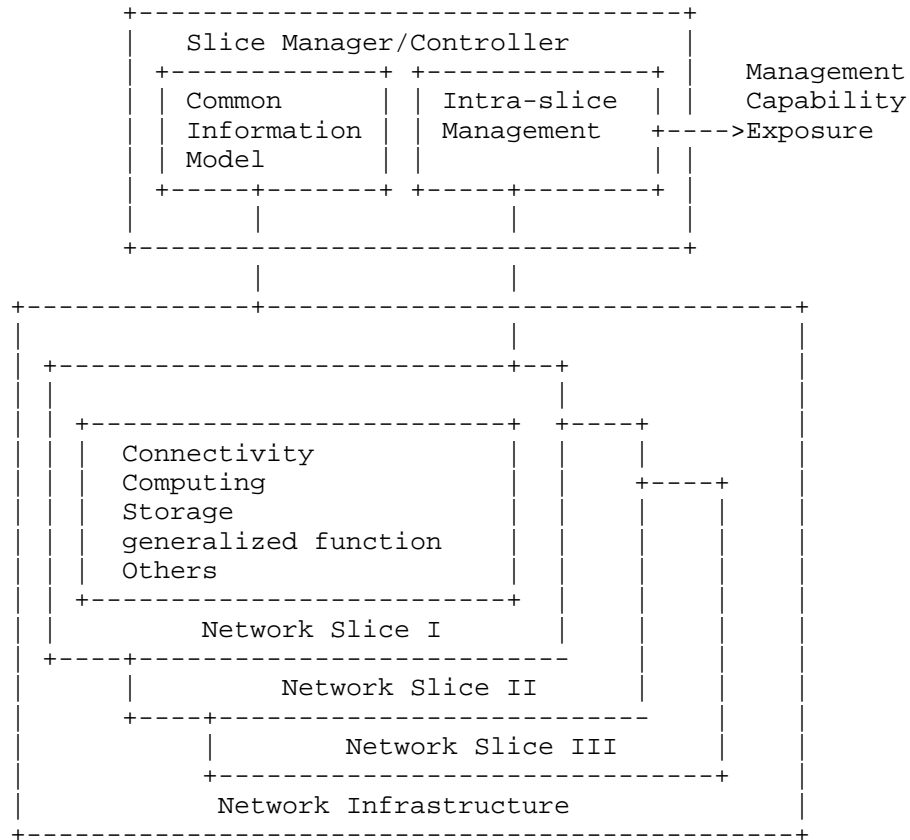


Figure 3: Common Information Model in Network Slicing

Given that common information models are set up for various network slice. The network-slice-level management is carried out by the slice controller accordingly. Meanwhile, an intra-network-slice controller is need for each network slice. The controller oversees the OAM of a single network slice and report to the heterogeneous transport network slice controller. As per agreement between NST and NSP, certain capabilities of intra-slice management may be exposed to NST. NST are authorized to use these capabilities to maintain its network slice in a view of dedicated networks and resources. The network slice-level information must not be exposed, which means the NST should not know he existence of any other network slices through intra-slice manager. The exposed controller capability should be supervised by the NSP, so that the network slice will not violate network slice-level policies.

6. IANA Considerations

This document makes no request of IANA.

7. Security Considerations

Each layer of the system has its own security requirements.

8. Acknowledgements

9. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.

Authors' Addresses

Liang Geng
China Mobile
Beijing
China

Email: gengliang@chinamobile.com

Lei Wang
China Mobile
Beijing
China

Email: wangleiyjy@chinamobile.com

Slawomir Kuklinski
Orange

Email: slawomir.kuklinski@orange.com

Li Qiang
Huawei Technologies
Huawei Campus, No. 156 Beiqing Rd.
Beijing 100095

Email: qiangli3@huawei.com

Satoru Matsushima
Softbank

Email: satoru.matsushima@g.softbank.co.jp

Alex Galis
University College London

Email: a.galis@ucl.ac.uk

Luis Miguel Contreras Murillo
Telefonica

Email: luismiguel.contrerasmurillo@telefonica.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 17, 2018

E. Lear
Cisco Systems
R. Droms
Google
D. Romascanu
June 15, 2018

Manufacturer Usage Description Specification
draft-ietf-opsawg-mud-25

Abstract

This memo specifies a component-based architecture for manufacturer usage descriptions (MUD). The goal of MUD is to provide a means for end devices to signal to the network what sort of access and network functionality they require to properly function. The initial focus is on access control. Later work can delve into other aspects.

This memo specifies two YANG modules, IPv4 and IPv6 DHCP options, an LLDP TLV, a URL, an X.509 certificate extension and a means to sign and verify the descriptions.

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 17, 2018.

Copyright Notice

Copyright (c) 2018 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
1.1. What MUD Doesn't Do	5
1.2. A Simple Example	5
1.3. Terminology	5
1.4. Determining Intended Use	6
1.5. Finding A Policy: The MUD URL	6
1.6. Processing of the MUD URL	7
1.7. Types of Policies	8
1.8. The Manufacturer Usage Description Architecture	10
1.9. Order of operations	11
2. The MUD Model and Semantic Meaning	12
2.1. The IETF-MUD YANG Module	13
3. MUD model definitions for the root mud container	14
3.1. mud-version	14
3.2. mud-url	15
3.3. to-device-policy and from-device-policy containers	15
3.4. last-update	15
3.5. cache-validity	15
3.6. is-supported	15
3.7. systeminfo	15
3.8. mfg-name, software-rev, model-name firmware-rev	16
3.9. extensions	16
4. Augmentation to the ACL Model	16
4.1. manufacturer	16
4.2. same-manufacturer	16
4.3. documentation	17
4.4. model	17
4.5. local-networks	17
4.6. controller	17
4.7. my-controller	18
4.8. direction-initiated	18
5. Processing of the MUD file	18
6. What does a MUD URL look like?	18
7. The MUD YANG Model	19
8. The Domain Name Extension to the ACL Model	25
8.1. src-dnsname	26
8.2. dst-dnsname	26
8.3. The ietf-acldns Model	26
9. MUD File Example	28

10. The MUD URL DHCP Option	30
10.1. Client Behavior	31
10.2. Server Behavior	31
10.3. Relay Requirements	32
11. The Manufacturer Usage Description (MUD) URL X.509 Extension	32
12. The Manufacturer Usage Description LLDP extension	34
13. Creating and Processing of Signed MUD Files	35
13.1. Creating a MUD file signature	36
13.2. Verifying a MUD file signature	36
14. Extensibility	37
15. Deployment Considerations	37
16. Security Considerations	38
17. IANA Considerations	40
17.1. YANG Module Registrations	41
17.2. DHCPv4 and DHCPv6 Options	41
17.3. PKIX Extensions	41
17.4. MIME Media-type Registration for MUD files	42
17.5. LLDP IANA TLV Subtype Registry	42
17.6. The MUD Well Known Universal Resource Name (URNs)	43
17.7. Extensions Registry	43
18. Acknowledgments	43
19. References	44
19.1. Normative References	44
19.2. Informative References	47
Appendix A. Changes from Earlier Versions	49
Appendix B. Default MUD nodes	52
Appendix C. A Sample Extension: DETNET-indicator	57
Authors' Addresses	60

1. Introduction

The Internet has largely been constructed for general purpose computers, those devices that may be used for a purpose that is specified by those who own the device. [RFC1984] presumed that an end device would be most capable of protecting itself. This made sense when the typical device was a workstation or a mainframe, and it continues to make sense for general purpose computing devices today, including laptops, smart phones, and tablets.

[RFC7452] discusses design patterns for, and poses questions about, smart objects. Let us then posit a group of objects that are specifically not intended to be used for general purpose computing tasks. These devices, which this memo refers to as Things, have a specific purpose. By definition, therefore, all other uses are not intended. If a small number of communication patterns follows from those small number of uses, the combination of these two statements can be restated as a manufacturer usage description (MUD) that can be applied at various points within a network. MUD primarily addresses

threats to the device rather than the device as a threat. In some circumstances, however, MUD may offer some protection in the latter case, depending on the MUD-URL is communicated, and how devices and their communications are authenticated.

We use the notion of "manufacturer" loosely in this context to refer to the entity or organization that will state how a device is intended to be used. For example, in the context of a lightbulb, this might indeed be the lightbulb manufacturer. In the context of a smarter device that has a built in Linux stack, it might be an integrator of that device. The key points are that the device itself is assumed to serve a limited purpose, and that there exists an organization in the supply chain of that device that will take responsibility for informing the network about that purpose.

The intent of MUD is to provide the following:

- o Substantially reduce the threat surface on a device to those communications intended by the manufacturer.
- o Provide a means to scale network policies to the ever-increasing number of types of devices in the network.
- o Provide a means to address at least some vulnerabilities in a way that is faster than the time it might take to update systems. This will be particularly true for systems that are no longer supported.
- o Keep the cost of implementation of such a system to the bare minimum.
- o Provide a means of extensibility for manufacturers to express other device capabilities or requirements.

MUD consists of three architectural building blocks:

- o A URL that can be used to locate a description;
- o The description itself, including how it is interpreted, and;
- o A means for local network management systems to retrieve the description.

MUD is most effective when the network is able to identify in some way the remote endpoints that Things will talk to.

In this specification we describe each of these building blocks and how they are intended to be used together. However, they may also be

used separately, independent of this specification, by local deployments for their own purposes.

1.1. What MUD Doesn't Do

MUD is not intended to address network authorization of general purpose computers, as their manufacturers cannot envision a specific communication pattern to describe. In addition, even those devices that have a single or small number of uses might have very broad communication patterns. MUD on its own is not for them either.

Although MUD can provide network administrators with some additional protection when device vulnerabilities exist, it will never replace the need for manufacturers to patch vulnerabilities.

Finally, no matter what the manufacturer specifies in a MUD file, these are not directives, but suggestions. How they are instantiated locally will depend on many factors and will be ultimately up to the local network administrator, who must decide what is appropriate in a given circumstances.

1.2. A Simple Example

A light bulb is intended to light a room. It may be remotely controlled through the network, and it may make use of a rendezvous service of some form that an application on a smart phone. What we can say about that light bulb, then, is that all other network access is unwanted. It will not contact a news service, nor speak to the refrigerator, and it has no need of a printer or other devices. It has no social networking friends. Therefore, an access list applied to it that states that it will only connect to the single rendezvous service will not impede the light bulb in performing its function, while at the same time allowing the network to provide both it and other devices an additional layer of protection.

1.3. Terminology

MUD: manufacturer usage description.

MUD file: a file containing YANG-based JSON that describes a Thing and associated suggested specific network behavior.

MUD file server: a web server that hosts a MUD file.

MUD manager: the system that requests and receives the MUD file from the MUD server. After it has processed a MUD file, it may direct changes to relevant network elements.

MUD controller: a synonym that has been used in the past for MUD manager.

MUD URL: a URL that can be used by the MUD manager to receive the MUD file.

Thing: the device emitting a MUD URL.

Manufacturer: the entity that configures the Thing to emit the MUD URL and the one who asserts a recommendation in a MUD file. The manufacturer might not always be the entity that constructs a Thing. It could, for instance, be a systems integrator, or even a component provider.

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

1.4. Determining Intended Use

The notion of intended use is in itself not new. Network administrators apply access lists every day to allow for only such use. This notion of white listing was well described by Chapman and Zwicky in [FW95]. Profiling systems that make use of heuristics to identify types of systems have existed for years as well.

A Thing could just as easily tell the network what sort of access it requires without going into what sort of system it is. This would, in effect, be the converse of [RFC7488]. In seeking a general solution, however, we assume that a device will implement functionality necessary to fulfill its limited purpose. This is basic economic constraint. Unless the network would refuse access to such a device, its developers would have no reason to provide the network any information. To date, such an assertion has held true.

1.5. Finding A Policy: The MUD URL

Our work begins with the device emitting a Universal Resource Locator (URL) [RFC3986]. This URL serves both to classify the device type and to provide a means to locate a policy file.

MUD URLs MUST use the HTTPS scheme [RFC7230].

In this memo three means are defined to emit the MUD URL, as follows:

- o A DHCP option[RFC2131],[RFC3315] that the DHCP client uses to inform the DHCP server. The DHCP server may take further actions, such as act as the MUD manager or otherwise pass the MUD URL along to the MUD manager.
- o An X.509 constraint. The IEEE has developed [IEEE8021AR] that provides a certificate-based approach to communicate device characteristics, which itself relies on [RFC5280]. The MUD URL extension is non-critical, as required by IEEE 802.1AR. Various means may be used to communicate that certificate, including Tunnel Extensible Authentication Protocol (TEAP) [RFC7170].
- o Finally, a Link Layer Discovery Protocol (LLDP) frame is defined [IEEE8021AB].

It is possible that there may be other means for a MUD URL to be learned by a network. For instance, some devices may already be fielded or have very limited ability to communicate a MUD URL, and yet can be identified through some means, such as a serial number or a public key. In these cases, manufacturers may be able to map those identifiers to particular MUD URLs (or even the files themselves). Similarly, there may be alternative resolution mechanisms available for situations where Internet connectivity is limited or does not exist. Such mechanisms are not described in this memo, but are possible. Implementors are encouraged to allow for this sort of flexibility of how MUD URLs may be learned.

1.6. Processing of the MUD URL

MUD managers that are able to do so SHOULD retrieve MUD URLs and signature files as per [RFC7230], using the GET method [RFC7231]. They MUST validate the certificate using the rules in [RFC2818], Section 3.1.

Requests for MUD URLs SHOULD include an "Accept" header ([RFC7231], Section 5.3.2) containing "application/mud+json", an "Accept-Language" header field ([RFC7231], Section 5.3.5), and a "User-Agent" header ([RFC7231], Section 5.5.3).

MUD managers SHOULD automatically process 3xx response status codes.

If a MUD manager is not able to fetch a MUD URL, other means MAY be used to import MUD files and associated signature files. So long as the signature of the file can be validated, the file can be used. In such environments, controllers SHOULD warn administrators when cache-validity expiry is approaching so that they may check for new files.

It may not be possible for a MUD manager to retrieve a MUD file at any given time. Should a MUD manager fail to retrieve a MUD file, it SHOULD consider the existing one safe to use, at least for a time. After some period, it SHOULD log that it has been unable to retrieve the file. There may be very good reasons for such failures, including the possibility that the MUD manager is in an off-line environment, the local Internet connection has failed, or the remote Internet connection has failed. It is also possible that an attacker is attempting to interfere with the deployment of a device. It is a local decision as to how to handle such circumstances.

1.7. Types of Policies

When the MUD URL is resolved, the MUD manager retrieves a file that describes what sort of communications a device is designed to have. The manufacturer may specify either specific hosts for cloud based services or certain classes for access within an operational network. An example of a class might be "devices of a specified manufacturer type", where the manufacturer type itself is indicated simply by the authority component (e.g, the domain name) of the MUD URL. Another example might be to allow or disallow local access. Just like other policies, these may be combined. For example:

- o Allow access to devices of the same manufacturer
- o Allow access to and from controllers via Constrained Application Protocol (COAP)[RFC7252]
- o Allow access to local DNS/NTP
- o Deny all other access

A printer might have a description that states:

- o Allow access for port IPP or port LPD
- o Allow local access for port HTTP
- o Deny all other access

In this way anyone can print to the printer, but local access would be required for the management interface.

The files that are retrieved are intended to be closely aligned to existing network architectures so that they are easy to deploy. We make use of YANG [RFC7950] because it provides accurate and adequate models for use by network devices. JSON[RFC8259] is used as a

serialization format for compactness and readability, relative to XML. Other formats may be chosen with later versions of MUD.

While the policy examples given here focus on access control, this is not intended to be the sole focus. By structuring the model described in this document with clear extension points, other descriptions could be included. One that often comes to mind is quality of service.

The YANG modules specified here are extensions of [I-D.ietf-netmod-acl-model]. The extensions to this model allow for a manufacturer to express classes of systems that a manufacturer would find necessary for the proper function of the device. Two modules are specified. The first module specifies a means for domain names to be used in ACLs so that devices that have their controllers in the cloud may be appropriately authorized with domain names, where the mapping of those names to addresses may rapidly change.

The other module abstracts away IP addresses into certain classes that are instantiated into actual IP addresses through local processing. Through these classes, manufacturers can specify how the device is designed to communicate, so that network elements can be configured by local systems that have local topological knowledge. That is, the deployment populates the classes that the manufacturer specifies. The abstractions below map to zero or more hosts, as follows:

Manufacturer: A device made by a particular manufacturer, as identified by the authority component of its MUD URL

same-manufacturer: Devices that have the same authority component of their MUD URL.

controller: Devices that the local network administrator admits to the particular class.

my-controller: Devices intended to serve as controllers for the MUD-URL that the Thing emitted.

local: The class of IP addresses that are scoped within some administrative boundary. By default it is suggested that this be the local subnet.

The "manufacturer" classes can be easily specified by the manufacturer, whereas controller classes are initially envisioned to be specified by the administrator.

Because manufacturers do not know who will be using their devices, it is important for functionality referenced in usage descriptions to be relatively ubiquitous and mature. For these reasons the YANG-based configuration in a MUD file is limited to either the modules specified or referenced in this document, or those specified in documented extensions.

1.8. The Manufacturer Usage Description Architecture

With these components laid out we now have the basis for an architecture. This leads us to ASCII art.

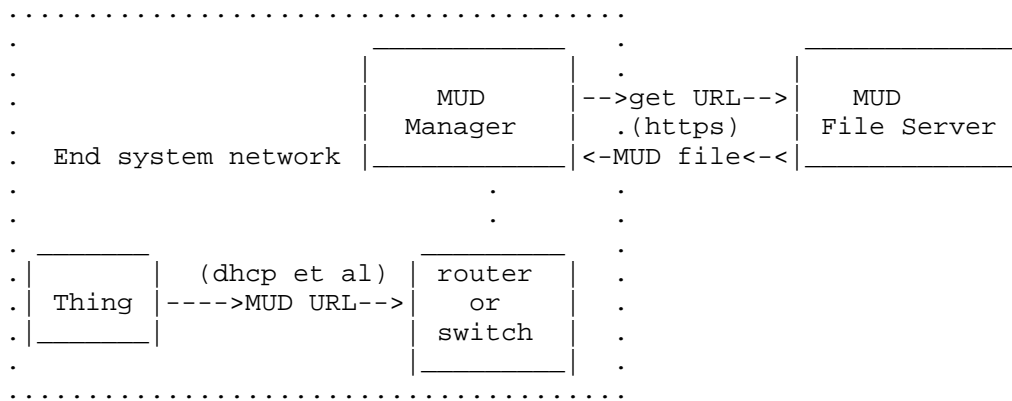


Figure 1: MUD Architecture

In the above diagram, the switch or router collects MUD URLs and forwards them to the MUD manager (a network management system) for processing. This happens in different ways, depending on how the URL is communicated. For instance, in the case of DHCP, the DHCP server might receive the URL and then process it. In the case of IEEE 802.1X [IEEE8021X], the switch would carry the URL via a certificate to the authentication server via EAP over Radius[RFC3748], which would then process it. One method to do this is TEAP, described in [RFC7170]. The certificate extension is described below.

The information returned by the MUD file server is valid for as long as the Thing is connected. There is no expiry. However, if the MUD manager has detected that the MUD file for a Thing has changed, it SHOULD update the policy expeditiously, taking into account whatever approval flow is required in a deployment. In this way, new recommendations from the manufacturer can be processed in a timely fashion.

The information returned by the MUD file server (a web server) is valid for the duration of the Thing's connection, or as specified in the description. Thus if the Thing is disconnected, any associated configuration in the switch can be removed. Similarly, from time to time the description may be refreshed, based on new capabilities or communication patterns or vulnerabilities.

The web server is typically run by or on behalf of the manufacturer. Its domain name is that of the authority found in the MUD URL. For legacy cases where Things cannot emit a URL, if the switch is able to determine the appropriate URL, it may proxy it. In the trivial case it may hardcode MUD-URL on a switch port or a map from some available identifier such as an L2 address or certificate hash to a MUD-URL.

The role of the MUD manager in this environment is to do the following:

- o receive MUD URLs,
- o fetch MUD files,
- o translate abstractions in the MUD files to specific network element configuration,
- o maintain and update any required mappings of the abstractions, and
- o update network elements with appropriate configuration.

A MUD manager may be a component of a AAA or network management system. Communication within those systems and from those systems to network elements is beyond the scope of this memo.

1.9. Order of operations

As mentioned above, MUD contains architectural building blocks, and so order of operation may vary. However, here is one clear intended example:

1. Thing emits URL.
2. That URL is forwarded to a MUD manager by the nearest switch (how this happens depends on the way in which the MUD URL is emitted).
3. The MUD manager retrieves the MUD file and signature from the MUD file server, assuming it doesn't already have copies. After validating the signature, it may test the URL against a web or domain reputation service, and it may test any hosts within the file against those reputation services, as it deems fit.

4. The MUD manager may query the administrator for permission to add the Thing and associated policy. If the Thing is known or the Thing type is known, it may skip this step.
5. The MUD manager instantiates local configuration based on the abstractions defined in this document.
6. The MUD manager configures the switch nearest the Thing. Other systems may be configured as well.
7. When the Thing disconnects, policy is removed.

2. The MUD Model and Semantic Meaning

A MUD file consists of a YANG model instance that has been serialized in JSON [RFC7951]. For purposes of MUD, the nodes that can be modified are access lists as augmented by this model. The MUD file is limited to the serialization of only the following YANG schema:

- o ietf-access-control-list [I-D.ietf-netmod-acl-model]
- o ietf-mud (this document)
- o ietf-acldns (this document)

Extensions may be used to add additional schema. This is described further on.

To provide the widest possible deployment, publishers of MUD files SHOULD make use of the abstractions in this memo and avoid the use of IP addresses. A MUD manager SHOULD NOT automatically implement any MUD file that contains IP addresses, especially those that might have local significance. The addressing of one side of an access list is implicit, based on whether it is applied as to-device-policy or from-device-policy.

With the exceptions of "name" of the ACL, "type", "name" of the ACE, and TCP and UDP source and destination port information, publishers of MUD files SHOULD limit the use of ACL model leaf nodes expressed to those found in this specification. Absent any extensions, MUD files are assumed to implement only the following ACL model features:

- o match-on-ipv4, match-on-ipv6, match-on-tcp, match-on-udp, match-on-icmp

Furthermore, only "accept" or "drop" actions SHOULD be included. A MUD manager MAY choose to interpret "reject" as "drop". A MUD manager SHOULD ignore all other actions. This is because

manufacturers do not have sufficient context within a local deployment to know whether reject is appropriate. That is a decision that should be left to a network administrator.

Given that MUD does not deal with interfaces, the support of the "ietf-interfaces" module [RFC8343] is not required. Specifically, the support of interface-related features and branches (e.g., interface-attachment and interface-stats) of the ACL YANG module is not required.

In fact, MUD managers MAY ignore any particular component of a description or MAY ignore the description in its entirety, and SHOULD carefully inspect all MUD descriptions. Publishers of MUD files MUST NOT include other nodes except as described in Section 3.9. See that section for more information.

2.1. The IETF-MUD YANG Module

This module is structured into three parts:

- o The first component, the "mud" container, holds information that is relevant to retrieval and validity of the MUD file itself, as well as policy intended to and from the Thing.
- o The second component augments the matching container of the ACL model to add several nodes that are relevant to the MUD URL, or otherwise abstracted for use within a local environment.
- o The third component augments the tcp-acl container of the ACL model to add the ability to match on the direction of initiation of a TCP connection.

A valid MUD file will contain two root objects, a "mud" container and an "acls" container. Extensions may add additional root objects as required. As a reminder, when parsing acls, elements within a "match" block are logically ANDed. In general, a single abstraction in a match statement should be used. For instance, it makes little sense to match both "my-controller" and "controller" with an argument, since they are highly unlikely to be the same value.

A simplified graphical representation of the data models is used in this document. The meaning of the symbols in these diagrams is explained in [RFC8340].

```

module: ietf-mud
  +--rw mud!
    +--rw mud-version          uint8
    +--rw mud-url              inet:uri
    +--rw last-update          yang:date-and-time
    +--rw mud-signature?       inet:uri
    +--rw cache-validity?      uint8
    +--rw is-supported         boolean
    +--rw systeminfo?          string
    +--rw mfg-name?            string
    +--rw model-name?          string
    +--rw firmware-rev?        string
    +--rw software-rev?        string
    +--rw documentation?       inet:uri
    +--rw extensions*          string
    +--rw from-device-policy
      | +--rw acls
      | | +--rw access-list* [name]
      | | | +--rw name      -> /acl:acls/acl/name
      | +--rw to-device-policy
      | | +--rw acls
      | | | +--rw access-list* [name]
      | | | | +--rw name      -> /acl:acls/acl/name
    augment /acl:acls/acl:acl/acl:aces/acl:ace/acl:matches:
      +--rw mud
        +--rw manufacturer?    inet:host
        +--rw same-manufacturer? empty
        +--rw model?           inet:uri
        +--rw local-networks?  empty
        +--rw controller?      inet:uri
        +--rw my-controller?    empty
      augment
        /acl:acls/acl:acl/acl:aces/acl:ace/acl:matches
        /acl:l4/acl:tcp/acl:tcp:
          +--rw direction-initiated? direction

```

3. MUD model definitions for the root mud container

3.1. mud-version

This node specifies the integer version of the MUD specification.
 This memo specifies version 1.

3.2. mud-url

This URL identifies the MUD file. This is useful when the file and associated signature are manually uploaded, say, in an offline mode.

3.3. to-device-policy and from-device-policy containers

[I-D.ietf-netmod-acl-model] describes access-lists. In the case of MUD, a MUD file must be explicit in describing the communication pattern of a Thing, and that includes indicating what is to be permitted or denied in either direction of communication. Hence each of these containers indicates the appropriate direction of a flow in association with a particular Thing. They contain references to specific access-lists.

3.4. last-update

This is a date-and-time value of when the MUD file was generated. This is akin to a version number. Its form is taken from [RFC6991] which, for those keeping score, in turn was taken from Section 5.6 of [RFC3339], which was taken from [ISO.8601.1988].

3.5. cache-validity

This uint8 is the period of time in hours that a network management station MUST wait since its last retrieval before checking for an update. It is RECOMMENDED that this value be no less than 24 and MUST NOT be more than 168 for any Thing that is supported. This period SHOULD be no shorter than any period determined through HTTP caching directives (e.g., "cache-control" or "Expires"). N.B., expiring of this timer does not require the MUD manager to discard the MUD file, nor terminate access to a Thing. See Section 16 for more information.

3.6. is-supported

This boolean is an indication from the manufacturer to the network administrator as to whether or not the Thing is supported. In this context a Thing is said to not be supported if the manufacturer intends never to issue a firmware or software update to the Thing or never update the MUD file. A MUD manager MAY still periodically check for updates.

3.7. systeminfo

This is a textual UTF-8 description of the Thing to be connected. The intent is for administrators to be able to see a brief

displayable description of the Thing. It SHOULD NOT exceed 60 characters worth of display space.

3.8. mfg-name, software-rev, model-name firmware-rev

These optional fields are filled in as specified by [RFC8348]. Note that firmware-rev and software-rev MUST NOT be populated in a MUD file if the device can be upgraded but the MUD-URL cannot be. This would be the case, for instance, with MUD-URLs that are contained in 802.1AR certificates.

3.9. extensions

This optional leaf-list names MUD extensions that are used in the MUD file. Note that MUD extensions MUST NOT be used in a MUD file without the extensions being declared. Implementations MUST ignore any node in this file that they do not understand.

Note that extensions can either extend the MUD file as described in the previous paragraph, or they might reference other work. An extension example can be found in Appendix C.

4. Augmentation to the ACL Model

Note that in this section, when we use the term "match" we are referring to the ACL model "matches" node.

4.1. manufacturer

This node consists of a hostname that would be matched against the authority component of another Thing's MUD URL. In its simplest form "manufacturer" and "same-manufacturer" may be implemented as access-lists. In more complex forms, additional network capabilities may be used. For example, if one saw the line "manufacturer" : "flobbidy.example.com", then all Things that registered with a MUD URL that contained flobbity.example.com in its authority section would match.

4.2. same-manufacturer

This null-valued node is an equivalent for when the manufacturer element is used to indicate the authority that is found in another Thing's MUD URL matches that of the authority found in this Thing's MUD URL. For example, if the Thing's MUD URL were <https://bl.example.com/ThingV1>, then all devices that had MUD URL with an authority section of bl.example.com would match.

4.3. documentation

This URI consists of a URL that points to documentation relating to the device and the MUD file. This can prove particularly useful when the "controller" class is used, so that its use can be explained.

4.4. model

This string matches the entire MUD URL, thus covering the model that is unique within the context of the authority. It may contain not only model information, but versioning information as well, and any other information that the manufacturer wishes to add. The intended use is for devices of this precise class to match, to permit or deny communication between one another.

4.5. local-networks

This null-valued node expands to include local networks. Its default expansion is that packets must not traverse toward a default route that is received from the router. However, administrators may expand the expression as is appropriate in their deployments.

4.6. controller

This URI specifies a value that a controller will register with the MUD manager. The node then is expanded to the set of hosts that are so registered. This node may also be a URN. In this case, the URN describes a well known service, such as DNS or NTP, that has been standardized. Both of those URNs may be found in Section 17.6.

When "my-controller" is used, it is possible that the administrator will be prompted to populate that class for each and every model. Use of "controller" with a named class allows the user to populate that class only once for many different models that a manufacturer may produce.

Controller URIs MAY take the form of a URL (e.g. "http[s]://"). However, MUD managers MUST NOT resolve and retrieve such files, and it is RECOMMENDED that there be no such file at this time, as their form and function may be defined at a point in the future. For now, URLs should serve simply as class names and may be populated by the local deployment administrator.

Great care should be taken by MUD managers when invoking the controller class in the form of URLs. For one thing, it requires some understanding by the administrator as to when it is appropriate. Pre-registration in such classes by controllers with the MUD server

is encouraged. The mechanism to do that is beyond the scope of this work.

4.7. my-controller

This null-valued node signals to the MUD manager to use whatever mapping it has for this MUD URL to a particular group of hosts. This may require prompting the administrator for class members. Future work should seek to automate membership management.

4.8. direction-initiated

This MUST only be applied to TCP. This matches the direction in which a TCP connection is initiated. When direction initiated is "from-device", packets that are transmitted in the direction of a thing MUST be dropped unless the thing has first initiated a TCP connection. By way of example, this node may be implemented in its simplest form by looking at naked SYN bits, but may also be implemented through more stateful mechanisms.

When applied this matches packets when the flow was initiated in the corresponding direction. [RFC6092] specifies IPv6 guidance best practices. While that document is scoped specifically to IPv6, its contents are applicable for IPv4 as well.

5. Processing of the MUD file

To keep things relatively simple in addition to whatever definitions exist, we also apply two additional default behaviors:

- o Anything not explicitly permitted is denied.
- o Local DNS and NTP are, by default, permitted to and from the Thing.

An explicit description of the defaults can be found in Appendix B. These are applied AFTER all other explicit rules. Thus, a default behavior can be changed with a "drop" action.

6. What does a MUD URL look like?

MUD URLs are required to use the HTTPS scheme, in order to establish the MUD file server's identity and assure integrity of the MUD file.

Any "https://" URL can be a MUD URL. For example:

```
https://things.example.org/product_abc123/v5
https://www.example.net/mudfiles/temperature_sensor/
https://example.com/lightbulbs/colour/v1
```

A manufacturer may construct a MUD URL in any way, so long as it makes use of the "https" schema.

7. The MUD YANG Model

```
<CODE BEGINS>file "ietf-mud@2018-06-15.yang"
module ietf-mud {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-mud";
  prefix ietf-mud;

  import ietf-access-control-list {
    prefix acl;
  }
  import ietf-yang-types {
    prefix yang;
  }
  import ietf-inet-types {
    prefix inet;
  }

  organization
    "IETF OPSAWG (Ops Area) Working Group";
  contact
    "WG Web: http://tools.ietf.org/wg/opsawg/
    WG List: opsawg@ietf.org
    Author: Eliot Lear
    lear@cisco.com
    Author: Ralph Droms
    rdroms@gmail.com
    Author: Dan Romascanu
    dromasca@gmail.com

    ";
  description
    "This YANG module defines a component that augments the
    IETF description of an access list.  This specific module
    focuses on additional filters that include local, model,
    and same-manufacturer.

    This module is intended to be serialized via JSON and stored
    as a file, as described in RFC XXXX [RFC Editor to fill in with
    this document #]."
```

Copyright (c) 2016,2017 IETF Trust and the persons identified as the document authors. 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 RFC XXXX; see the RFC itself for full legal notices.";

```
revision 2018-06-15 {
  description
    "Initial proposed standard.";
  reference
    "RFC XXXX: Manufacturer Usage Description
    Specification";
}

typedef direction {
  type enumeration {
    enum to-device {
      description
        "packets or flows destined to the target
        Thing";
    }
    enum from-device {
      description
        "packets or flows destined from
        the target Thing";
    }
  }
  description
    "Which way are we talking about?";
}

container mud {
  presence "Enabled for this particular MUD URL";
  description
    "MUD related information, as specified
    by RFC-XXXX [RFC Editor to fill in].";
  uses mud-grouping;
}

grouping mud-grouping {
  description
    "Information about when support end(ed), and
    when to refresh";
```

```
leaf mud-version {
  type uint8;
  mandatory true;
  description
    "This is the version of the MUD
    specification.  This memo specifies version 1.";
}
leaf mud-url {
  type inet:uri;
  mandatory true;
  description
    "This is the MUD URL associated with the entry found
    in a MUD file.";
}
leaf last-update {
  type yang:date-and-time;
  mandatory true;
  description
    "This is intended to be when the current MUD file
    was generated.  MUD Managers SHOULD NOT check
    for updates between this time plus cache validity";
}
leaf mud-signature {
  type inet:uri;
  description
    "A URI that resolves to a signature as
    described in this specification.";
}
leaf cache-validity {
  type uint8 {
    range "1..168";
  }
  units "hours";
  default "48";
  description
    "The information retrieved from the MUD server is
    valid for these many hours, after which it should
    be refreshed.  N.B. MUD manager implementations
    need not discard MUD files beyond this period.";
}
leaf is-supported {
  type boolean;
  mandatory true;
  description
    "This boolean indicates whether or not the Thing is
    currently supported by the manufacturer.";
}
leaf systeminfo {
```

```
    type string;
    description
        "A UTF-8 description of this Thing.  This
        should be a brief description that may be
        displayed to the user to determine whether
        to allow the Thing on the
        network.";
}
leaf mfg-name {
    type string;
    description
        "Manufacturer name, as described in
        the ietf-hardware YANG module.";
}
leaf model-name {
    type string;
    description
        "Model name, as described in the
        ietf-hardware YANG module.";
}
leaf firmware-rev {
    type string;
    description
        "firmware-rev, as described in the
        ietf-hardware YANG module.  Note this field MUST
        NOT be included when the device can be updated
        but the MUD-URL cannot.";
}
leaf software-rev {
    type string;
    description
        "software-rev, as described in the
        ietf-hardware YANG module.  Note this field MUST
        NOT be included when the device can be updated
        but the MUD-URL cannot.";
}
leaf documentation {
    type inet:uri;
    description
        "This URL points to documentation that
        relates to this device and any classes that it uses
        in its MUD file.  A caution: MUD managers need
        not resolve this URL on their own, but rather simply
        provide it to the administrator.  Parsing HTML is
        not an intended function of a MUD manager.";
}
leaf-list extensions {
    type string {
```

```
        length "1..40";
    }
    description
        "A list of extension names that are used in this MUD
        file. Each name is registered with the IANA and
        described in an RFC.";
    }
    container from-device-policy {
        description
            "The policies that should be enforced on traffic
            coming from the device. These policies are not
            necessarily intended to be enforced at a single
            point, but may be rendered by the controller to any
            relevant enforcement points in the network or
            elsewhere.";
        uses access-lists;
    }
    container to-device-policy {
        description
            "The policies that should be enforced on traffic
            going to the device. These policies are not
            necessarily intended to be enforced at a single
            point, but may be rendered by the controller to any
            relevant enforcement points in the network or
            elsewhere.";
        uses access-lists;
    }
}

grouping access-lists {
    description
        "A grouping for access lists in the context of device
        policy.";
    container access-lists {
        description
            "The access lists that should be applied to traffic
            to or from the device.";
        list access-list {
            key "name";
            description
                "Each entry on this list refers to an ACL that
                should be present in the overall access list
                data model. Each ACL is identified by name and
                type.";
            leaf name {
                type leafref {
                    path "/acl:acls/acl:acl/acl:name";
                }
            }
        }
    }
}
```

```
        description
          "The name of the ACL for this entry.";
      }
  }
}

augment "/acl:acls/acl:acl/acl:aces/acl:ace/acl:matches" {
  description
    "adding abstractions to avoid need of IP addresses";
  container mud {
    description
      "MUD-specific matches.";
    leaf manufacturer {
      type inet:host;
      description
        "A domain that is intended to match the authority
        section of the MUD URL. This node is used to specify
        one or more manufacturers a device should
        be authorized to access.";
    }
    leaf same-manufacturer {
      type empty;
      description
        "This node matches the authority section of the MUD URL
        of a Thing. It is intended to grant access to all
        devices with the same authority section.";
    }
    leaf model {
      type inet:uri;
      description
        "Devices of the specified model type will match if
        they have an identical MUD URL.";
    }
    leaf local-networks {
      type empty;
      description
        "IP addresses will match this node if they are
        considered local addresses. A local address may be
        a list of locally defined prefixes and masks
        that indicate a particular administrative scope.";
    }
    leaf controller {
      type inet:uri;
      description
        "This node names a class that has associated with it
        zero or more IP addresses to match against. These
        may be scoped to a manufacturer or via a standard
```

```

        URN.";
    }
    leaf my-controller {
        type empty;
        description
            "This node matches one or more network elements that
             have been configured to be the controller for this
             Thing, based on its MUD URL.";
    }
}
}
}
augment "/acl:acls/acl:acl/acl:aces/acl:ace/acl:matches" +
    "/acl:l4/acl:tcp/acl:tcp" {
    description
        "add direction-initiated";
    leaf direction-initiated {
        type direction;
        description
            "This node matches based on which direction a
             connection was initiated. The means by which that
             is determined is discussed in this document.";
    }
}
}
}
}
<CODE ENDS>

```

8. The Domain Name Extension to the ACL Model

This module specifies an extension to IETF-ACL model such that domain names may be referenced by augmenting the "matches" node. Different implementations may deploy differing methods to maintain the mapping between IP address and domain name, if indeed any are needed. However, the intent is that resources that are referred to using a name should be authorized (or not) within an access list.

The structure of the change is as follows:

```

module: ietf-acldns
    augment /acl:acls/acl:acl/acl:aces/acl:ace/
        acl:matches/acl:l3/acl:ipv4/acl:ipv4:
            +--rw src-dnsname?    inet:host
            +--rw dst-dnsname?    inet:host
    augment /acl:acls/acl:acl/acl:aces/acl:ace/
        acl:matches/acl:l3/acl:ipv6/acl:ipv6:
            +--rw src-dnsname?    inet:host
            +--rw dst-dnsname?    inet:host

```


The choice of these particular points in the access-list model is based on the assumption that we are in some way referring to IP-related resources, as that is what the DNS returns. A domain name in our context is defined in [RFC6991]. The augmentations are replicated across IPv4 and IPv6 to allow MUD file authors the ability to control the IP version that the Thing may utilize.

The following node are defined.

8.1. src-dnsname

The argument corresponds to a domain name of a source as specified by inet:host. A number of means may be used to resolve hosts. What is important is that such resolutions be consistent with ACLs required by Things to properly operate.

8.2. dst-dnsname

The argument corresponds to a domain name of a destination as specified by inet:host See the previous section relating to resolution.

Note when using either of these with a MUD file, because access is associated with a particular Thing, MUD files MUST NOT contain either a src-dnsname in an ACL associated with from-device-policy or a dst-dnsname associated with to-device-policy.

8.3. The ietf-acldns Model

```
<CODE BEGINS>file "ietf-acldns@2018-06-15.yang"
module ietf-acldns {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-acldns";
  prefix ietf-acldns;

  import ietf-access-control-list {
    prefix acl;
  }
  import ietf-inet-types {
    prefix inet;
  }

  organization
    "IETF OPSAWG (Ops Area) Working Group";
  contact
    "WG Web: http://tools.ietf.org/wg/opsawg/
    WG List: opsawg@ietf.org
    Author: Eliot Lear
```

```
    lear@cisco.com
    Author: Ralph Droms
    rdroms@gmail.com
    Author: Dan Romascanu
    dromasca@gmail.com
    ";
description
    "This YANG module defines a component that augments the
    IETF description of an access list to allow DNS names
    as matching criteria.";

revision 2018-06-15 {
    description
        "Base version of dnsname extension of ACL model";
    reference
        "RFC XXXX: Manufacturer Usage Description
        Specification";
}

grouping dns-matches {
    description
        "Domain names for matching.";
    leaf src-dnsname {
        type inet:host;
        description
            "domain name to be matched against";
    }
    leaf dst-dnsname {
        type inet:host;
        description
            "domain name to be matched against";
    }
}

augment "/acl:acls/acl:acl/acl:aces/acl:ace/acl:matches" +
"/acl:l3/acl:ipv4/acl:ipv4" {
    description
        "Adding domain names to matching";
    uses dns-matches;
}
augment "/acl:acls/acl:acl/acl:aces/acl:ace/acl:matches" +
"/acl:l3/acl:ipv6/acl:ipv6" {
    description
        "Adding domain names to matching";
    uses dns-matches;
}
}
<CODE ENDS>
```

9. MUD File Example

This example contains two access lists that are intended to provide outbound access to a cloud service on TCP port 443.

```
{
  "ietf-mud:mud": {
    "mud-version": 1,
    "mud-url": "https://lighting.example.com/lightbulb2000",
    "last-update": "2018-03-02T11:20:51+01:00",
    "cache-validity": 48,
    "is-supported": true,
    "systeminfo": "The BMS Example Lightbulb",
    "from-device-policy": {
      "access-lists": {
        "access-list": [
          {
            "name": "mud-76100-v6fr"
          }
        ]
      }
    },
    "to-device-policy": {
      "access-lists": {
        "access-list": [
          {
            "name": "mud-76100-v6to"
          }
        ]
      }
    }
  },
  "ietf-access-control-list:acls": {
    "acl": [
      {
        "name": "mud-76100-v6to",
        "type": "ipv6-acl-type",
        "aces": {
          "ace": [
            {
              "name": "cl0-todev",
              "matches": {
                "ipv6": {
                  "ietf-acldns:src-dnsname": "test.example.com",
                  "protocol": 6
                },
              },
              "tcp": {
                "ietf-mud:direction-initiated": "from-device",

```

```

        "source-port": {
            "operator": "eq",
            "port": 443
        }
    },
    "actions": {
        "forwarding": "accept"
    }
}
]
},
{
    "name": "mud-76100-v6fr",
    "type": "ipv6-acl-type",
    "aces": [
        "ace": [
            {
                "name": "cl0-frdev",
                "matches": {
                    "ipv6": {
                        "ietf-acldns:dst-dnsname": "test.example.com",
                        "protocol": 6
                    },
                    "tcp": {
                        "ietf-mud:direction-initiated": "from-device",
                        "destination-port": {
                            "operator": "eq",
                            "port": 443
                        }
                    }
                },
                "actions": {
                    "forwarding": "accept"
                }
            }
        ]
    }
}
]
```

In this example, two policies are declared, one from the Thing and the other to the Thing. Each policy names an access list that applies to the Thing, and one that applies from. Within each access

list, access is permitted to packets flowing to or from the Thing that can be mapped to the domain name of "service.bms.example.com". For each access list, the enforcement point should expect that the Thing initiated the connection.

10. The MUD URL DHCP Option

The IPv4 MUD URL client option has the following format:

```
+-----+-----+-----+
| code | len | MUDstring
+-----+-----+-----+
```

Code `OPTION_MUD_URL_V4` (161) is assigned by IANA. `len` is a single octet that indicates the length of MUD string in octets. The MUD string is defined as follows:

```
MUDstring = mudurl [ " " reserved ]
mudurl = URI; a URL [RFC3986] that uses the "https" schema [RFC7230]
reserved = 1*( OCTET ) ; from [RFC5234]
```

The entire option MUST NOT exceed 255 octets. If a space follows the MUD URL, a reserved string that will be defined in future specifications follows. MUD managers that do not understand this field MUST ignore it.

The IPv6 MUD URL client option has the following format:

```

0          1          2          3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|          OPTION_MUD_URL_V6          |          option-length          |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     MUDstring                             |
+-----+-----+-----+-----+-----+-----+-----+-----+
```

`OPTION_MUD_URL_V6` (112; assigned by IANA).

`option-length` contains the length of the MUDstring, as defined above, in octets.

The intent of this option is to provide both a new Thing classifier to the network as well as some recommended configuration to the routers that implement policy. However, it is entirely the purview

of the network system as managed by the network administrator to decide what to do with this information. The key function of this option is simply to identify the type of Thing to the network in a structured way such that the policy can be easily found with existing toolsets.

10.1. Client Behavior

A DHCPv4 client MAY emit a DHCPv4 option and a DHCPv6 client MAY emit DHCPv6 option. These options are singletons, as specified in [RFC7227]. Because clients are intended to have at most one MUD URL associated with them, they may emit at most one MUD URL option via DHCPv4 and one MUD URL option via DHCPv6. In the case where both v4 and v6 DHCP options are emitted, the same URL MUST be used.

10.2. Server Behavior

A DHCP server may ignore these options or take action based on receipt of these options. When a server consumes this option, it will either forward the URL and relevant client information (such as the gateway address or giaddr and requested IP address, and lease length) to a network management system, or it will retrieve the usage description itself by resolving the URL.

DHCP servers may implement MUD functionality themselves or they may pass along appropriate information to a network management system or MUD manager. A DHCP server that does process the MUD URL MUST adhere to the process specified in [RFC2818] and [RFC5280] to validate the TLS certificate of the web server hosting the MUD file. Those servers will retrieve the file, process it, create and install the necessary configuration on the relevant network element. Servers SHOULD monitor the gateway for state changes on a given interface. A DHCP server that does not provide MUD functionality and has forwarded a MUD URL to a MUD manager MUST notify the MUD manager of any corresponding change to the DHCP state of the client (such as expiration or explicit release of a network address lease).

Should the DHCP server fail, in the case when it implements the MUD manager functionality, any backup mechanisms SHOULD include the MUD state, and the server SHOULD resolve the status of clients upon its restart, similar to what it would do, absent MUD manager functionality. In the case where the DHCP server forwards information to the MUD manager, the MUD manager will either make use of redundant DHCP servers for information, or otherwise clear state based on other network information, such as monitoring port status on a switch via SNMP, Radius accounting, or similar mechanisms.

10.3. Relay Requirements

There are no additional requirements for relays.

11. The Manufacturer Usage Description (MUD) URL X.509 Extension

This section defines an X.509 non-critical certificate extension that contains a single Uniform Resource Locator (URL) that points to an on-line Manufacturer Usage Description concerning the certificate subject. URI must be represented as described in Section 7.4 of [RFC5280].

Any Internationalized Resource Identifiers (IRIs) MUST be mapped to URIs as specified in Section 3.1 of [RFC3987] before they are placed in the certificate extension.

The semantics of the URL are defined Section 6 of this document.

The choice of id-pe is based on guidance found in Section 4.2.2 of [RFC5280]:

These extensions may be used to direct applications to on-line information about the issuer or the subject.

The MUD URL is precisely that: online information about the particular subject.

In addition, a separate new extension is defined as id-pe-mudsigner. This contains the subject field of the signing certificate of the MUD file. Processing of this field is specified in Section 13.2.

The purpose of this signature is to make a claim that the MUD file found on the server is valid for a given device, independent of any other factors. There are several security considerations below in Section 16.

A new content-type id-ct-mud is also defined. While signatures are detached today, should a MUD file be transmitted as part of a CMS message, this content-type SHOULD be used.

The new extension is identified as follows:

<CODE BEGINS>

```
MUDURLExtnModule-2016 { iso(1) identified-organization(3) dod(6)
    internet(1) security(5) mechanisms(5) pkix(7)
    id-mod(0) id-mod-mudURLExtn2016(88) }
DEFINITIONS IMPLICIT TAGS ::= BEGIN
```

```
-- EXPORTS ALL --

IMPORTS

-- RFC 5912
EXTENSION
FROM PKIX-CommonTypes-2009
    { iso(1) identified-organization(3) dod(6) internet(1)
      security(5) mechanisms(5) pkix(7) id-mod(0)
      id-mod-pkixCommon-02(57) }

-- RFC 5912
id-ct
FROM PKIXCRMF-2009
    { iso(1) identified-organization(3) dod(6) internet(1)
      security(5) mechanisms(5) pkix(7) id-mod(0)
      id-mod-crmf2005-02(55) }

-- RFC 6268
CONTENT-TYPE
FROM CryptographicMessageSyntax-2010
    { iso(1) member-body(2) us(840) rsadsi(113549)
      pkcs(1) pkcs-9(9) smime(16) modules(0) id-mod-cms-2009(58) }

-- RFC 5912
id-pe, Name
FROM PKIX1Explicit-2009
    { iso(1) identified-organization(3) dod(6) internet(1)
      security(5) mechanisms(5) pkix(7) id-mod(0)
      id-mod-pkix1-explicit-02(51) } ;

--
-- Certificate Extensions
--

MUDCertExtensions EXTENSION ::=
    { ext-MUDURL | ext-MUDsigner, ... }

ext-MUDURL EXTENSION ::=
    { SYNTAX MUDURLSyntax IDENTIFIED BY id-pe-mud-url }

id-pe-mud-url OBJECT IDENTIFIER ::= { id-pe 25 }

MUDURLSyntax ::= IA5String

ext-MUDsigner EXTENSION ::=
    { SYNTAX MUDsignerSyntax IDENTIFIED BY id-pe-mudsigner }
```



```

id-pe-mudsigner OBJECT IDENTIFIER ::= { id-pe TBD1 }

MUDsignerSyntax ::= Name

--
-- CMS Content Types
--

MUDContentTypes CONTENT-TYPE ::=
    { ct-mud, ... }

ct-mud CONTENT-TYPE ::=
    { -- directly include the content
      IDENTIFIED BY id-ct-mudtype }
    -- The binary data that is in the form
    -- 'application/mud+json' is directly encoded as the
    -- signed data. No additional ASN.1 encoding is added.

id-ct-mudtype OBJECT IDENTIFIER ::= { id-ct TBD2 }

END
<CODE ENDS>

```

While this extension can appear in either an 802.AR manufacturer certificate (IDevID) or deployment certificate (LDevID), of course it is not guaranteed in either, nor is it guaranteed to be carried over. It is RECOMMENDED that MUD manager implementations maintain a table that maps a Thing to its MUD URL based on IDevIDs.

12. The Manufacturer Usage Description LLDP extension

The IEEE802.1AB Link Layer Discovery Protocol (LLDP) is a one hop vendor-neutral link layer protocol used by end hosts network Things for advertising their identity, capabilities, and neighbors on an IEEE 802 local area network. Its Type-Length-Value (TLV) design allows for 'vendor-specific' extensions to be defined. IANA has a registered IEEE 802 organizationally unique identifier (OUI) defined as documented in [RFC7042]. The MUD LLDP extension uses a subtype defined in this document to carry the MUD URL.

The LLDP vendor specific frame has the following format:

TLV Type	len	OUI	subtype	MUDString
=127		= 00 00 5E	= 1	
(7 bits)	(9 bits)	(3 octets)	(1 octet)	(1-255 octets)

where:

- o TLV Type = 127 indicates a vendor-specific TLV
- o len - indicates the TLV string length
- o OUI = 00 00 5E is the organizationally unique identifier of IANA
- o subtype = 1 (to be assigned by IANA for the MUD URL)
- o MUD URL - the length MUST NOT exceed 255 octets

The intent of this extension is to provide both a new Thing classifier to the network as well as some recommended configuration to the routers that implement policy. However, it is entirely the purview of the network system as managed by the network administrator to decide what to do with this information. The key function of this extension is simply to identify the type of Thing to the network in a structured way such that the policy can be easily found with existing toolsets.

Hosts, routers, or other network elements that implement this option are intended to have at most one MUD URL associated with them, so they may transmit at most one MUD URL value.

Hosts, routers, or other network elements that implement this option may ignore these options or take action based on receipt of these options. For example they may fill in information in the respective extensions of the LLDP Management Information Base (LLDP MIB). LLDP operates in a one-way direction. LLDPDUs are not exchanged as information requests by one Thing and response sent by another Thing. The other Things do not acknowledge LLDP information received from a Thing. No specific network behavior is guaranteed. When a Thing consumes this extension, it may either forward the URL and relevant remote Thing information to a MUD manager, or it will retrieve the usage description by resolving the URL in accordance with normal HTTP semantics.

13. Creating and Processing of Signed MUD Files

Because MUD files contain information that may be used to configure network access lists, they are sensitive. To ensure that they have not been tampered with, it is important that they be signed. We make use of DER-encoded Cryptographic Message Syntax (CMS) [RFC5652] for this purpose.

13.1. Creating a MUD file signature

A MUD file MUST be signed using CMS as an opaque binary object. In order to make successful verification more likely, intermediate certificates SHOULD be included. The signature is stored at the location specified in the MUD file. Signatures are transferred using content-type "application/pkcs7-signature".

For example:

```
% openssl cms -sign -signer mancrtfile -inkey mankey \  
-in mudfile -binary -outform DER -binary \  
-certfile intermediatecert -out mudfile.p7s
```

Note: A MUD file may need to be re-signed if the signature expires.

13.2. Verifying a MUD file signature

Prior to processing the rest of a MUD file, the MUD manager MUST retrieve the MUD signature file by retrieving the value of "mud-signature" and validating the signature across the MUD file. The Key Usage Extension in the signing certificate MUST be present and have the bit digitalSignature(0) set. When the id-pe-mudsigner extension is present in a device's X.509 certificate, the MUD signature file MUST have been generated by a certificate whose subject matches the contents of that id-pe-mudsigner extension. If these conditions are not met, or if it cannot validate the chain of trust to a known trust anchor, the MUD manager MUST cease processing the MUD file until an administrator has given approval.

The purpose of the signature on the file is to assign accountability to an entity, whose reputation can be used to guide administrators on whether or not to accept a given MUD file. It is already common place to check web reputation on the location of a server on which a file resides. While it is likely that the manufacturer will be the signer of the file, this is not strictly necessary, and may not be desirable. For one thing, in some environments, integrators may install their own certificates. For another, what is more important is the accountability of the recommendation, and not just the relationship between the Thing and the file.

An example:

```
% openssl cms -verify -in mudfile.p7s -inform DER -content mudfile
```

Note the additional step of verifying the common trust root.

14. Extensibility

One of our design goals is to see that MUD files are able to be understood by as broad a cross-section of systems as is possible. Coupled with the fact that we have also chosen to leverage existing mechanisms, we are left with no ability to negotiate extensions and a limited desire for those extensions in any event. A such, a two-tier extensibility framework is employed, as follows:

1. At a coarse grain, a protocol version is included in a MUD URL. This memo specifies MUD version 1. Any and all changes are entertained when this version is bumped. Transition approaches between versions would be a matter for discussion in future versions.
2. At a finer grain, only extensions that would not incur additional risk to the Thing are permitted. Specifically, adding nodes to the mud container is permitted with the understanding that such additions will be ignored by unaware implementations. Any such extensions SHALL be standardized through the IETF process, and MUST be named in the "extensions" list. MUD managers MUST ignore YANG nodes they do not understand and SHOULD create an exception to be resolved by an administrator, so as to avoid any policy inconsistencies.

15. Deployment Considerations

Because MUD consists of a number of architectural building blocks, it is possible to assemble different deployment scenarios. One key aspect is where to place policy enforcement. In order to protect the Thing from other Things within a local deployment, policy can be enforced on the nearest switch or access point. In order to limit unwanted traffic within a network, it may also be advisable to enforce policy as close to the Internet as possible. In some circumstances, policy enforcement may not be available at the closest hop. At that point, the risk of lateral infection (infection of devices that reside near one another) is increased to the number of Things that are able to communicate without protection.

A caution about some of the classes: admission of a Thing into the "manufacturer" and "same-manufacturer" class may have impact on access of other Things. Put another way, the admission may grow the access-list on switches connected to other Things, depending on how access is managed. Some care should be given on managing that access-list growth. Alternative methods such as additional network segmentation can be used to keep that growth within reason.

Because as of this writing MUD is a new concept, one can expect a great many devices to not have implemented it. It remains a local deployment decision as to whether a device that is first connected should be allowed broad or limited access. Furthermore, as mentioned in the introduction, a deployment may choose to ignore a MUD policy in its entirety, but simply taken into account the MUD URL as a classifier to be used as part of a local policy decision.

Finally, please see directly below regarding device lifetimes and use of domain names.

16. Security Considerations

Based on how a MUD URL is emitted, a Thing may be able to lie about what it is, thus gaining additional network access. This can happen in a number of ways when a device emits a MUD URL using DHCP or LLDP, such as being inappropriately admitted to a class such as "same-manufacturer", given access to a device such as "my-controller", or being permitted access to an Internet resource, where such access would otherwise be disallowed. Whether that is the case will depend on the deployment. Implementations SHOULD be configurable to disallow additive access for devices using MUD-URLs that are not emitted in a secure fashion such as in a certificate. Similarly, implementations SHOULD NOT grant elevated permissions (beyond those of devices presenting no MUD policy) to devices which do not strongly bind their identity to their L2/L3 transmissions. When insecure methods are used by the MUD Manager, the classes SHOULD NOT contain devices that use both insecure and secure methods, in order to prevent privilege escalation attacks, and MUST NOT contain devices with the same MUD-URL that are derived from both strong and weak authentication methods.

Devices may forge source (L2/L3) information. Deployments should apply appropriate protections to bind communications to the authentication that has taken place. For 802.1X authentication, IEEE 802.1AE (MACsec) [IEEE8021AE] is one means by which this may happen. A similar approach can be used with 802.11i (WPA2) [IEEE80211i]. Other means are available with other lower layer technologies. Implementations using session-oriented access that is not cryptographically bound should take care to remove state when any form of break in the session is detected.

A rogue CA may sign a certificate that contains the same subject name as is listed in the MUDsigner field in the manufacturer certificate, thus seemingly permitting a substitute MUD file for a device. There are two mitigations available: first, if the signer changes, this may be flagged as an exception by the MUD manager. If the MUD file also changes, the MUD manager SHOULD seek administrator approval (it

should do this in any case). In all circumstances, the MUD manager MUST maintain a cache of trusted CAs for this purpose. When such a rogue is discovered, it SHOULD be removed.

Additional mitigations are described below.

When certificates are not present, Things claiming to be of a certain manufacturer SHOULD NOT be included in that manufacturer grouping without additional validation of some form. This will be relevant when the MUD manager makes use of primitives such as "manufacturer" for the purpose of accessing Things of a particular type. Similarly, network management systems may be able to fingerprint the Thing. In such cases, the MUD URL can act as a classifier that can be proven or disproven. Fingerprinting may have other advantages as well: when 802.1AR certificates are used, because they themselves cannot change, fingerprinting offers the opportunity to add artifacts to the MUD string in the form of the reserved field discussed in Section 10. The meaning of such artifacts is left as future work.

MUD managers SHOULD NOT accept a usage description for a Thing with the same MAC address that has indicated a change of the URL authority without some additional validation (such as review by a network administrator). New Things that present some form of unauthenticated MUD URL SHOULD be validated by some external means when they would be given increased network access.

It may be possible for a rogue manufacturer to inappropriately exercise the MUD file parser, in order to exploit a vulnerability. There are three recommended approaches to address this threat. The first is to validate that the signer of the MUD file is known to and trusted by the MUD manager. The second is to have a system do a primary scan of the file to ensure that it is both parseable and believable at some level. MUD files will likely be relatively small, to start with. The number of ACEs used by any given Thing should be relatively small as well. It may also be useful to limit retrieval of MUD URLs to only those sites that are known to have decent web or domain reputations.

Use of a URL necessitates the use of domain names. If a domain name changes ownership, the new owner of that domain may be able to provide MUD files that MUD managers would consider valid. There are a few approaches that can mitigate this attack. First, MUD managers SHOULD cache certificates used by the MUD file server. When a new certificate is retrieved for whatever reason, the MUD manager should check to see if ownership of the domain has changed. A fair programmatic approximation of this is when the name servers for the domain have changed. If the actual MUD file has changed, the MUD manager MAY check the WHOIS database to see if registration ownership

of a domain has changed. If a change has occurred, or if for some reason it is not possible to determine whether ownership has changed, further review may be warranted. Note, this remediation does not take into account the case of a Thing that was produced long ago and only recently fielded, or the case where a new MUD manager has been installed.

The release of a MUD URL by a Thing reveals what the Thing is, and provides an attacker with guidance on what vulnerabilities may be present.

While the MUD URL itself is not intended to be unique to a specific Thing, the release of the URL may aid an observer in identifying individuals when combined with other information. This is a privacy consideration.

In addressing both of these concerns, implementors should take into account what other information they are advertising through mechanisms such as mDNS[RFC6872], how a Thing might otherwise be identified, perhaps through how it behaves when it is connected to the network, whether a Thing is intended to be used by individuals or carry personal identifying information, and then apply appropriate data minimization techniques. One approach is to make use of TEAP [RFC7170] as the means to share information with authorized components in the network. Network elements may also assist in limiting access to the MUD URL through the use of mechanisms such as DHCPv6-Shield [RFC7610].

There is the risk of the MUD manager itself being spied on to determine what things are connected to the network. To address this risk, MUD managers may choose to make use of TLS proxies that they trust that would aggregate other information.

Please note that the security considerations mentioned in Section 4.7 of [I-D.ietf-netmod-rfc6087bis] are not applicable in this case because the YANG serialization is not intended to be accessed via NETCONF. However, for those who try to instantiate this model in a network element via NETCONF, all objects in each model in this draft exhibit similar security characteristics as [I-D.ietf-netmod-acl-model]. The basic purpose of MUD is to configure access, and so by its very nature can be disruptive if used by unauthorized parties.

17. IANA Considerations

[There was originally a registry entry for .well-known suffixes. This has been removed from the draft and may be marked as deprecated in the registry. RFC Editor: please remove this comment.]

17.1. YANG Module Registrations

The following YANG modules are requested to be registered in the "IANA Module Names" registry:

The ietf-mud module:

- o Name: ietf-mud
- o URN: urn:ietf:params:xml:ns:yang:ietf-mud
- o Prefix: ietf-mud
- o Registrant contact: The IESG
- o Reference: [RFCXXXX]

The ietf-acldns module:

- o Name: ietf-acldns
- o URI: urn:ietf:params:xml:ns:yang:ietf-acldns
- o Prefix: ietf-acldns
- o Registrant: the IESG
- o Reference: [RFCXXXX]

17.2. DHCPv4 and DHCPv6 Options

The IANA has allocated option 161 in the Dynamic Host Configuration Protocol (DHCP) and Bootstrap Protocol (BOOTP) Parameters registry for the MUD DHCPv4 option, and option 112 for DHCPv6, as described in Section 10.

17.3. PKIX Extensions

IANA is kindly requested to make the following assignments for:

- o The MUDURLExtnModule-2016 ASN.1 module in the "SMI Security for PKIX Module Identifier" registry (1.3.6.1.5.5.7.0).
- o id-pe-mud-url object identifier from the "SMI Security for PKIX Certificate Extension" registry (1.3.6.1.5.5.7.1).
- o id-pe-mudsigner object identifier from the "SMI Security for PKIX Certificate Extension" registry (TBD1).

- o id-ct-mudtype object identifier from the "SMI Security for S/MIME CMS Content Type" registry (TBD2).

The use of these values is specified in Section 11.

17.4. MIME Media-type Registration for MUD files

The following media-type is defined for transfer of MUD file:

- o Type name: application
- o Subtype name: mud+json
- o Required parameters: n/a
- o Optional parameters: n/a
- o Encoding considerations: 8bit; application/mud+json values are represented as a JSON object; UTF-8 encoding MUST be employed. [RFC3629]
- o Security considerations: See Security Considerations of RFCXXXX and [RFC8259] Section 12.
- o Interoperability considerations: n/a
- o Published specification: [RFCXXXX]
- o Applications that use this media type: MUD managers as specified by [RFCXXXX].
- o Fragment identifier considerations: n/a
- o Additional information:
 - Magic number(s): n/a
 - File extension(s): n/a
 - Macintosh file type code(s): n/a
- o Person & email address to contact for further information: Eliot Lear <lear@cisco.com>, Ralph Droms <rdroms@gmail.com>
- o Intended usage: COMMON
- o Restrictions on usage: none
- o Author:
 - Eliot Lear <lear@cisco.com>
 - Ralph Droms <rdroms@gmail.com>
- o Change controller: IESG
- o Provisional registration? (standards tree only): No.

17.5. LLDP IANA TLV Subtype Registry

IANA is requested to create a new registry for IANA Link Layer Discovery Protocol (LLDP) TLV subtype values. The recommended policy for this registry is Expert Review. The maximum number of entries in the registry is 256.

IANA is required to populate the initial registry with the value:

LLDP subtype value = 1 (All the other 255 values should be initially marked as 'Unassigned'.)

Description = the Manufacturer Usage Description (MUD) Uniform Resource Locator (URL)

Reference = < this document >

17.6. The MUD Well Known Universal Resource Name (URNs)

The following parameter registry is requested to be added in accordance with [RFC3553]

Registry name: "urn:ietf:params:mud" is requested.
Specification: this document
Repository: this document
Index value: Encoded identically to a TCP/UDP port service name, as specified in Section 5.1 of [RFC6335]

The following entries should be added to the "urn:ietf:params:mud" name space:

"urn:ietf:params:mud:dns" refers to the service specified by [RFC1123]. "urn:ietf:params:mud:ntp" refers to the service specified by [RFC5905].

17.7. Extensions Registry

The IANA is requested to establish a registry of extensions as follows:

Registry name: MUD extensions registry
Registry policy: Standards action
Standard reference: document
Extension name: UTF-8 encoded string, not to exceed 40 characters.

Each extension MUST follow the rules specified in this specification. As is usual, the IANA issues early allocations based in accordance with [RFC7120].

18. Acknowledgments

The authors would like to thank Einar Nilsen-Nygaard, who singlehandedly updated the model to match the updated ACL model, Bernie Volz, Tom Gindin, Brian Weis, Sandeep Kumar, Thorsten Dahm, John Bashinski, Steve Rich, Jim Bieda, Dan Wing, Joe Clarke, Henk Birkholz, Adam Montville, Jim Schaad, and Robert Sparks for their valuable advice and reviews. Russ Housley entirely rewrote

Section 11 to be a complete module. Adrian Farrel provided the basis for privacy considerations text. Kent Watsen provided a thorough review of the architecture and the YANG model. The remaining errors in this work are entirely the responsibility of the authors.

19. References

19.1. Normative References

- [I-D.ietf-netmod-acl-model]
Jethanandani, M., Huang, L., Agarwal, S., and D. Blair,
"Network Access Control List (ACL) YANG Data Model",
draft-ietf-netmod-acl-model-19 (work in progress), April
2018.
- [IEEE8021AB]
Institute for Electrical and Electronics Engineers, "IEEE
Standard for Local and Metropolitan Area Networks--
Station and Media Access Control Connectivity Discovery",
n.d..
- [RFC1123] Braden, R., Ed., "Requirements for Internet Hosts -
Application and Support", STD 3, RFC 1123,
DOI 10.17487/RFC1123, October 1989,
<<https://www.rfc-editor.org/info/rfc1123>>.
- [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>>.
- [RFC2131] Droms, R., "Dynamic Host Configuration Protocol",
RFC 2131, DOI 10.17487/RFC2131, March 1997,
<<https://www.rfc-editor.org/info/rfc2131>>.
- [RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818,
DOI 10.17487/RFC2818, May 2000,
<<https://www.rfc-editor.org/info/rfc2818>>.
- [RFC3315] Droms, R., Ed., Bound, J., Volz, B., Lemon, T., Perkins,
C., and M. Carney, "Dynamic Host Configuration Protocol
for IPv6 (DHCPv6)", RFC 3315, DOI 10.17487/RFC3315, July
2003, <<https://www.rfc-editor.org/info/rfc3315>>.
- [RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO
10646", STD 63, RFC 3629, DOI 10.17487/RFC3629, November
2003, <<https://www.rfc-editor.org/info/rfc3629>>.

- [RFC3748] Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H. Levkowetz, Ed., "Extensible Authentication Protocol (EAP)", RFC 3748, DOI 10.17487/RFC3748, June 2004, <<https://www.rfc-editor.org/info/rfc3748>>.
- [RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <<https://www.rfc-editor.org/info/rfc3986>>.
- [RFC3987] Duerst, M. and M. Suignard, "Internationalized Resource Identifiers (IRIs)", RFC 3987, DOI 10.17487/RFC3987, January 2005, <<https://www.rfc-editor.org/info/rfc3987>>.
- [RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <<https://www.rfc-editor.org/info/rfc5234>>.
- [RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <<https://www.rfc-editor.org/info/rfc5280>>.
- [RFC5652] Housley, R., "Cryptographic Message Syntax (CMS)", STD 70, RFC 5652, DOI 10.17487/RFC5652, September 2009, <<https://www.rfc-editor.org/info/rfc5652>>.
- [RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, DOI 10.17487/RFC5905, June 2010, <<https://www.rfc-editor.org/info/rfc5905>>.
- [RFC5911] Hoffman, P. and J. Schaad, "New ASN.1 Modules for Cryptographic Message Syntax (CMS) and S/MIME", RFC 5911, DOI 10.17487/RFC5911, June 2010, <<https://www.rfc-editor.org/info/rfc5911>>.
- [RFC5912] Hoffman, P. and J. Schaad, "New ASN.1 Modules for the Public Key Infrastructure Using X.509 (PKIX)", RFC 5912, DOI 10.17487/RFC5912, June 2010, <<https://www.rfc-editor.org/info/rfc5912>>.

- [RFC6335] Cotton, M., Eggert, L., Touch, J., Westerlund, M., and S. Cheshire, "Internet Assigned Numbers Authority (IANA) Procedures for the Management of the Service Name and Transport Protocol Port Number Registry", BCP 165, RFC 6335, DOI 10.17487/RFC6335, August 2011, <<https://www.rfc-editor.org/info/rfc6335>>.
- [RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <<https://www.rfc-editor.org/info/rfc6991>>.
- [RFC7120] Cotton, M., "Early IANA Allocation of Standards Track Code Points", BCP 100, RFC 7120, DOI 10.17487/RFC7120, January 2014, <<https://www.rfc-editor.org/info/rfc7120>>.
- [RFC7227] Hankins, D., Mrugalski, T., Siodelski, M., Jiang, S., and S. Krishnan, "Guidelines for Creating New DHCPv6 Options", BCP 187, RFC 7227, DOI 10.17487/RFC7227, May 2014, <<https://www.rfc-editor.org/info/rfc7227>>.
- [RFC7230] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", RFC 7230, DOI 10.17487/RFC7230, June 2014, <<https://www.rfc-editor.org/info/rfc7230>>.
- [RFC7231] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content", RFC 7231, DOI 10.17487/RFC7231, June 2014, <<https://www.rfc-editor.org/info/rfc7231>>.
- [RFC7610] Gont, F., Liu, W., and G. Van de Velde, "DHCPv6-Shield: Protecting against Rogue DHCPv6 Servers", BCP 199, RFC 7610, DOI 10.17487/RFC7610, August 2015, <<https://www.rfc-editor.org/info/rfc7610>>.
- [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>>.
- [RFC7951] Lhotka, L., "JSON Encoding of Data Modeled with YANG", RFC 7951, DOI 10.17487/RFC7951, August 2016, <<https://www.rfc-editor.org/info/rfc7951>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

- [RFC8259] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", STD 90, RFC 8259, DOI 10.17487/RFC8259, December 2017, <<https://www.rfc-editor.org/info/rfc8259>>.
- [RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <<https://www.rfc-editor.org/info/rfc8340>>.
- [RFC8348] Bierman, A., Bjorklund, M., Dong, J., and D. Romascanu, "A YANG Data Model for Hardware Management", RFC 8348, DOI 10.17487/RFC8348, March 2018, <<https://www.rfc-editor.org/info/rfc8348>>.

19.2. Informative References

- [FW95] Chapman, D. and E. Zwicky, "Building Internet Firewalls", January 1995.
- [I-D.ietf-netmod-rfc6087bis]
Bierman, A., "Guidelines for Authors and Reviewers of YANG Data Model Documents", draft-ietf-netmod-rfc6087bis-20 (work in progress), March 2018.
- [IEEE80211i]
Institute for Electrical and Electronics Engineers, "IEEE Standard for information technology-Telecommunications and information exchange between systems-Local and metropolitan area networks-Specific requirements-Part 11-Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications- Amendment 6- Medium Access Control (MAC) Security Enhancements", 2004.
- [IEEE8021AE]
Institute for Electrical and Electronics Engineers, "IEEE Standard for Local and Metropolitan Area Networks- Media Access Control (MAC) Security", 2006.
- [IEEE8021AR]
Institute for Electrical and Electronics Engineers, "Secure Device Identity", 1998.
- [IEEE8021X]
Institute for Electrical and Electronics Engineers, "IEEE Standard for Local and metropolitan area networks--Port-Based Network Access Control", 2010.

- [ISO.8601.1988] International Organization for Standardization, "Data elements and interchange formats - Information interchange - Representation of dates and times", ISO Standard 8601, June 1988.
- [RFC1984] IAB and IESG, "IAB and IESG Statement on Cryptographic Technology and the Internet", BCP 200, RFC 1984, DOI 10.17487/RFC1984, August 1996, <<https://www.rfc-editor.org/info/rfc1984>>.
- [RFC3339] Klyne, G. and C. Newman, "Date and Time on the Internet: Timestamps", RFC 3339, DOI 10.17487/RFC3339, July 2002, <<https://www.rfc-editor.org/info/rfc3339>>.
- [RFC3553] Mealling, M., Masinter, L., Hardie, T., and G. Klyne, "An IETF URN Sub-namespace for Registered Protocol Parameters", BCP 73, RFC 3553, DOI 10.17487/RFC3553, June 2003, <<https://www.rfc-editor.org/info/rfc3553>>.
- [RFC6092] Woodyatt, J., Ed., "Recommended Simple Security Capabilities in Customer Premises Equipment (CPE) for Providing Residential IPv6 Internet Service", RFC 6092, DOI 10.17487/RFC6092, January 2011, <<https://www.rfc-editor.org/info/rfc6092>>.
- [RFC6872] Gurbani, V., Ed., Burger, E., Ed., Anjali, T., Abdelnur, H., and O. Festor, "The Common Log Format (CLF) for the Session Initiation Protocol (SIP): Framework and Information Model", RFC 6872, DOI 10.17487/RFC6872, February 2013, <<https://www.rfc-editor.org/info/rfc6872>>.
- [RFC7042] Eastlake 3rd, D. and J. Abley, "IANA Considerations and IETF Protocol and Documentation Usage for IEEE 802 Parameters", BCP 141, RFC 7042, DOI 10.17487/RFC7042, October 2013, <<https://www.rfc-editor.org/info/rfc7042>>.
- [RFC7170] Zhou, H., Cam-Winget, N., Salowey, J., and S. Hanna, "Tunnel Extensible Authentication Protocol (TEAP) Version 1", RFC 7170, DOI 10.17487/RFC7170, May 2014, <<https://www.rfc-editor.org/info/rfc7170>>.
- [RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, DOI 10.17487/RFC7252, June 2014, <<https://www.rfc-editor.org/info/rfc7252>>.

- [RFC7452] Tschofenig, H., Arkko, J., Thaler, D., and D. McPherson, "Architectural Considerations in Smart Object Networking", RFC 7452, DOI 10.17487/RFC7452, March 2015, <<https://www.rfc-editor.org/info/rfc7452>>.
- [RFC7488] Boucadair, M., Penno, R., Wing, D., Patil, P., and T. Reddy, "Port Control Protocol (PCP) Server Selection", RFC 7488, DOI 10.17487/RFC7488, March 2015, <<https://www.rfc-editor.org/info/rfc7488>>.
- [RFC8343] Bjorklund, M., "A YANG Data Model for Interface Management", RFC 8343, DOI 10.17487/RFC8343, March 2018, <<https://www.rfc-editor.org/info/rfc8343>>.

Appendix A. Changes from Earlier Versions

RFC Editor to remove this section prior to publication.

Draft -19: * Edits after discussion with apps area to address reserved field for the future. * Correct systeminfo to be utf8. * Remove "hardware-rev" from list.

Draft -18: * Correct an error in the augment statement * Changes to the ACL model re ports.

Draft -17:

- o One editorial.

Draft -16

- o add mud-signature element based on review comments
- o redo mud-url
- o make clear that systeminfo uses UTF8

Draft -13 to -14:

- o Final WGLC comments and review comments
- o Move version from MUD-URL to Model
- o Have MUD-URL in model
- o Update based on update to draft-ietf-netmod-acl-model
- o Point to tree diagram draft instead of 6087bis.

Draft -12 to -13:

- o Additional WGLC comments

Draft -10 to -12:

These are based on WGLC comments:

- o Correct examples based on ACL model changes.
- o Change ordering nodes.
- o Additional explanatory text around systeminfo.
- o Change ordering in examples.
- o Make it VERY VERY VERY VERY clear that these are recommendations, not mandates.
- o DHCP -> NTP in some of the intro text.
- o Remove masa-server
- o "Things" to "network elements" in a few key places.
- o Reference to JSON YANG RFC added.

Draft -10 to -11:

- o Example corrections
- o Typo
- o Fix two lists.
- o Addition of 'any-acl' and 'mud-acl' in the list of allowed features.
- o Clarification of what should be in a MUD file.

Draft -09 to -10:

- o AD input.
- o Correct dates.
- o Add compliance sentence as to which ACL module features are implemented.

Draft -08 to -09:

- o Resolution of Security Area review, IoT directorate review, GenART review, YANG doctors review.
- o change of YANG structure to address mandatory nodes.
- o Terminology cleanup.
- o specify out extra portion of MUD-URL.
- o consistency changes.
- o improved YANG descriptions.
- o Remove extra revisions.
- o Track ACL model changes.
- o Additional cautions on use of ACL model; further clarifications on extensions.

Draft -07 to -08:

- o a number of editorials corrected.
- o definition of MUD file tweaked.

Draft -06 to -07:

- o Examples updated.
- o Additional clarification for direction-initiated.
- o Additional implementation guidance given.

Draft -06 to -07:

- o Update models to match new ACL model
- o extract directionality from the ACL, introducing a new device container.

Draft -05 to -06:

- o Make clear that this is a component architecture (Polk and Watson)
- o Add order of operations (Watson)

- o Add extensions leaf-list (Pritikin)
- o Remove previous-mud-file (Watson)
- o Modify text in last-update (Watson)
- o Clarify local networks (Weis, Watson)
- o Fix contact info (Watson)
- o Terminology clarification (Weis)
- o Advice on how to handle LDevIDs (Watson)
- o Add deployment considerations (Watson)
- o Add some additional text about fingerprinting (Watson)
- o Appropriate references to 6087bis (Watson)
- o Change systeminfo to a URL to be referenced (Lear)

Draft -04 to -05: * syntax error correction

Draft -03 to -04: * Re-add my-controller

Draft -02 to -03: * Additional IANA updates * Format correction in YANG. * Add reference to TEAP.

Draft -01 to -02: * Update IANA considerations * Accept Russ Housley rewrite of X.509 text * Include privacy considerations text * Redo the URL limit. Still 255 bytes, but now stated in the URL definition. * Change URI registration to be under urn:ietf:params

Draft -00 to -01: * Fix cert trust text. * change supportInformation to meta-info * Add an informational element in. * add urn registry and create first entry * add default elements

Appendix B. Default MUD nodes

What follows is the portion of a MUD file that permits DNS traffic to a controller that is registered with the URN "urn:ietf:params:mud:dns" and traffic NTP to a controller that is registered "urn:ietf:params:mud:ntp". This is considered the default behavior and the ACEs are in effect appended to whatever other "ace" entries that a MUD file contains. To block DNS or NTP one repeats the matching statement but replaces the "forwarding" action "accept" with "drop". Because ACEs are processed in the order they are

received, the defaults would not be reached. A MUD manager might further decide to optimize to simply not include the defaults when they are overridden.

Four "acl" list entries that implement default MUD nodes are listed below. Two are for IPv4 and two are for IPv6 (one in each direction for both versions of IP). Note that neither access-list name nor ace name need be retained or used in any way by local implementations, but are simply there for completeness' sake.

```
"ietf-access-control-list:acls": {
  "acl": [
    {
      "name": "mud-59776-v4to",
      "type": "ipv4-acl-type",
      "aces": {
        "ace": [
          {
            "name": "ent0-todev",
            "matches": {
              "ietf-mud:mud": {
                "controller": "urn:ietf:params:mud:dns"
              },
              "ipv4": {
                "protocol": 17
              },
              "udp": {
                "source-port": {
                  "operator": "eq",
                  "port": 53
                }
              }
            },
            "actions": {
              "forwarding": "accept"
            }
          },
          {
            "name": "ent1-todev",
            "matches": {
              "ietf-mud:mud": {
                "controller": "urn:ietf:params:mud:ntp"
              },
              "ipv4": {
                "protocol": 17
              },
              "udp": {
                "source-port": {
```

```

        "operator": "eq",
        "port": 123
      }
    },
    "actions": {
      "forwarding": "accept"
    }
  ]
},
{
  "name": "mud-59776-v4fr",
  "type": "ipv4-acl-type",
  "aces": {
    "ace": [
      {
        "name": "ent0-frdev",
        "matches": {
          "ietf-mud:mud": {
            "controller": "urn:ietf:params:mud:dns"
          },
          "ipv4": {
            "protocol": 17
          },
          "udp": {
            "destination-port": {
              "operator": "eq",
              "port": 53
            }
          }
        },
        "actions": {
          "forwarding": "accept"
        }
      }
    ],
    "name": "ent1-frdev",
    "matches": {
      "ietf-mud:mud": {
        "controller": "urn:ietf:params:mud:ntp"
      },
      "ipv4": {
        "protocol": 17
      },
      "udp": {
        "destination-port": {

```

```

        "operator": "eq",
        "port": 123
      }
    },
    "actions": {
      "forwarding": "accept"
    }
  ]
},
{
  "name": "mud-59776-v6to",
  "type": "ipv6-acl-type",
  "aces": {
    "ace": [
      {
        "name": "ent0-todev",
        "matches": {
          "ietf-mud:mud": {
            "controller": "urn:ietf:params:mud:dns"
          },
          "ipv6": {
            "protocol": 17
          },
          "udp": {
            "source-port": {
              "operator": "eq",
              "port": 53
            }
          }
        },
        "actions": {
          "forwarding": "accept"
        }
      },
      {
        "name": "ent1-todev",
        "matches": {
          "ietf-mud:mud": {
            "controller": "urn:ietf:params:mud:ntp"
          },
          "ipv6": {
            "protocol": 17
          },
          "udp": {
            "source-port": {

```

```

        "operator": "eq",
        "port": 123
      }
    },
    "actions": {
      "forwarding": "accept"
    }
  ]
},
{
  "name": "mud-59776-v6fr",
  "type": "ipv6-acl-type",
  "aces": {
    "ace": [
      {
        "name": "ent0-frdev",
        "matches": {
          "ietf-mud:mud": {
            "controller": "urn:ietf:params:mud:dns"
          },
          "ipv6": {
            "protocol": 17
          },
          "udp": {
            "destination-port": {
              "operator": "eq",
              "port": 53
            }
          }
        },
        "actions": {
          "forwarding": "accept"
        }
      },
      {
        "name": "ent1-frdev",
        "matches": {
          "ietf-mud:mud": {
            "controller": "urn:ietf:params:mud:ntp"
          },
          "ipv6": {
            "protocol": 17
          },
          "udp": {
            "destination-port": {

```

```

        "operator": "eq",
        "port": 123
      }
    },
    "actions": {
      "forwarding": "accept"
    }
  ]
}

```

Appendix C. A Sample Extension: DETNET-indicator

In this sample extension we augment the core MUD model to indicate whether the device implements DETNET. If a device claims not to use DETNET, but then later attempts to do so, a notification or exception might be generated. Note that this example is intended only for illustrative purposes.

Extension Name: "Example-Extension" (to be used in the extensions list)
 Standard: this document (but do not register the example)

This extension augments the MUD model to include a single node, using the following sample module that has the following tree structure:

```

module: ietf-mud-detext-example
  augment /ietf-mud:mud:
    +--rw is-detnet-required?  boolean

```

The model is defined as follows:

```

<CODE BEGINS>file "ietf-mud-detext-example@2018-06-15.yang"
module ietf-mud-detext-example {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-mud-detext-example";
  prefix ietf-mud-detext-example;

  import ietf-mud {
    prefix ietf-mud;
  }
}

```



```
organization
  "IETF OPSAWG (Ops Area) Working Group";
contact
  "WG Web: http://tools.ietf.org/wg/opsawg/
  WG List: opsawg@ietf.org
  Author: Eliot Lear
  lear@cisco.com
  Author: Ralph Droms
  rdroms@gmail.com
  Author: Dan Romascanu
  dromasca@gmail.com

  ";
description
  "Sample extension to a MUD module to indicate a need
  for DETNET support.";

revision 2018-06-15 {
  description
    "Initial revision.";
  reference
    "RFC XXXX: Manufacturer Usage Description
    Specification";
}

augment "/ietf-mud:mud" {
  description
    "This adds a simple extension for a manufacturer
    to indicate whether DETNET is required by a
    device.";
  leaf is-detnet-required {
    type boolean;
    description
      "This value will equal true if a device requires
      detnet to properly function";
  }
}
}
}
<CODE ENDS>
```

Using the previous example, we now show how the extension would be expressed:

```
{
  "ietf-mud:mud": {
    "mud-version": 1,
    "mud-url": "https://lighting.example.com/lightbulb2000",
    "last-update": "2018-03-02T11:20:51+01:00",
```

```
"cache-validity": 48,
"extensions": [
  "ietf-mud-detext-example"
],
"ietf-mud-detext-example:is-detnet-required": "false",
"is-supported": true,
"systeminfo": "The BMS Example Lightbulb",
"from-device-policy": {
  "access-lists": {
    "access-list": [
      {
        "name": "mud-76100-v6fr"
      }
    ]
  }
},
"to-device-policy": {
  "access-lists": {
    "access-list": [
      {
        "name": "mud-76100-v6to"
      }
    ]
  }
},
"ietf-access-control-list:acls": {
  "acl": [
    {
      "name": "mud-76100-v6to",
      "type": "ipv6-acl-type",
      "aces": {
        "ace": [
          {
            "name": "cl0-todev",
            "matches": {
              "ipv6": {
                "ietf-acldns:src-dnsname": "test.example.com",
                "protocol": 6
              },
              "tcp": {
                "ietf-mud:direction-initiated": "from-device",
                "source-port": {
                  "operator": "eq",
                  "port": 443
                }
              }
            }
          }
        ]
      }
    }
  ],
}
```

```

        "actions": {
            "forwarding": "accept"
        }
    ]
}
},
{
    "name": "mud-76100-v6fr",
    "type": "ipv6-acl-type",
    "aces": {
        "ace": [
            {
                "name": "cl0-frdev",
                "matches": {
                    "ipv6": {
                        "ietf-acldns:dst-dnsname": "test.example.com",
                        "protocol": 6
                    },
                    "tcp": {
                        "ietf-mud:direction-initiated": "from-device",
                        "destination-port": {
                            "operator": "eq",
                            "port": 443
                        }
                    }
                },
                "actions": {
                    "forwarding": "accept"
                }
            }
        ]
    }
}
]
}
}

```

Authors' Addresses

Eliot Lear
Cisco Systems
Richtistrasse 7
Wallisellen CH-8304
Switzerland

Phone: +41 44 878 9200
Email: lear@cisco.com

Ralph Droms
Google
355 Main St., 5th Floor
Cambridge

Phone: +1 978 376 3731
Email: rdroms@gmail.com

Dan Romascanu

Phone: +972 54 5555347
Email: dromasca@gmail.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: March 31, 2019

M. Boucadair, Ed.
Orange
S. Sivakumar
Cisco Systems
C. Jacquenet
Orange
S. Vinapamula
Juniper Networks
Q. Wu
Huawei
September 27, 2018

A YANG Module for Network Address Translation (NAT) and Network Prefix
Translation (NPT)
draft-ietf-opsawg-nat-yang-17

Abstract

This document defines a YANG module for the Network Address
Translation (NAT) function.

Network Address Translation from IPv4 to IPv4 (NAT44), Network
Address and Protocol Translation from IPv6 Clients to IPv4 Servers
(NAT64), Customer-side transLATOR (CLAT), Stateless IP/ICMP
Translation (SIIT), Explicit Address Mappings for Stateless IP/ICMP
Translation (SIIT EAM), IPv6 Network Prefix Translation (NPTv6), and
Destination NAT are covered in this document.

Editorial Note (To be removed by RFC Editor)

Please update these statements within the document with the RFC
number to be assigned to this document:

"This version of this YANG module is part of RFC XXXX;"

"RFC XXXX: A YANG Module for Network Address Translation (NAT) and
Network Prefix Translation (NPT)"

"reference: RFC XXXX"

Please update the "revision" date of the YANG module.

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 March 31, 2019.

Copyright Notice

Copyright (c) 2018 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
1.1. Terminology	3
2. Overview of the NAT YANG Data Model	5
2.1. Overview	5
2.2. Various Translation Flavors	6
2.3. TCP/UDP/ICMP NAT Behavioral Requirements	7
2.4. Other Transport Protocols	7
2.5. IP Addresses Used for Translation	8
2.6. Port Set Assignment	8
2.7. Port-Restricted IP Addresses	8
2.8. NAT Mapping Entries	9
2.9. Resource Limits	12
2.10. Binding the NAT Function to an External Interface	15
2.11. Relationship to NATV2-MIB	15
2.12. Tree Structure	16
3. NAT YANG Module	22
4. Security Considerations	71
5. IANA Considerations	73
6. Acknowledgements	73

7. References	74
7.1. Normative References	74
7.2. Informative References	77
Appendix A. Sample Examples	78
A.1. Traditional NAT44	79
A.2. Carrier Grade NAT (CGN)	80
A.3. CGN Pass-Through	83
A.4. NAT64	84
A.5. Stateless IP/ICMP Translation (SIIT)	84
A.6. Explicit Address Mappings for Stateless IP/ICMP Translation (EAM SIIT)	85
A.7. Static Mappings with Port Ranges	88
A.8. Static Mappings with IP Prefixes	89
A.9. Destination NAT	90
A.10. Customer-side Translator (CLAT)	93
A.11. IPv6 Network Prefix Translation (NPTv6)	93
Authors' Addresses	96

1. Introduction

This document defines a data model for Network Address Translation (NAT) and Network Prefix Translation (NPT) capabilities using the YANG data modeling language [RFC7950].

Traditional NAT is defined in [RFC2663], while Carrier Grade NAT (CGN) is defined in [RFC6888]. Unlike traditional NAT, the CGN is used to optimize the usage of global IP address space at the scale of a domain: a CGN is not managed by end users, but by service providers instead. This document covers both traditional NATs and CGNs.

This document also covers NAT64 [RFC6146], customer-side translator (CLAT) [RFC6877], Stateless IP/ICMP Translation (SIIT) [RFC7915], Explicit Address Mappings for Stateless IP/ICMP Translation (EAM) [RFC7757], IPv6 Network Prefix Translation (NPTv6) [RFC6296], and Destination NAT. The full set of translation schemes that are in scope is included in Section 2.2.

Sample examples are provided in Appendix A. These examples are not intended to be exhaustive.

1.1. Terminology

This document makes use of the following terms:

- o Basic Network Address Translation from IPv4 to IPv4 (NAT44): translation is limited to IP addresses alone (Section 2.1 of [RFC3022]).

- o Network Address/Port Translator (NAPT): translation in NAPT is extended to include IP addresses and transport identifiers (such as a TCP/UDP port or ICMP query ID); refer to Section 2.2 of [RFC3022]. A NAPT may use an extra identifier, in addition to the five transport tuple, to disambiguate bindings [RFC6619].
- o Destination NAT: is a translation that acts on the destination IP address and/or destination port number. This flavor is usually deployed in load balancers or at devices in front of public servers.
- o Port-restricted IPv4 address: An IPv4 address with a restricted port set. Multiple hosts may share the same IPv4 address; however, their port sets must not overlap [RFC7596].
- o Restricted port set: A non-overlapping range of allowed external ports to use for NAT operation. Source ports of IPv4 packets translated by a NAT must belong to the assigned port set. The port set is used for all port-aware IP protocols [RFC7596].
- o Internal Host: A host that may need to use a translation capability to send to and receive traffic from the Internet.
- o Internal Address/prefix: The IP address/prefix of an internal host.
- o External Address: The IP address/prefix assigned by a translator to an internal host; this is the address that will be seen by a remote host on the Internet.
- o Mapping: denotes a state at the translator that is necessary for network address and/or port translation.
- o Dynamic implicit mapping: is created implicitly as a side effect of processing a packet (e.g., an initial TCP SYN packet) that requires a new mapping. A validity lifetime is associated with this mapping.
- o Dynamic explicit mapping: is created as a result of an explicit request, e.g., PCP message [RFC6887]. A validity lifetime is associated with this mapping.
- o Static explicit mapping: is created using, e.g., a CLI interface. This mapping is likely to be maintained by the NAT function till an explicit action is executed to remove it.

The usage of the term NAT in this document refers to any translation flavor (NAT44, NAT64, etc.) indifferently.

This document uses the term "session" as defined in [RFC2663] and [RFC6146] for NAT64.

This document follows the guidelines of [RFC6087], uses the common YANG types defined in [RFC6991], and adopts the Network Management Datastore Architecture (NMDA). The meaning of the symbols in tree diagrams is defined in [RFC8340].

2. Overview of the NAT YANG Data Model

2.1. Overview

The NAT YANG module is designed to cover dynamic implicit mappings and static explicit mappings. The required functionality to instruct dynamic explicit mappings is defined in separate documents such as [I-D.boucadair-pcp-yang]. Considerations about instructing by explicit dynamic means (e.g., [RFC6887], [RFC6736], or [RFC8045]) are out of scope. As a reminder, REQ-9 of [RFC6888] requires that a CGN must implement a protocol giving subscribers explicit control over NAT mappings; that protocol should be the Port Control Protocol [RFC6887].

A single NAT device can have multiple NAT instances; each of these instances can be provided with its own policies (e.g., be responsible for serving a group of hosts). This document does not make any assumption about how internal hosts or flows are associated with a given NAT instance.

The NAT YANG module assumes that each NAT instance can be enabled/disabled, be provisioned with a specific set of configuration data, and maintains its own mapping tables.

The NAT YANG module allows for a NAT instance to be provided with multiple NAT policies (/nat/instances/instance/policy). The document does not make any assumption about how flows are associated with a given NAT policy of a given NAT instance. Classification filters are out of scope.

Defining multiple NAT instances or configuring multiple NAT policies within one single NAT instance is implementation- and deployment-specific.

This YANG module does not provide any method to instruct a NAT function to enable the logging feature or to specify the information to be logged for administrative or regulatory reasons (Section 2.3 of [RFC6908] and REQ-12 of [RFC6888]). Those considerations are out of the scope of this document.

2.2. Various Translation Flavors

The following translation modes are supported:

- o Basic NAT44
- o NAPT
- o Destination NAT
- o Port-restricted NAT
- o Stateful NAT64 (including with destination-based Pref64::/n [RFC7050])
- o SIIT
- o CLAT
- o EAM
- o NPTv6
- o Combination of Basic NAT/NAPT and Destination NAT
- o Combination of port-restricted and Destination NAT
- o Combination of NAT64 and EAM
- o Stateful and Stateless NAT64

[I-D.ietf-softwire-dslite-yang] specifies an extension to the NAT YANG module to support DS-Lite.

The YANG "feature" statement is used to indicate which of the different translation modes is relevant for a specific data node. Table 1 lists defined features:

Translation Mode	YANG Feature
Basic NAT44	basic-nat44
NAPT	napt44
Destination NAT	dst-nat
Stateful NAT64	nat64
Stateless IPv4/IPv6 translation	siit
CLAT	clat
EAM	eam
NPTv6	nptv6

Table 1: YANG NAT Features

The following translation modes do not require defining dedicated features:

- o Port-restricted NAT: This mode corresponds to supplying port restriction policies to a NAPT or NAT64 (port-set-restrict).
- o Combination of Basic NAT/NAPT and Destination NAT: This mode corresponds to setting 'dst-nat-enable' for Basic NAT44 or NAPT.

- o Combination of port-restricted and Destination NAT: This mode can be achieved by configuring a NAPT with port restriction policies (port-set-restrict) together with a destination IP address pool (dst-ip-address-pool).
- o Combination of NAT64 and EAM: This mode corresponds to configuring static mappings for NAT64.
- o Stateful and stateless NAT64: A NAT64 implementation can be instructed to behave in the stateless mode for a given prefix by setting the parameter (nat64-prefixes/stateless-enable). A NAT64 implementation may behave in both stateful and stateless modes if, in addition to appropriately setting the parameter (nat64-prefixes/stateless-enable), an external IPv4 address pool is configured.

The NAT YANG module provides a method to retrieve the capabilities of a NAT instance (including, list of supported translation modes, list of supported protocols, port restriction support status, supported NAT mapping types, supported NAT filtering types, port range allocation support status, port parity preservation support status, port preservation support status, the behavior for handling fragments (all, out-of-order, in-order)).

2.3. TCP/UDP/ICMP NAT Behavioral Requirements

This document assumes NAT behavioral recommendations for UDP [RFC4787], TCP [RFC5382], and ICMP [RFC5508] are enabled by default.

Furthermore, the NAT YANG module relies upon the recommendations detailed in [RFC6888] and [RFC7857].

2.4. Other Transport Protocols

The module is structured to support protocols other than UDP, TCP, and ICMP. Concretely, the module allows the operator to enable translation for other transport protocols when required (/nat/instances/instance/policy/transport-protocols). Moreover, the mapping table is designed so that it can indicate any transport protocol. For example, this module may be used to manage a DCCP-capable NAT that adheres to [RFC5597].

Future extensions may be needed to cover NAT-related considerations that are specific to other transport protocols such as SCTP [I-D.ietf-tsvwg-natsupp]. Typically, the mapping entry can be extended to record two optional SCTP-specific parameters: Internal Verification Tag (Int-VTag) and External Verification Tag (Ext-VTag).

This document only specifies transport protocol specific timers for UDP, TCP, and ICMP. While some timers could potentially be

generalized for other connection-oriented protocols, this document does not follow such an approach because there is no standard document specifying such generic behavior. Future documents may be edited to clarify how to reuse TCP-specific timers when needed.

2.5. IP Addresses Used for Translation

The NAT YANG module assumes that blocks of IP external addresses (external-ip-address-pool) can be provisioned to the NAT function. These blocks may be contiguous or not.

This behavior is aligned with [RFC6888] which specifies that a NAT function should not have any limitations on the size or the contiguity of the external address pool. In particular, the NAT function must be configurable with contiguous or non-contiguous external IPv4 address ranges. To accommodate traditional NAT, the module allows for a single IP address to be configured for external-ip-address-pool.

Likewise, one or multiple IP address pools may be configured for Destination NAT (dst-ip-address-pool).

2.6. Port Set Assignment

Port numbers can be assigned by a NAT individually (that is, a single port is assigned on a per session basis), but this port allocation scheme may not be optimal for logging purposes (Section 12 of [RFC6269]). A NAT function should be able to assign port sets (e.g., [RFC7753]) to optimize the volume of the logging data (REQ-14 of [RFC6888]). Both allocation schemes are supported in the NAT YANG module.

When port set assignment is activated (i.e., port-allocation-type==port-range-allocation), the NAT can be provided with the size of the port set to be assigned (port-set-size).

2.7. Port-Restricted IP Addresses

Some NATs restrict the source port numbers (e.g., Lightweight 4over6 [RFC7596], MAP-E [RFC7597]). Two schemes of port set assignments (port-set-restrict) are supported in this document:

- o Simple port range: is defined by two port values, the start and the end of the port range [RFC8045].
- o Algorithmic: an algorithm is defined in [RFC7597] to characterize the set of ports that can be used.

2.8. NAT Mapping Entries

A TCP/UDP mapping entry maintains an association between the following information:

```
(internal-src-address, internal-src-port) (internal-dst-address,  
internal-dst-port) <=> (external-src-address, external-src-port)  
(external-dst-address, external-dst-port)
```

An ICMP mapping entry maintains an association between the following information:

```
(internal-src-address, internal-dst-address, internal ICMP/ICMPv6  
identifier) <=> (external-src-address, external-dst-address,  
external ICMP/ICMPv6 identifier)
```

As a reminder, all the ICMP Query messages contain an 'Identifier' field, which is referred to in this document as the 'ICMP Identifier'.

To cover TCP, UDP, and ICMP, the NAT YANG module assumes the following structure of a mapping entry:

type: Indicates how the mapping was instantiated. For example, it may indicate whether a mapping is dynamically instantiated by a packet or statically configured.

transport-protocol: Indicates the transport protocol (e.g., UDP, TCP, ICMP) of a given mapping.

internal-src-address: Indicates the source IP address/prefix as used by an internal host.

internal-src-port: Indicates the source port number (or ICMP identifier) as used by an internal host.

external-src-address: Indicates the source IP address/prefix as assigned by the NAT.

external-src-port: Indicates the source port number (or ICMP identifier) as assigned by the NAT.

internal-dst-address: Indicates the destination IP address/prefix as used by an internal host when sending a packet to a remote host.

internal-dst-port: Indicates the destination port number as used by an internal host when sending a packet to a remote host.

external-dst-address: Indicates the destination IP address/prefix used by a NAT when processing a packet issued by an internal host towards a remote host.

external-dst-port: Indicates the destination port number used by a NAT when processing a packet issued by an internal host towards a remote host.

In order to cover both NAT64 and NAT44 flavors, the NAT mapping structure allows for the inclusion of an IPv4 or an IPv6 address as an internal IP address. Remaining fields are common to both NAT schemes.

For example, the mapping that will be created by a NAT64 upon receipt of a TCP SYN from source address 2001:db8:aaaa::1 and source port number 25636 to destination IP address 2001:db8:1234::198.51.100.1 and destination port number 8080 is shown in Table 2. This example assumes EDM (Endpoint-Dependent Mapping).

Mapping Entry Attribute	Value
type	dynamic implicit mapping
transport-protocol	6 (TCP)
internal-src-address	2001:db8:aaaa::1
internal-src-port	25636
external-src-address	T (an IPv4 address configured on the NAT64)
external-src-port	t (a port number that is chosen by the NAT64)
internal-dst-address	2001:db8:1234::198.51.100.1
internal-dst-port	8080
external-dst-address	198.51.100.1
external-dst-port	8080

Table 2: Example of an EDM NAT64 Mapping

The mappings that will be created by a NAT44 upon receipt of an ICMP request from source address 198.51.100.1 and ICMP identifier (ID1) to destination IP address 198.51.100.11 is depicted in Table 3. This example assumes EIM (Endpoint-Independent Mapping).

Mapping Entry Attribute	Value
type	dynamic implicit mapping
transport-protocol	1 (ICMP)
internal-src-address	198.51.100.1
internal-src-port	ID1
external-src-address	T (an IPv4 address configured on the NAT44)
external-src-port	ID2 (an ICMP identifier that is chosen by the NAT44)

Table 3: Example of an EIM NAT44 Mapping Entry

The mapping that will be created by a NAT64 (EIM mode) upon receipt of an ICMP request from source address 2001:db8:aaaa::1 and ICMP identifier (ID1) to destination IP address 2001:db8:1234::198.51.100.1 is shown in Table 4.

Mapping Entry Attribute	Value
type	dynamic implicit mapping
transport-protocol	58 (ICMPv6)
internal-src-address	2001:db8:aaaa::1
internal-src-port	ID1
external-src-address	T (an IPv4 address configured on the NAT64)
external-src-port	ID2 (an ICMP identifier that is chosen by the NAT64)

Table 4: Example of an EIM NAT64 Mapping Entry

Note that a mapping table is maintained only for stateful NAT functions. Particularly:

- o No mapping table is maintained for NPTv6 given that it is stateless and transport-agnostic.
- o The double translations are stateless in CLAT if a dedicated IPv6 prefix is provided for CLAT. If not, a stateful NAT44 will be required.
- o No per-flow mapping is maintained for EAM [RFC7757].

- o No mapping table is maintained for Stateless IPv4/IPv6 translation. As a reminder, in such deployments internal IPv6 nodes are addressed using IPv4-translatable IPv6 addresses, which enable them to be accessed by IPv4 nodes [RFC6052].

2.9. Resource Limits

In order to comply with CGN deployments in particular, the NAT YANG module allows limiting the number of external ports per subscriber (port-quota) and the amount of state memory allocated per mapping and per subscriber (mapping-limits and connection-limits). According to [RFC6888], the module is designed to allow for the following:

- o Per-subscriber limits are configurable by the NAT administrator.
- o Per-subscriber limits are configurable independently per transport protocol.
- o Administrator-adjustable thresholds to prevent a single subscriber from consuming excessive CPU resources from the NAT (e.g., rate-limit the subscriber's creation of new mappings) can be configured.

Table 5 lists the various limits that can be set using the NAT YANG module. Once a limit is reached, packets that would normally trigger new port mappings or be translated because they match existing mappings, are dropped by the translator.

Limit	Description
port-quota	Specifies a port quota to be assigned per subscriber. It corresponds to the maximum number of ports to be used by a subscriber. The port quota can be configured to apply to all protocols or to a specific protocol. Distinct port quota may be configured per protocol.
fragments-limit	In order to prevent denial of service attacks that can be caused by fragments, this parameter is used to limit the number of out-of-order fragments that can be handled by a translator.
mapping-limits	This parameter can be used to control the maximum number of subscribers that can be serviced by a NAT instance (limit-subscriber) and the maximum number of address and/or port mappings that can be maintained by a NAT instance (limit-address-mappings and limit-port-mappings). Also, limits specific to protocols (e.g., TCP, UDP, ICMP) can also be specified (limit-per-protocol).
connection-limits	In order to prevent exhausting the resources of a NAT implementation and to ensure fairness usage among subscribers, various rate-limits can be specified. Rate-limiting can be enforced per subscriber ((limit-subscriber), per NAT instance (limit-per-instance), and/or be specified for each supported protocol (limit-per-protocol).

Table 5: NAT Limits

Table 6 describes limits, that once exceeded, will trigger notifications to be generated:

Notification Threshold	Description
high-threshold	Used to notify high address utilization of a given pool. When exceeded, a nat-pool-event notification will be generated.
low-threshold	Used to notify low address utilization of a given pool. An administrator is supposed to configure low-threshold so that it can reflect an abnormal usage of NAT resources. When exceeded, a nat-pool-event notification will be generated.
notify-addresses-usage	Used to notify high address utilization of all pools configured to a NAT instance. When exceeded, a nat-instance-event will be generated.
notify-ports-usage	Used to notify high port allocation taking into account all pools configured to a NAT instance. When exceeded, a nat-instance-event notification will be generated.
notify-subscribers-limit	Used to notify a high number of active subscribers that are serviced by a NAT instance. When exceeded, a nat-instance-event notification will be generated.

Table 6: Notification Thresholds

In order to prevent a NAT implementation from generating frequent notifications, the NAT YANG module supports the following limits (Table 7) used to control how frequent notifications can be generated. That is, notifications are subject to rate-limiting imposed by these intervals.

Interval	Description
notify-pool-usage/notify-interval	Indicates the minimum number of seconds between successive notifications for a given address pool.
notification-limits/notify-interval	Indicates the minimum number of seconds between successive notifications for a NAT instance.

Table 7: Notification Intervals

2.10. Binding the NAT Function to an External Interface

The module is designed to specify an external realm on which the NAT function must be applied (external-realm). The module supports indicating an interface as an external realm [RFC8343], but the module is extensible so that other choices can be indicated in the future (e.g., Virtual Routing and Forwarding (VRF) instance).

Distinct external realms can be provided as a function of the NAT policy (see for example, Section 4 of [RFC7289]).

If no external realm is provided, this assumes that the system is able to determine the external interface (VRF instance, etc.) on which the NAT will be applied. Typically, the WAN and LAN interfaces of a CPE are determined by the CPE.

2.11. Relationship to NATV2-MIB

Section of 5.1 of [RFC7659] indicates that the NATV2-MIB assumes that the following information is configured on the NAT by some means, not specified in [RFC7659]:

- o The set of address realms to which the device connect.
- o For the CGN case, per-subscriber information including subscriber index, address realm, assigned prefix or address, and (possibly) policies regarding address pool selection in the various possible address realms to which the subscriber may connect.
- o The set of NAT instances running on the device, identified by NAT instance index and name.

- o The port mapping, filtering, pooling, and fragment behaviors for each NAT instance.
- o The set of protocols supported by each NAT instance.
- o Address pools for each NAT instance, including for each pool the pool index, address realm, and minimum and maximum port number.
- o Static address and port mapping entries.

All the above parameters can be configured by means of the NAT YANG module.

Unlike the NATV2-MIB, the NAT YANG module allows to configure multiple policies per NAT instance.

2.12. Tree Structure

The tree structure of the NAT YANG module is provided below:

```

module: ietf-nat
  +--rw nat
    +--rw instances
      +--rw instance* [id]
        +--rw id                               uint32
        +--rw name?                           string
        +--rw enable?                         boolean
        +--ro capabilities
          +--ro nat-flavor*
            | identityref
          +--ro per-interface-binding*
            | enumeration
          +--ro transport-protocols* [protocol-id]
            | +--ro protocol-id      uint8
            | +--ro protocol-name?  string
          +--ro restricted-port-support?
            | boolean
          +--ro static-mapping-support?
            | boolean
          +--ro port-randomization-support?
            | boolean
          +--ro port-range-allocation-support?
            | boolean
          +--ro port-preservation-suport?
            | boolean
          +--ro port-parity-preservation-support?
            | boolean
          +--ro address-roundrobin-support?

```

```

|         boolean
+--ro paired-address-pooling-support?
|         boolean
+--ro endpoint-independent-mapping-support?
|         boolean
+--ro address-dependent-mapping-support?
|         boolean
+--ro address-and-port-dependent-mapping-support?
|         boolean
+--ro endpoint-independent-filtering-support?
|         boolean
+--ro address-dependent-filtering?
|         boolean
+--ro address-and-port-dependent-filtering?
|         boolean
+--ro fragment-behavior?
|         enumeration
+--rw type?                               identityref
+--rw per-interface-binding?              enumeration
+--rw nat-pass-through* [id]
|     {basic-nat44 or napt44 or dst-nat}?
+--rw id                                  uint32
+--rw prefix                             inet:ip-prefix
+--rw port?                              inet:port-number
+--rw policy* [id]
|     +--rw id                                  uint32
+--rw clat-parameters {clat}?
|     | +--rw clat-ipv6-prefixes* [ipv6-prefix]
|     | | +--rw ipv6-prefix          inet:ipv6-prefix
|     | +--rw ipv4-prefixes* [ipv4-prefix]
|     | | +--rw ipv4-prefix          inet:ipv4-prefix
+--rw nptv6-prefixes* [internal-ipv6-prefix] {nptv6}?
|     +--rw internal-ipv6-prefix          inet:ipv6-prefix
|     +--rw external-ipv6-prefix          inet:ipv6-prefix
+--rw eam* [ipv4-prefix] {eam}?
|     +--rw ipv4-prefix                    inet:ipv4-prefix
|     +--rw ipv6-prefix                    inet:ipv6-prefix
+--rw nat64-prefixes* [nat64-prefix]
|     {siit or nat64 or clat}?
+--rw nat64-prefix                        inet:ipv6-prefix
+--rw destination-ipv4-prefix* [ipv4-prefix]
|     | +--rw ipv4-prefix              inet:ipv4-prefix
+--rw stateless-enable?                  boolean
+--rw external-ip-address-pool* [pool-id]
|     {basic-nat44 or napt44 or nat64}?
+--rw pool-id                            uint32
+--rw external-ip-pool                    inet:ipv4-prefix
+--rw port-set-restrict {napt44 or nat64}?

```

```

|   |--rw (port-type)?
|   |   |--:(port-range)
|   |   |   |--rw start-port-number?    inet:port-number
|   |   |   |--rw end-port-number?      inet:port-number
|   |   |--:(port-set-algo)
|   |   |   |--rw psid-offset?           uint8
|   |   |   |--rw psid-len               uint8
|   |   |   |--rw psid                   uint16
|--rw dst-nat-enable?                    boolean
|   {basic-nat44 or napt44}?
|--rw dst-ip-address-pool* [pool-id] {dst-nat}?
|   |--rw pool-id                       uint32
|   |--rw dst-in-ip-pool?               inet:ip-prefix
|   |--rw dst-out-ip-pool               inet:ip-prefix
|--rw transport-protocols* [protocol-id]
|   {napt44 or nat64 or dst-nat}?
|   |--rw protocol-id                   uint8
|   |--rw protocol-name?                string
|--rw subscriber-mask-v6?                uint8
|--rw subscriber-match* [match-id]
|   {basic-nat44 or napt44 or dst-nat}?
|   |--rw match-id                      uint32
|   |--rw subnet                        inet:ip-prefix
|--rw address-allocation-type?           enumeration
|--rw port-allocation-type?              enumeration
|   {napt44 or nat64}?
|--rw mapping-type?                      enumeration
|   {napt44 or nat64}?
|--rw filtering-type?                    enumeration
|   {napt44 or nat64}?
|--rw fragment-behavior?                 enumeration
|   {napt44 or nat64}?
|--rw port-quota* [quota-type] {napt44 or nat64}?
|   |--rw port-limit?                   uint16
|   |--rw quota-type                     uint8
|--rw port-set {napt44 or nat64}?
|   |--rw port-set-size                  uint16
|   |--rw port-set-timeout?              uint32
|--rw timers {napt44 or nat64}?
|   |--rw udp-timeout?                   uint32
|   |--rw tcp-idle-timeout?              uint32
|   |--rw tcp-trans-open-timeout?        uint32
|   |--rw tcp-trans-close-timeout?       uint32
|   |--rw tcp-in-syn-timeout?            uint32
|   |--rw fragment-min-timeout?          uint32
|   |--rw icmp-timeout?                  uint32
|   |--rw per-port-timeout* [port-number]
|   |   |--rw port-number                inet:port-number

```

```

| | | +-rw protocol?          uint32
| | | +-rw timeout           uint32
| | +-rw hold-down-timeout?   uint32
| | +-rw hold-down-max?      uint32
+--rw fragments-limit?       uint32
+--rw algs* [name]
| +-rw name                   string
| +-rw transport-protocol?   uint32
| +-rw dst-transport-port
| | +-rw start-port-number?   inet:port-number
| | +-rw end-port-number?    inet:port-number
| +-rw src-transport-port
| | +-rw start-port-number?   inet:port-number
| | +-rw end-port-number?    inet:port-number
| +-rw status?               boolean
+--rw all-algs-enable?        boolean
+--rw notify-pool-usage
| {basic-nat44 or napt44 or nat64}?
| +-rw pool-id?              uint32
| +-rw low-threshold?        percent
| +-rw high-threshold?       percent
| +-rw notify-interval?      uint32
+--rw external-realm
| +-rw (realm-type)?
| | +--:(interface)
| | | +-rw external-interface? if:interface-ref
+--rw mapping-limits {napt44 or nat64}?
| +-rw limit-subscribers?    uint32
| +-rw limit-address-mappings? uint32
| +-rw limit-port-mappings?   uint32
| +-rw limit-per-protocol* [protocol-id]
| | {napt44 or nat64 or dst-nat}?
| | +-rw protocol-id         uint8
| | +-rw limit?              uint32
+--rw connection-limits
| {basic-nat44 or napt44 or nat64}?
| +-rw limit-per-subscriber?  uint32
| +-rw limit-per-instance?    uint32
| +-rw limit-per-protocol* [protocol-id]
| | {napt44 or nat64}?
| | +-rw protocol-id         uint8
| | +-rw limit?              uint32
+--rw notification-limits
| +-rw notify-interval?       uint32
| | {basic-nat44 or napt44 or nat64}?
| +-rw notify-addresses-usage? percent
| | {basic-nat44 or napt44 or nat64}?
| +-rw notify-ports-usage?   percent

```

```

| | {napt44 or nat64}?
| +--rw notify-subscribers-limit? uint32
| | {basic-nat44 or napt44 or nat64}?
+--rw mapping-table
| | {basic-nat44 or napt44 or nat64 or clat or dst-nat}?
+--rw mapping-entry* [index]
| | +--rw index uint32
| | +--rw type? enumeration
| | +--rw transport-protocol? uint8
| | +--rw internal-src-address? inet:ip-prefix
| | +--rw internal-src-port
| | | +--rw start-port-number? inet:port-number
| | | +--rw end-port-number? inet:port-number
| | +--rw external-src-address? inet:ip-prefix
| | +--rw external-src-port
| | | +--rw start-port-number? inet:port-number
| | | +--rw end-port-number? inet:port-number
| | +--rw internal-dst-address? inet:ip-prefix
| | +--rw internal-dst-port
| | | +--rw start-port-number? inet:port-number
| | | +--rw end-port-number? inet:port-number
| | +--rw external-dst-address? inet:ip-prefix
| | +--rw external-dst-port
| | | +--rw start-port-number? inet:port-number
| | | +--rw end-port-number? inet:port-number
| | +--rw lifetime? uint32
+--ro statistics
| +--ro discontinuity-time yang:date-and-time
| +--ro traffic-statistics
| | +--ro sent-packets?
| | | yang:zero-based-counter64
| | +--ro sent-bytes?
| | | yang:zero-based-counter64
| | +--ro rcvd-packets?
| | | yang:zero-based-counter64
| | +--ro rcvd-bytes?
| | | yang:zero-based-counter64
| | +--ro dropped-packets?
| | | yang:zero-based-counter64
| | +--ro dropped-bytes?
| | | yang:zero-based-counter64
| | +--ro dropped-fragments?
| | | yang:zero-based-counter64
| | | {napt44 or nat64}?
| | +--ro dropped-address-limit-packets?
| | | yang:zero-based-counter64
| | | {basic-nat44 or napt44 or nat64}?
| | +--ro dropped-address-limit-bytes?

```



```

|         yang:zero-based-counter64
|         {basic-nat44 or napt44 or nat64}?
+--ro dropped-address-packets?
|         yang:zero-based-counter64
|         {basic-nat44 or napt44 or nat64}?
+--ro dropped-address-bytes?
|         yang:zero-based-counter64
|         {basic-nat44 or napt44 or nat64}?
+--ro dropped-port-limit-packets?
|         yang:zero-based-counter64
|         {napt44 or nat64}?
+--ro dropped-port-limit-bytes?
|         yang:zero-based-counter64
|         {napt44 or nat64}?
+--ro dropped-port-packets?
|         yang:zero-based-counter64
|         {napt44 or nat64}?
+--ro dropped-port-bytes?
|         yang:zero-based-counter64
|         {napt44 or nat64}?
+--ro dropped-subscriber-limit-packets?
|         yang:zero-based-counter64
|         {basic-nat44 or napt44 or nat64}?
+--ro dropped-subscriber-limit-bytes?
|         yang:zero-based-counter64
|         {basic-nat44 or napt44 or nat64}?
+--ro mappings-statistics
+--ro total-active-subscribers?  yang:gauge32
|         {basic-nat44 or napt44 or nat64}?
+--ro total-address-mappings?    yang:gauge32
|         {basic-nat44 or napt44 or nat64 or clat or dst-nat}?
+--ro total-port-mappings?       yang:gauge32
|         {napt44 or nat64}?
+--ro total-per-protocol* [protocol-id]
|         {napt44 or nat64}?
|         +--ro protocol-id      uint8
|         +--ro total?           yang:gauge32
+--ro pools-stats {basic-nat44 or napt44 or nat64}?
+--ro addresses-allocated?      yang:gauge32
+--ro addresses-free?           yang:gauge32
+--ro ports-stats {napt44 or nat64}?
|         +--ro ports-allocated? yang:gauge32
|         +--ro ports-free?       yang:gauge32
+--ro per-pool-stats* [pool-id]
|         {basic-nat44 or napt44 or nat64}?
+--ro pool-id                    uint32
+--ro discontinuity-time          yang:date-and-time
+--ro pool-stats

```

```

        |   +--ro addresses-allocated?   yang:gauge32
        |   +--ro addresses-free?       yang:gauge32
+--ro port-stats {napt44 or nat64}?
    +--ro ports-allocated?   yang:gauge32
    +--ro ports-free?       yang:gauge32

notifications:
+---n nat-pool-event {basic-nat44 or napt44 or nat64}?
|   +--ro id          -> /nat/instances/instance/id
|   +--ro policy-id?
|   |   -> /nat/instances/instance/policy/id
|   +--ro pool-id
|   |   -> /nat/instances/instance/policy/
|   |       external-ip-address-pool/pool-id
|   +--ro notify-pool-threshold    percent
+---n nat-instance-event {basic-nat44 or napt44 or nat64}?
    +--ro id
    |   -> /nat/instances/instance/id
    +--ro notify-subscribers-threshold?    uint32
    +--ro notify-addresses-threshold?     percent
    +--ro notify-ports-threshold?         percent

```

3. NAT YANG Module

<CODE BEGINS> file "ietf-nat@2018-09-27.yang"

```

module ietf-nat {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-nat";
  prefix "nat";

  import ietf-inet-types {
    prefix inet;
    reference
      "Section 4 of RFC 6991";
  }

  import ietf-yang-types {
    prefix yang;
    reference
      "Section 3 of RFC 6991";
  }

  import ietf-interfaces {
    prefix if;
    reference
      "RFC 8343: A YANG Data Model for Interface Management";
  }

```

```
}  
  
organization  
  "IETF OPSAWG (Operations and Management Area Working Group)";  
  
contact  
  
  "WG Web:    <https://datatracker.ietf.org/wg/opsawg/>  
  WG List:    <mailto:opsawg@ietf.org>  
  
  Editor:     Mohamed Boucadair  
              <mailto:mohamed.boucadair@orange.com>  
  
  Author:     Senthil Sivakumar  
              <mailto:ssenthil@cisco.com>  
  
  Author:     Christian Jacquenet  
              <mailto:christian.jacquenet@orange.com>  
  
  Author:     Suresh Vinapamula  
              <mailto:sureshk@juniper.net>  
  
  Author:     Qin Wu  
              <mailto:bill.wu@huawei.com>";  
  
description  
  "This module is a YANG module for NAT implementations.  
  
  NAT44, Network Address and Protocol Translation from IPv6  
  Clients to IPv4 Servers (NAT64), Customer-side transLATOR (CLAT),  
  Stateless IP/ICMP Translation (SIIT), Explicit Address Mappings  
  for Stateless IP/ICMP Translation (SIIT EAM), IPv6 Network  
  Prefix Translation (NPTv6), and Destination NAT are covered.  
  
  Copyright (c) 2018 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 RFC XXXX; see  
  the RFC itself for full legal notices.";  
  
revision 2018-09-27 {
```

```
description
  "Initial revision.";
reference
  "RFC XXXX: A YANG Module for Network Address Translation
    (NAT) and Network Prefix Translation (NPT)";
}

/*
 * Definitions
 */

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

/*
 * Features
 */

feature basic-nat44{
  description
    "Basic NAT44 translation is limited to IP addresses alone.";
  reference
    "RFC 3022: Traditional IP Network Address Translator
      (Traditional NAT)";
}

feature napt44 {
  description
    "Network Address/Port Translator (NAPT): translation is
    extended to include IP addresses and transport identifiers
    (such as a TCP/UDP port or ICMP query ID).

    If the internal IP address is not sufficient to uniquely
    disambiguate NAPT44 mappings, an additional attribute is
    required. For example, that additional attribute may
    be an IPv6 address (a.k.a., DS-Lite) or
    a Layer 2 identifier (a.k.a., Per-Interface NAT)";
  reference
    "RFC 3022: Traditional IP Network Address Translator
      (Traditional NAT)";
}

feature dst-nat {
```

```
description
  "Destination NAT is a translation that acts on the destination
   IP address and/or destination port number. This flavor is
   usually deployed in load balancers or at devices
   in front of public servers.";
}

feature nat64 {
  description
    "NAT64 translation allows IPv6-only clients to contact IPv4
     servers using, e.g., UDP, TCP, or ICMP. One or more
     public IPv4 addresses assigned to a NAT64 translator are
     shared among several IPv6-only clients.";
  reference
    "RFC 6146: Stateful NAT64: Network Address and Protocol
     Translation from IPv6 Clients to IPv4 Servers";
}

feature siit {
  description
    "The Stateless IP/ICMP Translation Algorithm (SIIT), which
     translates between IPv4 and IPv6 packet headers (including
     ICMP headers).

     In the stateless mode, an IP/ICMP translator converts IPv4
     addresses to IPv6 and vice versa solely based on the
     configuration of the stateless IP/ICMP translator and
     information contained within the packet being translated.

     The translator must support the stateless address mapping
     algorithm defined in RFC6052, which is the default behavior.";
  reference
    "RFC 7915: IP/ICMP Translation Algorithm";
}

feature clat {
  description
    "CLAT is customer-side translator that algorithmically
     translates 1:1 private IPv4 addresses to global IPv6 addresses,
     and vice versa.

     When a dedicated /64 prefix is not available for translation
     from DHCPv6-PD, the CLAT may perform NAT44 for all IPv4 LAN
     packets so that all the LAN-originated IPv4 packets appear
     from a single IPv4 address and are then statelessly translated
     to one interface IPv6 address that is claimed by the CLAT via
     the Neighbor Discovery Protocol (NDP) and defended with
     Duplicate Address Detection.";
```

```
    reference
      "RFC 6877: 464XLAT: Combination of Stateful and Stateless
        Translation";
  }

  feature eam {
    description
      "Explicit Address Mapping (EAM) is a bidirectional coupling
        between an IPv4 Prefix and an IPv6 Prefix.";
    reference
      "RFC 7757: Explicit Address Mappings for Stateless IP/ICMP
        Translation";
  }

  feature nptv6 {
    description
      "NPTv6 is a stateless transport-agnostic IPv6-to-IPv6
        prefix translation.";
    reference
      "RFC 6296: IPv6-to-IPv6 Network Prefix Translation";
  }

  /*
   * Identities
   */

  identity nat-type {
    description
      "Base identity for nat type.";
  }

  identity basic-nat44 {
    base nat:nat-type;
    description
      "Identity for Basic NAT support.";
    reference
      "RFC 3022: Traditional IP Network Address Translator
        (Traditional NAT)";
  }

  identity napt44 {
    base nat:nat-type;
    description
      "Identity for NAPT support.";
    reference
      "RFC 3022: Traditional IP Network Address Translator
        (Traditional NAT)";
  }
```

```
identity dst-nat {
  base nat:nat-type;
  description
    "Identity for Destination NAT support.";
}

identity nat64 {
  base nat:nat-type;
  description
    "Identity for NAT64 support.";
  reference
    "RFC 6146: Stateful NAT64: Network Address and Protocol
      Translation from IPv6 Clients to IPv4 Servers";
}

identity siit {
  base nat:nat-type;
  description
    "Identity for SIIT support.";
  reference
    "RFC 7915: IP/ICMP Translation Algorithm";
}

identity clat {
  base nat:nat-type;
  description
    "Identity for CLAT support.";
  reference
    "RFC 6877: 464XLAT: Combination of Stateful and Stateless
      Translation";
}

identity eam {
  base nat:nat-type;
  description
    "Identity for EAM support.";
  reference
    "RFC 7757: Explicit Address Mappings for Stateless IP/ICMP
      Translation";
}

identity nptv6 {
  base nat:nat-type;
  description
    "Identity for NPTv6 support.";
  reference
    "RFC 6296: IPv6-to-IPv6 Network Prefix Translation";
}
```

```
/*
 * Grouping
 */

grouping port-number {
  description
    "An individual port number or a range of ports.
    When only start-port-number is present,
    it represents a single port number.";

  leaf start-port-number {
    type inet:port-number;
    description
      "Beginning of the port range.";
    reference
      "Section 3.2.9 of RFC 8045.";
  }

  leaf end-port-number {
    type inet:port-number;

    must ". >= ../start-port-number"
    {
      error-message
        "The end-port-number must be greater than or
        equal to start-port-number.";
    }
    description
      "End of the port range.";
    reference
      "Section 3.2.10 of RFC 8045.";
  }
}

grouping port-set {
  description
    "Indicates a set of port numbers.

    It may be a simple port range, or use the Port Set ID (PSID)
    algorithm to represent a range of transport layer
    port numbers which will be used by a NAPT.";

  choice port-type {
    default port-range;
    description
      "Port type: port-range or port-set-algo.";
    case port-range {
      uses port-number;
    }
  }
}
```



```
    }

    case port-set-algo {
      leaf psid-offset {
        type uint8 {
          range 0..15;
        }

        description
          "The number of offset bits (a.k.a., 'a' bits).

          Specifies the numeric value for the excluded port
          range/offset bits.

          Allowed values are between 0 and 15.";

        reference
          "Section 5.1 of RFC 7597";
      }

      leaf psid-len {
        type uint8 {
          range 0..15;
        }
        mandatory true;

        description
          "The length of PSID, representing the sharing
          ratio for an IPv4 address.

          (also known as 'k').

          The address-sharing ratio would be 2^k.";
        reference
          "Section 5.1 of RFC 7597";
      }

      leaf psid {
        type uint16;
        mandatory true;
        description
          "Port Set Identifier (PSID) value, which
          identifies a set of ports algorithmically.";
        reference
          "Section 5.1 of RFC 7597";
      }
    }
  }
  reference
```

```
        "Section 7597: Mapping of Address and Port with
          Encapsulation (MAP-E)";
    }
}

grouping mapping-entry {
  description
    "NAT mapping entry.

    If an attribute is not stored in the mapping/session table,
    this means the corresponding field of a packet that
    matches this entry is not rewritten by the NAT or this
    information is not required for NAT filtering purposes.";

  leaf index {
    type uint32;
    description
      "A unique identifier of a mapping entry. This identifier can be
      automatically assigned by the NAT instance or be explicitly
      configured.";
  }

  leaf type {
    type enumeration {
      enum "static" {
        description
          "The mapping entry is explicitly configured
          (e.g., via command-line interface).";
      }

      enum "dynamic-implicit" {
        description
          "This mapping is created implicitly as a side effect
          of processing a packet that requires a new mapping.";
      }

      enum "dynamic-explicit" {
        description
          "This mapping is created as a result of an explicit
          request, e.g., a PCP message.";
      }
    }
  }

  description
    "Indicates the type of a mapping entry. E.g.,
    a mapping can be: static, implicit dynamic,
    or explicit dynamic.";
}
```

```
}

leaf transport-protocol {
  type uint8;
  description
    "Upper-layer protocol associated with this mapping.
    Values are taken from the IANA protocol registry::
    https://www.iana.org/assignments/protocol-numbers/
    protocol-numbers.xhtml

    For example, this field contains 6 for TCP,
    17 for UDP, 33 for DCCP, or 132 for SCTP.

    If this leaf is not instantiated, then the mapping
    applies to any protocol.";
}

leaf internal-src-address {
  type inet:ip-prefix;
  description
    "Corresponds to the source IPv4/IPv6 address/prefix
    of the packet received on an internal interface.";
}

container internal-src-port {
  description
    "Corresponds to the source port of the packet received
    on an internal interface.

    It is used also to indicate the internal source ICMP
    identifier.

    As a reminder, all the ICMP Query messages contain
    an 'Identifier' field, which is referred to in this
    document as the 'ICMP Identifier'.";

  uses port-number;
}

leaf external-src-address {
  type inet:ip-prefix;
  description
    "Source IP address/prefix of the packet sent on an
    external interface of the NAT.";
}

container external-src-port {
  description
```

"Source port of the packet sent on an external interface of the NAT.

It is used also to indicate the external source ICMP identifier.";

```
    uses port-number;
}
```

```
leaf internal-dst-address {
    type inet:ip-prefix;
    description
        "Corresponds to the destination IP address/prefix
        of the packet received on an internal interface
        of the NAT.

        For example, some NAT implementations support
        the translation of both source and destination
        addresses and port numbers, sometimes referred to
        as 'Twice NAT'.";
}
```

```
container internal-dst-port {
    description
        "Corresponds to the destination port of the
        IP packet received on the internal interface.

        It is used also to include the internal
        destination ICMP identifier.";

    uses port-number;
}
```

```
leaf external-dst-address {
    type inet:ip-prefix;
    description
        "Corresponds to the destination IP address/prefix
        of the packet sent on an external interface
        of the NAT.";
}
```

```
container external-dst-port {
    description
        "Corresponds to the destination port number of
        the packet sent on the external interface
        of the NAT.

        It is used also to include the external
```

```
        destination ICMP identifier.";

    uses port-number;
}

leaf lifetime {
    type uint32;
    units "seconds";
    description
        "When specified, it is used to track the connection that is
        fully-formed (e.g., once the three-way handshake
        TCP is completed) or the duration for maintaining
        an explicit mapping alive. The mapping entry will be
        removed by the NAT instance once this lifetime is expired.

        When reported in a get operation, the lifetime indicates
        the remaining validity lifetime.

        Static mappings may not be associated with a
        lifetime. If no lifetime is associated with a
        static mapping, an explicit action is required to
        remove that mapping.";
}
}

/*
 * NAT Module
 */

container nat {
    description
        "NAT module";

    container instances {
        description
            "NAT instances";

        list instance {
            key "id";

            description
                "A NAT instance. This identifier can be automatically assigned
                or explicitly configured.";

            leaf id {
                type uint32;
                must ". >= 1";
                description
```

```
    "NAT instance identifier.

    The identifier must be greater than zero.";
  reference
    "RFC 7659: Definitions of Managed Objects for Network
      Address Translators (NATs)";
}

leaf name {
  type string;
  description
    "A name associated with the NAT instance.";
  reference
    "RFC 7659: Definitions of Managed Objects for Network
      Address Translators (NATs)";
}

leaf enable {
  type boolean;
  description
    "Status of the NAT instance.";
}

container capabilities {
  config false;

  description
    "NAT capabilities";

  leaf-list nat-flavor {
    type identityref {
      base nat-type;
    }
    description
      "Supported translation type(s).";
  }

  leaf-list per-interface-binding {
    type enumeration {
      enum "unsupported" {
        description
          "No capability to associate a NAT binding with
            an extra identifier.";
      }

      enum "layer-2" {
        description
          "The NAT instance is able to associate a mapping with
```

```
        a layer-2 identifier.";
    }

    enum "dslite" {
        description
            "The NAT instance is able to associate a mapping with
            an IPv6 address (a.k.a., DS-Lite).";
    }
}
description
    "Indicates the capability of a NAT to associate a particular
    NAT session not only with the five tuples used for the
    transport connection on both sides of the NAT but also with
    the internal interface on which the user device is
    connected to the NAT.";
reference
    "Section 4 of RFC 6619";
}

list transport-protocols {
    key protocol-id;

    description
        "List of supported protocols.";

    leaf protocol-id {
        type uint8;
        mandatory true;
        description
            "Upper-layer protocol associated with a mapping.

            Values are taken from the IANA protocol registry.

            For example, this field contains 6 for TCP,
            17 for UDP, 33 for DCCP, or 132 for SCTP.";
    }

    leaf protocol-name {
        type string;
        description
            "The name of the Upper-layer protocol associated
            with this mapping.

            For example, TCP, UDP, DCCP, and SCTP.";
    }
}

leaf restricted-port-support {
```

```
    type boolean;
    description
        "Indicates source port NAT restriction support.";
    reference
        "RFC 7596: Lightweight 4over6: An Extension to
         the Dual-Stack Lite Architecture.";
}

leaf static-mapping-support {
    type boolean;
    description
        "Indicates whether static mappings are supported.";
}

leaf port-randomization-support {
    type boolean;
    description
        "Indicates whether port randomization is supported.";
    reference
        "Section 4.2.1 of RFC 4787.";
}

leaf port-range-allocation-support {
    type boolean;
    description
        "Indicates whether port range allocation is supported.";
    reference
        "Section 1.1 of RFC 7753.";
}

leaf port-preservation-suport {
    type boolean;
    description
        "Indicates whether port preservation is supported.";
    reference
        "Section 4.2.1 of RFC 4787.";
}

leaf port-parity-preservation-support {
    type boolean;
    description
        "Indicates whether port parity preservation is
         supported.";
    reference
        "Section 8 of RFC 7857.";
}

leaf address-roundrobin-support {
```



```
    type boolean;
    description
      "Indicates whether address allocation round robin is
       supported.";
  }

  leaf paired-address-pooling-support {
    type boolean;
    description
      "Indicates whether paired-address-pooling is
       supported";
    reference
      "REQ-2 of RFC 4787.";
  }

  leaf endpoint-independent-mapping-support {
    type boolean;
    description
      "Indicates whether endpoint-independent-
       mapping is supported.";
    reference
      "Section 4 of RFC 4787.";
  }

  leaf address-dependent-mapping-support {
    type boolean;
    description
      "Indicates whether address-dependent-mapping is
       supported.";
    reference
      "Section 4 of RFC 4787.";
  }

  leaf address-and-port-dependent-mapping-support {
    type boolean;
    description
      "Indicates whether address-and-port-dependent-mapping is
       supported.";
    reference
      "Section 4 of RFC 4787.";
  }

  leaf endpoint-independent-filtering-support {
    type boolean;
    description
      "Indicates whether endpoint-independent-filtering is
       supported.";
    reference
```

```
        "Section 5 of RFC 4787.";
    }

    leaf address-dependent-filtering {
        type boolean;
        description
            "Indicates whether address-dependent-filtering is
            supported.";
        reference
            "Section 5 of RFC 4787.";
    }

    leaf address-and-port-dependent-filtering {
        type boolean;
        description
            "Indicates whether address-and-port-dependent is
            supported.";
        reference
            "Section 5 of RFC 4787.";
    }

    leaf fragment-behavior {
        type enumeration {
            enum "unsupported" {
                description
                    "No capability to translate incoming fragments.
                    All received fragments are dropped.";
            }

            enum "in-order" {
                description
                    "The NAT instance is able to translate fragments only if
                    they are received in order. That is, in particular the
                    header is in the first packet. Fragments received
                    out of order are dropped. ";
            }

            enum "out-of-order" {
                description
                    "The NAT instance is able to translate a fragment even
                    if it is received out of order.

                    This behavior is recommended.";
                reference
                    "REQ-14 of RFC 4787";
            }
        }
        description
```

```
        "The fragment behavior is the NAT instance's capability to
        translate fragments received on the external interface of
        the NAT.";
    }
}

leaf type {
    type identityref {
        base nat-type;
    }
    description
        "Specify the translation type. Particularly useful when
        multiple translation flavors are supported.

        If one type is supported by a NAT, this parameter is by
        default set to that type.";
}

leaf per-interface-binding {
    type enumeration {
        enum "disabled" {
            description
                "Disable the capability to associate an extra identifier
                with NAT mappings.";
        }

        enum "layer-2" {
            description
                "The NAT instance is able to associate a mapping with
                a layer-2 identifier.";
        }

        enum "dslite" {
            description
                "The NAT instance is able to associate a mapping with
                an IPv6 address (a.k.a., DS-Lite).";
        }
    }
    description
        "A NAT that associates a particular NAT session not only with
        the five tuples used for the transport connection on both
        sides of the NAT but also with the internal interface on
        which the user device is connected to the NAT.

        If supported, this mode of operation should be configurable,
        and it should be disabled by default in general-purpose NAT
        devices."
```

```
        If one single per-interface binding behavior is supported by
        a NAT, this parameter is by default set to that behavior.";
    reference
        "Section 4 of RFC 6619";
}

list nat-pass-through {
    if-feature "basic-nat44 or napt44 or dst-nat";
    key id;

    description
        "IP prefix NAT pass through.";

    leaf id {
        type uint32;
        description
            "An identifier of the IP prefix pass through.";
    }

    leaf prefix {
        type inet:ip-prefix;
        mandatory true;
        description
            "The IP addresses that match should not be translated.

            It must be possible to administratively turn
            off translation for specific destination addresses
            and/or ports.";
        reference
            "REQ#6 of RFC 6888.";
    }

    leaf port {
        type inet:port-number;
        description
            "It must be possible to administratively turn off
            translation for specific destination addresses
            and/or ports.

            If no prefix is defined, the NAT pass through bound
            to a given port applies for any destination address.";
        reference
            "REQ#6 of RFC 6888.";
    }
}

list policy {
    key id;
```

```
description
  "NAT parameters for a given instance";

leaf id {
  type uint32;
  description
    "An identifier of the NAT policy. It must be unique
    within the NAT instance.";
}

container clat-parameters {
  if-feature clat;
  description
    "CLAT parameters.";

  list clat-ipv6-prefixes {
    key ipv6-prefix;
    description
      "464XLAT double translation treatment is stateless when a
      dedicated /64 is available for translation on the CLAT.
      Otherwise, the CLAT will have both stateful and stateless
      since it requires NAT44 from the LAN to a single IPv4
      address and then stateless translation to a single
      IPv6 address.";
    reference
      "RFC 6877: 464XLAT: Combination of Stateful and Stateless
      Translation";

    leaf ipv6-prefix {
      type inet:ipv6-prefix;
      description
        "An IPv6 prefix used for CLAT.";
    }
  }

  list ipv4-prefixes {
    key ipv4-prefix;
    description
      "Pool of IPv4 addresses used for CLAT.
      192.0.0.0/29 is the IPv4 service continuity prefix.";
    reference
      "RFC 7335: IPv4 Service Continuity Prefix";

    leaf ipv4-prefix {
      type inet:ipv4-prefix;
      description
        "464XLAT double translation treatment is
        stateless when a dedicated /64 is available
```

for translation on the CLAT. Otherwise, the CLAT will have both stateful and stateless since it requires NAT44 from the LAN to a single IPv4 address and then stateless translation to a single IPv6 address. The CLAT performs NAT44 for all IPv4 LAN packets so that all the LAN-originated IPv4 packets appear from a single IPv4 address and are then statelessly translated to one interface IPv6 address that is claimed by the CLAT.

An IPv4 address from this pool is also provided to an application that makes use of literals.";

```
reference
  "RFC 6877: 464XLAT: Combination of Stateful and Stateless
    Translation";
}
}
}

list nptv6-prefixes {
  if-feature nptv6;
  key internal-ipv6-prefix ;
  description
    "Provides one or a list of (internal IPv6 prefix,
      external IPv6 prefix) required for NPTv6.

    In its simplest form, NPTv6 interconnects two network
    links, one of which is an 'internal' network link
    attached to a leaf network within a single
    administrative domain and the other of which is an
    'external' network with connectivity to the global
    Internet.";
  reference
    "RFC 6296: IPv6-to-IPv6 Network Prefix Translation";

  leaf internal-ipv6-prefix {
    type inet:ipv6-prefix;
    mandatory true;
    description
      "An IPv6 prefix used by an internal interface of NPTv6.";
    reference
      "RFC 6296: IPv6-to-IPv6 Network Prefix Translation";
  }
}
```

```
    leaf external-ipv6-prefix {
      type inet:ipv6-prefix;
      mandatory true;
      description
        "An IPv6 prefix used by the external interface of NPTv6.";
      reference
        "RFC 6296: IPv6-to-IPv6 Network Prefix Translation";
    }
  }

  list eam {
    if-feature eam;
    key ipv4-prefix;
    description
      "The Explicit Address Mapping Table, a conceptual
       table in which each row represents an EAM.

       Each EAM describes a mapping between IPv4 and IPv6
       prefixes/addresses.";
    reference
      "Section 3.1 of RFC 7757.";

    leaf ipv4-prefix {
      type inet:ipv4-prefix;
      mandatory true;
      description
        "The IPv4 prefix of an EAM.";
      reference
        "Section 3.2 of RFC 7757.";
    }

    leaf ipv6-prefix {
      type inet:ipv6-prefix;
      mandatory true;
      description
        "The IPv6 prefix of an EAM.";
      reference
        "Section 3.2 of RFC 7757.";
    }
  }

  list nat64-prefixes {
    if-feature "siit or nat64 or clat";
    key nat64-prefix;
    description
      "Provides one or a list of NAT64 prefixes
       with or without a list of destination IPv4 prefixes.
       It allows mapping IPv4 address ranges to IPv6 prefixes.
```

```

    For example:
    192.0.2.0/24 is mapped to 2001:db8:122:300::/56.
    198.51.100.0/24 is mapped to 2001:db8:122::/48.";
reference
    "Section 5.1 of RFC 7050.";

leaf nat64-prefix {
    type inet:ipv6-prefix;
    mandatory true;
    description
        "A NAT64 prefix. Can be Network-Specific Prefix (NSP) or
        Well-Known Prefix (WKP).

        Organizations deploying stateless IPv4/IPv6 translation
        should assign a Network-Specific Prefix to their
        IPv4/IPv6 translation service.

        For stateless NAT64, IPv4-translatable IPv6 addresses
        must use the selected Network-Specific Prefix.

        Both IPv4-translatable IPv6 addresses and IPv4-converted
        IPv6 addresses should use the same prefix.";
    reference
        "Sections 3.3 and 3.4 of RFC 6052.";
}

list destination-ipv4-prefix {
    key ipv4-prefix;
    description
        "An IPv4 prefix/address.";

    leaf ipv4-prefix {
        type inet:ipv4-prefix;
        description
            "An IPv4 address/prefix.";
    }
}

leaf stateless-enable {
    type boolean;
    default false;
    description
        "Enable explicitly stateless NAT64.";
}

list external-ip-address-pool {
    if-feature "basic-nat44 or napt44 or nat64";
```



```
key pool-id;

description
  "Pool of external IP addresses used to service internal
  hosts.

  A pool is a set of IP prefixes.";

leaf pool-id {
  type uint32;
  must ". >= 1";
  description
    "An identifier that uniquely identifies the address pool
    within a NAT instance.

    The identifier must be greater than zero.";
  reference
    "RFC 7659: Definitions of Managed Objects for
    Network Address Translators (NATs)";
}

leaf external-ip-pool {
  type inet:ipv4-prefix;
  mandatory true;
  description
    "An IPv4 prefix used for NAT purposes.";
}
}

container port-set-restrict {
  if-feature "napt44 or nat64";
  description
    "Configures contiguous and non-contiguous port ranges.

    The port set is used to restrict the external source
    port numbers used by the translator.";

  uses port-set;
}

leaf dst-nat-enable {
  if-feature "basic-nat44 or napt44";
  type boolean;
  default false;
  description
    "Enable/Disable destination NAT.

    A NAT44 may be configured to enable Destination
```

```
    NAT, too.";
  }

  list dst-ip-address-pool {
    if-feature dst-nat;
    key pool-id;
    description
      "Pool of IP addresses used for destination NAT.";

    leaf pool-id {
      type uint32;
      description
        "An identifier of the address pool.";
    }

    leaf dst-in-ip-pool {
      type inet:ip-prefix;
      description
        "Is used to identify an internal destination
        IP prefix/address to be translated.";
    }

    leaf dst-out-ip-pool {
      type inet:ip-prefix;
      mandatory true;
      description
        "IP address/prefix used for destination NAT.";
    }
  }

  list transport-protocols {
    if-feature "napt44 or nat64 or dst-nat";
    key protocol-id;

    description
      "Configure the transport protocols to be handled by
      the translator.

      TCP and UDP are supported by default.";

    leaf protocol-id {
      type uint8;
      mandatory true;
      description
        "Upper-layer protocol associated with this mapping.

        Values are taken from the IANA protocol registry.
```

```
        For example, this field contains 6 for TCP,
        17 for UDP, 33 for DCCP, or 132 for SCTP.";
    }

    leaf protocol-name {
        type string;
        description
            "The name of the Upper-layer protocol associated
            with this mapping.

            For example, TCP, UDP, DCCP, and SCTP.";
    }
}

leaf subscriber-mask-v6 {
    type uint8 {
        range "0 .. 128";
    }

    description
        "The subscriber mask is an integer that indicates
        the length of significant bits to be applied on
        the source IPv6 address (internal side) to
        unambiguously identify a user device (e.g., CPE).

        Subscriber mask is a system-wide configuration
        parameter that is used to enforce generic
        per-subscriber policies (e.g., port-quota).

        The enforcement of these generic policies does not
        require the configuration of every subscriber's
        prefix.

        Example: suppose the 2001:db8:100:100::/56 prefix
        is assigned to a NAT64 serviced CPE. Suppose also
        that 2001:db8:100:100::1 is the IPv6 address used
        by the client that resides in that CPE. When the
        NAT64 receives a packet from this client,
        it applies the subscriber-mask-v6 (e.g., 56) on
        the source IPv6 address to compute the associated
        prefix for this client (2001:db8:100:100::/56).
        Then, the NAT64 enforces policies based on that
        prefix (2001:db8:100:100::/56), not on the exact
        source IPv6 address.";
}

list subscriber-match {
    if-feature "basic-nat44 or napt44 or dst-nat";
```

```
key match-id;

description
  "IP prefix match.
   A subscriber is identified by a subnet.";

leaf match-id {
  type uint32;
  description
    "An identifier of the subscriber match.";
}

leaf subnet {
  type inet:ip-prefix;
  mandatory true;
  description
    "The IP address subnets that match
     should be translated. E.g., all addresses
     that belong to the 192.0.2.0/24 prefix must
     be processed by the NAT.";
}

leaf address-allocation-type {
  type enumeration {
    enum "arbitrary" {
      if-feature "basic-nat44 or napt44 or nat64";
      description
        "Arbitrary pooling behavior means that the NAT
         instance may create the new port mapping using any
         address in the pool that has a free port for the
         protocol concerned.";
    }

    enum "roundrobin" {
      if-feature "basic-nat44 or napt44 or nat64";
      description
        "Round robin allocation.";
    }

    enum "paired" {
      if-feature "napt44 or nat64";
      description
        "Paired address pooling informs the NAT
         that all the flows from an internal IP
         address must be assigned the same external
         address. This is the recommended behavior for
         NAPT/NAT64.";
    }
  }
}
```

```
        reference
            "RFC 4787: Network Address Translation (NAT)
              Behavioral Requirements for Unicast UDP";
    }
}
description
    "Specifies how external IP addresses are allocated.";
}

leaf port-allocation-type {
    if-feature "napt44 or nat64";
    type enumeration {
        enum "random" {
            description
                "Port randomization is enabled. A NAT port allocation
                 scheme should make it hard for attackers to guess
                 port numbers";
            reference
                "REQ-15 of RFC 6888";
        }

        enum "port-preservation" {
            description
                "Indicates whether the NAT should preserve the internal
                 port number.";
        }

        enum "port-parity-preservation" {
            description
                "Indicates whether the NAT should preserve the port
                 parity of the internal port number.";
        }

        enum "port-range-allocation" {
            description
                "Indicates whether the NAT assigns a range of ports
                 for an internal host. This scheme allows to minimize
                 log volume.";
            reference
                "REQ-14 of RFC 6888";
        }
    }
    description
        "Indicates the type of port allocation.";
}

leaf mapping-type {
    if-feature "napt44 or nat64";
```

```
type enumeration {
  enum "eim" {
    description
      "endpoint-independent-mapping.";
    reference
      "Section 4 of RFC 4787.";
  }

  enum "adm" {
    description
      "address-dependent-mapping.";
    reference
      "Section 4 of RFC 4787.";
  }

  enum "edm" {
    description
      "address-and-port-dependent-mapping.";
    reference
      "Section 4 of RFC 4787.";
  }
}
description
  "Indicates the type of a NAT mapping.";
}

leaf filtering-type {
  if-feature "napt44 or nat64";
  type enumeration {
    enum "eif" {
      description
        "endpoint-independent-filtering.";
      reference
        "Section 5 of RFC 4787.";
    }

    enum "adf" {
      description
        "address-dependent-filtering.";
      reference
        "Section 5 of RFC 4787.";
    }

    enum "edf" {
      description
        "address-and-port-dependent-filtering";
      reference
        "Section 5 of RFC 4787.";
    }
  }
}
```

```
    }
  }
  description
    "Indicates the type of a NAT filtering.";
}

leaf fragment-behavior {
  if-feature "napt44 or nat64";
  type enumeration {
    enum "drop-all" {
      description
        "All received fragments are dropped.";
    }

    enum "in-order" {
      description
        "Translate fragments only if they are received
        in order.";
    }

    enum "out-of-order" {
      description
        "Translate a fragment even if it is received out
        of order.

        This behavior is recommended.";
      reference
        "REQ-14 of RFC 4787";
    }
  }
  description
    "The fragment behavior instructs the NAT about the
    behavior to follow to translate fragments received
    on the external interface of the NAT.";
}

list port-quota {
  if-feature "napt44 or nat64";
  key quota-type;
  description
    "Configures a port quota to be assigned per subscriber.
    It corresponds to the maximum number of ports to be
    used by a subscriber.";

  leaf port-limit {
    type uint16;
    description
      "Configures a port quota to be assigned per subscriber.
```

```
        It corresponds to the maximum number of ports to be
        used by a subscriber.";
    reference
        "REQ-4 of RFC 6888.";
}

leaf quota-type {
    type uint8;
    description
        "Indicates whether the port quota applies to
        all protocols (0) or to a specific protocol.";
}
}

container port-set {

    when "../port-allocation-type = 'port-range-allocation'";

    if-feature "napt44 or nat64";
    description
        "Manages port-set assignments.";

    leaf port-set-size {
        type uint16;
        mandatory true;
        description
            "Indicates the size of assigned port sets.";
    }

    leaf port-set-timeout {
        type uint32;
        units "seconds";
        description
            "inactivity timeout for port sets.";
    }
}

container timers {
    if-feature "napt44 or nat64";
    description
        "Configure values of various timeouts.";

    leaf udp-timeout {
        type uint32;
        units "seconds";
        default 300;
        description
            "UDP inactivity timeout. That is the time a mapping
```



```
        will stay active without packets traversing the NAT.";
    reference
        "RFC 4787: Network Address Translation (NAT)
        Behavioral Requirements for Unicast UDP";
}

leaf tcp-idle-timeout {
    type uint32;
    units "seconds";
    default 7440;
    description
        "TCP Idle timeout should be 2 hours and 4 minutes.";
    reference
        "RFC 5382: NAT Behavioral Requirements for TCP";
}

leaf tcp-trans-open-timeout {
    type uint32;
    units "seconds";
    default 240;
    description
        "The value of the transitory open connection
        idle-timeout.

        A NAT should provide different configurable
        parameters for configuring the open and
        closing idle timeouts.

        To accommodate deployments that consider
        a partially open timeout of 4 minutes as being
        excessive from a security standpoint, a NAT may
        allow the configured timeout to be less than
        4 minutes.

        However, a minimum default transitory connection
        idle-timeout of 4 minutes is recommended.";
    reference
        "Section 2.1 of RFC 7857.";
}

leaf tcp-trans-close-timeout {
    type uint32;
    units "seconds";
    default 240;
    description
        "The value of the transitory close connection
        idle-timeout.
```

```
        A NAT should provide different configurable
        parameters for configuring the open and
        closing idle timeouts.";
    reference
        "Section 2.1 of RFC 7857.";
}

leaf tcp-in-syn-timeout {
    type uint32;
    units "seconds";
    default 6;
    description
        "A NAT must not respond to an unsolicited
        inbound SYN packet for at least 6 seconds
        after the packet is received. If during
        this interval the NAT receives and translates
        an outbound SYN for the connection the NAT
        must silently drop the original unsolicited
        inbound SYN packet.";
    reference
        "RFC 5382 NAT Behavioral Requirements for TCP";
}

leaf fragment-min-timeout {
    when "../..//fragment-behavior='out-of-order'";
    type uint32;
    units "seconds";
    default 2;
    description
        "As long as the NAT has available resources,
        the NAT allows the fragments to arrive
        over fragment-min-timeout interval.
        The default value is inspired from RFC6146.";
}

leaf icmp-timeout {
    type uint32;
    units "seconds";
    default 60;
    description
        "An ICMP Query session timer must not expire
        in less than 60 seconds. It is recommended
        that the ICMP Query session timer be made
        configurable";
    reference
        "RFC 5508: NAT Behavioral Requirements for ICMP";
}
```

```
list per-port-timeout {
  key port-number;
  description
    "Some NATs are configurable with short timeouts
    for some ports, e.g., as 10 seconds on
    port 53 (DNS) and 123 (NTP) and longer timeouts
    on other ports.";

  leaf port-number {
    type inet:port-number;
    description
      "A port number.";
  }

  leaf protocol {
    type uint8;
    description
      "Upper-layer protocol associated with this port.

      Values are taken from the IANA protocol registry.

      If no protocol is indicated, this means 'any
      protocol'.";
  }

  leaf timeout {
    type uint32;
    units "seconds";
    mandatory true;
    description
      "Timeout for this port number";
  }
}

leaf hold-down-timeout {
  type uint32;
  units "seconds";
  default 120;
  description
    "Hold down timer.

    Ports in the hold down pool are not reassigned until
    hold-down-timeout expires.

    The length of time and the maximum number of ports in
    this state must be configurable by the administrator.

    This is necessary in order to prevent collisions
```

```
        between old and new mappings and sessions. It ensures
        that all established sessions are broken instead of
        redirected to a different peer.";
    reference
        "REQ#8 of RFC 6888.";
}

leaf hold-down-max {
    type uint32;
    description
        "Maximum ports in the hold down port pool.";
    reference
        "REQ#8 of RFC 6888.";
}

leaf fragments-limit{
    when "../fragment-behavior='out-of-order'";
    type uint32;
    description
        "Limits the number of out of order fragments that can
        be handled.";
    reference
        "Section 11 of RFC 4787.";
}

list algs {
    key name;
    description
        "ALG-related features.";

    leaf name {
        type string;
        description
            "The name of the ALG.";
    }

    leaf transport-protocol {
        type uint32;
        description
            "The transport protocol used by the ALG
            (e.g., TCP, UDP).";
    }

    container dst-transport-port {
        uses port-number;
        description
            "The destination port number(s) used by the ALG."
    }
}
```

```
        For example,
        - 21 for the FTP ALG
        - 53 for the DNS ALG.";
    }

    container src-transport-port {
        uses port-number;
        description
            "The source port number(s) used by the ALG.";
    }

    leaf status {
        type boolean;
        description
            "Enable/disable the ALG.";
    }
}

leaf all-algs-enable {
    type boolean;
    description
        "Disable/enable all ALGs.

        When specified, this parameter overrides the one
        that may be indicated, eventually, by the 'status'
        of an individual ALG.";
}

container notify-pool-usage {
    if-feature "basic-nat44 or napt44 or nat64";
    description
        "Notification of pool usage when certain criteria
        are met.";

    leaf pool-id {
        type uint32;
        description
            "Pool-ID for which the notification criteria
            is defined";
    }

    leaf low-threshold {
        type percent;
        description
            "Notification must be generated when the defined low
            threshold is reached.

            For example, if a notification is required when the
```

```
    pool utilization reaches below 10%, this
    configuration parameter must be set to 10.

    0% indicates that low-threshold notification is
    disabled.";
}

leaf high-threshold {
    type percent;
    must ". >= ../low-threshold" {
        error-message
            "The high threshold must be greater than or equal
            to the low threshold.";
    }
    description
        "Notification must be generated when the defined high
        threshold is reached.

        For example, if a notification is required when the
        pool utilization reaches 90%, this configuration
        parameter must be set to 90.

        Setting the same value as low-threshold is equivalent
        to disabling high-threshold notification.";
}

leaf notify-interval {
    type uint32 {
        range "1 .. 3600";
    }
    units "seconds";
    default '20';
    description
        "Minimum number of seconds between successive
        notifications for this pool.";

    reference
        "RFC 7659: Definitions of Managed Objects for
        Network Address Translators (NATs)";
}

container external-realm {
    description
        "Identifies the external realm of the NAT instance.";

    choice realm-type {
        description
```

```
        "Can be an interface, VRF instance, etc.";

    case interface {
        description
            "External interface.";

        leaf external-interface {
            type if:interface-ref;
            description
                "Name of the external interface.";
        }
    }
}

container mapping-limits {
    if-feature "napt44 or nat64";
    description
        "Information about the configuration parameters that
        limits the mappings based upon various criteria.";

    leaf limit-subscribers {
        type uint32;
        description
            "Maximum number of subscribers that can be serviced
            by a NAT instance.

            A subscriber is identified by a given prefix.";
        reference
            "RFC 7659: Definitions of Managed Objects for
            Network Address Translators (NATs)";
    }

    leaf limit-address-mappings {
        type uint32;
        description
            "Maximum number of address mappings that can be
            handled by a NAT instance.

            When this limit is reached, packets that would
            normally trigger translation, will be dropped.";
        reference
            "RFC 7659: Definitions of Managed Objects
            for Network Address Translators
            (NATs)";
    }
}
```

```
leaf limit-port-mappings {
  type uint32;
  description
    "Maximum number of port mappings that can be handled
    by a NAT instance.

    When this limit is reached, packets that would
    normally trigger translation, will be dropped.";
  reference
    "RFC 7659: Definitions of Managed Objects for
    Network Address Translators (NATs)";
}

list limit-per-protocol {
  if-feature "napt44 or nat64 or dst-nat";
  key protocol-id;

  description
    "Configure limits per transport protocol";

  leaf protocol-id {
    type uint8;
    mandatory true;
    description
      "Upper-layer protocol.

      Values are taken from the IANA protocol registry.

      For example, this field contains 6 for TCP,
      17 for UDP, 33 for DCCP, or 132 for SCTP.";
  }

  leaf limit {
    type uint32;
    description
      "Maximum number of protocol-specific NAT mappings
      per instance.";
  }
}

container connection-limits {
  if-feature "basic-nat44 or napt44 or nat64";
  description
    "Information about the configuration parameters that
    rate limit the translation based upon various criteria.";

  leaf limit-per-subscriber {
```



```
    type uint32;
    units "bits/second";
    description
        "Rate-limit the number of new mappings and sessions
        per subscriber.";
}

leaf limit-per-instance {
    type uint32;
    units "bits/second";
    description
        "Rate-limit the number of new mappings and sessions
        per instance.";
}

list limit-per-protocol {
    if-feature "napt44 or nat64";
    key protocol-id;
    description
        "Configure limits per transport protocol";

    leaf protocol-id {
        type uint8;
        mandatory true;
        description
            "Upper-layer protocol.

            Values are taken from the IANA protocol registry.

            For example, this field contains 6 for TCP,
            17 for UDP, 33 for DCCP, or 132 for SCTP.";
    }

    leaf limit {
        type uint32;
        description
            "Limit the number of protocol-specific mappings
            and sessions per instance.";
    }
}

container notification-limits {
    description "Sets notification limits.";

    leaf notify-interval {
        if-feature "basic-nat44 or napt44 or nat64";
        type uint32 {
```

```
        range "1 .. 3600";
    }
    units "seconds";
    default '10';
    description
        "Minimum number of seconds between successive
        notifications for this NAT instance.";
    reference
        "RFC 7659: Definitions of Managed Objects
        for Network Address Translators (NATs)";
}

leaf notify-addresses-usage {
    if-feature "basic-nat44 or napt44 or nat64";
    type percent;
    description
        "Notification of address mappings usage over
        the whole NAT instance.

        Notification must be generated when the defined
        threshold is reached.

        For example, if a notification is required when
        the address mappings utilization reaches 90%,
        this configuration parameter must be set
        to 90.";
}

leaf notify-ports-usage {
    if-feature "napt44 or nat64";
    type percent;
    description
        "Notification of port mappings usage over the
        whole NAT instance.

        Notification must be generated when the defined
        threshold is reached.

        For example, if a notification is required when
        the port mappings utilization reaches 90%, this
        configuration parameter must be set to 90.";
}

leaf notify-subscribers-limit {
    if-feature "basic-nat44 or napt44 or nat64";
    type uint32;
    description
        "Notification of active subscribers per NAT
```

```
        instance.

        Notification must be generated when the defined
        threshold is reached.";
    }
}

container mapping-table {
    if-feature "basic-nat44 or napt44 " +
        "or nat64 or clat or dst-nat";
    description
        "NAT mapping table. Applicable for functions which maintain
        static and/or dynamic mappings, such as NAT44, Destination
        NAT, NAT64, or CLAT.";

    list mapping-entry {
        key "index";
        description "NAT mapping entry.";
        uses mapping-entry;
    }
}

container statistics {
    config false;

    description
        "Statistics related to the NAT instance.";

    leaf discontinuity-time {
        type yang:date-and-time;
        mandatory true;
        description
            "The time on the most recent occasion at which the NAT
            instance suffered a discontinuity. This must be
            initialized when the NAT instance is configured
            or rebooted.";
    }
}

container traffic-statistics {
    description
        "Generic traffic statistics.";

    leaf sent-packets {
        type yang:zero-based-counter64;
        description
            "Number of packets sent.";
    }
}
```

```
leaf sent-bytes {
  type yang:zero-based-counter64;
  units 'bytes';
  description
    "Counter for sent traffic in bytes.";
}

leaf rcvd-packets {
  type yang:zero-based-counter64;
  description
    "Number of received packets.";
}

leaf rcvd-bytes {
  type yang:zero-based-counter64;
  units 'bytes';
  description
    "Counter for received traffic in bytes.";
}

leaf dropped-packets {
  type yang:zero-based-counter64;
  description
    "Number of dropped packets.";
}

leaf dropped-bytes {
  type yang:zero-based-counter64;
  units 'bytes';
  description
    "Counter for dropped traffic in bytes.";
}

leaf dropped-fragments {
  if-feature "napt44 or nat64";
  type yang:zero-based-counter64;
  description
    "Number of dropped fragments on the external realm.";
}

leaf dropped-address-limit-packets {
  if-feature "basic-nat44 or napt44 or nat64";
  type yang:zero-based-counter64;
  description
    "Number of dropped packets because an address limit
    is reached.";
}
```

```
leaf dropped-address-limit-bytes {
  if-feature "basic-nat44 or napt44 or nat64";
  type yang:zero-based-counter64;
  units 'bytes';
  description
    "Counter of dropped packets because an address limit
     is reached, in bytes.";
}

leaf dropped-address-packets {
  if-feature "basic-nat44 or napt44 or nat64";
  type yang:zero-based-counter64;
  description
    "Number of dropped packets because no address is
     available for allocation.";
}

leaf dropped-address-bytes {
  if-feature "basic-nat44 or napt44 or nat64";
  type yang:zero-based-counter64;
  units 'bytes';
  description
    "Counter of dropped packets because no address is
     available for allocation, in bytes.";
}

leaf dropped-port-limit-packets {
  if-feature "napt44 or nat64";
  type yang:zero-based-counter64;
  description
    "Number of dropped packets because a port limit
     is reached.";
}

leaf dropped-port-limit-bytes {
  if-feature "napt44 or nat64";
  type yang:zero-based-counter64;
  units 'bytes';
  description
    "Counter of dropped packets because a port limit
     is reached, in bytes.";
}

leaf dropped-port-packets {
  if-feature "napt44 or nat64";
  type yang:zero-based-counter64;
  description
    "Number of dropped packets because no port is
```

```
        available for allocation.";
    }

    leaf dropped-port-bytes {
        if-feature "napt44 or nat64";
        type yang:zero-based-counter64;
        units 'bytes';
        description
            "Counter of dropped packets because no port is
             available for allocation, in bytes.";
    }

    leaf dropped-subscriber-limit-packets {
        if-feature "basic-nat44 or napt44 or nat64";
        type yang:zero-based-counter64;
        description
            "Number of dropped packets because the subscriber
             limit per instance is reached.";
    }

    leaf dropped-subscriber-limit-bytes {
        if-feature "basic-nat44 or napt44 or nat64";
        type yang:zero-based-counter64;
        units 'bytes';
        description
            "Counter of dropped packets because the subscriber
             limit per instance is reached, in bytes.";
    }
}

container mappings-statistics {
    description
        "Mappings statistics.";

    leaf total-active-subscribers {
        if-feature "basic-nat44 or napt44 or nat64";
        type yang:gauge32;
        description
            "Total number of active subscribers (that is,
             subscribers for which the NAT maintains active
             mappings.

             A subscriber is identified by a subnet,
             subscriber-mask, etc.";
    }

    leaf total-address-mappings {
        if-feature "basic-nat44 or napt44 " +
```

```
    "or nat64 or clat or dst-nat";
    type yang:gauge32;
    description
        "Total number of address mappings present at a given
        time. It includes both static and dynamic mappings.";
    reference
        "Section 3.3.8 of RFC 7659";
}

leaf total-port-mappings {
    if-feature "napt44 or nat64";
    type yang:gauge32;
    description
        "Total number of NAT port mappings present at
        a given time. It includes both static and dynamic
        mappings.";
    reference
        "Section 3.3.9 of RFC 7659";
}

list total-per-protocol {
    if-feature "napt44 or nat64";
    key protocol-id;
    description
        "Total mappings for each enabled/supported protocol.";

    leaf protocol-id {
        type uint8;
        mandatory true;
        description
            "Upper-layer protocol.
            For example, this field contains 6 for TCP,
            17 for UDP, 33 for DCCP, or 132 for SCTP.";
    }

    leaf total {
        type yang:gauge32;
        description
            "Total number of a protocol-specific mappings present
            at a given time. The protocol is identified by
            protocol-id.";
    }
}

container pools-stats {
    if-feature "basic-nat44 or napt44 or nat64";
    description
```

```
    "Statistics related to address/prefix pools
    usage";

    leaf addresses-allocated {
        type yang:gauge32;
        description
            "Number of all allocated addresses.";
    }

    leaf addresses-free {
        type yang:gauge32;
        description
            "Number of unallocated addresses of all pools at
            a given time. The sum of unallocated and allocated
            addresses is the total number of addresses of
            the pools.";
    }

    container ports-stats {
        if-feature "napt44 or nat64";

        description
            "Statistics related to port numbers usage.";

        leaf ports-allocated {
            type yang:gauge32;
            description
                "Number of allocated ports from all pools.";
        }

        leaf ports-free {
            type yang:gauge32;
            description
                "Number of unallocated addresses from all pools.";
        }
    }

    list per-pool-stats {
        if-feature "basic-nat44 or napt44 or nat64";
        key "pool-id";
        description
            "Statistics related to address/prefix pool usage";

        leaf pool-id {
            type uint32;
            description
                "Unique Identifier that represents a pool of
                addresses/prefixes.";
        }
    }
}
```



```
    }

    leaf discontinuity-time {
        type yang:date-and-time;
        mandatory true;
        description
            "The time on the most recent occasion at which this
            pool counters suffered a discontinuity. This must
            be initialized when the address pool is
            configured.";
    }

    container pool-stats {
        description
            "Statistics related to address/prefix pool usage";

        leaf addresses-allocated {
            type yang:gauge32;
            description
                "Number of allocated addresses from this pool.";
        }

        leaf addresses-free {
            type yang:gauge32;
            description
                "Number of unallocated addresses in this pool.";
        }
    }

    container port-stats {
        if-feature "napt44 or nat64";
        description
            "Statistics related to port numbers usage.";

        leaf ports-allocated {
            type yang:gauge32;
            description
                "Number of allocated ports from this pool.";
        }

        leaf ports-free {
            type yang:gauge32;
            description
                "Number of unallocated addresses from this pool.";
        }
    }
}
```

```
    }  
  }  
}  
  
/*  
 * Notifications  
 */  
  
notification nat-pool-event {  
  if-feature "basic-nat44 or napt44 or nat64";  
  description  
    "Notifications must be generated when the defined high/low  
    threshold is reached. Related configuration parameters  
    must be provided to trigger the notifications.";   
  
  leaf id {  
    type leafref {  
      path "/nat/instances/instance/id";  
    }  
    mandatory true;  
    description  
      "NAT instance Identifier.";  
  }  
  
  leaf policy-id {  
    type leafref {  
      path "/nat/instances/instance/policy/id";  
    }  
  
    description  
      "Policy Identifier.";  
  }  
  
  leaf pool-id {  
    type leafref {  
      path "/nat/instances/instance/policy/" +  
        "external-ip-address-pool/pool-id";  
    }  
    mandatory true;  
    description  
      "Pool Identifier.";  
  }  
  
  leaf notify-pool-threshold {  
    type percent;  
    mandatory true;  
    description
```

```
        "A threshold (high-threshold or low-threshold) has
        been fired.";
    }
}

notification nat-instance-event {
    if-feature "basic-nat44 or napt44 or nat64";
    description
        "Notifications must be generated when notify-addresses-usage
        and/or notify-ports-usage threshold are reached.";

    leaf id {
        type leafref {
            path "/nat/instances/instance/id";
        }
        mandatory true;
        description
            "NAT instance Identifier.";
    }

    leaf notify-subscribers-threshold {
        type uint32;
        description
            "The notify-subscribers-limit threshold has been fired.";
    }

    leaf notify-addresses-threshold {
        type percent;
        description
            "The notify-addresses-usage threshold has been fired.";
    }

    leaf notify-ports-threshold {
        type percent;
        description
            "The notify-ports-usage threshold has been fired.";
    }
}
}
<CODE ENDS>
```

4. Security Considerations

Security considerations related to address and prefix translation are discussed in [RFC6888], [RFC6146], [RFC6877], [RFC6296], and [RFC7757].

The YANG module defined in this document is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC5246].

The NETCONF access control model [RFC8341] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

All data nodes defined in the YANG module which can be created, modified and deleted (i.e., config true, which is the default) are considered sensitive. Write operations (e.g., edit-config) applied to these data nodes without proper protection can negatively affect network operations. The NAT YANG module provides a method to set parameters to prevent a user from aggressively using NAT resources (port-quota), rate-limit connections as a guard against Denial-of-Service, or to enable notifications so that appropriate measures are enforced to anticipate traffic drops. Nevertheless, an attacker who is able to access the NAT can undertake various attacks, such as:

- o Set a high or low resource limit to cause a DoS attack:
 - * /nat/instances/instance/policy/port-quota
 - * /nat/instances/instance/policy/fragments-limit
 - * /nat/instances/instance/mapping-limits
 - * /nat/instances/instance/connection-limits
- o Set a low notification threshold to cause useless notifications to be generated:
 - * /nat/instances/instance/policy/notify-pool-usage/high-threshold
 - * /nat/instances/instance/notification-limits/notify-addresses-usage
 - * /nat/instances/instance/notification-limits/notify-ports-usage
 - * /nat/instances/instance/notification-limits/notify-subscribers-limit
- o Set an arbitrarily high threshold, which may lead to the deactivation of notifications:

- * /nat/instances/instance/policy/notify-pool-usage/high-threshold
- * /nat/instances/instance/notification-limits/notify-addresses-usage
- * /nat/instances/instance/notification-limits/notify-ports-usage
- * /nat/instances/instance/notification-limits/notify-subscribers-limit
- o Set a low notification interval and a low notification threshold to induce useless notifications to be generated:
 - * /nat/instances/instance/policy/notify-pool-usage/notify-interval
 - * /nat/instances/instance/notification-limits/notify-interval
- o Access to privacy data maintained in the mapping table. Such data can be misused to track the activity of a host:
 - * /nat/instances/instance/mapping-table

5. IANA Considerations

This document requests IANA to register the following URI in the "IETF XML Registry" [RFC3688]:

URI: urn:ietf:params:xml:ns:yang:ietf-nat
Registrant Contact: The IESG.
XML: N/A; the requested URI is an XML namespace.

This document requests IANA to register the following YANG module in the "YANG Module Names" registry [RFC7950].

name: ietf-nat
namespace: urn:ietf:params:xml:ns:yang:ietf-nat
prefix: nat
reference: RFC XXXX

6. Acknowledgements

Many thanks to Dan Wing, Tianran Zhou, Tom Petch, Warren Kumari, and Benjamin Kaduk for the review.

Thanks to Juergen Schoenwaelder for the comments on the YANG structure and the suggestion to use NMDA. Mahesh Jethanandani provided useful comments.

Thanks to Lee Howard and Jordi Palet for the CLAT comments, Fred Baker for the NPTv6 comments, Tore Anderson for EAM SIIT review, and Kristian Poscic for the CGN review.

Special thanks to Maros Marsalek and Marek Gradzki for sharing their comments based on the FD.io implementation of this module (<https://git.fd.io/hc2vpp/tree/nat/nat-api/src/main/yang>).

Rajiv Asati suggested to clarify how the module applies for both stateless and stateful NAT64.

Juergen Schoenwaelder provided an early yandgoctors review. Many thanks to him.

Thanks to Roni Even, Mach Chen, Tim Chown, and Stephen Farrel for the directorates review. Igor Ryzhov identified a nit in one example.

Mirja Kuehlewind made a comment about the reuse of some TCP timers for any connection-oriented protocol.

7. References

7.1. Normative References

- [RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <<https://www.rfc-editor.org/info/rfc3688>>.
- [RFC4787] Audet, F., Ed. and C. Jennings, "Network Address Translation (NAT) Behavioral Requirements for Unicast UDP", BCP 127, RFC 4787, DOI 10.17487/RFC4787, January 2007, <<https://www.rfc-editor.org/info/rfc4787>>.
- [RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <<https://www.rfc-editor.org/info/rfc5246>>.
- [RFC5382] Guha, S., Ed., Biswas, K., Ford, B., Sivakumar, S., and P. Srisuresh, "NAT Behavioral Requirements for TCP", BCP 142, RFC 5382, DOI 10.17487/RFC5382, October 2008, <<https://www.rfc-editor.org/info/rfc5382>>.

- [RFC5508] Srisuresh, P., Ford, B., Sivakumar, S., and S. Guha, "NAT Behavioral Requirements for ICMP", BCP 148, RFC 5508, DOI 10.17487/RFC5508, April 2009, <<https://www.rfc-editor.org/info/rfc5508>>.
- [RFC6052] Bao, C., Huitema, C., Bagnulo, M., Boucadair, M., and X. Li, "IPv6 Addressing of IPv4/IPv6 Translators", RFC 6052, DOI 10.17487/RFC6052, October 2010, <<https://www.rfc-editor.org/info/rfc6052>>.
- [RFC6146] Bagnulo, M., Matthews, P., and I. van Beijnum, "Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers", RFC 6146, DOI 10.17487/RFC6146, April 2011, <<https://www.rfc-editor.org/info/rfc6146>>.
- [RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <<https://www.rfc-editor.org/info/rfc6241>>.
- [RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <<https://www.rfc-editor.org/info/rfc6242>>.
- [RFC6296] Wasserman, M. and F. Baker, "IPv6-to-IPv6 Network Prefix Translation", RFC 6296, DOI 10.17487/RFC6296, June 2011, <<https://www.rfc-editor.org/info/rfc6296>>.
- [RFC6619] Arkko, J., Eggert, L., and M. Townsley, "Scalable Operation of Address Translators with Per-Interface Bindings", RFC 6619, DOI 10.17487/RFC6619, June 2012, <<https://www.rfc-editor.org/info/rfc6619>>.
- [RFC6877] Mawatari, M., Kawashima, M., and C. Byrne, "464XLAT: Combination of Stateful and Stateless Translation", RFC 6877, DOI 10.17487/RFC6877, April 2013, <<https://www.rfc-editor.org/info/rfc6877>>.
- [RFC6888] Perreault, S., Ed., Yamagata, I., Miyakawa, S., Nakagawa, A., and H. Ashida, "Common Requirements for Carrier-Grade NATs (CGNs)", BCP 127, RFC 6888, DOI 10.17487/RFC6888, April 2013, <<https://www.rfc-editor.org/info/rfc6888>>.
- [RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <<https://www.rfc-editor.org/info/rfc6991>>.

- [RFC7596] Cui, Y., Sun, Q., Boucadair, M., Tsou, T., Lee, Y., and I. Farrer, "Lightweight 4over6: An Extension to the Dual-Stack Lite Architecture", RFC 7596, DOI 10.17487/RFC7596, July 2015, <<https://www.rfc-editor.org/info/rfc7596>>.
- [RFC7597] Troan, O., Ed., Dec, W., Li, X., Bao, C., Matsushima, S., Murakami, T., and T. Taylor, Ed., "Mapping of Address and Port with Encapsulation (MAP-E)", RFC 7597, DOI 10.17487/RFC7597, July 2015, <<https://www.rfc-editor.org/info/rfc7597>>.
- [RFC7757] Anderson, T. and A. Leiva Popper, "Explicit Address Mappings for Stateless IP/ICMP Translation", RFC 7757, DOI 10.17487/RFC7757, February 2016, <<https://www.rfc-editor.org/info/rfc7757>>.
- [RFC7857] Penno, R., Perreault, S., Boucadair, M., Ed., Sivakumar, S., and K. Naito, "Updates to Network Address Translation (NAT) Behavioral Requirements", BCP 127, RFC 7857, DOI 10.17487/RFC7857, April 2016, <<https://www.rfc-editor.org/info/rfc7857>>.
- [RFC7915] Bao, C., Li, X., Baker, F., Anderson, T., and F. Gont, "IP/ICMP Translation Algorithm", RFC 7915, DOI 10.17487/RFC7915, June 2016, <<https://www.rfc-editor.org/info/rfc7915>>.
- [RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <<https://www.rfc-editor.org/info/rfc7950>>.
- [RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <<https://www.rfc-editor.org/info/rfc8040>>.
- [RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", STD 91, RFC 8341, DOI 10.17487/RFC8341, March 2018, <<https://www.rfc-editor.org/info/rfc8341>>.
- [RFC8343] Bjorklund, M., "A YANG Data Model for Interface Management", RFC 8343, DOI 10.17487/RFC8343, March 2018, <<https://www.rfc-editor.org/info/rfc8343>>.

7.2. Informative References

- [I-D.boucadair-pcp-yang]
Boucadair, M., Jacquenet, C., Sivakumar, S., and S. Vinapamula, "YANG Modules for the Port Control Protocol (PCP)", draft-boucadair-pcp-yang-05 (work in progress), October 2017.
- [I-D.ietf-softwire-dslite-yang]
Boucadair, M., Jacquenet, C., and S. Sivakumar, "A YANG Data Model for Dual-Stack Lite (DS-Lite)", draft-ietf-softwire-dslite-yang-17 (work in progress), May 2018.
- [I-D.ietf-tsvwg-natsupp]
Stewart, R., Tuexen, M., and I. Ruengeler, "Stream Control Transmission Protocol (SCTP) Network Address Translation Support", draft-ietf-tsvwg-natsupp-12 (work in progress), July 2018.
- [RFC2663] Srisuresh, P. and M. Holdrege, "IP Network Address Translator (NAT) Terminology and Considerations", RFC 2663, DOI 10.17487/RFC2663, August 1999, <<https://www.rfc-editor.org/info/rfc2663>>.
- [RFC3022] Srisuresh, P. and K. Egevang, "Traditional IP Network Address Translator (Traditional NAT)", RFC 3022, DOI 10.17487/RFC3022, January 2001, <<https://www.rfc-editor.org/info/rfc3022>>.
- [RFC5597] Denis-Courmont, R., "Network Address Translation (NAT) Behavioral Requirements for the Datagram Congestion Control Protocol", BCP 150, RFC 5597, DOI 10.17487/RFC5597, September 2009, <<https://www.rfc-editor.org/info/rfc5597>>.
- [RFC6087] Bierman, A., "Guidelines for Authors and Reviewers of YANG Data Model Documents", RFC 6087, DOI 10.17487/RFC6087, January 2011, <<https://www.rfc-editor.org/info/rfc6087>>.
- [RFC6269] Ford, M., Ed., Boucadair, M., Durand, A., Levis, P., and P. Roberts, "Issues with IP Address Sharing", RFC 6269, DOI 10.17487/RFC6269, June 2011, <<https://www.rfc-editor.org/info/rfc6269>>.
- [RFC6736] Brockners, F., Bhandari, S., Singh, V., and V. Fajardo, "Diameter Network Address and Port Translation Control Application", RFC 6736, DOI 10.17487/RFC6736, October 2012, <<https://www.rfc-editor.org/info/rfc6736>>.

- [RFC6887] Wing, D., Ed., Cheshire, S., Boucadair, M., Penno, R., and P. Selkirk, "Port Control Protocol (PCP)", RFC 6887, DOI 10.17487/RFC6887, April 2013, <<https://www.rfc-editor.org/info/rfc6887>>.
- [RFC6908] Lee, Y., Maglione, R., Williams, C., Jacquenet, C., and M. Boucadair, "Deployment Considerations for Dual-Stack Lite", RFC 6908, DOI 10.17487/RFC6908, March 2013, <<https://www.rfc-editor.org/info/rfc6908>>.
- [RFC7050] Savolainen, T., Korhonen, J., and D. Wing, "Discovery of the IPv6 Prefix Used for IPv6 Address Synthesis", RFC 7050, DOI 10.17487/RFC7050, November 2013, <<https://www.rfc-editor.org/info/rfc7050>>.
- [RFC7289] Kuarsingh, V., Ed. and J. Cianfarani, "Carrier-Grade NAT (CGN) Deployment with BGP/MPLS IP VPNs", RFC 7289, DOI 10.17487/RFC7289, June 2014, <<https://www.rfc-editor.org/info/rfc7289>>.
- [RFC7335] Byrne, C., "IPv4 Service Continuity Prefix", RFC 7335, DOI 10.17487/RFC7335, August 2014, <<https://www.rfc-editor.org/info/rfc7335>>.
- [RFC7659] Perreault, S., Tsou, T., Sivakumar, S., and T. Taylor, "Definitions of Managed Objects for Network Address Translators (NATs)", RFC 7659, DOI 10.17487/RFC7659, October 2015, <<https://www.rfc-editor.org/info/rfc7659>>.
- [RFC7753] Sun, Q., Boucadair, M., Sivakumar, S., Zhou, C., Tsou, T., and S. Perreault, "Port Control Protocol (PCP) Extension for Port-Set Allocation", RFC 7753, DOI 10.17487/RFC7753, February 2016, <<https://www.rfc-editor.org/info/rfc7753>>.
- [RFC8045] Cheng, D., Korhonen, J., Boucadair, M., and S. Sivakumar, "RADIUS Extensions for IP Port Configuration and Reporting", RFC 8045, DOI 10.17487/RFC8045, January 2017, <<https://www.rfc-editor.org/info/rfc8045>>.
- [RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <<https://www.rfc-editor.org/info/rfc8340>>.

Appendix A. Sample Examples

This section provides a non-exhaustive set of examples to illustrate the use of the NAT YANG module.

A.1. Traditional NAT44

Traditional NAT44 is a Basic NAT44 or NAPT that is used to share the same IPv4 address among hosts that are owned by the same subscriber. This is typically the NAT that is embedded in CPE devices.

This NAT is usually provided with one single external IPv4 address; disambiguating connections is achieved by rewriting the source port number. The XML snippet to configure the external IPv4 address in such case together with a mapping entry is depicted below:

```
<instances>
  <instance>
    <id>1</id>
    <name>NAT_Subscriber_A</name>
    ....
    <external-ip-address-pool>
      <pool-id>1</pool-id>
      <external-ip-pool>
        198.51.100.1/32
      </external-ip-pool>
    </external-ip-address-pool>
    ....
    <mapping-table>
      ....
      <external-src-address>
        198.51.100.1/32
      </external-src-address>
      ....
    </mapping-table>
  </instance>
</instances>
```

The following shows the XML excerpt depicting a dynamic UDP mapping entry maintained by a traditional NAPT44. In reference to this example, the UDP packet received with a source IPv4 address (192.0.2.1) and source port number (1568) is translated into a UDP packet having a source IPv4 address (198.51.100.1) and source port (15000). The remaining lifetime of this mapping is 300 seconds.

```
<mapping-entry>
  <index>15</index>
  <type>
    dynamic-explicit
  </type>
  <transport-protocol>
    17
  </transport-protocol>
  <internal-src-address>
    192.0.2.1/32
  </internal-src-address>
  <internal-src-port>
    <start-port-number>
      1568
    </start-port-number>
  </internal-src-port>
  <external-src-address>
    198.51.100.1/32
  </external-src-address>
  <external-src-port>
    <start-port-number>
      15000
    </start-port-number>
  </external-src-port>
  <lifetime>
    300
  </lifetime>
</mapping-entry>
```

A.2. Carrier Grade NAT (CGN)

The following XML snippet shows the example of the capabilities supported by a CGN as retrieved using NETCONF.

```
<capabilities>
  <nat-flavor>napt44</nat-flavor>
  <transport-protocols>
    <protocol-id>1</protocol-id>
  </transport-protocols>
  <transport-protocols>
    <protocol-id>6</protocol-id>
  </transport-protocols>
  <transport-protocols>
    <protocol-id>17</protocol-id>
  </transport-protocols>
  <restricted-port-support>
    false
  </restricted-port-support>
```

```
<static-mapping-support>
  true
</static-mapping-support>
<port-randomization-support>
  true
</port-randomization-support>
<port-range-allocation-support>
  true
</port-range-allocation-support>
<port-preservation-suport>
  true
</port-preservation-suport>
<port-parity-preservation-support>
  false
</port-parity-preservation-support>
<address-roundrobin-support>
  true
</address-roundrobin-support>
<paired-address-pooling-support>
  true
</paired-address-pooling-support>
<endpoint-independent-mapping-support>
  true
</endpoint-independent-mapping-support>
<address-dependent-mapping-support>
  true
</address-dependent-mapping-support>
<address-and-port-dependent-mapping-support>
  true
</address-and-port-dependent-mapping-support>
<endpoint-independent-filtering-support>
  true
</endpoint-independent-filtering-support>
<address-dependent-filtering>
  true
</address-dependent-filtering>
<address-and-port-dependent-filtering>
  true
</address-and-port-dependent-filtering>
</capabilities>
```

The following XML snippet shows the example of a CGN that is provisioned with one contiguous pool of external IPv4 addresses (198.51.100.0/24). Further, the CGN is instructed to limit the number of allocated ports per subscriber to 1024. Ports can be allocated by the CGN by assigning ranges of 256 ports (that is, a subscriber can be allocated up to four port ranges of 256 ports each).

```
<instances>
  <instance>
    <id>1</id>
    <name>myCGN</name>
    ....
    <external-ip-address-pool>
      <pool-id>1</pool-id>
      <external-ip-pool>
        198.51.100.0/24
      </external-ip-pool>
    </external-ip-address-pool>
    <port-quota>
      <port-limit>
        1024
      </port-limit>
      <quota-type >
        all
      </quota-type >
    </port-quota>
    <port-allocation-type>
      port-range-allocation
    </port-allocation-type>
    <port-set>
      <port-set-size>
        256
      </port-set-size>
    </port-set>
    ....
  </instance>
</instances>
```

An administrator may decide to allocate one single port range per subscriber (e.g., port range of 1024 ports) as shown below:

```

<instances>
  <instance>
    <id>1</id>
    <name>myCGN</name>
    ....
    <external-ip-address-pool>
      <pool-id>1</pool-id>
      <external-ip-pool>
        198.51.100.0/24
      </external-ip-pool>
    </external-ip-address-pool>
    <port-quota>
      <port-limit>
        1024
      </port-limit>
      <quota-type >
        all
      </quota-type >
    </port-quota>
    <port-allocation-type>
      port-range-allocation
    </port-allocation-type>
    <port-set>
      <port-set-size>
        1024
      </port-set-size>
    </port-set>
    ....
  </instance>
</instances>

```

A.3. CGN Pass-Through

Figure 1 illustrates an example of the CGN pass-through feature.

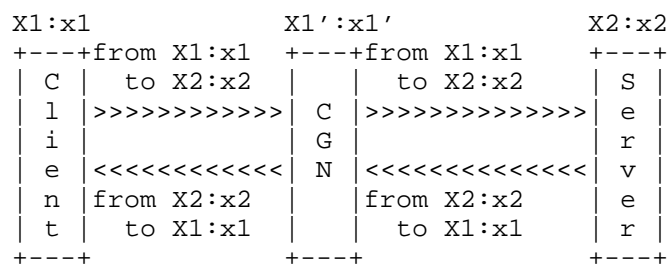


Figure 1: CGN Pass-Through

For example, in order to disable NAT for communications issued by the client (192.0.2.1), the following configuration parameter must be set:

```
<nat-pass-through>
  ...
  <prefix>192.0.2.1/32</prefix>
  ...
</nat-pass-through>
```

A.4. NAT64

Let's consider the example of a NAT64 that should use 2001:db8:122:300::/56 to perform IPv6 address synthesis [RFC6052]. The XML snippet to configure the NAT64 prefix in such case is depicted below:

```
<nat64-prefixes>
  <nat64-prefix>
    2001:db8:122:300::/56
  </nat64-prefix>
</nat64-prefixes>
```

Let's now consider the example of a NAT64 that should use 2001:db8:122::/48 to perform IPv6 address synthesis [RFC6052] only if the destination address matches 198.51.100.0/24. The XML snippet to configure the NAT64 prefix in such case is shown below:

```
<nat64-prefixes>
  <nat64-prefix>
    2001:db8:122::/48
  </nat64-prefix>
  <destination-ipv4-prefix>
    <ipv4-prefix>
      198.51.100.0/24
    </ipv4-prefix>
  </destination-ipv4-prefix>
</nat64-prefixes>
```

A.5. Stateless IP/ICMP Translation (SIIT)

Let's consider the example of a stateless translator that is configured with 2001:db8:100::/40 to perform IPv6 address synthesis [RFC6052]. Similar to the NAT64 case, the XML snippet to configure the NAT64 prefix in such case is depicted below:


```
<nat64-prefixes>
  <nat64-prefix>
    2001:db8:100::/40
  </nat64-prefix>
</nat64-prefixes>
```

When the translator receives an IPv6 packet, for example, with a source address (2001:db8:1c0:2:21::) and destination address (2001:db8:1c6:3364:2::), it extracts embedded IPv4 addresses following RFC6052 rules with 2001:db8:100::/40 as the NSP:

- o 192.0.2.33 is extracted from 2001:db8:1c0:2:21::
- o 198.51.100.2 is extracted from 2001:db8:1c6:3364:2::

The translator transforms the IPv6 header into an IPv4 header using the IP/ICMP Translation Algorithm [RFC7915]. The IPv4 packets will include 192.0.2.33 as the source address and 198.51.100.2 as the destination address.

Also, a NAT64 can be instructed to behave in the stateless mode by providing the following configuration. The same NAT64 prefix is used for constructing both IPv4-translatable IPv6 addresses and IPv4-converted IPv6 addresses (Section 3.3 of [RFC6052]).

```
<nat64-prefixes>
  <nat64-prefix>
    2001:db8:122:300::/56
  </nat64-prefix>
  <stateless-enable>
    true
  </stateless-enable>
</nat64-prefixes>
```

A.6. Explicit Address Mappings for Stateless IP/ICMP Translation (EAM SIIT)

As specified in [RFC7757], an EAM consists of an IPv4 prefix and an IPv6 prefix. Let's consider the set of EAM examples in Table 8.

IPv4 Prefix	IPv6 Prefix
192.0.2.1	2001:db8:aaaa::
192.0.2.2/32	2001:db8:bbbb::b/128
192.0.2.16/28	2001:db8:cccc::/124
192.0.2.128/26	2001:db8:dddd::/64
192.0.2.192/29	2001:db8:eeee:8::/62
192.0.2.224/31	64:ff9b::/127

Table 8: EAM Examples (RFC7757)

The following XML excerpt illustrates how these EAMs can be configured using the YANG NAT module:

```
<eam>
  <ipv4-prefix>
    192.0.2.1/32
  </ipv4-prefix>
  <ipv6-prefix>
    2001:db8:aaaa::/128
  </ipv6-prefix>
</eam>
<eam>
  <ipv4-prefix>
    192.0.2.2/32
  </ipv4-prefix>
  <ipv6-prefix>
    2001:db8:bbbb::b/128
  </ipv6-prefix>
</eam>
<eam>
  <ipv4-prefix>
    192.0.2.16/28
  </ipv4-prefix>
  <ipv6-prefix>
    2001:db8:cccc::/124
  </ipv6-prefix>
</eam>
<eam>
  <ipv4-prefix>
    192.0.2.128/26
  </ipv4-prefix>
  <ipv6-prefix>
    2001:db8:dddd::/64
  </ipv6-prefix>
</eam>
<eam>
  <ipv4-prefix>
    192.0.2.192/29
  </ipv4-prefix>
  <ipv6-prefix>
    2001:db8:eeee:8::/62
  </ipv6-prefix>
</eam>
<eam>
  <ipv4-prefix>
    192.0.2.224/31
  </ipv4-prefix>
  <ipv6-prefix>
    64:ff9b::/127
  </ipv6-prefix>
</eam>
```

EAMs may be enabled jointly with stateful NAT64. This example shows a NAT64 function that supports static mappings:

```
<capabilities
  <nat-flavor>
    nat64
  </nat-flavor>
  <static-mapping-support>
    true
  </static-mapping-support>
  <port-randomization-support>
    true
  </port-randomization-support>
  <port-range-allocation-support>
    true
  </port-range-allocation-support>
  <port-preservation-suport>
    true
  </port-preservation-suport>
  <address-roundrobin-support>
    true
  </address-roundrobin-support>
  <paired-address-pooling-support>
    true
  </paired-address-pooling-support>
  <endpoint-independent-mapping-support>
    true
  </endpoint-independent-mapping-support>
  <endpoint-independent-filtering-support>
    true
  </endpoint-independent-filtering-support>
</capabilities>
```

A.7. Static Mappings with Port Ranges

The following example shows a static mapping that instructs a NAT to translate packets issued from 192.0.2.1 and with source ports in the 100-500 range to 198.51.100.1:1100-1500.

```
<mapping-entry>
  <index>1</index>
  <type>
    static
  </type>
  <transport-protocol>
    6
  </transport-protocol>
  <internal-src-address>
    192.0.2.1/32
  </internal-src-address>
  <internal-src-port>
    <start-port-number>
      100
    </start-port-number>
    <end-port-number>
      500
    </end-port-number>
  </internal-dst-port>
  <external-src-address>
    198.51.100.1/32
  </external-src-address>
  <external-src-port>
    <start-port-number>
      1100
    </start-port-number>
    <end-port-number>
      1500
    </end-port-number>
  </external-src-port>
  ...
</mapping-entry>
```

A.8. Static Mappings with IP Prefixes

The following example shows a static mapping that instructs a NAT to translate TCP packets issued from 192.0.2.0/24 to 198.51.100.0/24.

```
<mapping-entry>
  <index>1</index>
  <type>
    static
  </type>
  <transport-protocol>
    6
  </transport-protocol>
  <internal-src-address>
    192.0.2.0/24
  </internal-src-address>
  <external-src-address>
    198.51.100.0/24
  </external-src-address>
  ...
</mapping-entry>
```

A.9. Destination NAT

The following XML snippet shows an example of a destination NAT that is instructed to translate all packets having 192.0.2.1 as a destination IP address to 198.51.100.1.

```
<dst-ip-address-pool>
  <pool-id>1</pool-id>
  <dst-in-ip-pool>
    192.0.2.1/32
  </dst-in-ip-pool>
  <dst-out-ip-pool>
    198.51.100.1/32
  </dst-out-ip-pool>
</dst-ip-address-pool>
```

In order to instruct a NAT to translate TCP packets destined to '192.0.2.1:80' to '198.51.100.1:8080', the following XML snippet shows the static mapping configured on the NAT:

```
<mapping-entry>
  <index>1568</index>
  <type>
    static
  </type>
  <transport-protocol>
    6
  </transport-protocol>
  <internal-dst-address>
    192.0.2.1/32
  </internal-dst-address>
  <internal-dst-port>
    <start-port-number>
      80
    </start-port-number>
  </internal-dst-port>
  <external-dst-address>
    198.51.100.1/32
  </external-dst-address>
  <external-dst-port>
    <start-port-number>
      8080
    </start-port-number>
  </external-dst-port>
</mapping-entry>
```

In order to instruct a NAT to translate TCP packets destined to '192.0.2.1:80' (http traffic) to 198.51.100.1 and '192.0.2.1:22' (ssh traffic) to 198.51.100.2, the following XML snippet shows the static mappings configured on the NAT:

```
<mapping-entry>
  <index>123</index>
  <type>
    static
  </type>
  <transport-protocol>
    6
  </transport-protocol>
  <internal-dst-address>
    192.0.2.1/32
  </internal-dst-address>
  <internal-dst-port>
    <start-port-number>
      80
    </start-port-number>
  </internal-dst-port>
  <external-dst-address>
    198.51.100.1/32
  </external-dst-address>
  ...
</mapping-entry>
<mapping-entry>
  <index>1236</index>
  <type>
    static
  </type>
  <transport-protocol>
    6
  </transport-protocol>
  <internal-dst-address>
    192.0.2.1/32
  </internal-dst-address>
  <internal-dst-port>
    <start-port-number>
      22
    </start-port-number>
  </internal-dst-port>
  <external-dst-address>
    198.51.100.2/32
  </external-dst-address>
  ...
</mapping-entry>
```

The NAT may also be instructed to proceed with both source and destination NAT. To do so, in addition to the above sample to configure destination NAT, the NAT may be provided, for example with a pool of external IP addresses (198.51.100.0/24) to use for source

address translation. An example of the corresponding XML snippet is provided hereafter:

```
<external-ip-address-pool>
  <pool-id>1</pool-id>
  <external-ip-pool>
    198.51.100.0/24
  </external-ip-pool>
</external-ip-address-pool>
```

Instead of providing an external IP address to share, the NAT may be configured with static mapping entries that modify the internal IP address and/or port number.

A.10. Customer-side Translator (CLAT)

The following XML snippet shows the example of a CLAT that is configured with 2001:db8:1234::/96 as PLAT-side IPv6 prefix and 2001:db8:aaaa::/96 as CLAT-side IPv6 prefix. The CLAT is also provided with 192.0.0.1/32 (which is selected from the IPv4 service continuity prefix defined in [RFC7335]).

```
<clat-ipv6-prefixes>
  <ipv6-prefix>
    2001:db8:aaaa::/96
  </ipv6-prefix>
</clat-ipv6-prefixes>
<clat-ipv4-prefixes>
  <ipv4-prefix>
    192.0.0.1/32
  </ipv4-prefix>
</clat-ipv4-prefixes>
<nat64-prefixes>
  <nat64-prefix>
    2001:db8:1234::/96
  </nat64-prefix>
</nat64-prefixes>
```

A.11. IPv6 Network Prefix Translation (NPTv6)

Let's consider the example of an NPTv6 translator that should rewrite packets with the source prefix (fd03:c03a:ecab::/48) with the external prefix (2001:db8:1::/48). The internal interface is "eth0" while the external interface is "eth1" (Figure 2).

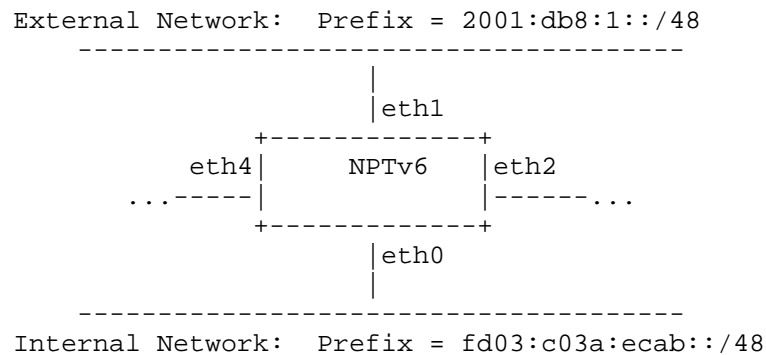


Figure 2: Example of NPTv6

The XML snippet to configure NPTv6 prefixes in such case is depicted below:

```

<nptv6-prefixes>
  <internal-ipv6-prefix>
    fd03:c03a:ecab::/48
  </internal-ipv6-prefix>
  <external-ipv6-prefix>
    2001:db8:1::/48
  </external-ipv6-prefix>
</nptv6-prefixes>
...
<external-realm>
  <external-interface>
    eth1
  </external-interface>
</external-realm>
  
```

Figure 3 shows an example of an NPTv6 translator that interconnects two internal networks (fd03:c03a:ecab::/48 and fda8:d5cb:14f3::/48); each is translated using a dedicated prefix (2001:db8:1::/48 and 2001:db8:6666::/48, respectively).

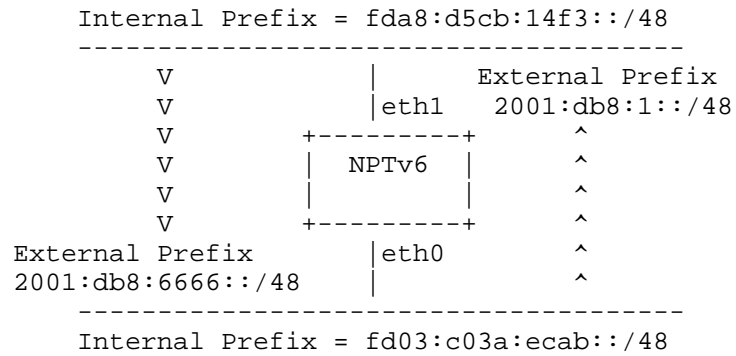


Figure 3: Connecting two Peer Networks

To that aim, the following configuration is provided to the NPTv6 translator:

```
<policy>
  <id>1</id>
  <nptv6-prefixes>
    <internal-ipv6-prefix>
      fd03:c03a:ecab::/48
    </internal-ipv6-prefix>
    <external-ipv6-prefix>
      2001:db8:1::/48
    </external-ipv6-prefix>
  </nptv6-prefixes>
  <external-realm>
    <external-interface>
      eth1
    </external-interface>
  </external-realm>
</policy>
<policy>
  <id>2</id>
  <nptv6-prefixes>
    <internal-ipv6-prefix>
      fda8:d5cb:14f3::/48
    </internal-ipv6-prefix>
    <external-ipv6-prefix>
      2001:db8:6666::/48
    </external-ipv6-prefix>
  </nptv6-prefixes>
  <external-realm>
    <external-interface>
      eth0
    </external-interface>
  </external-realm>
</policy>
```

Authors' Addresses

Mohamed Boucadair (editor)
Orange
Rennes 35000
France

Email: mohamed.boucadair@orange.com

Senthil Sivakumar
Cisco Systems
7100-8 Kit Creek Road
Research Triangle Park, North Carolina 27709
USA

Phone: +1 919 392 5158
Email: ssenthil@cisco.com

Christian Jacquenet
Orange
Rennes 35000
France

Email: christian.jacquenet@orange.com

Suresh Vinapamula
Juniper Networks
1133 Innovation Way
Sunnyvale 94089
USA

Email: sureshk@juniper.net

Qin Wu
Huawei
101 Software Avenue, Yuhua District
Nanjing, Jiangsu 210012
China

Email: bill.wu@huawei.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: September 16, 2018

K. Moriarty, Ed.
Dell EMC
A. Morton, Ed.
AT&T Labs
March 15, 2018

Effects of Pervasive Encryption on Operators
draft-mm-wg-effect-encrypt-25

Abstract

Pervasive Monitoring (PM) attacks on the privacy of Internet users are of serious concern to both the user and the operator communities. RFC7258 discussed the critical need to protect users' privacy when developing IETF specifications and also recognized making networks unmanageable to mitigate PM is not an acceptable outcome; an appropriate balance is needed. This document discusses current security and network operations and management practices that may be impacted by the shift to increased use of encryption to help guide protocol development in support of manageable and secure networks.

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 September 16, 2018.

Copyright Notice

Copyright (c) 2018 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
1.1. Additional Background on Encryption Changes	4
1.2. Examples of Attempts to Preserve Functions	6
2. Network Service Provider Monitoring	7
2.1. Passive Monitoring	8
2.1.1. Traffic Surveys	8
2.1.2. Troubleshooting	8
2.1.3. Traffic Analysis Fingerprinting	11
2.2. Traffic Optimization and Management	12
2.2.1. Load Balancers	12
2.2.2. Differential Treatment based on Deep Packet Inspection (DPI)	14
2.2.3. Network Congestion Management	15
2.2.4. Performance-enhancing Proxies	15
2.2.5. Caching and Content Replication Near the Network Edge	16
2.2.6. Content Compression	17
2.2.7. Service Function Chaining	18
2.3. Content Filtering, Network Access, and Accounting	18
2.3.1. Content Filtering	19
2.3.2. Network Access and Data Usage	20
2.3.3. Application Layer Gateways	21
2.3.4. HTTP Header Insertion	22
3. Encryption in Hosting and Application SP Environments	22
3.1. Management Access Security	22
3.1.1. Customer Access Monitoring	23
3.1.2. SP Content Monitoring of Applications	24
3.2. Hosted Applications	26
3.2.1. Monitoring Managed Applications	26
3.2.2. Mail Service Providers	27
3.3. Data Storage	27
3.3.1. Object-level Encryption	27
3.3.2. Disk Encryption, Data at Rest	28
3.3.3. Cross Data Center Replication Services	29
4. Encryption for Enterprises	29
4.1. Monitoring Practices of the Enterprise	30
4.1.1. Security Monitoring in the Enterprise	30
4.1.2. Application Performance Monitoring in the Enterprise	31
4.1.3. Enterprise Network Diagnostics and Troubleshooting	32
4.2. Techniques for Monitoring Internet Session Traffic	34
5. Security Monitoring for Specific Attack Types	36

5.1.	Mail Abuse and spam	36
5.2.	Denial of Service	37
5.3.	Phishing	37
5.4.	Botnets	38
5.5.	Malware	38
5.6.	Spoofed Source IP Address Protection	39
5.7.	Further work	39
6.	Application-based Flow Information Visible to a Network	39
6.1.	IP Flow Information Export	39
6.2.	TLS Server Name Indication	40
6.3.	Application Layer Protocol Negotiation (ALPN)	41
6.4.	Content Length, BitRate and Pacing	41
7.	Effect of Encryption on Mobile Network Evolution	41
8.	Response to Increased Encryption and Looking Forward	42
9.	Security Considerations	43
10.	IANA Considerations	43
11.	Acknowledgements	43
12.	Informative References	43
	Authors' Addresses	52

1. Introduction

In response to pervasive monitoring revelations and the IETF consensus that Pervasive Monitoring is an Attack [RFC7258], efforts are underway to increase encryption of Internet traffic. Pervasive Monitoring (PM) is of serious concern to users, operators, and application providers. RFC7258 discussed the critical need to protect users' privacy when developing IETF specifications and also recognized that making networks unmanageable to mitigate PM is not an acceptable outcome, but rather that an appropriate balance would emerge over time.

This document describes practices currently used by network operators to manage, operate, and secure their networks and how those practices may be impacted by a shift to increased use of encryption. It provides network operators' perspectives about the motivations and objectives of those practices as well as effects anticipated by operators as use of encryption increases. It is a summary of concerns of the operational community as they transition to managing networks with less visibility. The document does not endorse the use of the practices described herein. Nor does it aim to provide a comprehensive treatment of the effects of current practices, some of which have been considered controversial from a technical or business perspective or contradictory to previous IETF statements (e.g., [RFC1958], [RFC1984], [RFC2804]). The informational documents consider the end to end (e2e) architectural principle to be a guiding principle for the development of Internet protocols [RFC2775] [RFC3724] [RFC7754].

This document aims to help IETF participants understand network operators' perspectives about the impact of pervasive encryption, both opportunistic and strong end-to-end encryption, on operational practices. The goal is to help inform future protocol development to ensure that operational impact is part of the conversation. Perhaps, new methods could be developed to accomplish some of the goals of current practices despite changes in the extent to which cleartext will be available to network operators (including methods that rely on network endpoints where applicable). Discussion of current practices and the potential future changes is provided as a prerequisite to potential future cross-industry and cross-layer work to support the ongoing evolution towards a functional Internet with pervasive encryption.

Traditional network management, planning, security operations, and performance optimization have been developed in an Internet where a large majority of data traffic flows without encryption. While unencrypted traffic has made information that aids operations and troubleshooting at all layers accessible, it has also made pervasive monitoring by unseen parties possible. With broad support and increased awareness of the need to consider privacy in all aspects across the Internet, it is important to catalog existing management, operational, and security practices that have depended upon the availability of cleartext to function and to explore if critical operational practices can be met by less invasive means.

This document refers to several different forms of service providers, distinguished with adjectives. For example, network service providers (or network operators) provide IP-packet transport primarily, though they may bundle other services with packet transport. Alternatively, application service providers primarily offer systems that participate as an end-point in communications with the application user, and hosting service providers lease computing, storage, and communications systems in datacenters. In practice, many companies perform two or more service provider roles, but may be historically associated with one.

This document includes a sampling of current practices and does not attempt to describe every nuance. Some sections cover technologies used over a broad spectrum of devices and use cases.

1.1. Additional Background on Encryption Changes

Pervasive encryption in this document refers to all types of session encryption including Transport Layer Security (TLS), IP security (IPsec), TCPcrypt [TCPcrypt], QUIC [QUIC] and others that are increasing in deployment usage. It is well understood that session encryption helps to prevent both passive and active attacks on

transport protocols; more on pervasive monitoring can be found in Confidentiality in the Face of Pervasive Surveillance: A Threat Model and Problem Statement [RFC7624]. Active attacks have long been a motivation for increased encryption, and preventing pervasive monitoring became a focus just a few years ago. As such, the Internet Architecture Board (IAB) released a statement advocating for increased use of encryption in November 2014. Perspectives on encryption paradigms have shifted over time to incorporate ease of deployment as a high priority, and balance that against providing the maximum possible level of security regardless of deployment considerations.

One such shift is documented in "Opportunistic Security" (OS) [RFC7435], which suggests that when use of authenticated encryption is not possible, cleartext sessions should be upgraded to unauthenticated session encryption, rather than no encryption. OS encourages upgrading from cleartext, but cannot require or guarantee such upgrades. Once OS is used, it allows for an evolution to authenticated encryption. These efforts are necessary to improve end user's expectation of privacy, making pervasive monitoring cost prohibitive. With OS in use, active attacks are still possible on unauthenticated sessions. OS has been implemented as NULL Authentication with IPsec [RFC7619] and there are a number of infrastructure use cases such as server to server encryption where this mode is deployed. While OS is helpful in reducing pervasive monitoring by increasing the cost to monitor, it is recognized that risk profiles for some applications require authenticated and secure session encryption as well to prevent active attacks. IPsec, and other session encryption protocols, with authentication has many useful applications and usage has increased for infrastructure applications such as for virtual private networks between data centers. OS as well as other protocol developments, like the Automated Certificate Management Environment (ACME), have increased the usage of session encryption on the Internet.

Risk profiles vary and so do the types of session encryption deployed. To understand the scope of changes in visibility a few examples are highlighted. Work continues to improve the implementation, development and configuration of TLS and DTLS sessions to prevent active attacks used to monitor or intercept session data. The changes from TLS 1.2 to 1.3 enhance the security of TLS, while hiding more of the session negotiation and providing less visibility on the wire. The Using TLS in Applications (UTA) working group has been publishing documentation to improve the security of TLS and DTLS sessions. They have documented the known attack vectors in [RFC7457] and have documented Best Practices for TLS and DTLS in [RFC7525] and have other documents in the queue. The

recommendations from these documents were built upon for TLS 1.3 to provide a more inherently secure end-to-end protocol.

In addition to encrypted web site access (HTTP over TLS), there are other well-deployed application level transport encryption efforts such as mail transfer agent (MTA)-to-MTA session encryption transport for email (SMTP over TLS) and gateway-to-gateway for instant messaging (Extensible Messaging and Presence Protocol (XMPP) over TLS). Although this does provide protection from transport layer attacks, the servers could be a point of vulnerability if user-to-user encryption is not provided for these messaging protocols. User-to-user content encryption schemes, such as S/MIME and PGP for email and Off-the-Record (OTR) encryption for XMPP are used by those interested to protect their data as it crosses intermediary servers, preventing transport layer attacks by providing an end-to-end solution. User-to-user schemes are under review and additional options will emerge to ease the configuration requirements, making this type of option more accessible to non-technical users interested in protecting their privacy.

Increased use of encryption, either opportunistic or authenticated, at the transport, network or application layer, impacts how networks are operated, managed, and secured. In some cases, new methods to operate, manage, and secure networks will evolve in response. In other cases, currently available capabilities for monitoring or troubleshooting networks could become unavailable. This document lists a collection of functions currently employed by network operators that may be impacted by the shift to increased use of encryption. This draft does not attempt to specify responses or solutions to these impacts, but rather documents the current state.

1.2. Examples of Attempts to Preserve Functions

Following the Snowden [Snowden] revelations, application service providers responded by encrypting traffic between their data centers (IPsec) to prevent passive monitoring from taking place unbeknownst to them (Yahoo, Google, etc.). Infrastructure traffic carried over the public Internet has been encrypted for some time, this change for universal encryption was specific to their private backbones. Large mail service providers also began to encrypt session transport (TLS) to hosted mail services. This and other increases in the use of encryption had the immediate effect of providing confidentiality and integrity for protected data, but created a problem for some network management functions. Operators could no longer gain access to some session streams resulting in actions by several to regain their operational practices that previously depended on cleartext data sessions.

The EFF reported [EFF2014] several network service providers using a downgrade attack to prevent the use of SMTP over TLS by breaking STARTTLS (section 3.2 of [RFC7525]), essentially preventing the negotiation process resulting in fallback to the use of clear text. There have already been documented cases of service providers preventing STARTTLS to prevent session encryption negotiation on some session to inject a super cookie to enable tracking of users for advertisers, also considered an attack. These serve as examples of undesirable behavior that could be prevented through upfront discussions in protocol work for operators and protocol designers to understand the implications of such actions. In other cases, some service providers and enterprises have relied on middleboxes having access to clear text for the purposes of load balancing, monitoring for attack traffic, meeting regulatory requirements, or for other purposes. The implications for enterprises, who own the data on their networks or have explicit agreements that permit monitoring of user traffic is very different from service providers who may be accessing content in a way that violates privacy considerations. Additionally, service provider equipment is designed for accessing only the headers exposed for the data-link, network, and transport layers. Delving deeper into packets is possible, but there is typically a high degree of accuracy from the header information and packet sizes when limited to header information from these three layers. Service providers also have the option of adding routing overlay protocols to traffic. These middlebox implementations, whether performing functions considered legitimate by the IETF or not, have been impacted by increases in encrypted traffic. Only methods keeping with the goal of balancing network management and PM mitigation in [RFC7258] should be considered in solution work resulting from this document.

It is well known that national surveillance programs monitor traffic [JNSLP] [RFC2804] [RFC7258] monitor for criminal activities. Governments vary on their balance between monitoring versus the protection of user privacy, data, and assets. Those that favor unencrypted access to data ignore the real need to protect users' identity, financial transactions and intellectual property, which requires security and encryption to prevent crime. A clear understanding of technology, encryption, and monitoring goals will aid in the development of solutions as work continues towards finding an appropriate balance allowing for management while protecting users privacy with strong encryption solutions.

2. Network Service Provider Monitoring

Network Service Providers (SP) for this definition include the backbone Internet Service providers as well as those providing

infrastructure at scale for core Internet use (hosted infrastructure and services such as email).

Network service providers use various techniques to operate, manage, and secure their networks. The following subsections detail the purpose of several techniques and which protocol fields are used to accomplish each task. In response to increased encryption of these fields, some network service providers may be tempted to undertake undesirable security practices in order to gain access to the fields in unencrypted data flows. To avoid this situation, new methods could be developed to accomplish the same goals without service providers having the ability to see session data.

2.1. Passive Monitoring

2.1.1. Traffic Surveys

Internet traffic surveys are useful in many pursuits, such as input for Center for Applied Internet Data Analysis (CAIDA) studies [CAIDA], network planning and optimization. Tracking the trends in Internet traffic growth, from earlier peer-to-peer communication to the extensive adoption of unicast video streaming applications, has relied on a view of traffic composition with a particular level of assumed accuracy, based on access to cleartext by those conducting the surveys.

Passive monitoring makes inferences about observed traffic using the maximal information available, and is subject to inaccuracies stemming from incomplete sampling (of packets in a stream) or loss due to monitoring system overload. When encryption conceals more layers in each packet, reliance on pattern inferences and other heuristics grows, and accuracy suffers. For example, the traffic patterns between server and browser are dependent on browser supplier and version, even when the sessions use the same server application (e.g., web e-mail access). It remains to be seen whether more complex inferences can be mastered to produce the same monitoring accuracy.

2.1.2. Troubleshooting

Network operators use protocol-dissecting analyzers when responding to customer problems, to identify the presence of attack traffic, and to identify root causes of the problem such as misconfiguration. In limited cases, packet captures may also be used when a customer approves of access to their packets or provides packet captures close to the endpoint. The protocol dissection is generally limited to supporting protocols (e.g., DNS, DHCP), network and transport (e.g., IP, TCP), and some higher layer protocols (e.g., RTP, RTCP).

Troubleshooting will move closer to the endpoint with increased encryption and adjustments in practices to effectively troubleshoot using a 5-tuple may require education. Packet loss investigations, and those where access is limited to a 2-tuple (IPsec tunnel mode), rely on network and transport layer headers taken at the endpoint. In this case, captures on intermediate nodes are not reliable as there are far too many cases of aggregate interfaces and alternate paths in service provider networks.

Network operators are often the first ones called upon to investigate application problems (e.g., "my HD video is choppy"), to first rule out network and network services as a cause for the underlying issue. When diagnosing a customer problem, the starting point may be a particular application that isn't working. The ability to identify the problem application's traffic is important and packet capture provided from the customer close to the edge may be used for this purpose; IP address filtering is not useful for applications using content delivery networks (CDNs) or cloud providers. After identifying the traffic, an operator may analyze the traffic characteristics and routing of the traffic. This diagnostic step is important to help determine the root cause before exploring if the issue is directly with the application.

For example, by investigating packet loss (from TCP sequence and acknowledgement numbers), round-trip-time (from TCP timestamp options or application-layer transactions, e.g., DNS or HTTP response time), TCP receive-window size, packet corruption (from checksum verification), inefficient fragmentation, or application-layer problems, the operator can narrow the problem to a portion of the network, server overload, client or server misconfiguration, etc. Network operators may also be able to identify the presence of attack traffic as not conforming to the application the user claims to be using. In many instances, the exposed packet header is sufficient for this type of troubleshooting.

One way of quickly excluding the network as the bottleneck during troubleshooting is to check whether the speed is limited by the endpoints. For example, the connection speed might instead be limited by suboptimal TCP options, the sender's congestion window, the sender temporarily running out of data to send, the sender waiting for the receiver to send another request, or the receiver closing the receive window. All this information can be derived from the cleartext TCP header.

Packet captures and protocol-dissecting analyzers have been important tools. Automated monitoring has also been used to proactively identify poor network conditions, leading to maintenance and network upgrades before user experience declines. For example, findings of

loss and jitter in VoIP traffic can be a predictor of future customer dissatisfaction (supported by metadata from the RTP/RTCP protocol) [RFC3550], or increases in DNS response time can generally make interactive web browsing appear sluggish. But to detect such problems, the application or service stream must first be distinguished from others.

When increased encryption is used, operators lose a source of data that may be used to debug user issues. For example, IPsec obscures TCP and RTP header information, while TLS and SRTP do not. Because of this, application server operators using increased encryption might be called upon more frequently to assist with debugging and troubleshooting, and thus may want to consider what tools can be put in the hands of their clients or network operators.

Further, the performance of some services can be more efficiently managed and repaired when information on user transactions is available to the service provider. It may be possible to continue transaction monitoring activities without clear text access to the application layers of interest, but inaccuracy will increase and efficiency of repair activities will decrease. For example, an application protocol error or failure would be opaque to network troubleshooters when transport encryption is applied, making root cause location more difficult and therefore increasing the time-to-repair. Repair time directly reduces the availability of the service, and most network operators have made availability a key metric in their Service Level Agreements and/or subscription rebates. Also, there may be more cases of user communication failures when the additional encryption processes are introduced (e.g., key management at large scale), leading to more customer service contacts and (at the same time) less information available to network operations repair teams.

In mobile networks, knowledge about TCP's stream transfer progress (by observing ACKs, retransmissions, packet drops, and the Sector Utilization Level etc.) is further used to measure the performance of Network Segments (Sector, eNodeB (eNB) etc.). This information is used as key performance indicators (KPIs) and for the estimation of user/service key quality indicators at network edges for circuit emulation (CEM) as well as input for mitigation methods. If the make-up of active services per user and per sector are not visible to a server that provides Internet Access Point Names (APN), it cannot perform mitigation functions based on network segment view.

It is important to note that the push for encryption by application providers has been motivated by the application of the described techniques. Although network operators have noted performance improvements with network-based optimization or enhancement of user

traffic (otherwise, deployment would not have occurred), application providers have likewise noted some degraded performance and/or user experience, and such cases may result in additional operator troubleshooting. Further, encrypted application streams might avoid outdated optimization or enhancement techniques, where they exist.

A gap exists for vendors where built-in diagnostics and serviceability are not adequate to provide detailed logging and debugging capabilities that, when possible, could be accessed with cleartext network parameters. In addition to traditional logging and debugging methods, packet tracing and inspection along the service path provides operators the visibility to continue to diagnose problems reported both internally and by their customers. Logging of service path upon exit for routing overlay protocols will assist with policy management and troubleshooting capabilities for traffic flows on encrypted networks. Protocol trace logging and protocol data unit (PDU) logging should also be considered to improve visibility to monitor and troubleshoot application level traffic. Additional work on this gap would assist network operators to better troubleshoot and manage networks with increasing amounts of encrypted traffic.

2.1.1.3. Traffic Analysis Fingerprinting

Fingerprinting is used in traffic analysis and monitoring to identify traffic streams that match certain patterns. This technique can be used with both clear text or encrypted sessions. Some Distributed Denial of Service (DDoS) prevention techniques at the network provider level rely on the ability to fingerprint traffic in order to mitigate the effect of this type of attack. Thus, fingerprinting may be an aspect of an attack or part of attack countermeasures.

A common, early trigger for DDoS mitigation includes observing uncharacteristic traffic volumes or sources; congestion; or degradation of a given network or service. One approach to mitigate such an attack involves distinguishing attacker traffic from legitimate user traffic. The ability to examine layers and payloads above transport provides an increased range of filtering opportunities at each layer in the clear. If fewer layers are in the clear, this means that there are reduced filtering opportunities available to mitigate attacks. However, fingerprinting is still possible.

Passive monitoring of network traffic can lead to invasion of privacy by external actors at the endpoints of the monitored traffic. Encryption of traffic end-to-end is one method to obfuscate some of the potentially identifying information. For example, browser fingerprints are comprised of many characteristics, including User Agent, HTTP Accept headers, browser plug-in details, screen size and

color details, system fonts and time zone. A monitoring system could easily identify a specific browser, and by correlating other information, identify a specific user.

2.2. Traffic Optimization and Management

2.2.1. Load Balancers

A standalone load balancer is a function one can take off the shelf, place in front of a pool of servers, configure appropriately, and it will balance the traffic load among servers in the pool. This is a typical setup for load balancers. Standalone load balancers rely on the plainly observable information in the packets they are forwarding and rely on industry-accepted standards in interpreting the plainly observable information. Typically, this is a 5-tuple of the connection. This type of configuration terminates TLS sessions at the load balancer, making it the end point instead of the server. Standalone load balancers are considered middleboxes, but are an integral part of server infrastructure that scales.

In contrast, an integrated load balancer is developed to be an integral part of the service provided by the server pool behind that load balancer. These load balancers can communicate state with their pool of servers to better route flows to the appropriate servers. They rely on non-standard system-specific information and operational knowledge shared between the load balancer and its servers.

Both standalone and integrated load balancers can be deployed in pools for redundancy and load sharing. For high availability, it is important that when packets belonging to a flow start to arrive at a different load balancer in the load balancer pool, the packets continue to be forwarded to the original server in the server pool. The importance of this requirement increases as the chances of such load balancer change event increases.

Mobile operators deploy integrated load balancers to assist with maintaining connection state as devices migrate. With the proliferation of mobile connected devices, there is an acute need for connection-oriented protocols that maintain connections after a network migration by an endpoint. This connection persistence provides an additional challenge for multi-homed anycast-based services typically employed by large content owners and Content Distribution Networks (CDNs). The challenge is that a migration to a different network in the middle of the connection greatly increases the chances of the packets routed to a different anycast point-of-presence (POP) due to the new network's different connectivity and Internet peering arrangements. The load balancer in the new POP, potentially thousands of miles away, will not have information about

the new flow and would not be able to route it back to the original POP.

To help with the endpoint network migration challenges, anycast service operations are likely to employ integrated load balancers that, in cooperation with their pool servers, are able to ensure that client-to-server packets contain some additional identification in plainly-observable parts of the packets (in addition to the 5-tuple). As noted in Section 2 of [RFC7258], careful consideration in protocol design to mitigate PM is important, while ensuring manageability of the network.

An area for further research includes end-to-end solutions that would provide a simpler architecture and may solve the issue with CDN anycast. In this case, connections would be migrated to a CDN unicast address.

Current protocols, such as TCP, allow the development of stateless integrated load balancers by availing such load balancers of additional plain text information in client-to-server packets. In case of TCP, such information can be encoded by having server-generated sequence numbers (that are ACK'd by the client), segment values, lengths of the packet sent, etc. The use of some of these mechanisms for load balancing negates some of the security assumptions associated with those primitives (e.g., that an off-path attacker guessing valid sequence numbers for a flow is hard). Another possibility is a dedicated mechanism for storing load balancer state, such as QUIC's proposed connection ID to provide visibility to the load balancer. An identifier could be used for tracking purposes, but this may provide an option that is an improvement from bolting it on to an unrelated transport signal. This method allows for tight control by one of the endpoints and can be rotated to avoid roving client linkability: in other words, being a specific, separate signal, it can be governed in a way that is finely targeted at that specific use-case.

Some integrated load balancers have the ability to use additional plainly observable information even for today's protocols that are not network migration tolerant. This additional information allows for improved availability and scalability of the load balancing operation. For example, BGP reconvergence can cause a flow to switch anycast POPs even without a network change by any endpoint. Additionally, a system that is able to encode the identity of the pool server in plain text information available in each incoming packet is able to provide stateless load balancing. This ability confers great reliability and scalability advantages even if the flow remains in a single POP, because the load balancing system is not required to keep state of each flow. Even more importantly, there's

no requirement to continuously synchronize such state among the pool of load balancers. An integrated load balancer repurposing limited existing bits in transport flow state must maintain and synchronize per-flow state occasionally: using the sequence number as a cookie only works for so long given that there aren't that many bits available to divide across a pool of machines.

Mobile operators apply Self Organizing Networks (3GPP SON) for intelligent workflows such as content-aware MLB (Mobility Load Balancing). Where network load balancers have been configured to route according to application-layer semantics, an encrypted payload is effectively invisible. This has resulted in practices of intercepting TLS in front of load balancers to regain that visibility, but at a cost to security and privacy.

In future Network Function Virtualization (NFV) architectures, load balancing functions are likely to be more prevalent (deployed at locations throughout operators' networks). NFV environments will require some type of identifier (IPv6 flow identifiers, the proposed QUIC connection ID, etc.) for managing traffic using encrypted tunnels. The shift to increased encryption will have an impact to visibility of flow information and will require adjustments to perform similar load balancing functions within an NFV.

2.2.2. Differential Treatment based on Deep Packet Inspection (DPI)

Data transfer capacity resources in cellular radio networks tend to be more constrained than in fixed networks. This is a result of variance in radio signal strength as a user moves around a cell, the rapid ingress and egress of connections as users hand off between adjacent cells, and temporary congestion at a cell. Mobile networks alleviate this by queuing traffic according to its required bandwidth and acceptable latency: for example, a user is unlikely to notice a 20ms delay when receiving a simple Web page or email, or an instant message response, but will very likely notice a re-buffering pause in a video playback or a VoIP call de-jitter buffer. Ideally, the scheduler manages the queue so that each user has an acceptable experience as conditions vary, but inferences of the traffic type have been used to make bearer assignments and set scheduler priority.

Deep Packet Inspection (DPI) allows identification of applications based on payload signatures, in contrast to trusting well-known port numbers. Application and transport layer encryption make the traffic type estimation more complex and less accurate, and therefore it may not be effectual to use this information as input for queue management. With the use of WebSockets [RFC6455], for example, many forms of communications (from isochronous/real-time to bulk/elastic file transfer) will take place over HTTP port 80 or port 443, so only

the messages and higher-layer data will make application differentiation possible. If the monitoring system sees only "HTTP port 443", it cannot distinguish application streams that would benefit from priority queueing from others that would not.

Mobile networks especially rely on content/application based prioritization of Over-the-Top (OTT) services - each application-type or service has different delay/loss/throughput expectations, and each type of stream will be unknown to an edge device if encrypted; this impedes dynamic-QoS adaptation. An alternate way to achieve encrypted application separation is possible when the User Equipment (UE) requests a dedicated bearer for the specific application stream (known by the UE), using a mechanism such as the one described in Section 6.5 of 3GPP TS 24.301 [TS3GPP]. The UE's request includes the Quality Class Indicator (QCI) appropriate for each application, based on their different delay/loss/throughput expectations. However, UE requests for dedicated bearers and QCI may not be supported at the subscriber's service level, or in all mobile networks.

These effects and potential alternative solutions have been discussed at the accord BoF [ACCORD] at IETF95.

This section does not consider traffic discrimination by service providers related to NetNeutrality, where traffic may be favored according to the service provider preference as opposed to the user's preference. These use cases are considered out-of-scope for this document as controversial practices.

2.2.3. Network Congestion Management

For User Plane Congestion Management (3GPP UPCON) [UPCON], the ability to understand content and manage networks during periods of congestion is the focus of this 3GPP work item. Mitigating techniques such as deferred download, off-peak acceleration, and outbound roamers are a few examples of the areas explored in the associated 3GPP documents. The documents describe the issues, the data utilized in managing congestion, and make policy recommendations.

2.2.4. Performance-enhancing Proxies

Performance-enhancing TCP proxies may perform local retransmission at the network edge; this also applies to mobile networks. In TCP, duplicated ACKs are detected and potentially concealed when the proxy retransmits a segment that was lost on the mobile link without involvement of the far end (see section 2.1.1 of [RFC3135] and section 3.5 of [I-D.dolson-plus-middlebox-benefits]).

Operators report that this optimization at network edges improves real-time transmission over long delay Internet paths or networks with large capacity-variation (such as mobile/cellular networks). However, such optimizations can also cause problems with performance, for example if the characteristics of some packet streams begin to vary significantly from those considered in the proxy design.

In general some operators have stated that performance-enhancing proxies have a lower Round-Trip Time (RTT) to the client and therefore determine the responsiveness of flow control. A lower RTT makes the flow control loop more responsive to changes in the mobile network conditions and enables faster adaptation in a delay and capacity varying network due to user mobility.

Further, some use service-provider-operated proxies to reduce the control delay between the sender and a receiver on a mobile network where resources are limited. The RTT determines how quickly a user's attempt to cancel a video is recognized and therefore how quickly the traffic is stopped, thus keeping un-wanted video packets from entering the radio scheduler queue. If impacted by encryption, performance enhancing proxies could make use of routing overlay protocols to accomplish the same task, but this results in additional overhead.

An application-type-aware network edge (middlebox) can further control pacing, limit simultaneous HD videos, or prioritize active videos against new videos, etc. Services at this more granular level are limited with the use of encryption.

Performance enhancing proxies are primarily used on long delay links (satellite) with access to the TCP header to provide an early ACK and make the long delay link of the path seem shorter. With some specific forms of flow control, TCP can be more efficient than alternatives such as proxies. The editors cannot cite research on this point specific to the performance enhancing proxies described, but agree this area could be explored to determine if flow-control modifications could preserve the end-to-end performance on long delay paths session where the TCP header is exposed.

2.2.5. Caching and Content Replication Near the Network Edge

The features and efficiency of some Internet services can be augmented through analysis of user flows and the applications they provide. For example, network caching of popular content at a location close to the requesting user can improve delivery efficiency (both in terms of lower request response times and reduced use of International Internet links when content is remotely located), and the service provider through an authorized agreement acting on their

behalf use DPI in combination with content distribution networks to determine if they can intervene effectively. Encryption of packet contents at a given protocol layer usually makes DPI processing of that layer and higher layers impossible. That being said, it should be noted that some content providers prevent caching to control content delivery through the use of encrypted end-to-end sessions. CDNs vary in their deployment options of end-to-end encryption. The business risk of losing control of content is a motivation outside of privacy and pervasive monitoring that are driving end-to-end encryption for these content providers.

It should be noted that caching was first supported in [RFC1945] and continued in the recent update of "Hypertext Transfer Protocol (HTTP/1.1): Caching" in [RFC7234]. Some operators also operate transparent caches which neither the user nor the origin opt-in. The use of these caches is controversial within IETF and is generally precluded by the use of HTTPS.

Content replication in caches (for example live video, Digital Rights Management (DRM) protected content) is used to most efficiently utilize the available limited bandwidth and thereby maximize the user's Quality of Experience (QoE). Especially in mobile networks, duplicating every stream through the transit network increases backhaul cost for live TV. The Enhanced Multimedia Broadcast/Multicast Services (3GPP eMBMS) utilizes trusted edge proxies to facilitate delivering the same stream to different users, using either unicast or multicast depending on channel conditions to the user. There are on-going efforts to support multicast inside carrier networks while preserving end-to-end security: Automatic Multicast Tunneling (AMT), for instance, allows CDNs to deliver a single (potentially encrypted) copy of a live stream to a carrier network over the public internet and for the carrier to then distribute that live stream as efficiently as possible within its own network using multicast.

Alternate approaches are in the early phase of being explored to allow caching of encrypted content. These solutions require cooperation from content owners and fall outside the scope of what is covered in this document. Content delegation allows for replication with possible benefits, but any form of delegation has the potential to affect the expectation of client-server confidentiality.

2.2.6. Content Compression

In addition to caching, various applications exist to provide data compression in order to conserve the life of the user's mobile data plan or make delivery over the mobile link more efficient. The compression proxy access can be built into a specific user level

application, such as a browser, or it can be available to all applications using a system level application. The primary method is for the mobile application to connect to a centralized server as a transparent proxy (user does not opt-in), with the data channel between the client application and the server using compression to minimize bandwidth utilization. The effectiveness of such systems depends on the server having access to unencrypted data flows.

Aggregated data stream content compression that spans objects and data sources that can be treated as part of a unified compression scheme (e.g., through the use of a shared segment store) is often effective at providing data offload when there is a network element close to the receiver that has access to see all the content.

2.2.7. Service Function Chaining

Service Function Chaining (SFC) has been defined in RFC7665 [RFC7665] and RFC8300 [RFC8300]. As discussed in RFC7498 [RFC7498], common SFC deployments may use classifiers to direct traffic into VLANs instead of using NSH, as defined in RFC8300 [RFC8300]. As described in RFC7665 [RFC7665], the ordered steering of traffic to support specific optimizations depends upon the ability of a Classifier to determine the microflows. RFC2474 [RFC2474] defines "Microflow: a single instance of an application-to-application flow of packets which is identified by source address, destination address, protocol id, and source port, destination port (where applicable)." SFC currently depends upon a classifier to at least identify the microflow. As the classifier's visibility is reduced from a 5-tuple to a 2-tuple, or if information above the transport layer becomes inaccessible, then the SFC Classifier is not able to perform its job and the service functions of the path may be adversely affected.

There are also mechanisms provided to protect security and privacy. In the SFC case, the layer below a network service header can be protected with session encryption. A goal is protecting end-user data, while retaining the intended functions of RFC7665 [RFC7665] at the same time.

2.3. Content Filtering, Network Access, and Accounting

Mobile Networks and many ISPs operate under the regulations of their licensing government authority. These regulations include Lawful Intercept, adherence to Codes of Practice on content filtering, and application of court order filters. Such regulations assume network access to provide content filtering and accounting, as discussed below. As previously stated, the intent of this document is to document existing practices; the development of IETF protocols follows the guiding principles of [RFC1984] and [RFC2804] and

explicitly do not support tools and methods that could be used for wiretapping and censorship.

2.3.1. Content Filtering

There are numerous reasons why service providers might block content: to comply with requests from law enforcement or regulatory authorities, to effectuate parental controls, to enforce content-based billing, or for other reasons, possibly considered inappropriate by some. See RFC7754 [RFC7754] for a survey of Internet filtering techniques and motivations and the IAB consensus on those mechanisms. This section is intended to document a selection of current content blocking practices by operators and the effects of encryption on those practices. Content blocking may also happen at endpoints or at the edge of enterprise networks, but those are not addressed in this section.

In a mobile network content filtering usually occurs in the core network. With other networks, content filtering could occur in the core network or at the edge. A proxy is installed which analyses the transport metadata of the content users are viewing and either filters content based on a blacklist of sites or based on the user's pre-defined profile (e.g., for age sensitive content). Although filtering can be done by many methods, one commonly used method involves a trigger based on the proxy identifying a DNS lookup of a host name in a URL which appears on a blacklist being used by the operator. The subsequent requests to that domain will be re-routed to a proxy which checks whether the full URL matches a blocked URL on the list, and will return a 404 if a match is found. All other requests should complete. This technique does not work in situations where DNS traffic is encrypted (e.g., by employing [RFC7858]). This method is also used by other types of network providers enabling traffic inspection, but not modification.

Content filtering via a proxy can also utilize an intercepting certificate where the client's session is terminated at the proxy enabling for cleartext inspection of the traffic. A new session is created from the intercepting device to the client's destination; this is an opt-in strategy for the client, where the endpoint is configured to trust the intercepting certificate. Changes to TLSv1.3 do not impact this more invasive method of interception, that has the potential to expose every HTTPS session to an active man in the middle (MitM).

Another form of content filtering is called parental control, where some users are deliberately denied access to age-sensitive content as a feature to the service subscriber. Some sites involve a mixture of universal and age-sensitive content and filtering software. In these

cases, more granular (application layer) metadata may be used to analyze and block traffic. Methods that accessed cleartext application-layer metadata no longer work when sessions are encrypted. This type of granular filtering could occur at the endpoint or as a proxy service. However, the lack of ability to efficiently manage endpoints as a service reduces network service providers' ability to offer parental control.

2.3.2. Network Access and Data Usage

Approved access to a network is a prerequisite to requests for Internet traffic.

However, there are cases (beyond parental control) when a network service provider currently redirects customer requests for content (affecting content accessibility):

1. The network service provider is performing the accounting and billing for the content provider, and the customer has not (yet) purchased the requested content.
2. Further content may not be allowed as the customer has reached their usage limit and needs to purchase additional data service, which is the usual billing approach in mobile networks.

Currently, some network service providers redirect the customer using HTTP redirect to a captive portal page that explains to those customers the reason for the blockage and the steps to proceed. [RFC6108] describes one viable web notification system. When the HTTP headers and content are encrypted, this appropriately prevents mobile carriers from intercepting the traffic and performing an HTTP redirect. As a result, some mobile carriers block customer's encrypted requests, which impacts customer experience because the blocking reason must be conveyed by some other means. The customer may need to call customer care to find out the reason and/or resolve the issue, possibly extending the time needed to restore their network access. While there are well deployed alternate SMS-based solutions that do not involve out of specification protocol interception, this is still an unsolved problem for non-SMS users.

Further, when the requested service is about to consume the remainder of the user's plan limits, the transmission could be terminated and advance notifications may be sent to the user by their service provider to warn the user ahead of the exhausted plan. If web content is encrypted, the network provider cannot know the data transfer size at request time. Lacking this visibility of the application type and content size, the network would continue the transmission and stop the transfer when the limit was reached. A

partial transfer may not be usable by the client wasting both network and user resources, possibly leading to customer complaints. The content provider does not know user's service plans or current usage, and cannot warn the user of plan exhaustion.

In addition, some mobile network operators sell tariffs that allow free-data access to certain sites, known as 'zero rating'. A session to visit such a site incurs no additional cost or data usage to the user. For some implementations, zero rating is impacted if encryption hides the details of the content domain from the network.

2.3.3. Application Layer Gateways

Application Layer Gateways (ALG) assist applications to set connectivity across Network Address Translators (NAT), Firewalls, and/or Load Balancers for specific applications running across mobile networks. Section 2.9 of [RFC2663] describes the role of ALGs and their interaction with NAT and/or application payloads. ALG are deployed with an aim to improve connectivity. However, it is an IETF Best Common Practice recommendation that ALGs for UDP-based protocols should be turned off [RFC4787].

One example of an ALG in current use is aimed at video applications that use the Real Time Session Protocol (RTSP) [RFC7826] primary stream as a means to identify related Real Time Protocol/Real Time Control Protocol (RTP/RTCP) [RFC3550] flows at set-up. The ALG in this case relies on the 5-tuple flow information derived from RTSP to provision NAT or other middleboxes and provide connectivity. Implementations vary, and two examples follow:

1. Parse the content of the RTSP stream and identify the 5-tuple of the supporting streams as they are being negotiated.
2. Intercept and modify the 5-tuple information of the supporting media streams as they are being negotiated on the RTSP stream, which is more intrusive to the media streams.

When RTSP stream content is encrypted, the 5-tuple information within the payload is not visible to these ALG implementations, and therefore they cannot provision their associated middleboxes with that information.

The deployment of IPv6 may well reduce the need for NAT, and the corresponding requirement for Application Layer Gateways.

2.3.4. HTTP Header Insertion

Some mobile carriers use HTTP header insertion (see section 3.2.1 of [RFC7230]) to provide information about their customers to third parties or to their own internal systems [Enrich]. Third parties use the inserted information for analytics, customization, advertising, cross-site tracking of users, to bill the customer, or to selectively allow or block content. HTTP header insertion is also used to pass information internally between a mobile service provider's sub-systems, thus keeping the internal systems loosely coupled. When HTTP connections are encrypted to protect users privacy, mobile network service providers cannot insert headers to accomplish the, sometimes considered controversial, functions above.

Guidance from the Internet Architecture Board has been provided in RFC8165 [RFC8165] on Design Considerations for Metadata Insertion. The guidance asserts that designs that share metadata only by explicit actions at the host are preferable to designs in which middleboxes insert metadata. Alternate notification methods that follow this and other guidance would be helpful to mobile carriers.

3. Encryption in Hosting and Application SP Environments

Hosted environments have had varied requirements in the past for encryption, with many businesses choosing to use these services primarily for data and applications that are not business or privacy sensitive. A shift prior to the revelations on surveillance/passive monitoring began where businesses were asking for hosted environments to provide higher levels of security so that additional applications and service could be hosted externally. Businesses understanding the threats of monitoring in hosted environments increased that pressure to provide more secure access and session encryption to protect the management of hosted environments as well as for the data and applications.

3.1. Management Access Security

Hosted environments may have multiple levels of management access, where some may be strictly for the Hosting SP (infrastructure that may be shared among customers) and some may be accessed by a specific customer for application management. In some cases, there are multiple levels of hosting service providers, further complicating the security of management infrastructure and the associated requirements.

Hosting service provider management access is typically segregated from other traffic with a control channel and may or may not be encrypted depending upon the isolation characteristics of the

management session. Customer access may be through a dedicated connection, but discussion for that connection method is out-of-scope for this document.

In overlay networks (e.g. VXLAN, Geneve, etc.) that are used to provide hosted services, management access for a customer to support application management may depend upon the security mechanisms available as part of that overlay network. While overlay network data encapsulations may be used to indicate the desired isolation, this is not sufficient to prevent deliberate attacks that are aware of the use of the overlay network.

[I-D.mglt-nvo3-geneve-security-requirements] describes requirements to handle attacks. It is possible to use an overlay header in combination with IPsec or other encrypted traffic sessions, but this adds the requirement for authentication infrastructure and may reduce packet transfer performance. The use of an overlay header may also be deployed as a mechanism to manage encrypted traffic streams on the network by network service providers. Additional extension mechanisms to provide integrity and/or privacy protections are being investigated for overlay encapsulations. Section 7 of [RFC7348] describes some of the security issues possible when deploying VXLAN on Layer 2 networks. Rogue endpoints can join the multicast groups that carry broadcast traffic, for example.

3.1.1. Customer Access Monitoring

Hosted applications that allow some level of customer management access may also require monitoring by the hosting service provider. Monitoring could include access control restrictions such as authentication, authorization, and accounting for filtering and firewall rules to ensure they are continuously met. Customer access may occur on multiple levels, including user-level and administrative access. The hosting service provider may need to monitor access either through session monitoring or log evaluation to ensure security service level agreements (SLA) for access management are met. The use of session encryption to access hosted environments limits access restrictions to the metadata described below. Monitoring and filtering may occur at an:

2-tuple IP-level with source and destination IP addresses alone, or

5-tuple IP and protocol-level with source IP address, destination IP address, protocol number, source port number, and destination port number.

Session encryption at the application level, TLS for example, currently allows access to the 5-tuple. IP-level encryption, such as IPsec in tunnel mode prevents access to the original 5-tuple and may

limit the ability to restrict traffic via filtering techniques. This shift may not impact all hosting service provider solutions as alternate controls may be used to authenticate sessions or access may require that clients access such services by first connecting to the organization before accessing the hosted application. Shifts in access may be required to maintain equivalent access control management. Logs may also be used for monitoring that access control restrictions are met, but would be limited to the data that could be observed due to encryption at the point of log generation. Log analysis is out of scope for this document.

3.1.2. SP Content Monitoring of Applications

The following observations apply to any IT organization that is responsible for delivering services, whether to third-parties, for example as a web based service, or to internal customers in an enterprise, e.g. a data processing system that forms a part of the enterprise's business.

Organizations responsible for the operation of a data center have many processes which access the contents of IP packets (passive methods of measurement, as defined in [RFC7799]). These processes are typically for service assurance or security purposes as part of their data center operations.

Examples include:

- Network Performance Monitoring/Application Performance Monitoring
- Intrusion defense/prevention systems
- Malware detection
- Fraud Monitoring
- Application DDOS protection
- Cyber-attack investigation
- Proof of regulatory compliance
- Data Leakage Prevention

Many application service providers simply terminate sessions to/from the Internet at the edge of the data center in the form of SSL/TLS offload in the load balancer. Not only does this reduce the load on

application servers, it simplifies the processes to enable monitoring of the session content.

However, in some situations, encryption deeper in the data center may be necessary to protect personal information or in order to meet industry regulations, e.g. those set out by the Payment Card Industry (PCI). In such situations, various methods have been used to allow service assurance and security processes to access unencrypted data. These include SSL/TLS decryption in dedicated units, which then forward packets to SP-controlled tools, or by real-time or post-capture decryption in the tools themselves. A number of these tools provide passive decryption by providing the monitoring device with the server's private key. The move to increased use of forward-secret key exchange mechanism impacts the use of these techniques.

Data center operators may also maintain packet recordings in order to be able to investigate attacks, breach of internal processes, etc. In some industries, organizations may be legally required to maintain such information for compliance purposes. Investigations of this nature have used access to the unencrypted contents of the packet. Alternate methods to investigate attacks or breach of process will rely on endpoint information, such as logs. As previously noted, logs often lack complete information, and this is seen as a concern resulting in some relying on session access for additional information.

Application Service Providers may offer content-level monitoring options to detect intellectual property leakage, or other attacks. In service provider environments where Data Loss Prevention (DLP) has been implemented on the basis of the service provider having cleartext access to session streams, the use of encrypted streams prevents these implementations from conducting content searches for the keywords or phrases configured in the DLP system. DLP is often used to prevent the leakage of Personally Identifiable Information (PII) as well as financial account information, Personal Health Information (PHI), and Payment Card Information (PCI). If session encryption is terminated at a gateway prior to accessing these services, DLP on session data can still be performed. The decision of where to terminate encryption to hosted environments will be a risk decision made between the application service provider and customer organization according to their priorities. DLP can be performed at the server for the hosted application and on an end user's system in an organization as alternate or additional monitoring points of content; however, this is not frequently done in a service provider environment.

Application service providers, by their very nature, control the application endpoint. As such, much of the information gleaned from

sessions are still available on that endpoint. However, when a gap exists in the application's logging and debugging capabilities, this has led the application service provider to access data-in-transport for monitoring and debugging.

3.2. Hosted Applications

Organizations are increasingly using hosted applications rather than in-house solutions that require maintenance of equipment and software. Examples include Enterprise Resource Planning (ERP) solutions, payroll service, time and attendance, travel and expense reporting among others. Organizations may require some level of management access to these hosted applications and will typically require session encryption or a dedicated channel for this activity.

In other cases, hosted applications may be fully managed by a hosting service provider with service level agreement expectations for availability and performance as well as for security functions including malware detection. Due to the sensitive nature of these hosted environments, the use of encryption is already prevalent. Any impact may be similar to an enterprise with tools being used inside of the hosted environment to monitor traffic. Additional concerns were not reported in the call for contributions.

3.2.1. Monitoring Managed Applications

Performance, availability, and other aspects of a SLA are often collected through passive monitoring. For example:

- o Availability: ability to establish connections with hosts to access applications, and discern the difference between network or host-related causes of unavailability.
- o Performance: ability to complete transactions within a target response time, and discern the difference between network or host-related causes of excess response time.

Here, as with all passive monitoring, the accuracy of inferences are dependent on the cleartext information available, and encryption would tend to reduce the information and therefore, the accuracy of each inference. Passive measurement of some metrics will be impossible with encryption that prevents inferring packet correspondence across multiple observation points, such as for packet loss metrics.

Application logging currently lacks detail sufficient to make accurate inferences in an environment with increased encryption, and

so this constitutes a gap for passive performance monitoring (which could be closed if log details are enhanced in the future).

3.2.2. Mail Service Providers

Mail (application) service providers vary in what services they offer. Options may include a fully hosted solution where mail is stored external to an organization's environment on mail service provider equipment or the service offering may be limited to monitor incoming mail to remove spam [Section 5.1], malware [Section 5.6], and phishing attacks [Section 5.3] before mail is directed to the organization's equipment. In both of these cases, content of the messages and headers is monitored to detect spam, malware, phishing, and other messages that may be considered an attack.

STARTTLS should have zero effect on anti-spam efforts for SMTP traffic. Anti-spam services could easily be performed on an SMTP gateway, eliminating the need for TLS decryption services. The impact to anti-spam service providers should be limited to a change in tools, where middleboxes were deployed to perform these functions.

Many efforts are emerging to improve user-to-user encryption, including promotion of PGP and newer efforts such as Dark Mail [DarkMail]. Of course, content-based spam filtering will not be possible on encrypted content.

3.3. Data Storage

Numerous service offerings exist that provide hosted storage solutions. This section describes the various offerings and details the monitoring for each type of service and how encryption may impact the operational and security monitoring performed.

Trends in data storage encryption for hosted environments include a range of options. The following list is intentionally high-level to describe the types of encryption used in coordination with data storage that may be hosted remotely, meaning the storage is physically located in an external data center requiring transport over the Internet. Options for monitoring will vary with each encryption approach described below. In most cases, solutions have been identified to provide encryption while ensuring management capabilities were maintained through logging or other means.

3.3.1. Object-level Encryption

For higher security and/or privacy of data and applications, options that provide end-to-end encryption of the data from the user's desktop or server to the storage platform may be preferred. This

description includes any solution that encrypts data at the object level, not transport level. Encryption of data may be performed with libraries on the system or at the application level, which includes file encryption services via a file manager. Object-level encryption is useful when data storage is hosted, or scenarios when the storage location is determined based on capacity or based on a set of parameters to automate decisions. This could mean that large data sets accessed infrequently could be sent to an off-site storage platform at an external hosting service, data accessed frequently may be stored locally, or the decision could be based on the transaction type. Object-level encryption is grouped separately for the purpose of this document since data may be stored in multiple locations including off-site remote storage platforms. If session encryption is also used, the protocol is likely to be TLS.

Impacts to monitoring may include access to content inspection for data leakage prevention and similar technologies, depending on their placement in the network.

3.3.1.1. Monitoring for Hosted Storage

Monitoring of hosted storage solutions that use host-level (object) encryption is described in this subsection. Host-level encryption can be employed for backup services, and occasionally for external storage services (operated by a third party) when internal storage limits are exceeded.

Monitoring of data flows to hosted storage solutions is performed for security and operational purposes. The security monitoring may be to detect anomalies in the data flows that could include changes to destination, the amount of data transferred, or alterations in the size and frequency of flows. Operational considerations include capacity and availability monitoring.

3.3.2. Disk Encryption, Data at Rest

There are multiple ways to achieve full disk encryption for stored data. Encryption may be performed on data to be stored while in transit close to the storage media with solutions like Controller Based Encryption (CBE) or in the drive system with Self-Encrypting Drives (SED). Session encryption is typically coupled with encryption of these data at rest (DAR) solutions to also protect data in transit. Transport encryption is likely via TLS.

3.3.2.1. Monitoring Session Flows for Data at Rest (DAR) Solutions

Monitoring for transport of data to storage platforms, where object level encryption is performed close to or on the storage platform are similar to those described in the section on Monitoring for Hosted Storage. The primary difference for these solutions is the possible exposure of sensitive information, which could include privacy related data, financial information, or intellectual property if session encryption via TLS is not deployed. Session encryption is typically used with these solutions, but that decision would be based on a risk assessment. There are use cases where DAR or disk-level encryption is required. Examples include preventing exposure of data if physical disks are stolen or lost. In the case where TLS is in use, monitoring and the exposure of data is limited to a 5-tuple.

3.3.3. Cross Data Center Replication Services

Storage services also include data replication which may occur between data centers and may leverage Internet connections to tunnel traffic. The traffic may use iSCSI [RFC7143] or FC/IP [RFC7146] encapsulated in IPsec. Either transport or tunnel mode may be used for IPsec depending upon the termination points of the IPsec session, if it is from the storage platform itself or from a gateway device at the edge of the data center respectively.

3.3.3.1. Monitoring Of IPsec for Data Replication Services

Monitoring of data flows between data centers (for data replication) may be performed for security and operational purposes and would typically concentrate more on operational aspects since these flows are essentially virtual private networks (VPN) between data centers. Operational considerations include capacity and availability monitoring. The security monitoring may be to detect anomalies in the data flows, similar to what was described in the "Monitoring for Hosted Storage Section". If IPsec tunnel mode is in use, monitoring is limited to a 2-tuple, or with transport mode, a 5-tuple.

4. Encryption for Enterprises

Encryption of network traffic within the private enterprise is a growing trend, particularly in industries with audit and regulatory requirements. Some enterprise internal networks are almost completely TLS and/or IPsec encrypted.

For each type of monitoring, different techniques and access to parts of the data stream are part of current practice. As we transition to an increased use of encryption, alternate methods of monitoring for operational purposes may be necessary to reduce the practice of

breaking encryption (other policies may apply in some enterprise settings).

4.1. Monitoring Practices of the Enterprise

Large corporate enterprises are the owners of the platforms, data, and network infrastructure that provide critical business services to their user communities. As such, these enterprises are responsible for all aspects of the performance, availability, security, and quality of experience for all user sessions. In many such enterprises, users are required to consent to the enterprise monitoring all their activities as a condition of employment. Subsections of 4. Encryption for Enterprises may discuss techniques that access data beyond the data-link, network, and transport level headers typically used in SP networks since the corporate enterprise owns the data. These responsibilities break down into three basic areas:

1. Security Monitoring and Control
2. Application Performance Monitoring and Reporting
3. Network Diagnostics and Troubleshooting

In each of the above areas, technical support teams utilize collection, monitoring, and diagnostic systems. Some organizations currently use attack methods such as replicated TLS server RSA private keys to decrypt passively monitored copies of encrypted TLS packet streams.

For an enterprise to avoid costly application down time and deliver expected levels of performance, protection, and availability, some forms of traffic analysis, sometimes including examination of packet payloads, are currently used.

4.1.1. Security Monitoring in the Enterprise

Enterprise users are subject to the policies of their organization and the jurisdictions in which the enterprise operates. As such, proxies may be in use to:

1. intercept outbound session traffic to monitor for intellectual property leakage (by users, malware, and trojans),
2. detect viruses/malware entering the network via email or web traffic,

3. detect malware/Trojans in action, possibly connecting to remote hosts,
4. detect attacks (Cross site scripting and other common web related attacks),
5. track misuse and abuse by employees,
6. restrict the types of protocols permitted to/from the entire corporate environment,
7. detect and defend against Internet DDoS attacks, including both volumetric and layer 7 attacks.

A significant portion of malware hides its activity within TLS or other encryption protocols. This includes lateral movement, Command and Control, and Data Exfiltration.

The impact to a fully encrypted internal network would include cost and possible loss of detection capabilities associated with the transformation of the network architecture and tools for monitoring. The capabilities of detection through traffic fingerprinting, logs, host-level transaction monitoring, and flow analysis would vary depending on access to a 2-tuple or 5-tuple in the network as well.

Security monitoring in the enterprise may also be performed at the endpoint with numerous current solutions that mitigate the same problems as some of the above mentioned solutions. Since the software agents operate on the device, they are able to monitor traffic before it is encrypted, monitor for behavior changes, and lock down devices to use only the expected set of applications. Session encryption does not affect these solutions. Some might argue that scaling is an issue in the enterprise, but some large enterprises have used these tools effectively.

Use of Bring-your-own-device (BYOD) policies within organizations may limit the scope of monitoring permitted with these alternate solutions. Network endpoint assessment (NEA) or the use of virtual hosts could help to bridge the monitoring gap.

4.1.2. Application Performance Monitoring in the Enterprise

There are two main goals of monitoring:

1. Assess traffic volume on a per-application basis, for billing, capacity planning, optimization of geographical location for servers or proxies, and other goals.

2. Assess performance in terms of application response time and user perceived response time.

Network-based Application Performance Monitoring tracks application response time by user and by URL, which is the information that the application owners and the lines of business request. Content Delivery Networks (CDNs) add complexity in determining the ultimate endpoint destination. By their very nature, such information is obscured by CDNs and encrypted protocols -- adding a new challenge for troubleshooting network and application problems. URL identification allows the application support team to do granular, code level troubleshooting at multiple tiers of an application.

New methodologies to monitor user perceived response time and to separate network from server time are evolving. For example, the IPv6 Destination Option Header (DOH) implementation of Performance and Diagnostic Metrics (PDM) will provide this [RFC8250]. Using PDM with IPsec Encapsulating Security Payload (ESP) Transport Mode requires placement of the PDM DOH within the ESP encrypted payload to avoid leaking timing and sequence number information that could be useful to an attacker. Use of PDM DOH also may introduce some security weaknesses, including a timing attack, as described in Section 7 of [RFC8250]. For these and other reasons, [RFC8250] requires that the PDM DOH option be explicitly turned on by administrative action in each host where this measurement feature will be used.

4.1.3. Enterprise Network Diagnostics and Troubleshooting

One primary key to network troubleshooting is the ability to follow a transaction through the various tiers of an application in order to isolate the fault domain. A variety of factors relating to the structure of the modern data center and multi-tiered application have made it difficult to follow a transaction in network traces without the ability to examine some of the packet payload. Alternate methods, such as log analysis need improvement to fill this gap.

4.1.3.1. Address Sharing (NAT)

Content Delivery Networks (CDNs) and NATs and Network Address and Port Translators (NAPT) obscure the ultimate endpoint designation (See [RFC6269] for types of address sharing and a list of issues). Troubleshooting a problem for a specific end user requires finding information such as the IP address and other identifying information so that their problem can be resolved in a timely manner.

NAT is also frequently used by lower layers of the data center infrastructure. Firewalls, Load Balancers, Web Servers, App Servers,

and Middleware servers all regularly NAT the source IP of packets. Combine this with the fact that users are often allocated randomly by load balancers to all these devices, the network troubleshooter is often left with very few options in today's environment due to poor logging implementations in applications. As such, network troubleshooting is used to trace packets at a particular layer, decrypt them, and look at the payload to find a user session.

This kind of bulk packet capture and bulk decryption is frequently used when troubleshooting a large and complex application. Endpoints typically don't have the capacity to handle this level of network packet capture, so out-of-band networks of robust packet brokers and network sniffers that use techniques such as copies of TLS RSA private keys accomplish this task today.

4.1.3.2. TCP Pipelining/Session Multiplexing

TCP pipelining/session multiplexing used mainly by middleboxes today allows for multiple end user sessions to share the same TCP connection. This raises several points of interest with an increased use of encryption. TCP session multiplexing should still be possible when TLS or TCPcrypt is in use since the TCP header information is exposed leaving the 5-tuple accessible. The use of TCP session multiplexing of an IP layer encryption, e.g. IPsec, that only exposes a 2-tuple would not be possible. Troubleshooting capabilities with encrypted sessions from the middlebox may limit troubleshooting to the use of logs from the end points performing the TCP multiplexing or from the middleboxes prior to any additional encryption that may be added to tunnel the TCP multiplexed traffic.

Increased use of HTTP/2 will likely further increase the prevalence of session multiplexing, both on the Internet and in the private data center. HTTP pipelining requires both the client and server to participate; visibility of packets once encrypted will hide the use of HTTP pipelining for any monitoring that takes place outside of the endpoint or proxy solution. Since HTTP pipelining is between a client and server, logging capabilities may require improvement in some servers and clients for debugging purposes if this is not already possible. Visibility for middleboxes includes anything exposed by TLS and the 5-tuple.

4.1.3.3. HTTP Service Calls

When an application server makes an HTTP service call to back end services on behalf of a user session, it uses a completely different URL and a completely different TCP connection. Troubleshooting via network trace involves matching up the user request with the HTTP service call. Some organizations do this today by decrypting the TLS

packet and inspecting the payload. Logging has not been adequate for their purposes.

4.1.3.4. Application Layer Data

Many applications use text formats such as XML to transport data or application level information. When transaction failures occur and the logs are inadequate to determine the cause, network and application teams work together, each having a different view of the transaction failure. Using this troubleshooting method, the network packet is correlated with the actual problem experienced by an application to find a root cause. The inability to access the payload prevents this method of troubleshooting.

4.2. Techniques for Monitoring Internet Session Traffic

Corporate networks commonly monitor outbound session traffic to detect or prevent attacks as well as to guarantee service level expectations. In some cases, alternate options are available when encryption is in use through a proxy or a shift to monitoring at the endpoint. In both cases, scaling is a concern and advancements to support this shift in monitoring practices will assist the deployment of end-to-end encryption.

Some DLP tools intercept traffic at the Internet gateway or proxy services with the ability to man-in-the-middle (MiTM) encrypted session traffic (HTTP/TLS). These tools may monitor for key words important to the enterprise including business sensitive information such as trade secrets, financial data, personally identifiable information (PII), or personal health information (PHI). Various techniques are used to intercept HTTP/TLS sessions for DLP and other purposes, and can be misused as described in "Summarizing Known Attacks on TLS and DTLS" [RFC7457] Section 2.8. Note: many corporate policies allow access to personal financial and other sites for users without interception. Another option is to terminate a TLS session prior to the point where monitoring is performed. Aside from exposing user information to the enterprise MITM devices often are subject to severe security defects which can lead to exposure of user data to attackers outside the enterprise UserData [UserData]. In addition, implementation errors in middleboxes have led to major difficulties in deploying new versions of security protocols such as TLS [Ben17a][Ben17b][Res17a][Res17b]

Monitoring traffic patterns for anomalous behavior such as increased flows of traffic that could be bursty at odd times or flows to unusual destinations (small or large amounts of traffic) is common. This traffic may or may not be encrypted and various methods of encryption or just obfuscation may be used.

Web filtering devices are sometimes used to allow only access to well-known sites found to be legitimate and free of malware on last check by a web filtering service company. One common example of web filtering in a corporate environment is blocking access to sites that are not well-known to these tools for the purpose of blocking malware; this may be noticeable to those in research who are unable to access colleague's individual sites or new web sites that have not yet been screened. In situations where new sites are required for access, they can typically be added after notification by the user or log alerts and review. Home mail account access may be blocked in corporate settings to prevent another vector for malware to enter as well as for intellectual property to leak out of the network. This method remains functional with increased use of encryption and may be more effective at preventing malware from entering the network. Some enterprises may be more aggressive in their filtering and monitoring policy, causing undesirable outcomes. Web filtering solutions monitor and potentially restrict access based on the destination URL when available, server name, IP address, or the DNS name. A complete URL may be used in cases where access restrictions vary for content on a particular site or for the sites hosted on a particular server. In some cases, the enterprise may use a proxy to access this additional information based on their policy. This type of restriction is intended to be transparent to users in a corporate setting as the typical corporate user does not access sites which are not well-known to these tools. However, the mechanisms which these web filters use to do monitoring and enforcement have the potential to cause access issues or other user-visible failures.

Desktop DLP tools are used in some corporate environments as well. Since these tools reside on the desktop, they can intercept traffic before it is encrypted and may provide a continued method of monitoring intellectual property leakage from the desktop to the Internet or attached devices.

DLP tools can also be deployed by Network Service providers, as they have the vantage point of monitoring all traffic paired with destinations off the enterprise network. This makes an effective solution for enterprises that allow "bring-your-own" devices when the traffic is not encrypted, and for devices outside the desktop category (such as mobile phones) that are used on corporate networks nonetheless.

Enterprises may wish to reduce the traffic on their Internet access facilities by monitoring requests for within-policy content and caching it. In this case, repeated requests for Internet content spawned by URLs in e-mail trade newsletters or other sources can be served within the enterprise network. Gradual deployment of end to end encryption would tend to reduce the cacheable content over time,

owing to concealment of critical headers and payloads. Many forms of enterprise performance management may be similarly affected. It should be noted that transparent caching is considered an anti-pattern.

5. Security Monitoring for Specific Attack Types

Effective incident response today requires collaboration at Internet scale. This section will only focus on efforts of collaboration at Internet scale that are dedicated to specific attack types. They may require new monitoring and detection techniques in an increasingly encrypted Internet. As mentioned previously, some service providers have been interfering with STARTTLS to prevent session encryption to be able to perform functions they are used to (injecting ads, monitoring, etc.). By detailing the current monitoring methods used for attack detection and response, this information can be used to devise new monitoring methods that will be effective in the changed Internet via collaboration and innovation.

Changes to improve encryption or to deploy OS methods have little impact on the detection of malicious actors. Malicious actors have had access to strong encryption for quite some time. Incident responders, in many cases, have developed techniques to locate malicious traffic within encrypted sessions. The following section will note some examples where detection and mitigation of such traffic has been successful.

5.1. Mail Abuse and spam

The largest operational effort to prevent mail abuse is through the Messaging, Malware, Mobile Anti-Abuse Working Group (M3AAWG)[M3AAWG]. Mail abuse is combatted directly with mail administrators who can shut down or stop continued mail abuse originating from large scale providers that participate in using the Abuse Reporting Format (ARF) agents standardized in the IETF [RFC5965], [RFC6430], [RFC6590], [RFC6591], [RFC6650], [RFC6651], and [RFC6652]. The ARF agent directly reports abuse messages to the appropriate service provider who can take action to stop or mitigate the abuse. Since this technique uses the actual message, the use of SMTP over TLS between mail gateways will not affect its usefulness. As mentioned previously, SMTP over TLS only protects data while in transit and the messages may be exposed on mail servers or mail gateways if a user-to-user encryption method is not used. Current user-to-user message encryption methods on email (S/MIME and PGP) do not encrypt the email header information used by ARF and the service provider operators in their abuse mitigation efforts.

Another effort, Domain-based Message Authentication, Reporting, and Conformance (DMARC) [RFC7489] is a mechanism for policy distribution that enables increasingly strict handling of messages that fail authentication checks, ranging from no action, through altered delivery, up to message rejection. DMARC is also not affected by the use of STARTTLS.

5.2. Denial of Service

Response to Denial of Service (DoS) attacks are typically coordinated by the SP community with a few key vendors who have tools to assist in the mitigation efforts. Traffic patterns are determined from each DoS attack to stop or rate limit the traffic flows with patterns unique to that DoS attack.

Data types used in monitoring traffic for DDoS are described in the DDoS Open Threat Signaling (DOTS) [DOTS] working group documents in development. The impact of encryption can be understood from their documented use cases[I-D.ietf-dots-use-cases].

Data types used in DDoS attacks have been detailed in the IODEF Guidance draft [RFC8274], Appendix A.2, with the help of several members of the service provider community. The examples provided are intended to help identify the useful data in detecting and mitigating these attacks independent of the transport and protocol descriptions in the drafts.

5.3. Phishing

Investigations and response to phishing attacks follow well-known patterns, requiring access to specific fields in email headers as well as content from the body of the message. When reporting phishing attacks, the recipient has access to each field as well as the body to make content reporting possible, even when end-to-end encryption is used. The email header information is useful to identify the mail servers and accounts used to generate or relay the attack messages in order to take the appropriate actions. The content of the message often contains an embedded attack that may be in an infected file or may be a link that results in the download of malware to the user's system.

Administrators often find it helpful to use header information to track down similar message in their mail queue or users inboxes to prevent further infection. Combinations of To:, From:, Subject:, Received: from header information might be used for this purpose. Administrators may also search for document attachments of the same name, size, or containing a file with a matching hash to a known phishing attack. Administrators might also add URLs contained in

messages to block lists locally or this may also be done by browser vendors through larger scales efforts like that of the Anti-Phishing Working Group (APWG). See the Coordinating Attack Response at Internet Scale (CARIS) workshop Report [RFC8073] for additional information and pointers to the APWG's efforts on anti-phishing.

A full list of the fields used in phishing attack incident response can be found in RFC5901. Future plans to increase privacy protections may limit some of these capabilities if some email header fields are encrypted, such as To:, From:, and Subject: header fields. This does not mean that those fields should not be encrypted, only that we should be aware of how they are currently used.

Some products protect users from phishing by maintaining lists of known phishing domains (such as misspelled bank names) and blocking access. This can be done by observing DNS, clear-text HTTP, or server name indication (SNI) in TLS, in addition to analyzing email. Alternate options to detect and prevent phishing attacks may be needed. More recent examples of data exchanged in spear phishing attacks has been detailed in the IODEF Guidance draft [RFC8274], Appendix A.3.

5.4. Botnets

Botnet detection and mitigation is complex as botnets may involve hundreds or thousands of hosts with numerous Command and Control (C&C) servers. The techniques and data used to monitor and detect each may vary. Connections to C&C servers are typically encrypted, therefore a move to an increasingly encrypted Internet may not affect the detection and sharing methods used.

5.5. Malware

Malware monitoring and detection techniques vary. As mentioned in the enterprise section, malware monitoring may occur at gateways to the organization analyzing email and web traffic. These services can also be provided by service providers, changing the scale and location of this type of monitoring. Additionally, incident responders may identify attributes unique to types of malware to help track down instances by their communication patterns on the Internet or by alterations to hosts and servers.

Data types used in malware investigations have been summarized in an example of the IODEF Guidance draft [RFC8274], Appendix A.1.

5.6. Spoofed Source IP Address Protection

The IETF has reacted to spoofed source IP address-based attacks, recommending the use of network ingress filtering BCP 38 [RFC2827] and the unicast Reverse Path Forwarding (uRPF) mechanism [RFC2504]. But uRPF suffers from limitations regarding its granularity: a malicious node can still use a spoofed IP address included inside the prefix assigned to its link. The Source Address Validation Improvements (SAVI) mechanisms try to solve this issue. Basically, a SAVI mechanism is based on the monitoring of a specific address assignment/management protocol (e.g., SLAAC [RFC4862], SEND [RFC3971], DHCPv4/v6 [RFC2131][RFC3315]) and, according to this monitoring, set-up a filtering policy allowing only the IP flows with a correct source IP address (i.e., any packet with a source IP address, from a node not owning it, is dropped). The encryption of parts of the address assignment/management protocols, critical for SAVI mechanisms, can result in a dysfunction of the SAVI mechanisms.

5.7. Further work

Although incident response work will continue, new methods to prevent system compromise through security automation and continuous monitoring [SACM] may provide alternate approaches where system security is maintained as a preventative measure.

6. Application-based Flow Information Visible to a Network

This section describes specific techniques used in monitoring applications that is visible to the network if a 5-tuple is exposed and as such can potentially be used as an input for future network management approaches. It also includes an overview of IPFIX, a flow-based protocol used to export information about network flows.

6.1. IP Flow Information Export

Many of the accounting, monitoring and measurement tasks described in this document, especially Section 2.3.2, Section 3.1.1, Section 4.1.3, Section 4.2, and Section 5.2 use the IPFIX protocol [RFC7011] for export and storage of the monitored information. IPFIX evolved from the widely-deployed NetFlow protocol [RFC3954], which exports information about flows identified by 5-tuple. While NetFlow was largely concerned with exporting per-flow byte and packet counts for accounting purposes, IPFIX's extensible information model [RFC7012] provides a variety of Information Elements (IEs) [IPFIX-IANA] for representing information above and below the traditional network layer flow information. Enterprise-specific IEs allow exporter vendors to define their own non-standard IEs, as well,

and many of these are driven by header and payload inspection at the metering process.

While the deployment of encryption has no direct effect on the use of IPFIX, certain defined IEs may become unavailable when the metering process observing the traffic cannot decrypt formerly cleartext information. For example, HTTPS renders HTTP header analysis impossible, so IEs derived from the header (e.g. `httpContentType`, `httpUserAgent`) cannot be exported.

The collection of IPFIX data itself, of course, provides a point of centralization for potentially business- and privacy-critical information. The IPFIX File Format specification [RFC5655] recommends encryption for this data at rest, and the IP Flow Anonymization specification [RFC6235] defines a metadata format for describing the anonymization functions applied to an IPFIX dataset, if anonymization is employed for data sharing of IPFIX information between enterprises or network operators.

6.2. TLS Server Name Indication

When initiating the TLS handshake, the Client may provide an extension field (`server_name`) which indicates the server to which it is attempting a secure connection. TLS SNI was standardized in 2003 to enable servers to present the "correct TLS certificate" to clients in a deployment of multiple virtual servers hosted by the same server infrastructure and IP-address. Although this is an optional extension, it is today supported by all modern browsers, web servers and developer libraries. Akamai [Nygren] reports that many of their customer see client TLS SNI usage over 99%. It should be noted that HTTP/2 introduces the Alt-SVC method for upgrading the connection from HTTP/1 to either unencrypted or encrypted HTTP/2. If the initial HTTP/1 request is unencrypted, the destination alternate service name can be identified before the communication is potentially upgraded to encrypted HTTP/2 transport. HTTP/2 requires the TLS implementation to support the Server Name Indication (SNI) extension (see section 9.2 of [RFC7540]). It is also worth noting that [RFC7838] "allows an origin server to nominate additional means of interacting with it on the network", while [RFC8164] allows for a URI to be accessed with HTTP/2 and TLS using Opportunistic Security (on an experimental basis).

This information is only available if the client populates the Server Name Indication extension. Doing so is an optional part of the TLS standard and as stated above this has been implemented by all major browsers. Due to its optional nature, though, existing network filters that examine a TLS ClientHello for a SNI extension cannot expect to always find one. The SNI Encryption in TLS Through

Tunneling [I-D.ietf-tls-sni-encryption] draft has been adopted by the TLS working group, which provides solutions to encrypt SNI. As such, there will be an option to encrypt SNI in future versions of TLS. The per-domain nature of SNI may not reveal the specific service or media type being accessed, especially where the domain is of a provider offering a range of email, video, Web pages etc. For example, certain blog or social network feeds may be deemed 'adult content', but the Server Name Indication will only indicate the server domain rather than a URL path.

There are additional issues for identification of content using SNI: [RFC7540] includes connection coalescing, [I-D.ietf-httpbis-origin-frame] defines the ORIGIN frame, and the [I-D.bishop-httpbis-http2-additional-certs] proposal will increase the difficulty of passive monitoring.

6.3. Application Layer Protocol Negotiation (ALPN)

ALPN is a TLS extension which may be used to indicate the application protocol within the TLS session. This is likely to be of more value to the network where it indicates a protocol dedicated to a particular traffic type (such as video streaming) rather than a multi-use protocol. ALPN is used as part of HTTP/2 'h2', but will not indicate the traffic types which may make up streams within an HTTP/2 multiplex. ALPN is sent clear text in the ClientHello and the server returns it in Encrypted Extensions in TLS 1.3.

6.4. Content Length, BitRate and Pacing

The content length of encrypted traffic is effectively the same as that of the cleartext. Although block ciphers utilize padding, this makes a negligible difference. Bitrate and pacing are generally application specific, and do not change much when the content is encrypted. Multiplexed formats (such as HTTP/2 and QUIC [QUIC]) may however incorporate several application streams over one connection, which makes the bitrate/pacing no longer application-specific. Also, packet padding is available in HTTP/2, TLS 1.3, and many other protocols. Traffic analysis is made more difficult by such countermeasures.

7. Effect of Encryption on Mobile Network Evolution

Transport header encryption prevents the use of transit proxies in center of the network and the use of some edge proxies by preventing the proxies from taking action on the stream. It may be that the claimed benefits of such proxies could be achieved by end-to-end client and server optimizations, distribution using CDNs, plus the ability to continue connections across different access technologies

(across dynamic user IP addresses). The following aspects should be considered in this approach:

1. In a wireless mobile network, the delay and channel capacity per user and sector varies due to coverage, contention, user mobility, scheduling balances fairness, capacity, and service QoE. If most users are at the cell edge, the controller cannot use more complex QAM, thus reducing total cell capacity; similarly if a UMTS edge is serving some number of CS-Voice Calls, the remaining capacity for packet services is reduced.
 2. Mobile wireless networks service in-bound roamers (Users of Operator A in a foreign operator Network B) by backhauling their traffic through Operator B's network to Operator A's Network and then serving through the P-Gateway (PGW), General GPRS Support Node (GGSN), Content Distribution Network (CDN) etc., of Operator A (User's Home Operator). Increasing window sizes to compensate for the path RTT will have the limitations outlined earlier for TCP. The outbound roamer scenario has a similar TCP performance impact.
 3. Issues in deploying CDNs in Radio Access Networks (RAN) include decreasing client-server control loop that requires deploying CDNs/Cloud functions that terminate encryption closer to the edge. In Cellular RAN, the user IP traffic is encapsulated into General Packet Radio Service (GPRS) Tunneling Protocol-User Plane (GTP-U in UMTS and LTE) tunnels to handle user mobility; the tunnels terminate in APN/GGSN/PGW that are in central locations. One user's traffic may flow through one or more APN's (for example Internet APN, Roaming APN for Operator X, Video-Service APN, OnDeckAPN etc.). The scope of operator private IP addresses may be limited to specific APNs. Since CDNs generally operate on user IP flows, deploying them would require enhancing them with tunnel translation, tunnel management functions etc.
 4. While CDNs that de-encrypt flows or split-connection proxy (similar to split-tcp) could be deployed closer to the edges to reduce control loop RTT, with transport header encryption, such CDNs perform optimization functions only for partner client flows. Therefore, content from some Small-Medium Businesses (SMBs) would not get such CDN benefits.
8. Response to Increased Encryption and Looking Forward

As stated in [RFC7258], "an appropriate balance (between network management and PM mitigations) will emerge over time as real instances of this tension are considered." Numerous operators made it clear in their response to this document that they fully support

strong encryption and providing privacy for end users, this is a common goal. Operators recognize not all the practices documented need to be supported going forward, either because of the risk to end user privacy or alternate technologies and tools have already emerged. This document is intended to support network engineers and other innovators to work toward solving network and security management problems with protocol designers and application developers in new ways that facilitate adoption of strong encryption rather than preventing the use of encryption. By having the discussions on network and security management practices with application developers and protocol designers, each side of the debate can understand each others goals, work toward alternate solutions, and disband with practices that should no longer be supported. A goal of this document is to assist the IETF to understand some of the current practices so as to identify new work items for IETF-related use cases which can help facilitate the adoption of strong session encryption and support network and security management.

9. Security Considerations

There are no additional security considerations as this is a summary and does not include a new protocol or functionality.

10. IANA Considerations

This memo makes no requests of IANA.

11. Acknowledgements

Thanks to our reviewers, Natasha Rooney, Kevin Smith, Ashutosh Dutta, Brandon Williams, Jean-Michel Combes, Nalini Elkins, Paul Barrett, Badri Subramanyan, Igor Lubashev, Suresh Krishnan, Dave Dolson, Mohamed Boucadair, Stephen Farrell, Warren Kumari, Alia Atlas, Roman Danyliw, Mirja Kuhlewind, Ines Robles, Joe Clarke, Kyle Rose, Christian Huitema, and Chris Morrow for their editorial and content suggestions. Surya K. Kovvali provided material for section 7. Chris Morrow and Nik Teague provided reviews and updates specific to the DoS fingerprinting text. Brian Trammell provided the IPFIX text.

12. Informative References

[ACCORD] "Acord BoF IETF95
<https://www.ietf.org/proceedings/95/accord.html>".

- [CAIDA] "CAIDA *Anonymized Internet Traces*
[<http://www.caida.org/data/overview/> and
[http://www.caida.org/data/passive/
passive_2016_dataset.xml](http://www.caida.org/data/passive/passive_2016_dataset.xml)]".
- [DarkMail] "The Dark Mail Technical Alliance <https://darkmail.info/>".
- [DOTS] <https://datatracker.ietf.org/wg/dots/charter/>, "DDoS Open Threat Signaling IETF Working Group".
- [EFF2014] "EFF Report on STARTTLS Downgrade Attacks
[https://www.eff.org/deeplinks/2014/11/
starttls-downgrade-attacks](https://www.eff.org/deeplinks/2014/11/starttls-downgrade-attacks)".
- [Enrich] Narseo Vallina-Rodriguez, et al., "Header Enrichment or ISP Enrichment, Emerging Privacy Threats in Mobile Networks, Hot Middlebox, August 17-21 2015, London, United Kingdom", 2015.
- [I-D.bishop-httpbis-http2-additional-certs]
Bishop, M., Sullivan, N., and M. Thomson, "Secondary Certificate Authentication in HTTP/2", draft-bishop-httpbis-http2-additional-certs-05 (work in progress), October 2017.
- [I-D.dolson-plus-middlebox-benefits]
Dolson, D., Snellman, J., Boucadair, M., and C. Jacquenet, "Beneficial Functions of Middleboxes", draft-dolson-plus-middlebox-benefits-03 (work in progress), March 2017.
- [I-D.ietf-dots-use-cases]
Dobbins, R., Migault, D., Fouant, S., Moskowitz, R., Teague, N., Xia, L., and K. Nishizuka, "Use cases for DDoS Open Threat Signaling", draft-ietf-dots-use-cases-09 (work in progress), November 2017.
- [I-D.ietf-httpbis-origin-frame]
Nottingham, M. and E. Nygren, "The ORIGIN HTTP/2 Frame", draft-ietf-httpbis-origin-frame-06 (work in progress), January 2018.
- [I-D.ietf-tls-sni-encryption]
Huitema, C. and E. Rescorla, "SNI Encryption in TLS Through Tunneling", draft-ietf-tls-sni-encryption-02 (work in progress), March 2018.

- [I-D.mglt-nvo3-geneve-security-requirements]
Migault, D., Boutros, S., Wing, D., and S. Krishnan,
"Geneve Protocol Security Requirements", draft-mglt-nvo3-
geneve-security-requirements-03 (work in progress),
February 2018.
- [IPFIX-IANA]
"IP Flow Information Export (IPFIX) Entities
<https://www.iana.org/assignments/ipfix/>".
- [JNSLP] Surveillance, Vol. 8 No. 3, "10 Standards for Oversight
and Transparency of National Intelligence Services
<http://jnslp.com/>".
- [M3AAWG] "Messaging, Malware, Mobile Anti-Abuse Working Group
(M3AAWG) <https://www.maawg.org/>".
- [Nygren] <https://blogs.akamai.com/2017/03/reaching-toward-universal-tls-sni.html>, "Erik Nygren, personal reference".
- [QUIC] <https://datatracker.ietf.org/wg/quic/charter/>, "QUIC
(quic)".
- [RFC1945] Berners-Lee, T., Fielding, R., and H. Frystyk, "Hypertext
Transfer Protocol -- HTTP/1.0", RFC 1945,
DOI 10.17487/RFC1945, May 1996,
<<https://www.rfc-editor.org/info/rfc1945>>.
- [RFC1958] Carpenter, B., Ed., "Architectural Principles of the
Internet", RFC 1958, DOI 10.17487/RFC1958, June 1996,
<<https://www.rfc-editor.org/info/rfc1958>>.
- [RFC1984] IAB and IESG, "IAB and IESG Statement on Cryptographic
Technology and the Internet", BCP 200, RFC 1984,
DOI 10.17487/RFC1984, August 1996,
<<https://www.rfc-editor.org/info/rfc1984>>.
- [RFC2131] Droms, R., "Dynamic Host Configuration Protocol",
RFC 2131, DOI 10.17487/RFC2131, March 1997,
<<https://www.rfc-editor.org/info/rfc2131>>.
- [RFC2474] Nichols, K., Blake, S., Baker, F., and D. Black,
"Definition of the Differentiated Services Field (DS
Field) in the IPv4 and IPv6 Headers", RFC 2474,
DOI 10.17487/RFC2474, December 1998,
<<https://www.rfc-editor.org/info/rfc2474>>.

- [RFC2504] Guttman, E., Leong, L., and G. Malkin, "Users' Security Handbook", FYI 34, RFC 2504, DOI 10.17487/RFC2504, February 1999, <<https://www.rfc-editor.org/info/rfc2504>>.
- [RFC2663] Srisuresh, P. and M. Holdrege, "IP Network Address Translator (NAT) Terminology and Considerations", RFC 2663, DOI 10.17487/RFC2663, August 1999, <<https://www.rfc-editor.org/info/rfc2663>>.
- [RFC2775] Carpenter, B., "Internet Transparency", RFC 2775, DOI 10.17487/RFC2775, February 2000, <<https://www.rfc-editor.org/info/rfc2775>>.
- [RFC2804] IAB and IESG, "IETF Policy on Wiretapping", RFC 2804, DOI 10.17487/RFC2804, May 2000, <<https://www.rfc-editor.org/info/rfc2804>>.
- [RFC2827] Ferguson, P. and D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing", BCP 38, RFC 2827, DOI 10.17487/RFC2827, May 2000, <<https://www.rfc-editor.org/info/rfc2827>>.
- [RFC3135] Border, J., Kojo, M., Griner, J., Montenegro, G., and Z. Shelby, "Performance Enhancing Proxies Intended to Mitigate Link-Related Degradations", RFC 3135, DOI 10.17487/RFC3135, June 2001, <<https://www.rfc-editor.org/info/rfc3135>>.
- [RFC3315] Droms, R., Ed., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, DOI 10.17487/RFC3315, July 2003, <<https://www.rfc-editor.org/info/rfc3315>>.
- [RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, DOI 10.17487/RFC3550, July 2003, <<https://www.rfc-editor.org/info/rfc3550>>.
- [RFC3724] Kempf, J., Ed., Austein, R., Ed., and IAB, "The Rise of the Middle and the Future of End-to-End: Reflections on the Evolution of the Internet Architecture", RFC 3724, DOI 10.17487/RFC3724, March 2004, <<https://www.rfc-editor.org/info/rfc3724>>.
- [RFC3954] Claise, B., Ed., "Cisco Systems NetFlow Services Export Version 9", RFC 3954, DOI 10.17487/RFC3954, October 2004, <<https://www.rfc-editor.org/info/rfc3954>>.

- [RFC3971] Arkko, J., Ed., Kempf, J., Zill, B., and P. Nikander, "SECure Neighbor Discovery (SEND)", RFC 3971, DOI 10.17487/RFC3971, March 2005, <<https://www.rfc-editor.org/info/rfc3971>>.
- [RFC4787] Audet, F., Ed. and C. Jennings, "Network Address Translation (NAT) Behavioral Requirements for Unicast UDP", BCP 127, RFC 4787, DOI 10.17487/RFC4787, January 2007, <<https://www.rfc-editor.org/info/rfc4787>>.
- [RFC4862] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, DOI 10.17487/RFC4862, September 2007, <<https://www.rfc-editor.org/info/rfc4862>>.
- [RFC5655] Trammell, B., Boschi, E., Mark, L., Zseby, T., and A. Wagner, "Specification of the IP Flow Information Export (IPFIX) File Format", RFC 5655, DOI 10.17487/RFC5655, October 2009, <<https://www.rfc-editor.org/info/rfc5655>>.
- [RFC5965] Shafranovich, Y., Levine, J., and M. Kucherawy, "An Extensible Format for Email Feedback Reports", RFC 5965, DOI 10.17487/RFC5965, August 2010, <<https://www.rfc-editor.org/info/rfc5965>>.
- [RFC6108] Chung, C., Kasyanov, A., Livingood, J., Mody, N., and B. Van Lieu, "Comcast's Web Notification System Design", RFC 6108, DOI 10.17487/RFC6108, February 2011, <<https://www.rfc-editor.org/info/rfc6108>>.
- [RFC6235] Boschi, E. and B. Trammell, "IP Flow Anonymization Support", RFC 6235, DOI 10.17487/RFC6235, May 2011, <<https://www.rfc-editor.org/info/rfc6235>>.
- [RFC6269] Ford, M., Ed., Boucadair, M., Durand, A., Levis, P., and P. Roberts, "Issues with IP Address Sharing", RFC 6269, DOI 10.17487/RFC6269, June 2011, <<https://www.rfc-editor.org/info/rfc6269>>.
- [RFC6430] Li, K. and B. Leiba, "Email Feedback Report Type Value: not-spam", RFC 6430, DOI 10.17487/RFC6430, November 2011, <<https://www.rfc-editor.org/info/rfc6430>>.
- [RFC6455] Fette, I. and A. Melnikov, "The WebSocket Protocol", RFC 6455, DOI 10.17487/RFC6455, December 2011, <<https://www.rfc-editor.org/info/rfc6455>>.

- [RFC6590] Falk, J., Ed. and M. Kucherawy, Ed., "Redaction of Potentially Sensitive Data from Mail Abuse Reports", RFC 6590, DOI 10.17487/RFC6590, April 2012, <<https://www.rfc-editor.org/info/rfc6590>>.
- [RFC6591] Fontana, H., "Authentication Failure Reporting Using the Abuse Reporting Format", RFC 6591, DOI 10.17487/RFC6591, April 2012, <<https://www.rfc-editor.org/info/rfc6591>>.
- [RFC6650] Falk, J. and M. Kucherawy, Ed., "Creation and Use of Email Feedback Reports: An Applicability Statement for the Abuse Reporting Format (ARF)", RFC 6650, DOI 10.17487/RFC6650, June 2012, <<https://www.rfc-editor.org/info/rfc6650>>.
- [RFC6651] Kucherawy, M., "Extensions to DomainKeys Identified Mail (DKIM) for Failure Reporting", RFC 6651, DOI 10.17487/RFC6651, June 2012, <<https://www.rfc-editor.org/info/rfc6651>>.
- [RFC6652] Kitterman, S., "Sender Policy Framework (SPF) Authentication Failure Reporting Using the Abuse Reporting Format", RFC 6652, DOI 10.17487/RFC6652, June 2012, <<https://www.rfc-editor.org/info/rfc6652>>.
- [RFC7011] Claise, B., Ed., Trammell, B., Ed., and P. Aitken, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information", STD 77, RFC 7011, DOI 10.17487/RFC7011, September 2013, <<https://www.rfc-editor.org/info/rfc7011>>.
- [RFC7012] Claise, B., Ed. and B. Trammell, Ed., "Information Model for IP Flow Information Export (IPFIX)", RFC 7012, DOI 10.17487/RFC7012, September 2013, <<https://www.rfc-editor.org/info/rfc7012>>.
- [RFC7143] Chadalapaka, M., Satran, J., Meth, K., and D. Black, "Internet Small Computer System Interface (iSCSI) Protocol (Consolidated)", RFC 7143, DOI 10.17487/RFC7143, April 2014, <<https://www.rfc-editor.org/info/rfc7143>>.
- [RFC7146] Black, D. and P. Koning, "Securing Block Storage Protocols over IP: RFC 3723 Requirements Update for IPsec v3", RFC 7146, DOI 10.17487/RFC7146, April 2014, <<https://www.rfc-editor.org/info/rfc7146>>.

- [RFC7230] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", RFC 7230, DOI 10.17487/RFC7230, June 2014, <<https://www.rfc-editor.org/info/rfc7230>>.
- [RFC7234] Fielding, R., Ed., Nottingham, M., Ed., and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Caching", RFC 7234, DOI 10.17487/RFC7234, June 2014, <<https://www.rfc-editor.org/info/rfc7234>>.
- [RFC7258] Farrell, S. and H. Tschofenig, "Pervasive Monitoring Is an Attack", BCP 188, RFC 7258, DOI 10.17487/RFC7258, May 2014, <<https://www.rfc-editor.org/info/rfc7258>>.
- [RFC7348] Mahalingam, M., Dutt, D., Duda, K., Agarwal, P., Kreeger, L., Sridhar, T., Bursell, M., and C. Wright, "Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks", RFC 7348, DOI 10.17487/RFC7348, August 2014, <<https://www.rfc-editor.org/info/rfc7348>>.
- [RFC7435] Dukhovni, V., "Opportunistic Security: Some Protection Most of the Time", RFC 7435, DOI 10.17487/RFC7435, December 2014, <<https://www.rfc-editor.org/info/rfc7435>>.
- [RFC7457] Sheffer, Y., Holz, R., and P. Saint-Andre, "Summarizing Known Attacks on Transport Layer Security (TLS) and Datagram TLS (DTLS)", RFC 7457, DOI 10.17487/RFC7457, February 2015, <<https://www.rfc-editor.org/info/rfc7457>>.
- [RFC7489] Kucherawy, M., Ed. and E. Zwicky, Ed., "Domain-based Message Authentication, Reporting, and Conformance (DMARC)", RFC 7489, DOI 10.17487/RFC7489, March 2015, <<https://www.rfc-editor.org/info/rfc7489>>.
- [RFC7498] Quinn, P., Ed. and T. Nadeau, Ed., "Problem Statement for Service Function Chaining", RFC 7498, DOI 10.17487/RFC7498, April 2015, <<https://www.rfc-editor.org/info/rfc7498>>.
- [RFC7525] Sheffer, Y., Holz, R., and P. Saint-Andre, "Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", BCP 195, RFC 7525, DOI 10.17487/RFC7525, May 2015, <<https://www.rfc-editor.org/info/rfc7525>>.

- [RFC7540] Belshé, M., Peon, R., and M. Thomson, Ed., "Hypertext Transfer Protocol Version 2 (HTTP/2)", RFC 7540, DOI 10.17487/RFC7540, May 2015, <<https://www.rfc-editor.org/info/rfc7540>>.
- [RFC7619] Smyslov, V. and P. Wouters, "The NULL Authentication Method in the Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 7619, DOI 10.17487/RFC7619, August 2015, <<https://www.rfc-editor.org/info/rfc7619>>.
- [RFC7624] Barnes, R., Schneier, B., Jennings, C., Hardie, T., Trammell, B., Huitema, C., and D. Borkmann, "Confidentiality in the Face of Pervasive Surveillance: A Threat Model and Problem Statement", RFC 7624, DOI 10.17487/RFC7624, August 2015, <<https://www.rfc-editor.org/info/rfc7624>>.
- [RFC7665] Halpern, J., Ed. and C. Pignataro, Ed., "Service Function Chaining (SFC) Architecture", RFC 7665, DOI 10.17487/RFC7665, October 2015, <<https://www.rfc-editor.org/info/rfc7665>>.
- [RFC7754] Barnes, R., Cooper, A., Kolkman, O., Thaler, D., and E. Nordmark, "Technical Considerations for Internet Service Blocking and Filtering", RFC 7754, DOI 10.17487/RFC7754, March 2016, <<https://www.rfc-editor.org/info/rfc7754>>.
- [RFC7799] Morton, A., "Active and Passive Metrics and Methods (with Hybrid Types In-Between)", RFC 7799, DOI 10.17487/RFC7799, May 2016, <<https://www.rfc-editor.org/info/rfc7799>>.
- [RFC7826] Schulzrinne, H., Rao, A., Lanphier, R., Westerlund, M., and M. Stiemerling, Ed., "Real-Time Streaming Protocol Version 2.0", RFC 7826, DOI 10.17487/RFC7826, December 2016, <<https://www.rfc-editor.org/info/rfc7826>>.
- [RFC7838] Nottingham, M., McManus, P., and J. Reschke, "HTTP Alternative Services", RFC 7838, DOI 10.17487/RFC7838, April 2016, <<https://www.rfc-editor.org/info/rfc7838>>.
- [RFC7858] Hu, Z., Zhu, L., Heidemann, J., Mankin, A., Wessels, D., and P. Hoffman, "Specification for DNS over Transport Layer Security (TLS)", RFC 7858, DOI 10.17487/RFC7858, May 2016, <<https://www.rfc-editor.org/info/rfc7858>>.

- [RFC8073] Moriarty, K. and M. Ford, "Coordinating Attack Response at Internet Scale (CARIS) Workshop Report", RFC 8073, DOI 10.17487/RFC8073, March 2017, <<https://www.rfc-editor.org/info/rfc8073>>.
- [RFC8164] Nottingham, M. and M. Thomson, "Opportunistic Security for HTTP/2", RFC 8164, DOI 10.17487/RFC8164, May 2017, <<https://www.rfc-editor.org/info/rfc8164>>.
- [RFC8165] Hardie, T., "Design Considerations for Metadata Insertion", RFC 8165, DOI 10.17487/RFC8165, May 2017, <<https://www.rfc-editor.org/info/rfc8165>>.
- [RFC8250] Elkins, N., Hamilton, R., and M. Ackermann, "IPv6 Performance and Diagnostic Metrics (PDM) Destination Option", RFC 8250, DOI 10.17487/RFC8250, September 2017, <<https://www.rfc-editor.org/info/rfc8250>>.
- [RFC8274] Kampanakis, P. and M. Suzuki, "Incident Object Description Exchange Format Usage Guidance", RFC 8274, DOI 10.17487/RFC8274, November 2017, <<https://www.rfc-editor.org/info/rfc8274>>.
- [RFC8300] Quinn, P., Ed., Elzur, U., Ed., and C. Pignataro, Ed., "Network Service Header (NSH)", RFC 8300, DOI 10.17487/RFC8300, January 2018, <<https://www.rfc-editor.org/info/rfc8300>>.
- [SACM] <https://datatracker.ietf.org/wg/sacm/charter/>, "Security Automation and Continuous Monitoring (sacm) IETF Working Group".
- [Snowden] <http://www.jjsylvia.com/bigdatacourse/wp-content/uploads/2016/04/pl4-verble-1.pdf>, "The NSA and Edward Snowden: Surveillance In The 21st Century", 2014.
- [TCPcrypt] <https://datatracker.ietf.org/wg/tcpinc/charter/>, "TCPcrypt".
- [TLS100Proceedings] IETF 100, TLS Working Group Session, "Presentation before the TLS WG at IETF" <https://datatracker.ietf.org/meeting/100/materials/slides-100-tls-sessa-tls13/>, 2017.
- [TS3GPP] "3GPP TS 24.301, "Non-Access-Stratum (NAS) protocol for Evolved Packet System (EPS); Stage 3"", 2017.

[UPCON] 3GPP, "User Plane Congestion Management
[http://www.3gpp.org/DynaReport/
FeatureOrStudyItemFile-570029.htm](http://www.3gpp.org/DynaReport/FeatureOrStudyItemFile-570029.htm)", 2014.

[UserData] Network and Distributed Systems Symposium, The Internet
Society, "The Security Impact of HTTPS Interception",
2017.

Authors' Addresses

Kathleen Moriarty (editor)
Dell EMC
176 South St
Hopkinton, MA
USA

Phone: +1
Email: Kathleen.Moriarty@dell.com

Al Morton (editor)
AT&T Labs
200 Laurel Avenue South
Middletown,, NJ 07748
USA

Phone: +1 732 420 1571
Fax: +1 732 368 1192
Email: acmorton@att.com

None
Internet-Draft
Intended status: Informational
Expires: April 21, 2018

K. Makhijani, ed
J. Qin
R. Ravindran
Huawei Technologies
L. Geng
China Mobile
L. Qiang
S. Peng
Huawei Technologies
X. de Foy
A. Rahman
InterDigital Inc.
A. Galis
University College London
G. Fioccola
Telecom Italia
October 18, 2017

Network Slicing Use Cases: Network Customization and Differentiated
Services
draft-netslices-usecases-02

Abstract

Network Slicing is meant to enable creating (end-to-end) partitioned network infrastructure that may include the user equipment, access/core transport networks, edge and central data center resources to provide differentiated connectivity behaviors to fulfill the requirements of distinct services, applications and customers. In this context, connectivity is not restricted to differentiated forwarding capabilities but it covers also advanced service functions that will be invoked when transferring data within a given domain.

The purpose of this document is to focus on use cases that benefit from the use of 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 April 21, 2018.

Copyright Notice

Copyright (c) 2017 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
1.1. Requirements Language	3
1.2. Terminology	3
2. Scope	4
3. A Generalized Network Slice as a Service	6
3.1. Resource Centric Service Concept	6
3.2. Strict Resource Demand	7
3.3. Network Customization	7
3.4. NSaaS of Different Granularity	7
3.5. Service customization across multi-provider multi-domains as NSaaS	8
4. Network Slicing in 3GPP Mobile Network	10
4.1. Network Slices in 3GPP Systems	10
4.2. Creating, Managing and Operating 3GPP Network Slices	11
5. Role of Virtualization in Network slicing	12
5.1. Virtualized Customer Premise Equipment	12
5.2. Enhanced Broadband	14
6. Services with Resource Assurance	16
6.1. Massive Machine to Machine Communication	16
6.2. Ultra-reliable Low Latency Communication	18
6.3. Critical Communications	20
7. Network Infrastructure for new technologies	23
7.1. ICN as a Network Slice	23
7.2. New Verticals - ICN based service delivery	24

7.2.1. Required Characteristics	25
8. Overall Use Case Analysis	26
8.1. Requirements Reference	26
8.2. Mapping Common characteristics to Requirements	26
9. Conclusion	28
10. Security Considerations	28
11. IANA Considerations	29
12. Acknowledgements	29
13. References	29
13.1. Normative References	29
13.2. Informative References	30
Authors' Addresses	31

1. Introduction

Network Slicing enables the creation of (end-to-end) partitioned network infrastructure that may include the user equipment, access/core transport networks, edge and central data center resources to provide differentiated connectivity behaviors to fulfill the requirements of distinct services, applications and customers. In this context, connectivity is not restricted to differentiated forwarding capabilities but it also spans service, management and control plane support offered to a slice instance.

1.1. Requirements Language

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

1.2. Terminology

Please refer to [I-D.geng-netslices-architecture] for related terminologies and definitions.

Additionally, the following terms are used:

- o V2X (Vehicle-to-everything): Is a communication of information from a vehicle to any other entity that may be another vehicle, road-side network element or application end point.
- o ITS (Intelligent Transportation Systems): Considered as an aspect of how using Internet of Things resource like road sensors can creates a smart transport network. The network offers services related to transport and traffic management systems through flow of information between road-side sensors, vehicles, smart devices and humans.

- o Over-the-top (OTT): A service, e.g., content delivery using a CDN or a social networking service, operated by a different service providers to which the users of the NSP service are attached to, and to whom it serves as a communication (or bit pipe) provider
- o Industry vertical: A collection of services or tools specific to an industry, trade or market sector. also, referred to as Service Verticals in this document.
- o TETRA: Terrestrial trunked radio is a digital trunked mobile radio standard to meet needs of public safety, transportation and utilities like organizations.
- o SLA: Service Level Agreement - A contract between a service provider and an end user that stipulates a specified level of service, support option, a guaranteed level of system performance as relates to downtime or up-time.

2. Scope

To maximize resource utilization and minimize infrastructure cost, services will need to operate over a shared network infrastructure, as against the traditional monolithic model operated either as dedicated network or as an overlay. Service operators can utilize or benefit from Network Slicing through multi-tenancy, enabling different customized network infrastructures for different group of services across different network domains and operating them independently.

In this document, multi-domain refers to combination of different kinds of connection-technology network domains. For example, it may be a RAN, DSL etc. in access; a fixed, wireless or mobile service provider network; as well as different technology domains, in transport networks such as carrier Ethernet, optical, MPLS, TE-tunnel etc. Often, a combination of technology domains is under the same administrator's control but may also belong to different administrative systems and may require cross-domain coordination.

The document covers generalized as well as resource guaranteed service scenarios that can benefit by applying Network Slicing principles as below in Figure 1

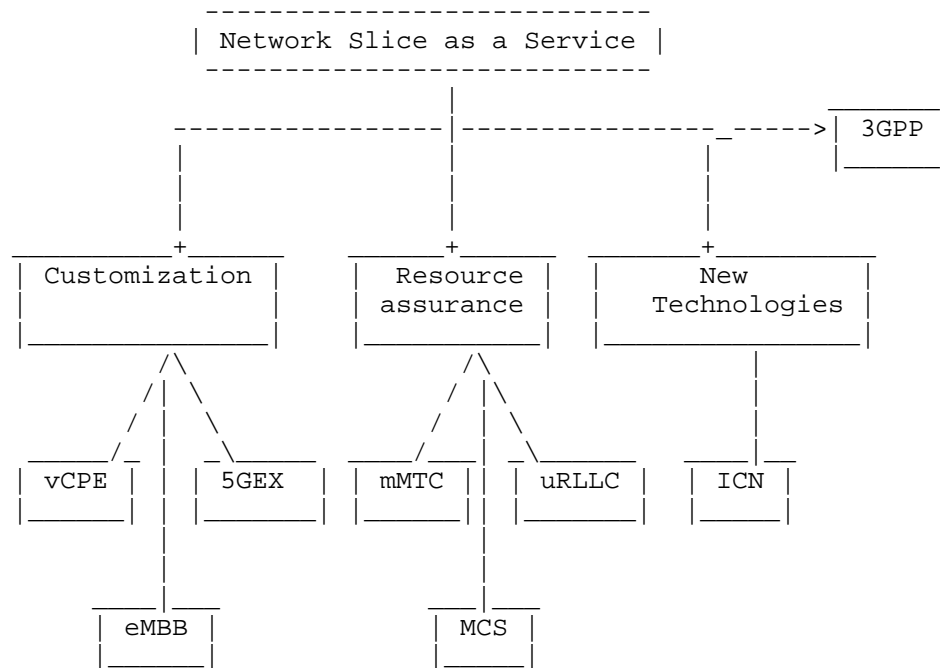


Figure 1: Use case organization in the document

The remaining document is organized as below:

- o In Section 3, Network Slice as a Service(NSaaS) delivery model is described.
- o Section 3.5, is a scenario for multi-domain network slice coordination.
- o In Section 4, 3GPP architecture for 5G is discussed as a use case so that those requirements may be taken into consideration during slicing activities in IETF.
- o Other use cases are discussed from 2 perspectives
 - a Existing scenarios: Several already deployed use cases that would further benefit operators when deployed through Network Slice paradigm are discussed in Section 5.
 - b Differentiated service scenarios: that must absolutely meet strict resource requirements, as if they use a dedicated

infrastructure. The example use cases are categorized in Section 6.

- o Section 7, has an example use case of cases where new technologies can be verified or deployed using network slicing concept.
- o In Section 8, the use case requirements are summarized which are inputs to the [I-D.qiang-netslices-gap-analysis].

3. A Generalized Network Slice as a Service

Network slicing instances share a common infrastructure, which provide flexible design of a logical network with specific network functions customized to support differentiated performance requirements of vertical industry through logical or physical system isolation and certain OAM tools.

Traditionally, vertical industries run their services in a shared network environment upon which infrastructure owners and service providers offer standalone network capabilities including connections, storage and etc. Network slicing paradigm enables supporting the requirements of a network slicing tenant to be met individually. Hence it is anticipated that this type of new business model where network slice instances are leased to industry verticals as a service (i.e. Network Slicing as a Service, NSaaS) may become a norm in the near future.

3.1. Resource Centric Service Concept

Network services specify a set of resource requirements to offer desired Quality of Experience (QoE) to its consumers, using features offered by the control and forwarding planes. Traditional service guarantees are associated with resource attributes such as throughput, packet loss, latency, network bandwidth/burst or other bit rates and security. In addition, redundancy and reliability are provided by the infrastructure to improve overall QoE. More recently, concepts such as edge computing allow opportunistic placement of services to meet stringent requirements of low latency and/or high bandwidth applications.

Clearly the description of service delivery is more diverse now than before and demands higher degree of resource engineering and agility. The motivation behind Network slicing paradigm is to enable new service deployments without having to build new network infrastructures or causing disruptions to already deployed services in the network. In this regard, there are two primary characteristics NS should satisfy, a) Strict demand for network resource, b) Network Customization.

3.2. Strict Resource Demand

Several services are sensitive to response times and/or amount of bandwidth, e.g. real time interactive multimedia, high bandwidth video feed or remote access to an enterprise network. Failure to meet these criteria lead to service degradation. Moreover, new industry verticals are evolving due to technological advancements in sensors, IoT, robotics and multi-media, along with new type of network interactions (both human-human or human-machine). These impose even stricter resource and connectivity requirements. The challenge lies in utilizing common network infrastructure and judiciously allocating available infrastructure resources.

3.3. Network Customization

To a network slice tenant, the ability to customize services dynamically is important. Customization gives control to the operator of a slice to create, provision and change network resources to suit their service demands. Customization enables decomposition of resources from an underlying network infrastructure and logically aggregate them as part of a slice. These customizations also include placement and logical connection of the network functions based on the service requirements.

3.4. NSaaS of Different Granularity

In order to meet various requirements from the network slice tenant, NSaaS should be provided with different granularities. Some typical examples of granularities that a provider may offer are as follows.

- o Network Domain - Network slice instances of different networks i.e. access (wireless, fixed) network, transport network and core network.
- o Access technologies - Network slice instances of different generations of cellular and fixed network technologies, i.e. 4G, WiFi, Passive Optical Network (PON) and DSL.
- o SLA requirements - Network slice instances of different SLA requirements, i.e. low-latency network, legacy best-effort network and network with guaranteed-bandwidth.
- o Vertical applications - Network slice instances of different industry verticals. i.e. manufacturing site, V2X, industrial IoT and smart city.

- o OTT services - Network slice instances of different applications provided by OTT, i.e. messaging, payment, video streaming and gaming.
- o Cross domain services - Network slice instances of different services across multi-provider domains such as L2, L3 VPN services.

During the realization of network slice instance, it is also very important that sub-instance of a more general one can be provided with a finer granularity. In practice, it is up to the provider to decide the granularity to lease the network slice instances.

The customization of different granularities of a network slice introduce many challenges, especially in terms of network management and orchestration. As a network slice provider (provider of end-to-end slice service), it is essential to have a comprehensive understanding of the network capability. This requires that network connectivity and resources can be exposed to the network slice tenants (as the differentiated services). Accordingly, network slice provider is able to orchestrate specific instances based on these exposed capabilities.

3.5. Service customization across multi-provider multi-domains as NSaaS

L2 and L3 connectivity services can be deployed in a multi-provider multi-domain scenario and, in the SDN era, this implies the decoupling of network resources for different service provider and domain orchestrators. The allocation of network resources within the domain of each service provider, involved by the end-to-end service, can be defined as a network slice.

Within a single domain, provider is aware of the entire topology and its own resource availability and has complete control over those resources. However, in a multi domain scenario, the overall knowledge of the resources and topologies cannot be made across providers. Therefore, the exchange of information across these providers have to be enabled, as shown in Figure 2, inspired by [I-D.bernardos-nfvrg-multidomain] and [I-D.ietf-opsawg-service-model-explained].

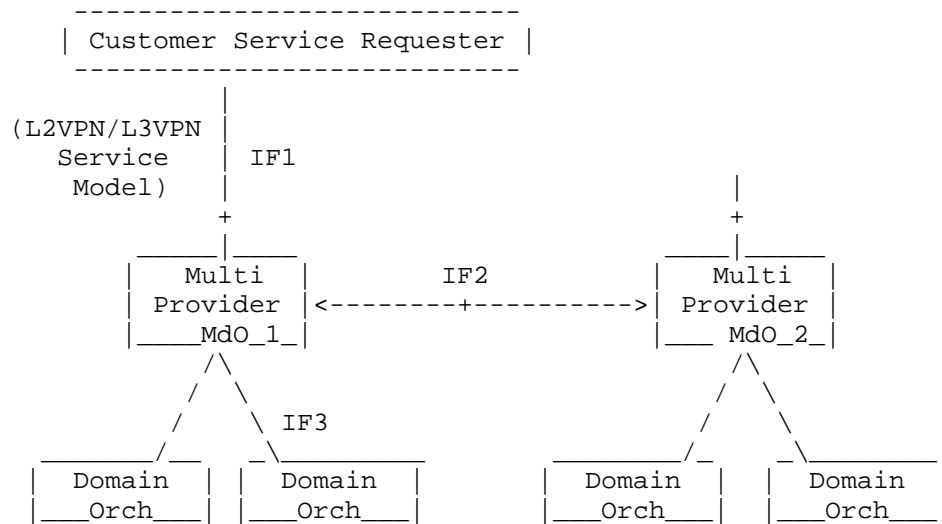


Figure 2: Multi-domain, multi-provider connectivity services

The Figure 2 shows a multi provider MdO (MP-MdO) exposing an interface 1 (IF1) to the tenant, interface 2 (IF2) to other multi provider MdO (multi domain orchestrator) and an interface 3 (IF3) to individual domain orchestrators. IF1 is exposed to the tenant who could request his specific services and/or slices to be deployed. IF2 is between the orchestrators and is a key interface to enable multi-provider operation. IF3 focuses on abstracting the technology or vendor dependent implementation details to support orchestration in each network domain (see [!@I-D.bernardos-nfvrg-multidomain] for details). The coordination alternatives between MP-MdOs are: *

- * Bilateral Cascading: providers can have long-lasting business agreements only with their direct neighbors.
- * Full Mesh between MP-MdOs: Providers can have long-lasting business agreement with any provider (neighboring or remote).

This reference architecture is the main focus of the 5GEx European Project.

Among applications, L2VPN and L3VPN wholesale end-to-end services in a multi-provider and multi-domain scenario needs the following characteristics for network slice management

- o An automatic activation test and verification functionality (by customer or orchestrator).

- o Interface to modify parameters of L2VPN or L3VPN service such as bandwidth or path redundancy.

Looking at Figure 2, the customer needs a new L2 end-to-end service between CPEs across two domains (MP-Md01 and MP-Md02). As MP-Md01 receives the service request, it is deployed as a network slice. In this regards MP-Md01 has prior knowledge of topology and resource across domains in some form, it then splits the service request into a slice across each of the involved domain. Once service is set up: MP-Md01 allocates resources for the slice on SP1 domain while MP-Md02 allocates on SP2 domain respectively.

[I-D.ietf-l2sm-l2vpn-service-model] and [RFC8049] can describe IF1 For L2 and L3 end-to-end services respectively. The ability to map such services as network slice will be considerable opportunity for dynamic cross-domain operations.

4. Network Slicing in 3GPP Mobile Network

Network Slicing is a core capability of the currently under development 3GPP 5G phase 1 mobile system, as it makes it possible for different service verticals, such as IoT and broadband applications, to be deployed over a common shared infrastructure. More details can be found in [TS_3GPP.23.501], [TS_3GPP.23.502], [TR_3GPP.38.801], [TR_3GPP.33.899] and [TS_3GPP.28.500].

3GPP is currently defining its own solution for network slicing. An IETF effort in this field may, however, still be complementary in the long run as IETF focuses on the IP infrastructure and protocols which are generally out of scope of 3GPP. Challenges relevant to the IETF include isolation between network slices, supporting sharing network functions between several slices, building slices recursively from smaller slice subnets, implementing slicing across different domains for roaming, etc.

4.1. Network Slices in 3GPP Systems

In 3GPP systems a network slice is a complete logical network which provides telecommunication services and network capabilities. Distinct Radio Access Network (RAN) slices and core network slices interwork to provide mobile connectivity. A device may access multiple NS simultaneously through a single RAN. 3GPP defines slice IDs (NSSAI) composed of a Slice Service Type (SST) and a Slice Differentiator (SD). SST refers to an expected network behavior in terms of features and services (e.g. specialized for broadband or massive IoT), while SD helps distinguishing among several NS instances.

Figure 3 describes the general layout of Network Slicing in mobile networks. A core network slice includes, a Session Management Function (SMF), which manages PDU sessions, and a User Plane Function (UPF). Some functions such as the Access and Mobility management Function (AMF) are common and shared between multiple RAN and core network slices.

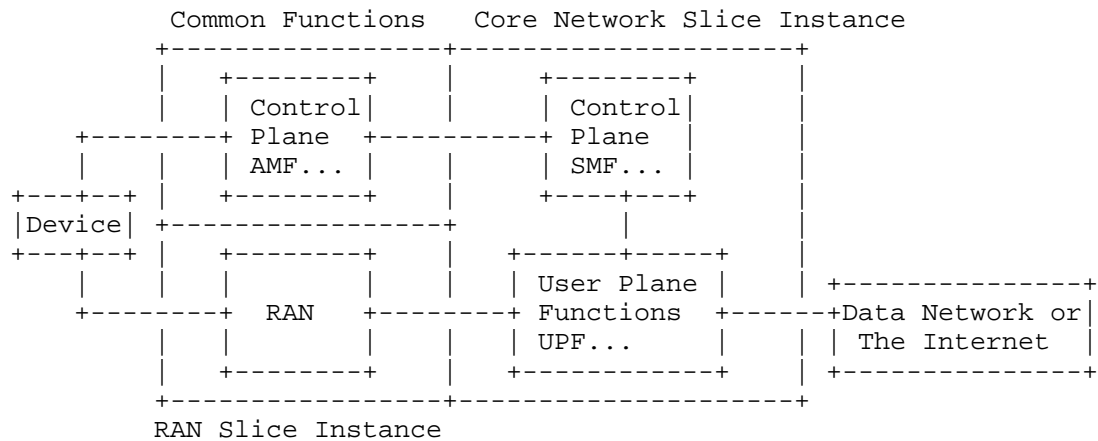


Figure 3: 3GPP Network Slices

4.2. Creating, Managing and Operating 3GPP Network Slices

To create a network slice instance, mobile network operators define "Network Slice Subnets" into OSS/BSS management system. NS subnets are NS components including NFs and reserved network resources. OSS/BSS communicates with the orchestrator, which, through the rest of the NFV-MANO system, configures compute and network elements to create, compose and activate slices.

Mobile network operators can modify the configuration of a RAN or core network slice, while it is in use. To support this, the operator needs to measure QoS/SLA data for hosted network services, and associate results with the relevant network slice. Example of operations include increase or decrease network capacity or compute capacity of NFs; update the configuration of NFs; add, replace or remove a NFs or a Network Slice Subnet.

Slice selection occurs in 2 phases: first, common functions (including AMF) and available network slices are pre-selected when the device registers with the network. Later on, the network dynamically selects network slices when a device initiates

communication, based on a slice ID associated with the application (on the device) that requests a new flow.

5. Role of Virtualization in Network slicing

Virtualization is a key enabler of network slices; Many network services can be easily deployed using components of NFV framework like network functions, hardware decoupling and resource placement [1]. When deployed as a network slice, the resources associated with virtualized network services are managed uniformly by network slice provider. One such use case is described below.

5.1. Virtualized Customer Premise Equipment

A CPE is an equipment that connects the customer premises to the provider's network. A CPE may either be a layer-2 or a layer-3 device (the routing gateway) performing different network functions depending on the access technology (DSL modem, PON modem, etc.). Any services provided such as Internet access, IPTV, VoIP, etc. or network functions for example, local NAT, local DHCP, IGMP proxy-routing, PPP sessions, routing, etc. are also part of CPE. The installation of different on-premise devices, entails a high cost for service providers in terms of both initial installation and operational support, since they are typically responsible for the end-to-end service.

Traditional CPE deployments are service provider network functions installed on customer site to provide above mentioned functionalities along with remote site connectivity. Communication Service provider (CSP) is responsible for management and administration of connections and state with proper policy, bandwidth, security and QoS requirements.

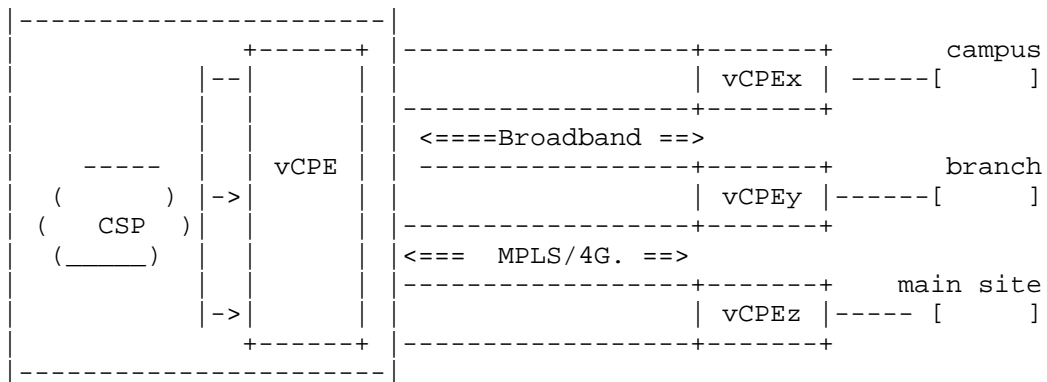


Figure 4: Virtualized CPE with distributed architecture

Figure 4 shows a virtualized architecture in which many functions are moved to CSP's cloud simplifying CPE on premises tremendously. Additional details of deployment architecture models are captured in [I-D.pularikkal-virtual-cpe] where full dissemination of data path and control plane functions is described. The figure shows vCPE_x, vCPE_y, vCPE_z are virtualized CPEs on multiple sites of a specific customer, there may be set of different network functions in each x, y and z CPE. The vCPE instance in CSP cloud is integrated to each site performing service chains of network functions and resource allocations specific for ingress and egress path of each site.

A vCPE is a well-known concept[VCPEBBF] which when combined with WAN technologies provides end to end visibility and reachability to remote sites. However, there is no standard approach to connectivity or management of various CPE functions. Using network slicing, a greater level of agility can be achieved, with each customer dynamically managing its own network with the assistance of network slicing framework.

The benefit of self-managing a vCPE network slice is the capability to move network functions on premise of to the cloud. An obvious use case will be customer initiated gradual migration of network functions from a site to CSP cloud.

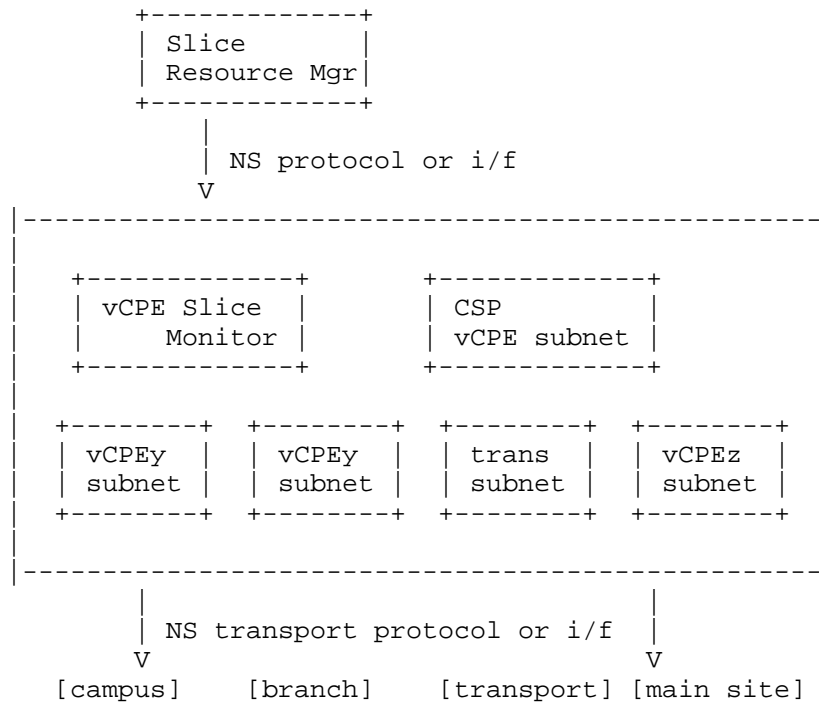


Figure 5: vCPE as a Network Slice

In Figure 5, a slice for vCPE is shown. Using slice subnet approach, each vCPE site instance may be considered as an abstracted subnet, along with the WAN transport as another subnet. The network functions are chained in a distributed fashion between site vCPEs and CSP vCPE subnet. A monitoring function interfaces with CSP's global slice manager for resource management. A south-bound interface through network slice transport protocol, realizes these functions on the infrastructure.

5.2. Enhanced Broadband

Today, video consumes the largest amount of bandwidth over the Internet. As the higher resolution formats enter mainstream, even more bandwidth will be needed to stream 4K/8K/360 degree formats. For example, connected Virtual Reality(VR)/Augmented Reality(AR) is the future use case of eMBB services. Notably, media processing for AR/VR will require in-network processing functions and high latencies between components could lead to downgrade of user experience. Therefore, an AR/VR stream requires a special infrastructure that differs from best-effort network.

A purpose-built network slice for eMBB streaming shall ensure to minimize processing overheads, it may be done by placement of network functions closer to subscribers. Resource scaling for eMBB should be dynamic because bandwidth is expensive and such vertical service operators may not want to pay for unutilized bandwidth. Therefore, slices should be able to monitor, negotiate and adjust the scale for both bandwidth and service functions. Latency guarantees vary from general services, therefore, as a first step, monitoring for quality of service is needed and more advanced operation would involve recovery and reparation of paths.

A typical eMBB slice Figure 6 from a network operator is a performance oriented service customization. An eMBB service slice template will allow a tenant to request or specify

- (1) CDN components (as service functions)
 - * Regional network locations of CDN, encoders etc.
 - * Location of acquired content.
 - * Describes transport constraints for its own distribution network comprising of connectivity between content acquisition and Fan-out points.
- (2) An interface to subscriber database perhaps as a network function, from multiple access network types (cellular, fixed).
- (3) Live performance monitoring and resource negotiation loop.
- (4) A well-coordinated network slice protocol that enables resource allocation across different network domains.

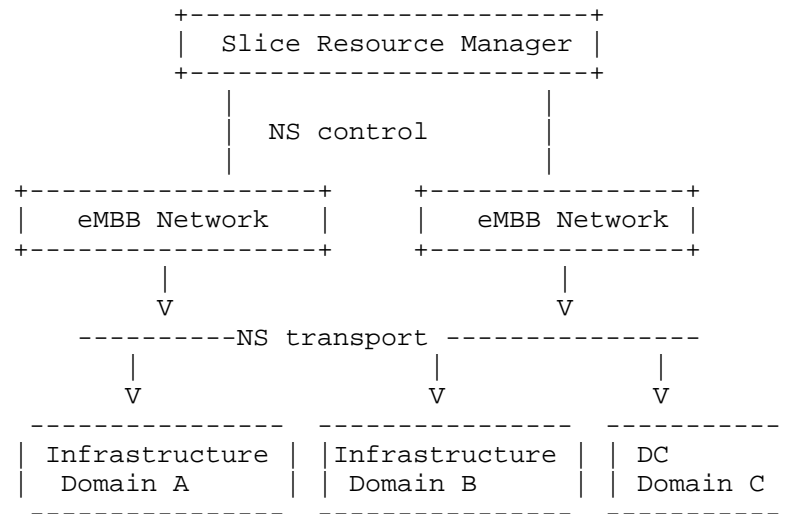


Figure 6: Transport provider network operator view.

6. Services with Resource Assurance

6.1. Massive Machine to Machine Communication

Sensor networks are widely deployed in industries such as agriculture, environmental monitoring and manufacturing. The general workflow of wireless sensor network is provided in Figure 7.

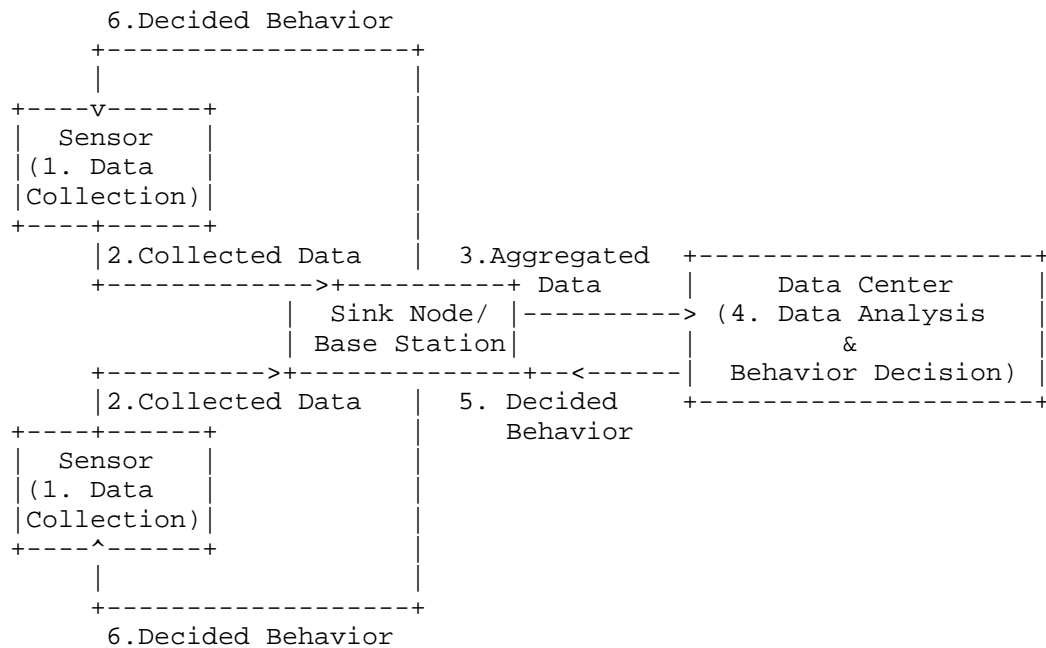


Figure 7: Workflow of wireless sensor network

Figure 7 shows, control of sensor data & behavior at scale, requiring wide area coverage and power constrained communication. A few new types of scenarios that require unique infrastructure are:

- o Smart city networks: an integration of several public infrastructures together through M2M communications. For example Automatic metering (for gas, energy, water, etc.), environment monitoring (for pollution, temperature, humidity, etc.), traffic signal control etc.
- o E-health communications that remote monitor the physical conditions (e.g., heart rate, pulse, blood pressure etc.), and accordingly take necessary measures remotely. E-health communication network must be secure, reliable and fast but small-size of data exchange.

mMTC Type Slices involves potentially a large number of small and power-constrained devices, therefore, resource allocation at scale is of particular importance in mMTC type slices. Furthermore, different kind of IoT devices may exhibit delay sensitivity in industry operations etc. The mMTC type slices should be conscious of requirements of scale, variable data pattern, and energy efficient communications.

6.2. Ultra-reliable Low Latency Communication

In uRLLC scenarios, data loss is not acceptable. Both data and control planes may require significant enhancements to transmission or information distribution protocols. [TR_3GPP_38.913] specifies access network user plane latency as 1ms and reliability factor of 99.999% for transmission of a packet of size 32 bytes. The slices of this type must be ensured that shared infrastructure absolutely does not cause any adverse effects.

In the following sections three new uRLLC scenarios are described.

- (1) Industrial operation: Operations in remote sites usually need combined support of cellular and transport network. Operational accuracy is characterized by
 - * Requires high-quality communication links between the control site.
 - * Low latency and low jitter in communication path
 - * Closed control loop (Sensor -Controller - Actuator) as shown in Figure 8, a typical control cycle time where network is involved should be below 10ms [Tactile-Internet].

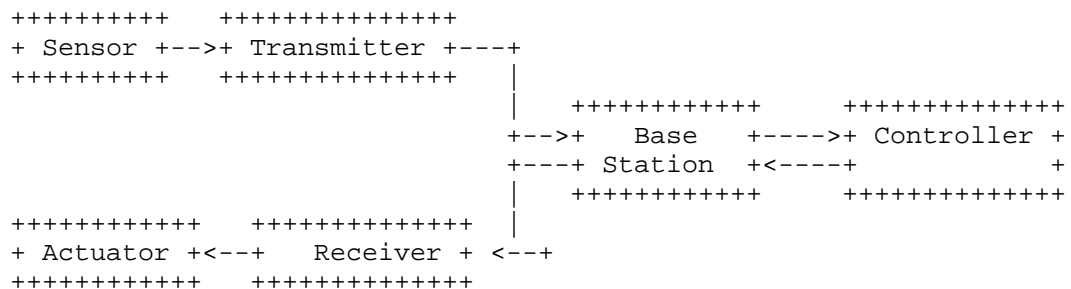


Figure 8: Industrial closed control loop

- (2) Remote surgery enables surgeons to perform critical specialized medical procedures remotely, providing accurate control and haptic feedback.

A uRLLC network slice only accepts service specific traffic and must not receive any other type of traffic to avoid negative impact on the service operation. Capabilities required by uRLLC service provider include

- o Locations of the access nodes for terminals (devices, vehicles) to the transport network and locations of the controller to construct its own network topology within the network slice. In high mobility scenario such as automotive verticals, the dynamic topology adjustments are required without loss of data.
- o Each service vertical has different performance requirements in terms of latency, reliability and data rate etc., therefore, the uRLLC network slice should allow customization for these parameters.
- o A uRLLC service provider should be able to registers self with access rights to resource monitoring and negotiation loop.

A network slice provider offers a uRLLC Slice with the following considerations

- o Should support/provide specific data and control planes protocols with significant enhancements for deterministic latency and reliability (e.g. DetNet[I-D.dt-detnet-dp-sol] in data plane).
- o Allow uRLLC service operator to access user admission and authentication to its network slice in advance.
- o The network coverage for a uRLLC service provisioning may be limited to a confined area, either indoor or outdoor, network operator needs to be able to coordinate resource allocation across different access types and network domains.

A high-level Figure 9, shows a uRLLC slice provider and service view of the network. The monitoring of resources is done in the context of performance. A performance degradation would require resource adjustment. As shown in Figure 9, in one possible sliced model will have its own customizer that uses internal performance observing logic with in its slice by coordinating with different subnets/ domains using southbound NS transport protocol and transfers this information to operator via a northbound NS protocol for resource adjustment.

It is implied that domains maybe different access technologies and need for a common performance metric propagation and resource allocation is important for a uRLLC slice to function properly.

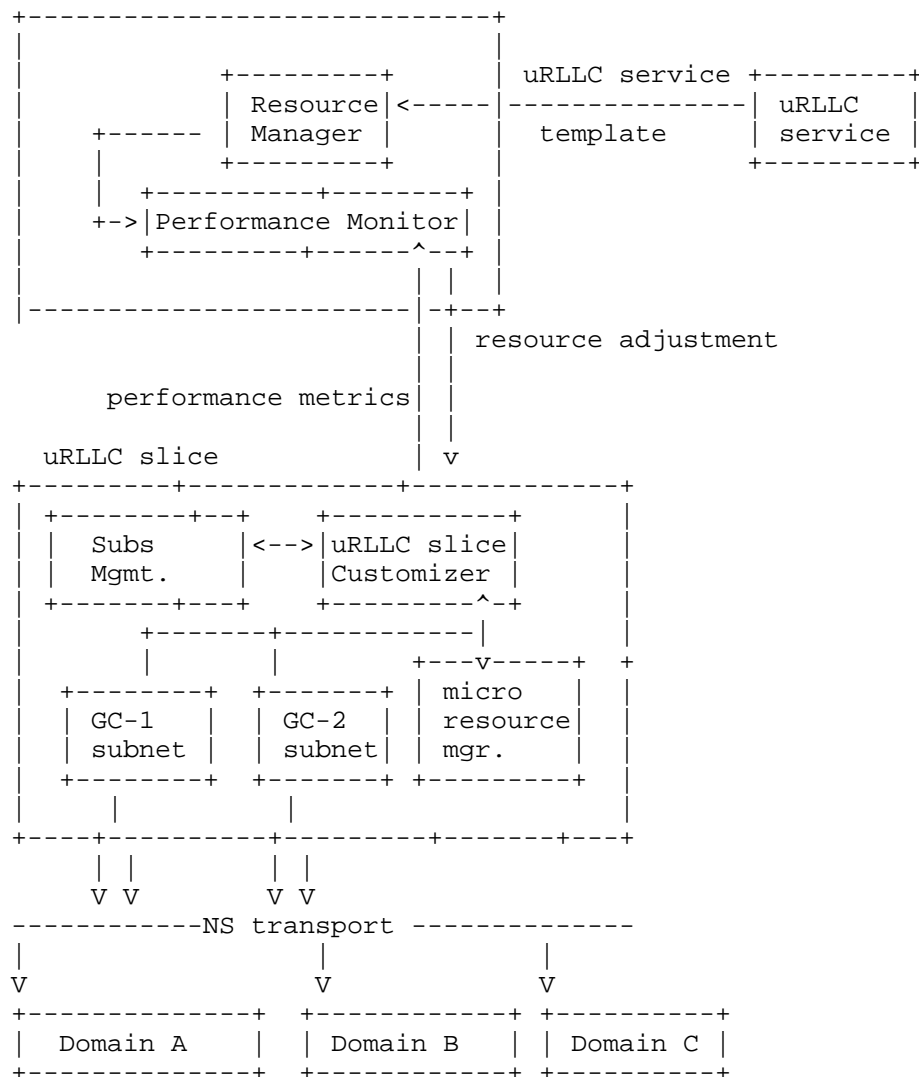


Figure 9: Reference for uRLLC Network Slice.

6.3. Critical Communications

Critical communications are associated with emergency situations. Often referred to as mission critical, the communication has to be reliable and non-disruptive. Different scenarios of critical communications relate to public safety responders (e.g. fire fighters, paramedics), military, utility or commercial applications,

mainly using reliable voice or short data messaging over wireless communication systems.

Next-generation public safety communications are planned to be built with enhanced broadband voice, data and video communications services beyond narrowband LMR with broadband LTE networks for high speed data (ref 22.179 and FirstNet).

3GPP defined on-network critical communication can be established both via (a) over the network infrastructure to manage the call, (b) off-network, where the terminals communicate directly to each other. In the network slicing context, over the network, involves transport networks for an always available, reliable, and zero packet loss quality of traffic support to meet critical services requirements.

Maintaining a separate broadband infrastructure for critical communications incurs a heavy deployment cost. Especially, as the coverage of this separate network has to be extended to large-scale nationwide geographies and remain interoperable is too expensive. As new communication technologies emerge, public safety systems will have to bear the state of the art adoption cost. A separate infrastructure lacks flexibility to add new value-added services or to take advantage of available commercial services.

While shared infrastructure, brings out challenges of these kind:

- (1) Reliable support: Of basic mission critical services: Such as loss of information in voice communication is not acceptable in emergency services, if common infrastructure is to be used, it must assure no loss of information.
- (2) Zero congestion: It is not acceptable for critical calls to be delayed at call setup times or be subjected to any other congestion scenarios.

Having the Mission Critical Service (MCS) as a network slice benefit from the following:

- o Insertion and authorization of subscribers in a group communication: In a critical infrastructure, the subscriber authentication may be done earlier at the entry point automatically through slice selection functional entity.
- o Pre-allocated QoS Class Identifiers (QCIs): Generally, QCIs are requested on per session basis which could slow down overall call control setup and is undesirable for emergency services. When operating in a slice, these resources maybe reserved ahead of time in a coarse-grained manner instead of per session.

MCS network slices are relatively straight forward as it only concerns with guaranteed bit rate (GBR) on per media basis and management of groups. From transport they should be able to request transport services based on GBR for reliable communication. A reference network slice in Figure 10 below, shows a mission critical (MC) organization providing service agreement through a network slice template with resource specification. The MCS slice sets up different subnetworks of different subscriber groups and manages its membership. These subnets are realized into the infrastructure across different domains through a network slice transport mechanism. The MCS must be capable of active resource monitoring to prevent congestions to ever occur as well as request additional transport resources in case of emergency event occurrence.

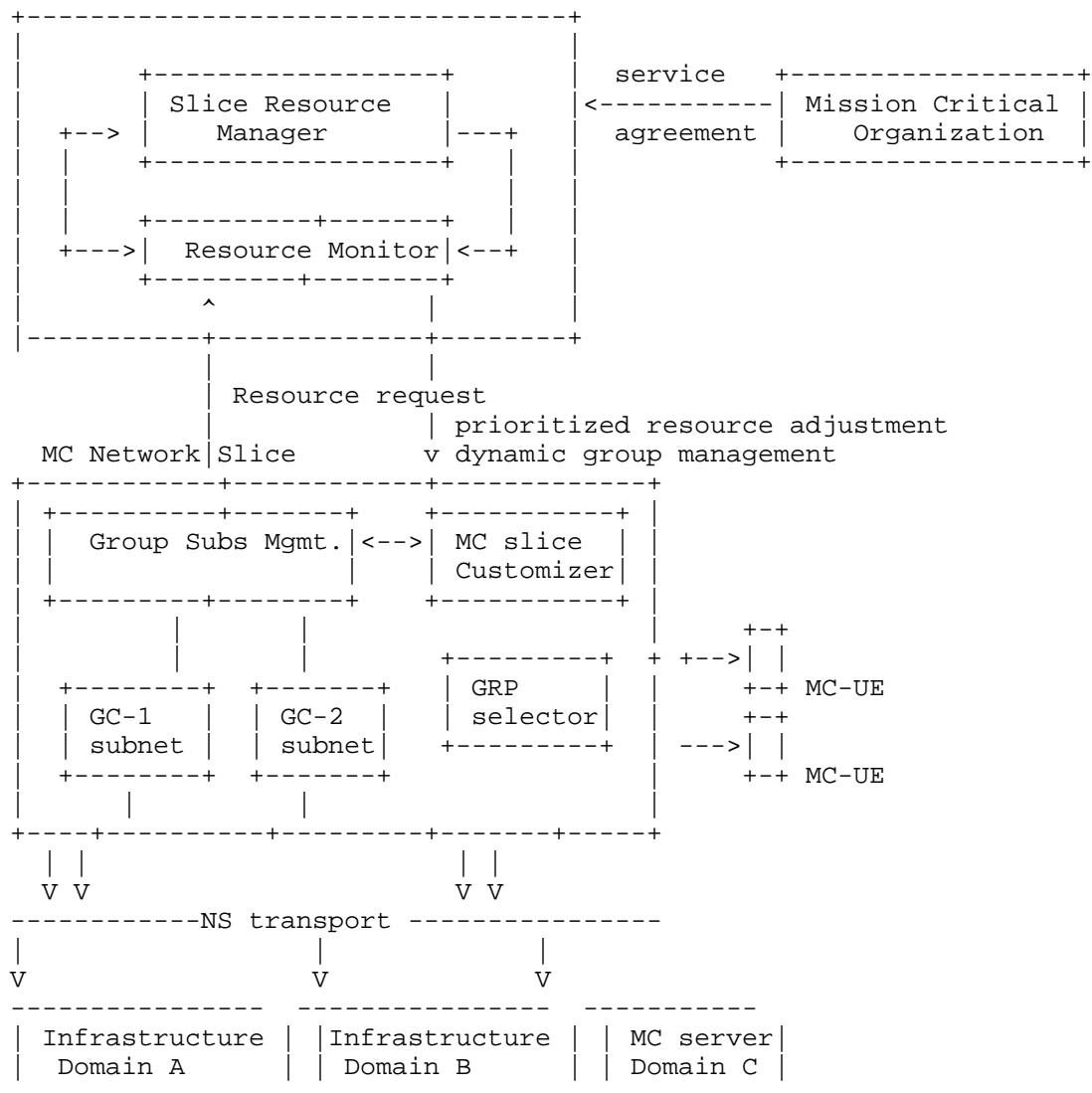


Figure 10: Reference for Mission Critical Network Slice.

7. Network Infrastructure for new technologies

7.1. ICN as a Network Slice

ICN as in Information-Centric Networking is a culmination of multiple future Internet research efforts in various parts of the world, now being pursued under IRTF's research task group called [ICNRG].

Information-Centric Networking (ICN) addresses Internet's network architectural design gaps based on evolving applications requirements and end user behavior that is significantly different from what IP was designed for - which was optimized for host-to-host communication paradigm. ICN is a non-IP paradigm based on name-based routing and offers many desirable networking features to applications such as naming, security, caching, mobility, multicasting and computing in a manner different from traditional host-centric communication model. ICN's name-based abstraction to application minimizes bootstrap configuration from the network, making it suitable to several communication modalities such as multi-point-to-multi-point, AR/VR, D2D and Ad hoc communication.

7.2. New Verticals - ICN based service delivery

Services over ICN slices can take advantage of its features such as:

- (1) In ICN, applications, services and content are addressed using names, hence end host resolution services like DNS can be avoided, this achieves name resolution to edge content or services without incurring additional RTT delays.
- (2) Service flows will be offered mobility and multicasting support, as the networking is session-less and optimized towards efficient movement of named data or networking named services and host level communication.
- (3) Services can be deployed at the very edges with ease as ICN routers are compute friendly, this is because states in the forwarding table can be that of either content or service resources.
- (4) Further saving bandwidth in the upstream link through opportunistic caching is an inherent feature of ICN, this also leads to energy efficient networking.

When offered as a programmable and customizable logical network slice, ICN based services can be offered as a network slice in parallel with traditional IP based services. ICN can be realized as a slice [_5GICN_] based on the choice of data plane resource offered by the operators in different domains of the network such as the access, core network or main data centers. While the same resources can be used to support services over IP, proper resource isolation shall allow it to co-exist with ICN slices as well. ICN slices can be offered over a network slicing framework built upon a programmable pool of software and/or hardware based data plane resources.

7.2.1. Required Characteristics

In ICN, applications use Interest/Data or Get/Put abstractions over named resources resolved by ICN's routing plane. An ICN slice shall be a programmable ICN-domain, in which content learning and distribution will be done using existing or new ICN aware distributed routing logic or through centralized application controllers. As a result, it should be possible to deploy software or hardware based network functions such as ICN routers and content producers and distributors that serve and speak ICN protocols, or enabled through service gateways at the edges of the network. Just as multiple service instances can be part of a slice, an ICN slices can multiplex heterogeneous services; on the other hand an ICN slice can be as granular as a single service instance too. The latter approach has implications with respect to consumer privacy, access control of name data objects, and granularity of mobility handling [_5GICN_].

A basic ICN slice can be manifested as a resource isolated logical network while sharing resources with other connectivity or IP based service slices. An ICN slice relies on programmability and virtualization framework to manage the service slices, to allow maximum flexibility through ICN aware logically centralized control plane for ICN service and slice management.

- o Through a network slice template -ICN service providing entity could specify specific locations (edge of network domains) to deploy ICN-routers or other ICN-NFs (ICN aware network functions). Its service definition varies with the type of service.
- o Application driven connectivity between ICN network elements in all segments and create an ICN based virtual topology.
- o Mechanisms to deliver ICN user traffic over the infrastructure such as overlay or, ICN NFs can be tightly integrated with the RAN such as the eNodeB or implicitly using traffic classification function at the edge and tunneled to ICN User Plane Function (UPF).
- o In addition, bandwidth and other network resources may be requested from the underlay depending on its capability of providing deterministic or statistically guarantees.

How multiple services will be deployed within an ICN aware slice may or may not be exposed to the network operator, depending on if the ICN slices are natively managed by it or a by other service providers.

8. Overall Use Case Analysis

The discussion in above use cases can be summarized as following in terms of the requirements for network slicing framework.

8.1. Requirements Reference

The following functional requirements are derived from discussions in above sections. They are described in details in [I-D.qiang-netslices-gap-analysis] document.

The differentiated services described in this document demonstrate several common functionalities. Therefore, a homogeneous approach towards deployment and management is absolutely necessary.

8.2. Mapping Common characteristics to Requirements

- (1) Resource Reservation: Compute and network resources are reserved as part of initial creation and subsequently during the maintenance of a slice. For example, a service may initially reserve resources for its own control plane, and then later it may reserve user plane flows for applications on demand. Reference use cases: Differentiated services discussed in section "Services with Resource Assurance". A network slice aware infrastructure shall be able to support mechanisms for elastic scaling (up/down) of resources and their non-disruptive provisioning.
- (2) Resource Assurance: A network slice aware infrastructure allows operators to allocate part of the network resources to meet stringent resource characteristics. Scenarios in both Section 5 and Section 6 require on demand and dynamic adjustments. It may not be possible to achieve this using centralize or API approach with finer granularity of resources participating in constrained path computation.
- (3) Multi-dimensional service vertical: Network slicing supports dynamic multi-services, multi-tenancy and the means for backing vertical market players.
- (4) Multi-domain coordination: Multi-domain refers to different technology related network domains. For example, it may be RAN, DSL etc., mobile core network, ISP or different domains in transport networks such as carrier Ethernet, MPLS, TE-tunnel etc. Often, they are under same administrator's control but may require coordination across different administrations. Furthermore, capabilities of each domain must be known in order to validate if a slice can be created or not. All scenarios

mentioned require multi-domain coordination to connect and administer different subnets.

- (5) Operational Isolation: A network slice represents logical group of network resources, functions and corresponding configurations separating its behavior and hence operation from the underlying physical network. Each network slice may have its own operator that sees this slice as a complete network (i.e. with router instances, policies, programmability, placement of virtual network functions according to traffic patterns etc.) and can manage as its own network.
- (6) Transparency: Network slicing does not change the functionality of a scenario; It only facilitates creation of an isolated, an independently run infrastructure for that use case over a common network. Transparency promotes inter-operability and a common resource specification enables it.
- (7) Reliability: It is an important resource attribute in the type of service verticals described above. Many services verticals cannot deliver functionality unless the network is reliable (See remote industry operation, remote surgery and other uRLLC applications). In this regard, monitoring probes are needed of each network slice and resources associated with it.

Requirements Illustrated above	Aggregated Requirements
1) Resource reservation 6) Transparency 4) Multi-access knowledge 3) Multi-dimensional service vertical	Req 1. Network Slicing Specification
4) Multi-Domain coordination 2) Resource Assurance	Req 2. Network Slicing Cross-Domain Coordination
5) Operational/performance Isolation	Req 3. Network Slicing Performance Guarantee and Isolation
7) Reliability	Req 4. Network Slicing OAM

Figure 11: Mapping Common Characteristics to Requirements

NSaaS is a key for network operators to deploy network slices. Having standard means to realize these use cases, enables (a)

different usecases to be uniformly understood by a network slice provider, and (b) similar use cases to be understood in a similar fashion by different network slice providers. Both these cases should allow common mechanisms to map and allocate network slices over the network infrastructure.

Due to the availability of diverse technologies in control and data planes; the first step should be a top-down means to realize a slice with a common technology independent information model. It may describe a resource-centric slice with connectivity, storage, and compute resources, network functions, and operational requirements, that further get mapped to infrastructure resources and capabilities for run-time operations and monitoring. This model may be used by an orchestrator onboarding function for creating instances of network slice services and distributing to network infrastructure providers.

9. Conclusion

A service should typically need a network slice for one of those reasons:

- (1) The service cannot provide optimal experience on a best-effort network.
- (2) It is inefficient and expensive to build a separate infrastructure.

The separation from a generalized network, should allow new services to use newer or different protocols in network, transport and management layer/plane for that service (as in the case of ICN, mMTC, uRLL). The goal of Network slices is to offer enriched service verticals with very different network capability and performance demands but also simplify from the traditional service delivery models.

There is need for a uniform framework for end to end network slicing specifications that spans across multiple technology domains and can drive extensions in those technology-areas for support of Network slices.

10. Security Considerations

The security considerations apply to each kind of slice. In addition general security considerations of underlying infrastructure whether isolated communication with in a slice apply for links using wireless technologies.

11. IANA Considerations

There are no IANA actions requested at this time.

12. Acknowledgements

Note, the 5GEX L2VPN and L3VPN usecase is an independent contribution by authors and is not endorsed by 5GEX. Many thanks to the following reviewers for providing details for several use cases and for helping with the review of the document.

Stewart Bryant (stewart.bryant@gmail.com), Hannu Flinck (hannu.flinck@nokia-bell-labs.com), Med Boucadair (mohamed.boucadair@orange.com), Dong Jie (dong.jie@huawei.com).

13. References

13.1. Normative References

- [I-D.bernardos-nfvrg-multidomain]
Bernardos, C., Contreras, L., Vaishnavi, I., and R. Szabo, "Multi-domain Network Virtualization", draft-bernardos-nfvrg-multidomain-03 (work in progress), September 2017.
- [I-D.dt-detnet-dp-sol]
Korhonen, J., Andersson, L., Jiang, Y., Finn, N., Varga, B., Farkas, J., Bernardos, C., Mizrahi, T., and L. Berger, "DetNet Data Plane Encapsulation", draft-dt-detnet-dp-sol-02 (work in progress), September 2017.
- [I-D.geng-netslices-architecture]
67, 4., Dong, J., Bryant, S., kiran.makhijani@huawei.com, k., Galis, A., Foy, X., and S. Kuklinski, "Network Slicing Architecture", draft-geng-netslices-architecture-02 (work in progress), July 2017.
- [I-D.ietf-l2sm-l2vpn-service-model]
Wen, B., Fioccola, G., Xie, C., and L. Jalil, "A YANG Data Model for L2VPN Service Delivery", draft-ietf-l2sm-l2vpn-service-model-03 (work in progress), September 2017.
- [I-D.ietf-opsawg-service-model-explained]
Wu, Q., LIU, W., and A. Farrel, "Service Models Explained", draft-ietf-opsawg-service-model-explained-05 (work in progress), October 2017.

- [I-D.pularikkal-virtual-cpe]
Pularikkal, B., Fu, Q., Hui, D., Sundaram, G., and S. Gundavelli, "Virtual CPE Deployment Considerations", draft-pularikkal-virtual-cpe-02 (work in progress), February 2017.
- [I-D.qiang-netslices-gap-analysis]
Qiang, L., Martinez-Julia, P., 67, 4., Dong, J., kiran.makhijani@huawei.com, k., Galis, A., Hares, S., and S. Slawomir, "Gap Analysis for Transport Network Slicing", draft-qiang-netslices-gap-analysis-01 (work in progress), July 2017.
- [RFC8049] Litkowski, S., Tomotaki, L., and K. Ogaki, "YANG Data Model for L3VPN Service Delivery", RFC 8049, DOI 10.17487/RFC8049, February 2017, <<https://www.rfc-editor.org/info/rfc8049>>.

13.2. Informative References

- [_5GICN_] IEEE Communication, "Delivering ICN Services in 5G using Network Slicing. 'Asit Chakraborti, Syed Obaid Amin, Aytac Azgin, Ravi Ravindran, G.Q.Wang'", May 2017, <<https://arxiv.org/abs/1610.01182>>.
- [ICNRG] IRTF, "ICN Routing Group", November 2016, <<https://irtf.org/icnrg>>.
- [Tactile-Internet]
ITU-T, "Technology Watch Report, The Tactile Internet", August 2014, <<https://www.itu.int/oth/T2301000023/en>>.
- [TR_3GPP.33.899]
3GPP, "Study on the security aspects of the next generation system", 3GPP TR 33.899 0.6.0, November 2016, <<http://www.3gpp.org/ftp/Specs/html-info/33899.htm>>.
- [TR_3GPP.38.801]
3GPP, "Study on new radio access technology Radio access architecture and interfaces", 3GPP TR 38.801 1.0.0, March 2017, <<http://www.3gpp.org/ftp/Specs/html-info/38801.htm>>.
- [TR_3GPP_38.913]
3GPP, "Study on scenarios and requirements for next generation access technologies", 3GPP TR 38.913 14.2.0, March 2017, <http://www.3gpp.org/ftp/Specs/archive/38_series/38.913>.

- [TS_3GPP.23.501]
3GPP, "System Architecture for the 5G System", 3GPP
TS 23.501 0.2.0, February 2017,
<<http://www.3gpp.org/ftp/Specs/html-info/23501.htm>>.
- [TS_3GPP.23.502]
3GPP, "Procedures for the 5G System", 3GPP TS 23.502
0.2.0, February 2017,
<<http://www.3gpp.org/ftp/Specs/html-info/23502.htm>>.
- [TS_3GPP.28.500]
3GPP, "Telecommunication management; Management concept,
architecture and requirements for mobile networks that
include virtualized network functions", 3GPP TS 28.500
1.3.0, 11 2016,
<<http://www.3gpp.org/ftp/Specs/html-info/28500.htm>>.
- [VCPEBBF] Broadband Forum, "TR-345 Broadband Network Gateway and
Network Function Virtualization", Dec 2016,
<[https://www.broadband-forum.org/technical/download/
TR-345.pdf](https://www.broadband-forum.org/technical/download/TR-345.pdf)>.

Authors' Addresses

Kiran Makhijani
Huawei Technologies
2890 Central Expressway
Santa Clara CA 95050
USA

Email: kiran.makhijani@huawei.com

Jun Qin
Huawei Technologies
Huawei Campus, No. 156 Beiqing Rd.
Beijing 100095

Email: qinjun4@huawei.com

Ravi Ravindran
Huawei Technologies
2890 Central Expressway
Santa Clara CA 95050
USA

Email: ravi.ravindran@huawei.com

Liang Geng
China Mobile
Beijing 100095
China

Email: gengliang@chinamobile.com

Li Qiang
Huawei Technologies
Huawei Campus, No. 156 Beiqing Rd.
Beijing 100095
China

Email: qiangli3@huawei.com

Shuping Peng
Huawei Technologies
Huawei Campus, No. 156 Beiqing Rd.
Beijing 100095
China

Email: pengshuping@huawei.com

Xavier de Foy
InterDigital Inc.
1000 Sherbrooke West
Montreal
Canada

Email: Xavier.Defoy@InterDigital.com

Akbar Rahman
InterDigital Inc.
1000 Sherbrooke West
Montreal
Canada

Email: Akbar.Rahman@InterDigital.com

Alex Galis
University College London
London
U.K.

Email: a.galis@ucl.ac.uk

Giuseppe Fioccola
Telecom Italia
Italy

Email: giuseppe.fioccola@telecomitalia.it

none
Internet-Draft
Intended status: Informational
Expires: May 3, 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.
October 30, 2017

Technology Independent Information Model for Network Slicing
draft-qiag-coms-netslicing-information-model-01

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 May 3, 2018.

Copyright Notice

Copyright (c) 2017 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	2
2. Terminology	4
3. Network Slice Tree Structure	4
3.1. resources	5
3.1.1. nodes	6
3.1.2. links	7
3.1.3. storage-units	8
3.1.4. compute-units	9
3.2. generalized-function-block	9
3.3. slice-level-attributes	9
4. Operations	12
5. Yang Module	13
6. Security Considerations	24
7. IANA Considerations	24
8. Acknowledgements	24
9. References	24
9.1. Normative References	24
9.2. Informative References	24
Authors' Addresses	25

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.

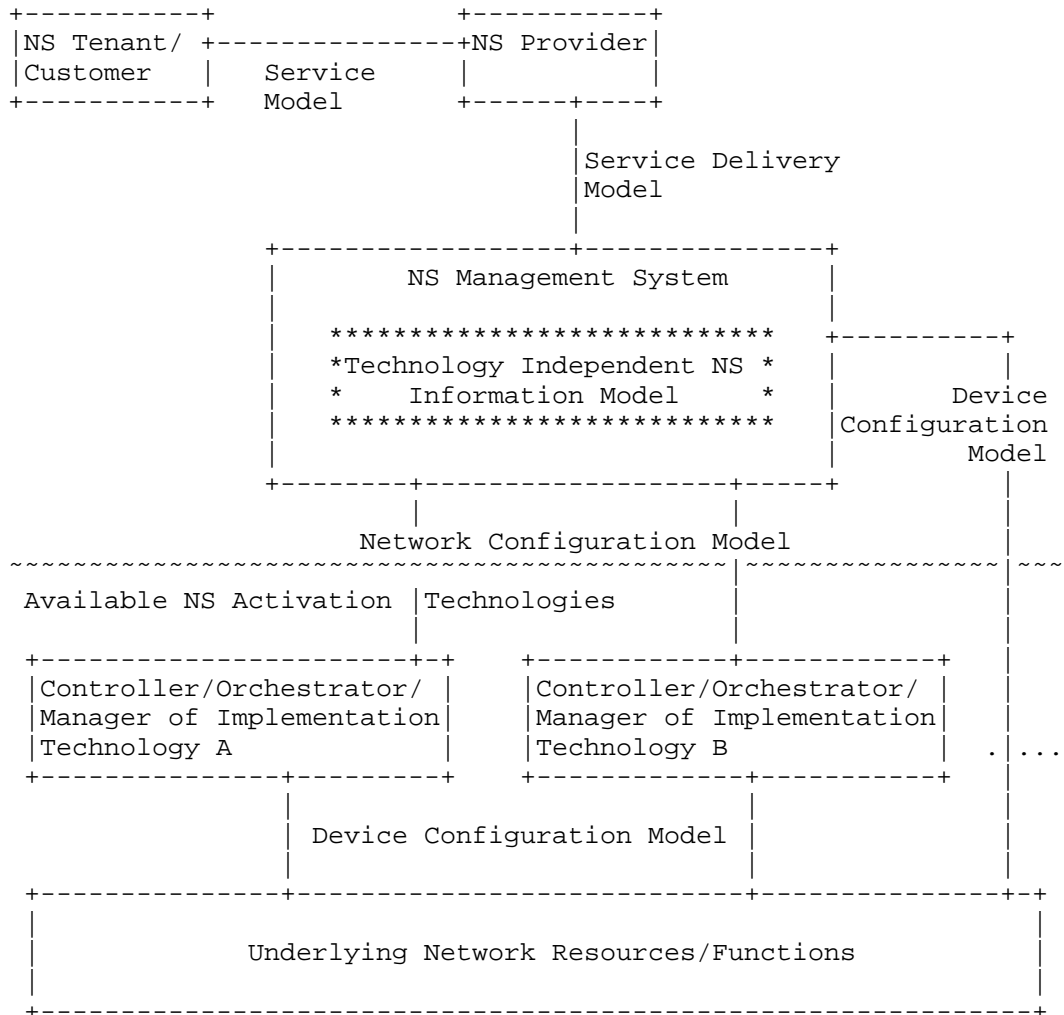


Figure 1: Technology Independent NS Information Model

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.

3.1.1.1. 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 calculated by algorithm.

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 calculated by algorithm.

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 calculated by algorithm.

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:isolation-mode?           isolation-mode-type
|   +--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 M.

isolation-mode: two options include shared 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:isolation-mode?    isolation-mode-type
|   +-rw netslice:location?          string
```

num-cores: the number of arithmetic logic unit.

ram: RAM in bytes.

isolation-mode: two options include shared or dedicated.

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

3.2. generalized-function-block

```
+++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:ip-address?     string
|   |   +-rw netslice:port?           string
|   +-rw netslice:load-balancer
|   |   +-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.

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

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-preemptive	A certain of resource reservation, the free reserved resources could be used by other slice instances, and unable to be retrieved if other slice instances are using them	Shared and non-preemptive
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> file "ietf-coms-core@2017-10-28.yang"
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 "2017-10-27" {
    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 isolation-mode-type {
    type enumeration {
        enum isolation-mode-shared {
            description "Underlying compute or storage can be
            shared with other instances";
        }
        enum isolation-mode-dedicated {
            description "Underlying compute or storage is not
            shared with other instances";
        }
    }
    description "Isolation mode";
}

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";
        }
        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 M";
    }
    leaf isolation-mode {
      type isolation-mode-type;
      description "isolation 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 isolation-mode {
      type isolation-mode-type;
      description "isolation 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";
}
}

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 function blocks TBD)";

  container domain-agent {
    description "a network slice agent to receive manager request";
    leaf agent-name {
      type string;
      description "agent name";
    }
    leaf ip-address {
      type string;
      description "IP Address of the server (type TBD)";
    }
    leaf port {
      type string;
      description "Port of the server (type TBD)";
    }
  }
}
```

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

  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

TBD

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>.
- [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

Alex Galis
University College London

Email: a.galis@ucl.ac.uk

Liang Geng
China Mobile

Email: 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