

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

M. Boucadair  
Orange  
S. Sivakumar  
Cisco Systems  
C. Jacquenet  
Orange  
S. Vinapamula  
Juniper Networks  
Q. Wu  
Huawei  
October 30, 2017

**A YANG Data Model for Network Address Translation (NAT) and Network  
Prefix Translation (NPT)  
draft-ietf-opsawg-nat-yang-07**

**Abstract**

For the sake of network automation and the need for programming Network Address Translation (NAT) function in particular, a data model for configuring and managing the NAT is essential. This document defines a YANG module for the NAT function.

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

**Editorial Note (To be removed by RFC Editor)**

Please update these statements 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 Data Model for Network Address Translation (NAT) and Network Prefix Translation (NPT)";

"reference: RFC XXXX"

**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

<a href="#">1.</a>	Introduction . . . . .	<a href="#">3</a>
<a href="#">1.1.</a>	Terminology . . . . .	<a href="#">3</a>
<a href="#">1.2.</a>	Tree Diagrams . . . . .	<a href="#">5</a>
<a href="#">2.</a>	Overview of the NAT YANG Data Model . . . . .	<a href="#">5</a>
<a href="#">2.1.</a>	Overview . . . . .	<a href="#">5</a>
<a href="#">2.2.</a>	Various NAT Flavors . . . . .	<a href="#">6</a>
<a href="#">2.3.</a>	TCP, UDP and ICMP NAT Behavioral Requirements . . . . .	<a href="#">6</a>
<a href="#">2.4.</a>	Other Transport Protocols . . . . .	<a href="#">6</a>
<a href="#">2.5.</a>	IP Addresses Used for Translation . . . . .	<a href="#">7</a>
<a href="#">2.6.</a>	Port Set Assignment . . . . .	<a href="#">7</a>
<a href="#">2.7.</a>	Port-Restricted IP Addresses . . . . .	<a href="#">7</a>
<a href="#">2.8.</a>	NAT Mapping Entries . . . . .	<a href="#">7</a>
<a href="#">2.9.</a>	Resource Limits . . . . .	<a href="#">10</a>
<a href="#">2.10.</a>	Binding the NAT Function to an External Interface or VRF	<a href="#">10</a>
<a href="#">2.11.</a>	Tree Structure . . . . .	<a href="#">11</a>
<a href="#">3.</a>	NAT YANG Module . . . . .	<a href="#">15</a>
<a href="#">4.</a>	Security Considerations . . . . .	<a href="#">52</a>
<a href="#">5.</a>	IANA Considerations . . . . .	<a href="#">53</a>
<a href="#">6.</a>	Acknowledgements . . . . .	<a href="#">53</a>
<a href="#">7.</a>	References . . . . .	<a href="#">54</a>
<a href="#">7.1.</a>	Normative References . . . . .	<a href="#">54</a>



<a href="#">7.2.</a>	Informative References . . . . .	<a href="#">55</a>
<a href="#">Appendix A.</a>	Sample Examples . . . . .	<a href="#">57</a>
<a href="#">A.1.</a>	Traditional NAT44 . . . . .	<a href="#">58</a>
<a href="#">A.2.</a>	CGN . . . . .	<a href="#">59</a>
<a href="#">A.3.</a>	CGN Pass-Through . . . . .	<a href="#">62</a>
<a href="#">A.4.</a>	NAT64 . . . . .	<a href="#">63</a>
<a href="#">A.5.</a>	Explicit Address Mappings for Stateless IP/ICMP Translation . . . . .	<a href="#">64</a>
<a href="#">A.6.</a>	Static Mappings with Port Ranges . . . . .	<a href="#">67</a>
<a href="#">A.7.</a>	Static Mappings with IP Prefixes . . . . .	<a href="#">67</a>
<a href="#">A.8.</a>	Destination NAT . . . . .	<a href="#">68</a>
<a href="#">A.9.</a>	CLAT . . . . .	<a href="#">71</a>
<a href="#">A.10.</a>	NPTv6 . . . . .	<a href="#">71</a>
	Authors' Addresses . . . . .	<a href="#">74</a>

## [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](#)], Explicit Address Mappings for Stateless IP/ICMP Translation (EAM) [[RFC7757](#)], and IPv6 Network Prefix Translation (NPTv6) [[RFC6296](#)]. 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 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 \[RFC3022\]](#).



- 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 solicit a NAT or an NPTv6 (or both) 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 NAT/NPTv6 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 NAT 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 NAT flavor (NAT44, NAT64, etc.) indifferently.

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



## **1.2. Tree Diagrams**

The meaning of the symbols in these diagrams is as follows:

- o Brackets "[" and "]" enclose list keys.
- o Curly braces "{" and "}" contain names of optional features that make the corresponding node conditional.
- o Abbreviations before data node names: "rw" means configuration (read-write), "ro" state data (read-only).
- o Symbols after data node names: "?" means an optional node, "!" a container with presence, and "\*" denotes a "list" or "leaf-list".
- o Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").
- o Ellipsis ("...") stands for contents of subtrees that are not shown.

## **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 explicit dynamic means (e.g., [[RFC6887](#)], [[RFC6736](#)], or [[RFC8045](#)]) are out of scope.

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.

Further, the NAT YANG module allows for a NAT instance to be provided with multiple NAT policies (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.

To accommodate deployments where [\[RFC6302\]](#) is not enabled, this YANG module allows to instruct a NAT function to log the destination port number. The reader may refer to [\[I-D.ietf-behave-ipfix-nat-logging\]](#) which provides the templates to log the destination ports.

## **[2.2.](#) Various NAT Flavors**

The following modes are supported:

1. Basic NAT44
2. NAPT
3. Destination NAT
4. Port-restricted NAT
5. Stateful and stateless NAT64
6. EAM SIIT
7. CLAT
8. NPTv6
9. Combination of Basic NAT/NAPT and Destination NAT
10. Combination of port-restricted and Destination NAT
11. Combination of NAT64 and EAM

[\[I-D.ietf-softwire-dslite-yang\]](#) specifies an extension to support DS-Lite.

## **[2.3.](#) TCP, UDP and ICMP NAT Behavioral Requirements**

This document assumes [\[RFC4787\]](#)[\[RFC5382\]](#)[\[RFC5508\]](#) 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 other protocols than UDP, TCP, and ICMP. 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 can be defined 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).



### **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.

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). Nevertheless, this port allocation scheme may not be optimal for logging purposes.

Therefore, 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 features 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 require to restrict the 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)
```

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 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 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 as used by an internal host when sending a packet to a remote host.

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

**external-dst-address:** Indicates the destination IP address 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 in particular, the NAT mapping structure allows to include 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 characterized as follows:

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

The mapping 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 characterized as follows:

- o type: dynamic implicit mapping.
- o transport-protocol: ICMP (1)
- o internal-src-address: 198.51.100.1
- o internal-src-port: ID1
- o external-src-address: T (an IPv4 address configured on the NAT44)
- o external-src-port: ID2 (an ICMP identifier that is chosen by the NAT44)
- o internal-dst-address: 198.51.100.11

The mapping that will be created by a NAT64 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 characterized as follows:

- o type: dynamic implicit mapping.
- o transport-protocol: ICMPv6 (58)
- o internal-src-address: 2001:db8:aaaa::1
- o internal-src-port: ID1
- o external-src-address: T (an IPv4 address configured on the NAT64)
- o external-src-port: ID2 (an ICMP identifier that is chosen by the NAT64)
- o internal-dst-address: 2001:db8:1234::198.51.100.1
- o external-dst-address: 198.51.100.1

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 NAT64. 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-limit and connection-limit). According to [[RFC6888](#)], the model allows 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.

## **[2.10.](#) Binding the NAT Function to an External Interface or VRF**

The model allows to specify the interface or Virtual Routing and Forwarding (VRF) instance on which the NAT function must be applied (external-realm). Distinct interfaces/VRFs can be provided as a function of the NAT policy (see for example, [Section 4 of \[RFC7289\]](#)).

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



### 2.11. 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
        +--rw capabilities
          | +--rw nat-flavor*          identityref
          | +--rw nat44-flavor*        identityref
          | +--rw restricted-port-support? boolean
          | +--rw static-mapping-support? boolean
          | +--rw port-randomization-support? boolean
          | +--rw port-range-allocation-support? boolean
          | +--rw port-preservation-support? boolean
          | +--rw port-parity-preservation-support? boolean
          | +--rw address-roundrobin-support? boolean
          | +--rw paired-address-pooling-support? boolean
          | +--rw endpoint-independent-mapping-support? boolean
          | +--rw address-dependent-mapping-support? boolean
          | +--rw address-and-port-dependent-mapping-support? boolean
          | +--rw endpoint-independent-filtering-support? boolean
          | +--rw address-dependent-filtering? boolean
          | +--rw address-and-port-dependent-filtering? boolean
        +--rw nat-pass-through* [id]
          | +--rw id          uint32
          | +--rw prefix?     inet:ip-prefix
          | +--rw port?       inet:port-number
        +--rw policy* [id]
          | +--rw id          uint32
          | +--rw clat-parameters
          | | +--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* [translation-id]
          | | +--rw translation-id     uint32
          | | +--rw internal-ipv6-prefix? inet:ipv6-prefix
          | | +--rw external-ipv6-prefix? inet:ipv6-prefix
          | +--rw eam* [ipv4-prefix]
          | | +--rw ipv4-prefix     inet:ipv4-prefix
          | | +--rw ipv6-prefix?    inet:ipv6-prefix
          | +--rw nat64-prefixes* [nat64-prefix]
          | | +--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]
| | +--rw pool-id              uint32
| | +--rw external-ip-pool?    inet:ipv4-prefix
| +--rw port-set-restrict
| | +--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
| +--rw dst-ip-address-pool* [pool-id]
| | +--rw pool-id              uint32
| | +--rw dst-in-ip-pool?      inet:ip-prefix
| | +--rw dst-out-ip-pool?     inet:ip-prefix
| +--rw supported-transport-protocols* [transport-protocol-id]
| | +--rw transport-protocol-id    uint8
| | +--rw transport-protocol-name?  string
| +--rw subscriber-mask-v6?        uint8
| +--rw subscriber-match* [sub-match-id]
| | +--rw sub-match-id          uint32
| | +--rw sub-mask              inet:ip-prefix
| +--rw paired-address-pooling?    boolean
| +--rw mapping-type?              enumeration
| +--rw filtering-type?            enumeration
| +--rw port-quota* [quota-type]
| | +--rw port-limit?            uint16
| | +--rw quota-type            uint8
| +--rw port-allocation-type?      enumeration
| +--rw address-roundrobin-enable?  boolean
| +--rw port-set
| | +--rw port-set-size?          uint16
| | +--rw port-set-timeout?       uint32
| +--rw timers
| | +--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 port-timeout      uint32
| | +--rw hold-down-timeout?   uint32
| | +--rw hold-down-max?      uint32
| +--rw algs* [name]
| | +--rw name                  string
| | +--rw transport-protocol?  uint32
| | +--rw transport-port?      inet:port-number
| | +--rw status?              boolean
| +--rw all-algs-enable?       boolean
| +--rw notify-pool-usage
| | +--rw pool-id?             uint32
| | +--rw high-threshold       percent
| | +--rw low-threshold?       percent
| +--rw external-realm
|   +--rw (realm-type)?
|     +--:(interface)
|       | +--rw external-interface?  if:interface-ref
|       +--:(vrf)
|         +--rw external-vrf-instance?  identityref
+--rw mapping-limit
| +--rw limit-per-subscriber?  uint32
| +--rw limit-per-vrf?        uint32
| +--rw limit-per-instance    uint32
| +--rw limit-per-udp          uint32
| +--rw limit-per-tcp          uint32
| +--rw limit-per-icmp         uint32
+--rw connection-limit
| +--rw limit-per-subscriber?  uint32
| +--rw limit-per-vrf?        uint32
| +--rw limit-per-instance    uint32
| +--rw limit-per-udp          uint32
| +--rw limit-per-tcp          uint32
| +--rw limit-per-icmp         uint32
+--rw logging-info
| +--rw logging-enable?       boolean
| +--rw destination-address    inet:ip-prefix
| +--rw destination-port      inet:port-number
| +--rw (protocol)?
|   +--:(syslog)
|     | +--rw syslog?          boolean
|     +--:(ipfix)
|       | +--rw ipfix?          boolean
|       +--:(ftp)
|         +--rw ftp?            boolean
+--rw mapping-table
| +--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 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 mapping-statistics
    |   +--ro total-mappings?         yang:gauge32
    |   +--ro total-tcp-mappings?     yang:gauge32
    |   +--ro total-udp-mappings?     yang:gauge32
    |   +--ro total-icmp-mappings?    yang:gauge32
  +--ro pool-stats
    +--ro pool-id?                   uint32
    +--ro addresses-allocated?        yang:gauge32
    +--ro addresses-free?             yang:gauge32
    +--ro port-stats
      +--ro ports-allocated?          yang:gauge32
      +--ro ports-free?               yang:gauge32

```

#### notifications:

##### +--n nat-event

```

+--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

```



### 3. NAT YANG Module

```
<CODE BEGINS> file "ietf-nat@2017-10-30.yang"

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

  //namespace to be assigned by IANA
  prefix "nat";

  import ietf-inet-types { prefix inet; }
  import ietf-yang-types { prefix yang; }
  import ietf-interfaces { prefix if; }

  organization "IETF OPSAWG (Operations and Management Area Working Group)";

  contact

    "WG Web:  <https://datatracker.ietf.org/wg/opsawg/>
    WG List:  <mailto:opsawg@ietf.org>

    WG Chair: Ignas Bagdonas
               <mailto:ibagdona@gmail.com>

    WG Chair: Joe Clarke
               <mailto:jclarke@cisco.com>

    WG Chair: Tianran Zhou
               <mailto:zhoutianran@huawei.com>

    Editor:  Mohamed Boucadair
             <mailto:mohamed.boucadair@orange.com>

    Editor:  Senthil Sivakumar
             <mailto:ssenthil@cisco.com>

    Editor:  Chritsian Jacquenet
             <mailto:christian.jacquenet@orange.com>

    Editor:  Suresh Vinapamula
             <mailto:sureshk@juniper.net>

    Editor:  Qin Wu
             <mailto:bill.wu@huawei.com>";
```



`description`

"This module is a YANG module for NAT implementations (including NAT44 and NAT64 flavors).

Copyright (c) 2017 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 2017-10-30 {``description`

"Initial revision.";

`reference`

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

`}``/*``* Definitions``*/``typedef percent {``type uint8 {`

range "0 .. 100";

`}``description`

"Percentage";

`}``/*``* Identities``*/``identity nat-type {``description`

"Base identity for nat type.";

`}``identity nat44 {`

base nat:nat-type;



```
    description
      "Identity for traditional NAT support.";
    reference
      "RFC 3022: Traditional IP Network Address Translator
        (Traditional NAT)";
  }

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

  identity napt {
    base nat:nat44;
    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 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";
}

identity vrf-routing-instance {
  description
    "This identity represents a VRF routing instance.";
  reference
    "Section 8.9 of RFC 4026.";
}

/*
 * Grouping
 */

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

  leaf start-port-number {
    type inet:port-number;
    description
      "Begining 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 ports.

        It may be a simple port range, or use the Port Set ID (PSID)
        algorithm to represent a range of transport layer
        ports 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.";

  leaf index {
    type uint32;
    description
      "A unique identifier of a mapping entry.";
  }

  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.
    For example, this field contains 6 (TCP) for a TCP
    mapping or 17 (UDP) for a UDP mapping.

    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 ports, 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.";

    leaf id {
      type uint32;
      description
        "NAT instance identifier.";
      reference
        "RFC 7659.";
    }

    leaf name {
      type string;
      description
        "A name associated with the NAT instance.";
    }

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

  container capabilities {
    description
      "NAT capabilities";

    leaf-list nat-flavor {
      type identityref {
        base nat-type;
      }
      description
        "Type of NAT.";
    }

    leaf-list nat44-flavor {
      when "../nat-flavor = 'nat44'";
      type identityref {
        base nat44;
      }
      description
```



```
    "Type of NAT44: Basic NAT or NAPT.";
}

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 in Section 4 of RFC 4787 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

Boucadair, et al.

Expires May 3, 2018

[Page 26]

```
    "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.";
}
}

list nat-pass-through {
    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;
        description
            "The IP addresses that match
            should not be translated. According to
            REQ#6 of RFC6888, it must be possible
            to administratively turn off translation
            for specific destination addresses
            and/or ports.";
        reference
            "REQ#6 of RFC6888.";
    }
}

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



```
    description
      "According to REQ#6 of RFC6888, 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 RFC6888.";
  }
}

list policy {
  key id;
  description
    "NAT parameters for a given instance";

  leaf id {
    type uint32;
    description
      "An identifier of the NAT policy.";
  }

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

    list clat-ipv6-prefixes {
      when "../../capabilities/nat-flavor = 'clat' ";
      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 {
    when "../..../capabilities/nat-flavor = 'clat'";
    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 {
    when "../..../capabilities/nat-flavor = 'nptv6' ";
    key translation-id;
    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 translation-id {

type uint32;

description

"An identifier of the NPTv6 prefixes.";

}

leaf internal-ipv6-prefix {

type inet:ipv6-prefix;

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;

description

"An IPv6 prefix used by the external interface of NPTv6.";

reference

["RFC 6296: IPv6-to-IPv6 Network Prefix Translation"](#);

}

}

list eam {

when "../capabilities/nat-flavor = '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;



```
    description
      "The IPv4 prefix of an EAM.";
    reference
      "Section 3.2 of RFC 7757.";
  }

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

list nat64-prefixes {
  when "../capabilities/nat-flavor = 'nat64' " +
    " or ../capabilities/nat-flavor = 'clat'";
  key nat64-prefix;
  description
    "Provides one or a list of NAT64 prefixes
     with or without a list of destination IPv4 prefixes.

     Destination-based Pref64::/n is discussed in
     Section 5.1 of \[RFC7050\]). 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 RFC7050.";

  leaf nat64-prefix {
    type inet:ipv6-prefix;
    description
      "A NAT64 prefix. Can be NSP or a 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;
  description
    "Enable explicitly statless NAT64.";
}

list external-ip-address-pool {
  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;
    description
      "An identifier of the address pool.";
  }

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

container port-set-restrict {
  when "../capabilities/restricted-port-support = 'true'";

  description
    "Configures contiguous and non-contiguous port ranges.";

  uses port-set;
```



```
}

leaf dst-nat-enable {
  type boolean;
  default false;
  description
    "Enable/Disable destination NAT.
     A NAT44 may be configured to enable
     Destination NAT, too.";
}

list dst-ip-address-pool {
  when "../capabilities/nat-flavor = '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
      "Internal IP prefix/address";
  }

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

list supported-transport-protocols {
  key transport-protocol-id;

  description
    "Supported transport protocols.
     TCP and UDP are supported by default.";

  leaf transport-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 (TCP) for a TCP
      mapping or 17 (UDP) for a UDP mapping.";
  }

  leaf transport-protocol-name {
    type string;
    description
      "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 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 (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 {
```



```
    key sub-match-id;

    description
      "IP prefix match.";

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

    leaf sub-mask {
      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 paired-address-pooling {
    type boolean;
    default true;
    description
      "Paired address pooling informs the NAT
      that all the flows from an internal IP
      address must be assigned the same external
      address.";

    reference
      "RFC 4787: Network Address Translation (NAT) Behavioral
Requirements
      for Unicast UDP";
  }

  leaf mapping-type {
    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 {
    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.";
  }

  list port-quota {
    when "../capabilities/nat44-flavor = "+"
      "'napt' or "+"
      "../capabilities/nat-flavor = "+"
      "'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 transport.";
}
}

leaf port-allocation-type {
  type enumeration {
    enum "random" {
      description
        "Port randomization is enabled.";
    }

    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.";
    }
}
description
    "Indicates the type of a port allocation.";
}

leaf address-roundrobin-enable {
    type boolean;

    description
        "Enable/disable address allocation
        round robin.";
}

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

    description
        "Manages port-set assignments.";

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

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

container timers {
    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.

       Section 2.1 of \[RFC7857\] clarifies that 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.
```



```
    Section 2.1 of \[RFC7857\] clarifies that 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 {
    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 NTP (123) and longer timeouts
    on other ports.";

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

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

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 timer pool.

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](#).";

```
}
}

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.";
  }

  leaf transport-port {
    type inet:port-number;
    description
      "The port number used by the ALG.";
  }

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



```
leaf all-algs-enable {
  type boolean;
  description
    "Enable/disable 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 {
  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 high-threshold {
    type percent;
    mandatory true;
    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%.";
  }

  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%.";
  }
}
```



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

  choice realm-type {
    description
      "Interface or VRF.";

    case interface {
      description
        "External interface.";

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

    case vrf {
      description
        "External VRF instance.";

      leaf external-vrf-instance {
        type identityref {
          base vrf-routing-instance;
        }
        description
          "A VRF instance.";
      }
    }
  }
}

container mapping-limit {
  description
    "Information about the configuration parameters that
    limits the mappings based upon various criteria.";

  leaf limit-per-subscriber {
    type uint32;
    description
      "Maximum number of NAT mappings per subscriber.

      A subscriber is identifier by a given prefix.";
  }
}
```



```
leaf limit-per-vrf {
  type uint32;
  description
    "Maximum number of NAT mappings per VLAN/VRF.";
}

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

leaf limit-per-udp {
  type uint32;
  mandatory true;
  description
    "Maximum number of UDP NAT mappings per subscriber.";
}

leaf limit-per-tcp {
  type uint32;
  mandatory true;
  description
    "Maximum number of TCP NAT mappings per subscriber.";
}

leaf limit-per-icmp {
  type uint32;
  mandatory true;
  description
    "Maximum number of ICMP NAT mappings per subscriber.";
}
}

container connection-limit {
  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-vrf {
  type uint32;
  units "bits/second";
  description
    "Rate-limit the number of new mappings
    and sessions per VLAN/VRF.";
}

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

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

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

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

container logging-info {
  description
    "Information about logging NAT events";
```



```
leaf logging-enable {
  type boolean;
  description
    "Enable logging features.";
  reference
    "Section 2.3 of RFC 6908.";
}

leaf destination-address {
  type inet:ip-prefix;
  mandatory true;
  description
    "Address of the collector that receives
    the logs";
}

leaf destination-port {
  type inet:port-number;
  mandatory true;
  description
    "Destination port of the collector.";
}

choice protocol {

  description
    "Enable the protocol to be used for
    the retrieval of logging entries.";

  case syslog {
    leaf syslog {
      type boolean;
      description
        "If SYSLOG is in use.";
    }
  }

  case ipfix {
    leaf ipfix {
      type boolean;
      description
        "If IPFIX is in use.";
    }
  }

  case ftp {
    leaf ftp {
      type boolean;
    }
  }
}
```



```
        description
            "If FTP is in use.";
    }
}
}
}

container mapping-table {
    when "../capabilities/nat-flavor = "+
        "'nat44' or "+
        "../capabilities/nat-flavor = "+
        "'nat64' or "+
        "../capabilities/nat-flavor = "+
        "'clat' or "+
        "../capabilities/nat-flavor = 'dst-nat'";

    description
        "NAT mapping table. Applicable for functions
        which maintains 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.";

    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;
            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;
        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;
        description
            "Counter for dropped traffic in
            bytes.";
    }
}

container mapping-statistics {
    when "../capabilities/nat-flavor = '"+
        "'nat44' or '"+
        "../capabilities/nat-flavor = '"+
        "'nat64' or '"+
        "../capabilities/nat-flavor = 'dst-nat'";

    description
        "Mapping statistics.";

    leaf total-mappings {
        type yang:gauge32;
        description
            "Total number of NAT mappings present
            at a given time. This variable includes
            all the static and dynamic mappings.";
    }
}
```





```
leaf total-tcp-mappings {
  type yang:gauge32;
  description
    "Total number of TCP mappings present
    at a given time.";
}

leaf total-udp-mappings {
  type yang:gauge32;
  description
    "Total number of UDP mappings present
    at a given time.";
}

leaf total-icmp-mappings {
  type yang:gauge32;
  description
    "Total number of ICMP mappings present
    at a given time.";
}
}

container pool-stats {

  when "../capabilities/nat-flavor = '"+
    "'nat44' or '"+
    "../capabilities/nat-flavor = '"+
    "'nat64'";

  description
    "Statistics related to address/prefix
    pool usage";

  leaf pool-id {
    type uint32;
    description
      "Unique Identifier that represents
      a pool of addresses/prefixes.";
  }

  leaf addresses-allocated {
    type yang:gauge32;
    description
      "Number of allocated addresses in
      the pool";
  }

  leaf addresses-free {
```



```
    type yang:gauge32;
    description
      "Number of unallocated addresses in
       the pool at a given time.The sum of
       unallocated and allocated
       addresses is the total number of
       addresses of the pool.";
  }

  container port-stats {

    description
      "Statistics related to port
       usage.";

    leaf ports-allocated {
      type yang:gauge32;
      description
        "Number of allocated ports
         in the pool.";
    }

    leaf ports-free {
      type yang:gauge32;
      description
        "Number of unallocated addresses
         in the pool.";
    }
  }
}

/*
 * Notifications
 */

notification nat-event {
  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";
        }
    description
        "NAT instance ID.";
}

leaf policy-id {
    type leafref {
        path
            "/nat/instances/"
            + "instance/policy/id";
    }

    description
        "Policy ID.";
}

leaf pool-id {
    type leafref {
        path
            "/nat/instances/"
            + "instance/policy/"
            + "external-ip-address-pool/pool-id";
    }
    description
        "Pool ID.";
}

leaf notify-pool-threshold {
    type percent;
    mandatory true;
    description
        "A treshhold has been fired.";
}
}
}
<CODE ENDS>
```

#### 4. Security Considerations

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 [[RFC6536](#)] 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). These data nodes are considered sensitive. Write operations (e.g., edit-config) applied to these data nodes without proper protection can negatively affect network operations.

Security considerations related to address and prefix translation are discussed in [[RFC6888](#)], [[RFC6146](#)], [[RFC6877](#)], [[RFC7757](#)], and [[RFC6296](#)].

## 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 and Tianran Zhou for the review.

Thanks to Juergen Schoenwaelder for the comments on the YANG structure and the suggestion to use NMDA.

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 an earlier version of this module.





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.

## **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>>.
- [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>>.



- [RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", [RFC 6536](#), DOI 10.17487/RFC6536, March 2012, <<https://www.rfc-editor.org/info/rfc6536>>.
- [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>>.
- [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>>.
- [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>>.

## **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-behave-ipfix-nat-logging]  
Sivakumar, S. and R. Penno, "IPFIX Information Elements for logging NAT Events", [draft-ietf-behave-ipfix-nat-logging-13](#) (work in progress), January 2017.



[I-D.ietf-softwire-dslite-yang]

Boucadair, M., Jacquenet, C., and S. Sivakumar, "YANG Data Modules for the DS-Lite", [draft-ietf-softwire-dslite-yang-07](#) (work in progress), October 2017.

[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-11](#) (work in progress), July 2017.

[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>>.

[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>>.

[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>>.

[RFC6302] Durand, A., Gashinsky, I., Lee, D., and S. Sheppard, "Logging Recommendations for Internet-Facing Servers", [BCP 162](#), [RFC 6302](#), DOI 10.17487/RFC6302, June 2011, <<https://www.rfc-editor.org/info/rfc6302>>.

[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>>.
- [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>>.
- [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>>.
- [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>>.

## **[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>
        192.0.2.1
      </external-ip-pool>
    </external-ip-address-pool>
    ....
    <mapping-table>
      ....
      <external-src-address>
        192.0.2.1
      </external-src-address>
      ....
    </mapping-table>
  </instance>
</instances>
```

The following shows the XML excerpt depicting a dynamic UDP mapping entry maintained by a traditional NAT44. 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 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
  </internal-dst-address>
  <internal-src-port>
    <start-port-number>
      1568
    </start-port-number>
  </internal-dst-port>
  <external-dst-address>
    198.51.100.1
  </external-dst-address>
  <external-dst-port>
    <start-port-number>
      15000
    </start-port-number>
  </external-dst-port>
  <lifetime>
    300
  </lifetime>
</mapping-entry>
```

#### [A.2.](#) CGN

The following XML snippet shows the example of the capabilities supported by a CGN as retrieved using NETCONF.



```
<capabilities
  <nat-flavor>
    nat44
  </nat44-flavor>
  <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-support>
    true
  </port-preservation-support>
  <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>
    false
  </address-dependent-mapping-support>
  <address-and-port-dependent-mapping-support>
    false
  </address-and-port-dependent-mapping-support>
  <endpoint-independent-filtering-support>
    true
  </endpoint-independent-filtering-support>
  <address-dependent-filtering>
    false
  </address-dependent-filtering>
  <address-and-port-dependent-filtering>
    false
  </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 (192.0.2.0/24). 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>
        192.0.2.0/24
      </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 (port range of 1024 ports) as shown below:





```
<instances>
  <instance>
    <id>1</id>
    <name>myotherCGN</name>
    ....
    <external-ip-address-pool>
      <pool-id>1</pool-id>
      <external-ip-pool>
        192.0.2.0/24
      </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.

X1:x1		X1':x1'		X2:x2
+---+from X1:x1		+---+from X1:x1		+---+
C     to X2:x2		to X2:x2		S
l   >>>>>>>>>>		C   >>>>>>>>>>		e
i		G		r
e   <<<<<<<<<<		N   <<<<<<<<<<		v
n   from X2:x2		from X2:x2		e
t     to X1:x1		to X1:x1		r
+---+		+---+		+---+

Figure 1: CGN Pass-Through



For example, in order to disable NAT for communications issued by the client (192.0.2.25), the following configuration parameter must be set:

```
<nat-pass-through>
  ...
  <prefix>192.0.2.25</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>
```

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>
```

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.](#) Explicit Address Mappings for Stateless IP/ICMP Translation**

As specified in [[RFC7757](#)], an EAM consists of an IPv4 prefix and an IPv6 prefix. Let's consider the set of EAM examples in Figure 2.

+---+-----+-----+-----+			
#	IPv4 Prefix		IPv6 Prefix
+---+-----+-----+-----+			
1	192.0.2.1		2001:db8:aaaa::
2	192.0.2.2/32		2001:db8:bbbb::b/128
3	192.0.2.16/28		2001:db8:cccc::/124
4	192.0.2.128/26		2001:db8:dddd::/64
5	192.0.2.192/29		2001:db8:eeee:8::/62
6	192.0.2.224/31		64:ff9b::/127
+---+-----+-----+-----+			

Figure 2: 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
  </ipv4-prefix>
  <ipv6-prefix>
    2001:db8:aaaa::
  </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 statefull NAT64. This example shows a NAT64 function that supports static mappings:

```
<capabilities
  <nat-flavor>
    nat64
  </nat44-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-support>
    true
  </port-preservation-support>
  <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>
    false
  </address-dependent-mapping-support>
  <address-and-port-dependent-mapping-support>
    false
  </address-and-port-dependent-mapping-support>
  <endpoint-independent-filtering-support>
    true
  </endpoint-independent-filtering-support>
  <address-dependent-filtering>
    false
  </address-dependent-filtering>
  <address-and-port-dependent-filtering>
    false
  </address-and-port-dependent-filtering>
</capabilities>
```



#### [A.6.](#) 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
  </internal-dst-address>
  <internal-dst-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
  </external-dst-address>
  <external-src-port>
    <start-port-number>
      1100
    </start-port-number>
    <end-port-number>
      1500
    </end-port-number>
  </external-dst-port>
  ...
</mapping-entry>
```

#### [A.7.](#) Static Mappings with IP Prefixes

The following example shows a static mapping that instructs a NAT to translate packets issued from 192.0.2.1/24 to 198.51.100.1/24.



```
<mapping-entry>
  <index>1</index>
  <type>static</type>
  <transport-protocol>6</transport-protocol>
  <internal-src-address>
    192.0.2.1/24
  </internal-dst-address>
  <external-src-address>
    198.51.100.1/24
  </external-dst-address>
  ...
</mapping-entry>
```

#### [A.8.](#) Destination NAT

The following XML snippet shows an example a destination NAT that is instructed to translate 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
  </dst-in-ip-pool>
  <dst-out-ip-pool>
    198.51.100.1
  </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 to be configured on the NAT:



```
<mapping-entry>
  <index>1</index>
  <type>static</type>
  <transport-protocol>6</transport-protocol>
  <internal-dst-address>
    192.0.2.1
  </internal-dst-address>
  <internal-dst-port>
    <start-port-number>80</start-port-number>
  </internal-dst-port>
  <external-dst-address>
    198.51.100.1
  </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 to be configured on the NAT:





```
<mapping-entry>
  <index>1</index>
  <type>static</type>
  <transport-protocol>6</transport-protocol>
  <internal-dst-address>
    192.0.2.1
  </internal-dst-address>
  <internal-dst-port>
    <start-port-number>
      80
    </start-port-number>
  </internal-dst-port>
  <external-dst-address>
    198.51.100.1
  </external-dst-address>
  ...
</mapping-entry>
<mapping-entry>
  <index>2</index>
  <type>static</type>
  <transport-protocol>
    6
  </transport-protocol>
  <internal-dst-address>
    192.0.2.1
  </internal-dst-address>
  <internal-dst-port>
    <start-port-number>
      22
    </start-port-number>
  </internal-dst-port>
  <external-dst-address>
    198.51.100.2
  </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 modifies the internal IP address and/or port number.

#### [A.9.](#) CLAT

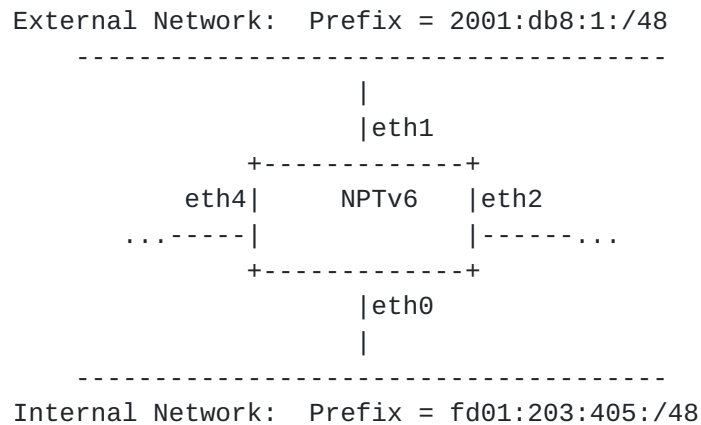
The following XML snippet shows the example of a CLAT that is configured with 2001:db8:1234::/96 as PLAT-side IPv6 prefix and 2001:db8:aaaa::/96 as CLAT-side IPv6 prefix. The CLAT is also provided with 192.0.0.1/32 (which is selected from the IPv4 service continuity prefix defined in [[RFC7335](#)]).

```
<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.10.](#) NPTv6

Let's consider the example of a NPTv6 translator that should rewrite packets with the source prefix (fd01:203:405:/48) with the external prefix (2001:db8:1:/48). The internal interface is "eth0" while the external interface is "eth1".





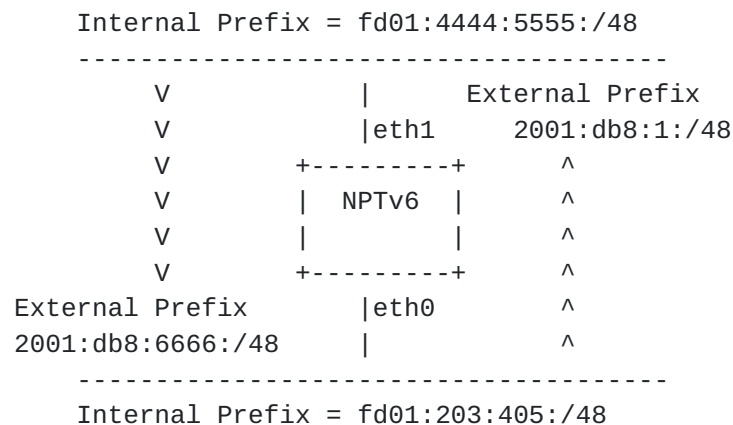
Example of NPTv6 ([RFC6296](https://tools.ietf.org/html/rfc6296))

The XML snippet to configure NPTv6 prefixes in such case is depicted below:

```
<nptv6-prefixes>
  <translation-id>1</translation-id>
  <internal-ipv6-prefix>
    fd01:203:405:/48
  </internal-ipv6-prefix>
  <external-ipv6-prefix>
    2001:db8:1:/48
  </external-ipv6-prefix>
</nptv6-prefixes>
...
<external-interfaces>
  <external-interface>
    eth1
  </external-interface>
</external-interfaces>
```

Figure 3 shows an example of an NPTv6 that interconnects two internal networks (fd01:203:405:/48 and fd01:4444:5555:/48); each is translated using a dedicated prefix (2001:db8:1:/48 and 2001:db8:6666:/48, respectively).



Figure 3: Connecting two Peer Networks ([RFC6296](#))

To that aim, the following configuration is provided to the NPTv6:

```

<policy>
  <id>1</id>
  <nptv6-prefixes>
    <translation-id>1</translation-id>
    <internal-ipv6-prefix>
      fd01:203:405:/48
    </internal-ipv6-prefix>
    <external-ipv6-prefix>
      2001:db8:1:/48
    </external-ipv6-prefix>
  </nptv6-prefixes>
  <external-interface>
    eth1
  </external-interface>
</policy>
<policy>
  <id>2</id>
  <nptv6-prefixes>
    <translation-id>2</translation-id>
    <internal-ipv6-prefix>
      fd01:4444:5555:/48
    </internal-ipv6-prefix>
    <external-ipv6-prefix>
      2001:db8:6666:/48
    </external-ipv6-prefix>
  </nptv6-prefixes>
  <external-interface>
    eth0
  </external-interface>
</policy>
  
```





Authors' Addresses

Mohamed Boucadair  
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

