Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 11, 2016

S. Sivakumar
Cisco Systems
M. Boucadair
France Telecom
S. Vinapamula
Juniper Networks
March 10, 2016

YANG Data Model for Network Address Translation (NAT)
draft-sivakumar-yang-nat-04

Abstract

For the sake of network automation and the need for programming NAT function in particular, a data model for configuring and managing the NAT device is essential. This document defines a YANG data model for the NAT function. Both the NAT44 and NAT64 are covered in this document.

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 http://datatracker.ietf.org/drafts/current/.

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

This Internet-Draft will expire on September 11, 2016.

Table of Contents

1. Introduction

This document defines a data model for Network Address Translation (NAT) using the YANG data modeling language [RFC6020]. Traditional NAT is defined in [RFC2663] and Carrier Grade NAT is defined in [RFC6888]. This document covers the NAT features in both documents. This document also covers the NAT64 as defined in [RFC6146].

This document assumes [RFC4787][RFC5382][RFC5508] are enabled by default.

1.1. Requirements Language

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

The usage of the term "NAT device" in this document refer to any NAT44 and NAT64 devices. This document uses the term "Session" as it is defined in [RFC2663] and the term BIB as it is defined in [RFC6146].

1.2. Tree Diagrams

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

- Brackets "[" and "]" enclose list keys.

- Curly braces "{" and "}" contain names of optional features that make the corresponding node conditional.

- Abbreviations before data node names: "rw" means configuration (read-write), "ro" state data (read-only).

- Symbols after data node names: "?" means an optional node, "!" a container with presence, and "*" denotes a "list" or "leaf-list".

- Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").

- Ellipsis ("...") stands for contents of subtrees that are not shown.

2. Overview of the NAT YANG Data Model

The NAT data model is designed to cover both configuration and state retrieval, nevertheless this document covers dynamic (implicit) mapping while PCP-related functionality to instruct dynamic explicit mapping is defined in [I-D.boucadair-pcp-yang].

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

A NAT function can either assign individual port numbers or port sets. Both features are supported in the YANG data model.

To accommodate deployments where [RFC6302] is not enabled, the NAT function can be configured to log the destination port number.

This data model assumes that pools of IPv4 addresses can be provisioned to NAT function. These pools may be contiguous or non-contiguous.

A NAT device can enabled multiple NAT instances; each responsible to service a group of internal hosts. This document does make any assumption how internal hosts are attached to a given NAT instance.

The data model assumes that each NAT instance can: be enable/ disabled, be provisioned with a dedicated configuration data, and maintain its own mapping table.

This version of the document does not cover the following functionalities:

- DSCP-related operations.
- Exclude/include ports (e.g.; system port) from the port assignment pool.
- Deterministic NAT assignment scheme.

The tree structure of the NAT data model is provided below:

```
module: ietf-nat
   +--rw nat-config
   |  +--rw nat-instances
   |     +--rw nat-instance* [id]
   |        +--rw id                                  uint32
   |        +--rw enable?                             boolean
   |        +--rw external-ip-address-pool* [pool-id]
   |        |  +--rw pool-id             uint32
   |        |  +--rw external-ip-pool?   inet:ipv4-prefix
   |        +--rw subscriber-mask-v6?                 uint8
   |        +--rw subscriber-mask-v4* [sub-mask-id]
   |        |  +--rw sub-mask-id    uint32
   |        |  +--rw sub-mask       inet:ipv4-prefix
   |        +--rw paired-address-pooling?             boolean
   |        +--rw nat-mapping-type?                   enumeration
   |        +--rw nat-filtering-type?                 enumeration
   |        +--rw port-quota?                         uint16
   |        +--rw port-set
   |        |  +--rw port-set-enable?    boolean
   |        |  +--rw port-set-size?      uint16
   |        |  +--rw port-set-timeout?   uint32
   |        +--rw port-randomization-enable?          boolean
   |        +--rw port-preservation-enable?           boolean
   |        +--rw port-range-preservation-enable?     boolean
   |        +--rw port-parity-preservation-enable?    boolean
   |        +--rw address-roundrobin-enable?          boolean
   |        +--rw udp-timeouts?                       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 logging-info
   |        |  +--rw destination-address    inet:ipv4-prefix
   |        |  +--rw destination-port       inet:port-number
   |        +--rw connection-limit
   |        |  +--rw limit-per-subscriber?   uint32
   |        |  +--rw limit-per-vrf?          uint32
   |        |  +--rw limit-per-subnet?       inet:ipv4-prefix
   |        |  +--rw limit-per-instance      uint32
   |        +--rw mapping-limit
   |        |  +--rw limit-per-subscriber?   uint32
   |        |  +--rw limit-per-vrf?          uint32
   |        |  +--rw limit-per-subnet?       inet:ipv4-prefix
   |        |  +--rw limit-per-instance      uint32
```

```
|       +--rw ftp-alg-enable?                   boolean
|       +--rw dns-alg-enable?                   boolean
|       +--rw tftp-alg-enable?                  boolean
|       +--rw msrpc-alg-enable?                 boolean
|       +--rw netbios-alg-enable?               boolean
|       +--rw rcmd-alg-enable?                  boolean
|       +--rw ldap-alg-enable?                  boolean
|       +--rw sip-alg-enable?                   boolean
|       +--rw rtsp-alg-enable?                  boolean
|       +--rw h323-alg-enable?                  boolean
|       +--rw all-algs-enable?                  boolean
|       +--rw notify-pool-usage
|       |  +--rw pool-id?                      uint32
|       |  +--rw notify-pool-hi-threshold     percent
|       |  +--rw notify-pool-low-threshold?   percent
|       +--rw nat64-prefixes* [nat64-prefix-id]
|       |  +--rw nat64-prefix-id           uint32
|       |  +--rw nat64-prefix?             inet:ipv6-prefix
|       |  +--rw destination-ipv4-prefix* [ipv4-prefix-id]
|       |     +--rw ipv4-prefix-id    uint32
|       |     +--rw ipv4-prefix?      inet:ipv4-prefix
|       +--rw mapping-table
|          +--rw mapping-entry* [index]
|             +--rw index                  uint32
|             +--rw type?                  enumeration
|             +--rw internal-src-address    inet:ip-address
|             +--rw internal-src-port
|             |  +--rw (port-type)?
|             |     +--:(single-port-number)
|             |     |  +--rw single-port-number?   inet:port-number
|             |     +--:(port-range)
|             |        +--rw start-port-number?    inet:port-number
|             |        +--rw end-port-number?      inet:port-number
|             +--rw external-src-address    inet:ipv4-address
|             +--rw external-src-port
|             |  +--rw (port-type)?
|             |     +--:(single-port-number)
|             |     |  +--rw single-port-number?   inet:port-number
|             |     +--:(port-range)
|             |        +--rw start-port-number?    inet:port-number
|             |        +--rw end-port-number?      inet:port-number
|             +--rw transport-protocol      uint8
|             +--rw internal-dst-address?   inet:ipv4-prefix
|             +--rw internal-dst-port
|             |  +--rw (port-type)?
|             |     +--:(single-port-number)
|             |     |  +--rw single-port-number?   inet:port-number
|             |     +--:(port-range)
```

```
   |              |       +--rw start-port-number?    inet:port-number
   |              |       +--rw end-port-number?      inet:port-number
   |              +--rw external-dst-address?   inet:ipv4-address
   |              +--rw external-dst-port
   |              |  +--rw (port-type)?
   |              |     +--:(single-port-number)
   |              |     |  +--rw single-port-number?   inet:port-number
   |              |     +--:(port-range)
   |              |        +--rw start-port-number?    inet:port-number
   |              |        +--rw end-port-number?      inet:port-number
   |              +--rw lifetime               uint32
   +--ro nat-state
      +--ro nat-instances
         +--ro nat-instance* [id]
            +--ro id                    int32
            +--ro nat-capabilities
            |  +--ro nat44-support?                                boolean
            |  +--ro nat64-support?                                boolean
            |  +--ro static-mapping-support?                       boolean
            |  +--ro port-set-support?                             boolean
            |  +--ro port-randomization-support?                   boolean
            |  +--ro port-range-preservation-support?              boolean
            |  +--ro port-preservation-suport?                     boolean
            |  +--ro port-parity-preservation-support?             boolean
            |  +--ro address-roundrobin-support?                   boolean
            |  +--ro ftp-alg-support?                              boolean
            |  +--ro dns-alg-support?                              boolean
            |  +--ro tftp-support?                                 boolean
            |  +--ro msrpc-alg-support?                            boolean
            |  +--ro netbios-alg-support?                          boolean
            |  +--ro rcmd-alg-support?                             boolean
            |  +--ro ldap-alg-support?                             boolean
            |  +--ro sip-alg-support?                              boolean
            |  +--ro rtsp-alg-support?                             boolean
            |  +--ro h323-alg-support?                             boolean
            |  +--ro paired-address-pooling-support?               boolean
            |  +--ro endpoint-independent-mapping-support?         boolean
            |  +--ro address-dependent-mapping-support?            boolean
            |  +--ro address-and-port-dependent-mapping-support?   boolean
            |  +--ro endpoint-independent-filtering-support?       boolean
            |  +--ro address-dependent-filtering?                  boolean
            |  +--ro address-and-port-dependent-filtering?         boolean
            |  +--ro stealth-mode-support?                         boolean
            +--ro nat-current-config
            |  +--ro external-ip-address-pool* [pool-id]
            |  |  +--ro pool-id             uint32
            |  |  +--ro external-ip-pool?   inet:ipv4-prefix
            |  +--ro subscriber-mask-v6?               uint8
```

```
|  +--ro subscriber-mask-v4* [sub-mask-id]
|  |  +--ro sub-mask-id    uint32
|  |  +--ro sub-mask       inet:ipv4-prefix
|  +--ro paired-address-pooling?            boolean
|  +--ro nat-mapping-type?                  enumeration
|  +--ro nat-filtering-type?                enumeration
|  +--ro port-quota?                        uint16
|  +--ro port-set
|  |  +--ro port-set-enable?    boolean
|  |  +--ro port-set-size?      uint16
|  |  +--ro port-set-timeout?   uint32
|  +--ro port-randomization-enable?         boolean
|  +--ro port-preservation-enable?          boolean
|  +--ro port-range-preservation-enable?    boolean
|  +--ro port-parity-preservation-enable?   boolean
|  +--ro address-roundrobin-enable?         boolean
|  +--ro udp-timeouts?                      uint32
|  +--ro tcp-idle-timeout?                  uint32
|  +--ro tcp-trans-open-timeout?            uint32
|  +--ro tcp-trans-close-timeout?           uint32
|  +--ro tcp-in-syn-timeout?                uint32
|  +--ro fragment-min-timeout?              uint32
|  +--ro icmp-timeout?                      uint32
|  +--ro logging-info
|  |  +--ro destination-address    inet:ipv4-prefix
|  |  +--ro destination-port       inet:port-number
|  +--ro connection-limit
|  |  +--ro limit-per-subscriber?   uint32
|  |  +--ro limit-per-vrf?          uint32
|  |  +--ro limit-per-subnet?       inet:ipv4-prefix
|  |  +--ro limit-per-instance      uint32
|  +--ro mapping-limit
|  |  +--ro limit-per-subscriber?   uint32
|  |  +--ro limit-per-vrf?          uint32
|  |  +--ro limit-per-subnet?       inet:ipv4-prefix
|  |  +--ro limit-per-instance      uint32
|  +--ro ftp-alg-enable?                    boolean
|  +--ro dns-alg-enable?                    boolean
|  +--ro tftp-alg-enable?                   boolean
|  +--ro msrpc-alg-enable?                  boolean
|  +--ro netbios-alg-enable?                boolean
|  +--ro rcmd-alg-enable?                   boolean
|  +--ro ldap-alg-enable?                   boolean
|  +--ro sip-alg-enable?                    boolean
|  +--ro rtsp-alg-enable?                   boolean
|  +--ro h323-alg-enable?                   boolean
|  +--ro all-algs-enable?                   boolean
|  +--ro notify-pool-usage
```

```
|  |  +--ro pool-id?                    uint32
|  |  +--ro notify-pool-hi-threshold    percent
|  |  +--ro notify-pool-low-threshold?  percent
|  +--ro nat64-prefixes* [nat64-prefix-id]
|     +--ro nat64-prefix-id           uint32
|     +--ro nat64-prefix?             inet:ipv6-prefix
|     +--ro destination-ipv4-prefix* [ipv4-prefix-id]
|        +--ro ipv4-prefix-id    uint32
|        +--ro ipv4-prefix?      inet:ipv4-prefix
+--ro mapping-table
|  +--ro mapping-entry* [index]
|     +--ro index                   uint32
|     +--ro type?                   enumeration
|     +--ro internal-src-address    inet:ip-address
|     +--ro internal-src-port
|     |  +--ro (port-type)?
|     |     +--:(single-port-number)
|     |     |  +--ro single-port-number?   inet:port-number
|     |     +--:(port-range)
|     |        +--ro start-port-number?    inet:port-number
|     |        +--ro end-port-number?      inet:port-number
|     +--ro external-src-address    inet:ipv4-address
|     +--ro external-src-port
|     |  +--ro (port-type)?
|     |     +--:(single-port-number)
|     |     |  +--ro single-port-number?   inet:port-number
|     |     +--:(port-range)
|     |        +--ro start-port-number?    inet:port-number
|     |        +--ro end-port-number?      inet:port-number
|     +--ro transport-protocol      uint8
|     +--ro internal-dst-address?   inet:ipv4-prefix
|     +--ro internal-dst-port
|     |  +--ro (port-type)?
|     |     +--:(single-port-number)
|     |     |  +--ro single-port-number?   inet:port-number
|     |     +--:(port-range)
|     |        +--ro start-port-number?    inet:port-number
|     |        +--ro end-port-number?      inet:port-number
|     +--ro external-dst-address?   inet:ipv4-address
|     +--ro external-dst-port
|     |  +--ro (port-type)?
|     |     +--:(single-port-number)
|     |     |  +--ro single-port-number?   inet:port-number
|     |     +--:(port-range)
|     |        +--ro start-port-number?    inet:port-number
|     |        +--ro end-port-number?      inet:port-number
|     +--ro lifetime                uint32
+--ro statistics
```

```
               +--ro total-mappings?        uint32
               +--ro total-tcp-mappings?    uint32
               +--ro total-udp-mappings?    uint32
               +--ro total-icmp-mappings?   uint32
               +--ro pool-stats
                  +--ro pool-id?             uint32
                  +--ro address-allocated?   uint32
                  +--ro address-free?        uint32
                  +--ro port-stats
                     +--ro ports-allocated?   uint32
                     +--ro ports-free?        uint32

notifications:
   +---n nat-event
      +--ro id?                     -> /nat-state/nat-instances/
      |                                nat-instance/id
      +--ro notify-pool-threshold   percent
```

3. NAT YANG Module

```
       <CODE BEGINS> file "ietf-nat@2015-09-08.yang"


       module ietf-nat {
           namespace "urn:ietf:params:xml:ns:yang:ietf-nat";
           //namespace to be assigned by IANA
           prefix "nat";
               import ietf-inet-types {
              prefix "inet";
           }
       organization "IETF NetMod Working Group";
       contact
         "Senthil Sivakumar <ssenthil@cisco.com>
          Mohamed Boucadair <mohamed.boucadair@orange.com>
          Suresh Vinapamula <sureshk@juniper.net>";

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

           Copyright (c) 2015 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 2015-09-08 {
          description "Fixes few YANG errors.";
          reference "-02";
        }

        revision 2015-09-07 {
          description "Completes the NAT64 model.";
          reference "01";
        }

        revision 2015-08-29 {
          description "Initial version.";
          reference "00";
        }

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

        /*
         * Grouping
         */

        grouping timeouts {
            description
            "Configure values of various timeouts.";

            leaf udp-timeouts {
              type uint32;
              default 300;
              description
               "UDP inactivity timeout.";
            }

            leaf tcp-idle-timeout {
                type uint32;
                default 7440;
                description
```

```
                "TCP Idle timeout, as per RFC 5382 should be no
                 2 hours and 4 minutes.";
            }

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

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

             leaf tcp-in-syn-timeout {
                 type uint32;
                 default 6;
                 description
                   "6 seconds, as defined in [RFC5382].";
             }

             leaf fragment-min-timeout {
                 type uint32;
                 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;
                 default 60;
                 description
                     "60 seconds, as defined in [RFC5508].";
             }
        }

        // port numbers: single or port range

        grouping port-number {
```

```
            description
           "Individual port or a range of ports.";

            choice port-type {
                default single-port-number;
                description
                    "Port type: single or port-range.";

                case single-port-number {
                    leaf single-port-number {
                        type inet:port-number;
                        description
                            "Used for single port numbers.";
                    }
                }

                case port-range {
                    leaf start-port-number {
                        type inet:port-number;
                        description
                            "Begining of the port range.";
                    }

                    leaf end-port-number {
                        type inet:port-number;
                        description
                            "End of the port range.";
                    }
                }
            }
        }

        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 manually configured.";
                      }
```

```
                enum "dynamic" {
                   description
                    "This mapping is created by an outgoing
                    packet.";
                }
             }
             description
               "Indicates the type of a mapping entry. E.g.,
               a mapping can be: static or dynamic";
          }

          leaf internal-src-address {
              type inet:ip-address;
              mandatory true;
              description
               "Corresponds to the source IPv4/IPv6 address
                of the IPv4 packet";
          }

          container internal-src-port {
              description
                 "Corresponds to the source port of the
                  IPv4 packet.";
              uses port-number;
          }

          leaf external-src-address {
               type inet:ipv4-address;
               mandatory true;
               description
                "External IPv4 address assigned by NAT";
          }

          container external-src-port {
             description
            "External source port number assigned by NAT.";
             uses port-number;
          }

          leaf transport-protocol {
              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 internal-dst-address {
                type inet:ipv4-prefix;
                description
                 "Corresponds to the destination IPv4 address
                  of the IPv4 packet, for example, some NAT
                  implementation support translating both source
                  and destination address and ports referred to as
                  Twice NAT";
            }

            container internal-dst-port {
                description
                   "Corresponds to the destination port of the
                    IPv4 packet.";
                 uses port-number;
            }

            leaf external-dst-address {
                 type inet:ipv4-address;
                 description
                  "External destination IPv4 address";
            }

            container external-dst-port {
               description
              "External source port number.";
               uses port-number;
            }

            leaf lifetime {
                 type uint32;
                 mandatory true;
                 description
                   "Lifetime of the mapping.";
            }
        }

        grouping nat-parameters {
            description
              "NAT parameters for a given instance";

                list external-ip-address-pool {
                     key pool-id;


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

```
                    Both contiguous and non-contiguous pools
                    can be configured for NAT.";

                     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.";
                      }
                }


                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 IP address (internal side) to
                     unambiguously identify a CPE.

                     Subscriber-mask is a system-wide configuration
                     parameter that is used to enforce generic
                     per-subscriberpolicies (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-mask-v4 {

                     key sub-mask-id;

                     description
                        "IPv4 subscriber mask.";

                     leaf sub-mask-id {
                          type uint32;
                          description
                            "An identifier of the subscriber masks.";
                     }
                     leaf sub-mask {
                           type inet:ipv4-prefix;
                           mandatory true;
                           description
                            "The IP address subnets that matches
                            should be translated. E.g., If the
                            private realms that are to be translated
                            by NAT would be 192.0.2.0/24";
                     }
                }

                leaf paired-address-pooling {
                     type boolean;
                     default true;
                     description
                      "Paired address pooling is indicating to NAT
                      that all the flows from an internal IP
                      address must be assigned the same external
                      address. This is defined in RFC 4007.";
                }

                leaf nat-mapping-type {
                      type enumeration {
                          enum "eim"  {
                             description
                                "endpoint-independent-mapping.
                                Refer section 4 of RFC 4787.";
                          }

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

                          enum "edm"  {
```

```
                    description
                       "address-and-port-dependent-mapping.
                       Refer section 4 of RFC 4787.";
                    }
                 }
               description
                 "Indicates the type of a NAT mapping.";
             }
             leaf nat-filtering-type {
               type enumeration {
                    enum "eif"  {
                       description
                          "endpoint-independent- filtering.
                          Refer section 5 of RFC 4787.";
                    }

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

                    enum "edf"  {
                       description
                          "address-and-port-dependent- filtering.
                          Refer section 5 of RFC 4787.";
                    }
                   }
               description
                 "Indicates the type of a NAT filtering.";
             }

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

             container port-set {
               description
                "Manages port-set assignments.";

               leaf port-set-enable {
                    type boolean;
                    description
                       "Enable/Disable port set assignment.";
               }
```

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

                leaf port-set-timeout {
                   type uint32;
                   description
                       "Inactivty timeout for port sets.";
                }
            }

           leaf port-randomization-enable {
              type boolean;
              description
                "Enable/disable port randomization
                 feature.";
           }

           leaf port-preservation-enable {
              type boolean;
              description
                "Indicates whether the PCP server should
                 preserve the internal port number.";
           }

           leaf port-range-preservation-enable {
                type boolean;
                description
                 "Indicates whether the NAT device should
                 preserve the internal port range.";
            }

           leaf port-parity-preservation-enable {
              type boolean;
              description
                "Indicates whether the PCP server should
                 preserve the port parity of the
                 internal port number.";
           }
           leaf address-roundrobin-enable {
              type boolean;
              description
                "Enable/disable address allocation
                round robin.";
            }
```

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

                 leaf destination-address {
                      type inet:ipv4-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.";
                 }

            }
            container connection-limit {
                 description
                   "Information on the config parameters that
                    rate limit the translations based on various
                    criteria";

                 leaf limit-per-subscriber {
                      type uint32;
                      description
                        "Maximum number of NAT mappings per
                        subscriber.";
                 }
                 leaf limit-per-vrf {
                      type uint32;
                      description
                        "Maximum number of NAT mappings per
                        VLAN/VRF.";
                 }
                 leaf limit-per-subnet {
                      type inet:ipv4-prefix;
                      description
                       "Maximum number of NAT mappings per
                        subnet.";
                 }
                 leaf limit-per-instance {
                      type uint32;
                      mandatory true;
                      description
```

```
                    "Maximum number of NAT mappings per
                    instance.";
               }
          }
          container mapping-limit {
               description
                 "Information on the config parameters that
                  rate limit the mappings based on various
                  criteria";

               leaf limit-per-subscriber {
                    type uint32;
                    description
                      "Maximum number of NAT mappings per
                      subscriber.";
               }
               leaf limit-per-vrf {
                    type uint32;
                    description
                      "Maximum number of NAT mappings per
                      VLAN/VRF.";
               }
               leaf limit-per-subnet {
                    type inet:ipv4-prefix;
                    description
                     "Maximum number of NAT mappings per
                      subnet.";
               }
               leaf limit-per-instance {
                    type uint32;
                    mandatory true;
                    description
                      "Maximum number of NAT mappings per
                      instance.";
               }
          }
          leaf ftp-alg-enable {
               type boolean;
               description
                  "Enable/Disable FTP ALG";
          }

          leaf dns-alg-enable {
               type boolean;
               description
                  "Enable/Disable DNSALG";
          }
```

```
            leaf tftp-alg-enable {
                 type boolean;
                 description
                    "Enable/Disable TFTP ALG";
            }

            leaf msrpc-alg-enable {
                 type boolean;
                 description
                    "Enable/Disable MS-RPC ALG";
            }

            leaf netbios-alg-enable {
                 type boolean;
                 description
                    "Enable/Disable NetBIOS ALG";
            }

            leaf rcmd-alg-enable {
                 type boolean;
                 description
                    "Enable/Disable rcmd ALG";
            }

            leaf ldap-alg-enable {
                 type boolean;
                 description
                    "Enable/Disable LDAP ALG";
            }

            leaf sip-alg-enable {
                 type boolean;
                 description
                    "Enable/Disable SIP ALG";
            }

            leaf rtsp-alg-enable {
                 type boolean;
                 description
                    "Enable/Disable RTSP ALG";
            }

            leaf h323-alg-enable {
                 type boolean;
                 description
                    "Enable/Disable H323 ALG";
            }
```

```
            leaf all-algs-enable {
                 type boolean;
                 description
                    "Enable/Disable all the ALGs";
            }

            container notify-pool-usage {
                 description
                    "Notification of Pool usage when certain criteria
                     is met";

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

                 leaf notify-pool-hi-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 notify-pool-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%";
                 }
            }
            list nat64-prefixes {
                 key nat64-prefix-id;

                 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.";

               leaf nat64-prefix-id {
                   type uint32;
                   description
                     "An identifier of the NAT64 prefix.";
               }

               leaf nat64-prefix {
                   type inet:ipv6-prefix;
                   default "64:ff9b::/96";
                   description
                     "A NAT64 prefix. Can be NSP or WKP [RFC6052].";
               }

               list destination-ipv4-prefix {

                    key ipv4-prefix-id;

                    description
                      "An IPv4 prefix/address.";

                    leaf ipv4-prefix-id {
                       type uint32;
                       description
                        "An identifier of the IPv4 prefix/address.";
                    }

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

       container nat-config {
           description
            "NAT";

           container nat-instances {
              description
                "nat instances";
```

```
            list nat-instance {

                key "id";

                description
                   "A NAT instance.";

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

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

                uses nat-parameters;

                container mapping-table {
                   description
                     "NAT dynamic mapping table used to track
                     sessions";

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

    /*
     * NAT State
     */

    container nat-state {

         config false;

         description
            "nat-state";
```

```
            container nat-instances {
                description
                    "nat instances";

                list nat-instance {
                    key "id";

                    description
                     "nat instance";

                    leaf id {
                        type int32;
                         description
                          "The identifier of the nat instance.";
                    }

                    container nat-capabilities {
                       description
                          "NAT Capabilities";

                        leaf nat44-support {
                            type boolean;
                             description
                               "Indicates NAT44 support";
                         }

                         leaf nat64-support {
                             type boolean;
                              description
                               "Indicates NAT64 support";
                         }

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

                         leaf port-set-support {
                             type boolean;
                              description
                               "Indicates port set assignment
                               support ";
                         }

                         leaf port-randomization-support {
                            type boolean;
```

```
                        description
                         "Indicates whether port randomization is
                           supported.";
                      }

                      leaf port-range-preservation-support {
                           type boolean;
                           description
                           "Indicates whether port range
                           preservation is supported.";
                      }

                      leaf port-preservation-suport {
                           type boolean;
                           description
                            "Indicates whether port preservation
                              is supported.";
                      }

                      leaf port-parity-preservation-support {
                           type boolean;
                           description
                            "Indicates whether port parity
                            preservation is supported.";
                      }

                      leaf address-roundrobin-support {
                           type boolean;
                           description
                            "Indicates whether address allocation
                            round robin is supported.";
                      }

               leaf ftp-alg-support {
                    type boolean;
                    description
                       "Indicates whether FTP ALG is supported";
               }

               leaf dns-alg-support {
                    type boolean;
                    description
                       "Indicates whether DNSALG is supported";
               }

               leaf tftp-support {
                    type boolean;
                    description
```

```
                "Indicates whether TFTP ALG is supported";
            }

            leaf msrpc-alg-support {
                 type boolean;
                 description
                    "Indicates whether MS-RPC ALG is supported";
            }

            leaf netbios-alg-support {
                 type boolean;
                 description
                    "Indicates whether NetBIOS ALG is supported";
            }

            leaf rcmd-alg-support {
                 type boolean;
                 description
                    "Indicates whether rcmd ALG is supported";
            }

            leaf ldap-alg-support {
                 type boolean;
                 description
                    "Indicates whether LDAP ALG is supported";
            }

            leaf sip-alg-support {
                 type boolean;
                 description
                    "Indicates whether SIP ALG is supported";
            }

            leaf rtsp-alg-support {
                 type boolean;
                 description
                    "Indicates whether RTSP ALG is supported";
            }

            leaf h323-alg-support {
                 type boolean;
                 description
                    "Indicates whether H323 ALG is supported";
            }

            leaf paired-address-pooling-support {
                 type boolean;
                 description
```

```
                "Indicates whether paired-address-pooling is
                supported";
            }

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

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

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

            leaf endpoint-independent-filtering-support {
                type boolean;
                description
                "Indicates whether endpoint-independent-mapping in
                section 5 of RFC 4787 is supported.";
            }

            leaf address-dependent-filtering {
                type boolean;
                description
                "Indicates whether endpoint-independent-mapping in
                section 5 of RFC 4787 is supported.";
            }

            leaf address-and-port-dependent-filtering {
                type boolean;
                description
                "Indicates whether endpoint-independent-mapping in
                section 5 of RFC 4787 is supported.";
            }

            leaf stealth-mode-support {
                type boolean;
```

```
                description
                "Indicates whether to respond for unsolicited
                traffic.";
            }

               }

                container nat-current-config {
                   description
                        "current config";

                   uses nat-parameters;
                }

                container mapping-table {
                    description
                        "Mapping table";
                    list mapping-entry {
                        key "index";
                        description
                      "mapping entry";
                        uses mapping-entry;
                    }
                }

                container statistics {
                     description
                       "Statistics related to the NAT instance";

                     leaf total-mappings {
                          type uint32;
                          description
                           "Total number of NAT Mappings present
                           at the time. This includes all the
                           static and dynamic mappings";
                     }
                     leaf total-tcp-mappings {
                          type uint32;
                          description
                           "Total number of TCP Mappings present
                           at the time.";
                     }
                     leaf total-udp-mappings {
                          type uint32;
                          description
                           "Total number of UDP Mappings present
                           at the time.";
                     }
```

```
                    leaf total-icmp-mappings {
                         type uint32;
                         description
                          "Total number of ICMP Mappings present
                          at the time.";
                    }
                    container pool-stats {
                         description
                            "Statistics related to Pool usage";
                         leaf pool-id {
                              type uint32;
                              description
                               "Unique Identifier that represents
                               a pool";
                         }
                         leaf address-allocated {
                              type uint32;
                              description
                                 "Number of allocated addresses in
                                 the pool";
                         }
                         leaf address-free {
                              type uint32;
                              description
                                "Number of free addresses in
                                the pool.The sum of free
                                addresses and allocated
                                addresses are the total
                                addresses in the pool";
                         }
                         container port-stats {
                              description
                                "Statistics related to port
                                usage.";

                              leaf ports-allocated {
                                   type uint32;
                                   description
                                      "Number of allocated ports
                                      in the pool";
                              }

                              leaf ports-free {
                                   type uint32;
                                   description
                                      "Number of free addresses
                                      in the pool";
                              }
```

```
                              }
                            }
                          } //statistics
                        } //nat-instance
                    } //nat-instances
               } //nat-state
               /*
                * 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-state/nat-instances/"
                         + "nat-instance/id";
                          }
                          description
                           "NAT instance ID.";
                    }

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

4. Security Considerations

The YANG module defined in this memo is designed to be accessed via the NETCONF protocol [RFC6241]. The lowest NETCONF layer is the secure transport layer and the support of SSH is mandatory to implement secure transport [RFC6242]. The NETCONF access control model [RFC6536] provides means to restrict access for particular NETCONF users to a pre-configured subset of all available NETCONF protocol operations and contents.

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.

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 [RFC6020].

```
name: ietf-nat
namespace: urn:ietf:params:xml:ns:yang:ietf-nat
prefix: nat
reference: RFC XXXX
```

6. References

6.1. Normative References

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

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <http://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, <http://www.rfc-editor.org/info/rfc4787>.

[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, <http://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, <http://www.rfc-editor.org/info/rfc5508>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <http://www.rfc-editor.org/info/rfc6020>.

[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, <http://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, <http://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, <http://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, <http://www.rfc-editor.org/info/rfc6536>.

6.2. Informative References

[I-D.boucadair-pcp-yang] Boucadair, M., Jacquenet, C., Sivakumar, S., and S. Vinapamula, "YANG Data Models for the Port Control Protocol (PCP)", draft-boucadair-pcp-yang-01 (work in progress), December 2015.

[RFC2663] Srisuresh, P. and M. Holdrege, "IP Network Address Translator (NAT) Terminology and Considerations", RFC 2663, DOI 10.17487/RFC2663, August 1999, <http://www.rfc-editor.org/info/rfc2663>.

[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, <http://www.rfc-editor.org/info/rfc6302>.

[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, <http://www.rfc-editor.org/info/rfc6888>.

Authors' Addresses

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

Mohamed Boucadair
France Telecom
Rennes 35000
France

Email: mohamed.boucadair@orange.com

Suresh Vinapamula
Juniper Networks
1133 Innovation Way
Sunnyvale 94089
USA