NTP Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 24, 2018

N. Wu
Huawei
A. Kumar S N
RtBrick Inc.
Y. Zhao
Ericsson
D. Dhody
Huawei
A. Sinha
RtBrick Inc.
June 22, 2018

A YANG Data Model for NTP
draft-ietf-ntp-yang-data-model-03

Abstract

This document defines a YANG data model for Network Time Protocol (NTP) implementations. The data model includes configuration data and state data.

Requirements Language

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

Status of This Memo

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

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

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

This Internet-Draft will expire on December 24, 2018.

Copyright Notice

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

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

Table of Contents

1. Introduction 3
  1.1. Operational State 3
  1.2. Terminology 3
  1.3. Tree Diagrams 3
  1.4. Prefixes in Data Node Names 3
2. NTP data model 4
3. Relationship with NTPv4-MIB 7
4. Relationship with RFC 7317 9
5. NTP YANG Module 9
6. Usage Example 29
  6.1. Unicast association 29
  6.2. Refclock master 31
  6.3. Authentication configuration 32
  6.4. Access configuration 34
  6.5. Multicast configuration 34
  6.6. Manycast configuration 38
  6.7. Clock state 41
  6.8. Get all association 41
  6.9. Global statistic 43
7. IANA Considerations 43
8. Security Considerations 44
9. Acknowledgments 45
10. References 45
  10.1. Normative References 45
  10.2. Informative References 47
Authors' Addresses 47

1. Introduction

This document defines a YANG [RFC6020] data model for Network Time Protocol [RFC5905] implementations.

The data model convers configuration of system parameters of NTP, such as access rules, authentication and VPN Routing and Forwarding (VRF) binding, and also associations of NTP in different modes and parameters of per-interface. It also provides information about running state of NTP implementations.

1.1. Operational State

NTP Operational State is included in the same tree as NTP configuration, consistent with Network Management Datastore Architecture [RFC8342]. NTP current state and statistics are also maintained in the operational state. Additionally, the operational state also include the associations state.

1.2. Terminology

The terminology used in this document is aligned to [RFC5905].

1.3. Tree Diagrams

A simplified graphical representation of the data model is used in this document. This document uses the graphical representation of data models defined in [RFC8340].

1.4. Prefixes in Data Node Names

In this document, names of data nodes and other data model objects are often used without a prefix, as long as it is clear from the context in which YANG module each name is defined. Otherwise, names are prefixed using the standard prefix associated with the corresponding YANG module, as shown in Table 1.

Prefix	YANG module	Reference
yang	ietf-yang-types	[RFC6991]
inet	ietf-inet-types	[RFC6991]
if	ietf-interfaces	[RFC8343]
ianach	iana-crypt-hash	[RFC7317]
key-chain	ietf-key-chain	[RFC8177]
acl	ietf-access-control-list	[I-D.ietf-netmod-acl-model]
rt-types	ietf-routing-types	[RFC8294]

Table 1: Prefixes and corresponding YANG modules

2. NTP data model

This document defines the YANG module "ietf-ntp", which has the following structure:

```
module: ietf-ntp
  +--rw ntp!
     +--rw port?                    uint16 {ntp-port}?
     +--rw refclock-master!
     |  +--rw master-stratum?   ntp-stratum
     +--rw authentication
     |  +--rw auth-enabled?          boolean
     |  +--rw trusted-keys* [key-id]
     |  |  +--rw key-id
     |  |          -> /ntp/authentication/authentication-keys/key-id
     |  +--rw authentication-keys* [key-id]
     |     +--rw key-id       uint32
     |     +--rw algorithm?   identityref
     |     +--rw password?    ianach:crypt-hash
     +--rw access-rules
     |  +--rw access-rule* [access-mode]
     |     +--rw access-mode    access-modes
     |     +--rw acl?           -> /acl:acls/acl/name
     +--ro clock-state
     |  +--ro system-status
     |     +--ro clock-state                  ntp-clock-status
     |     +--ro clock-stratum                ntp-stratum
     |     +--ro clock-refid                  union
     |     +--ro associations-address?
     |     |       -> /ntp/associations/address
     |     +--ro associations-local-mode?
     |     |       -> /ntp/associations/local-mode
     |     +--ro associations-isConfigured?
```

```
        |     |       -> /ntp/associations/isConfigured
        |     +--ro nominal-freq                 decimal64
        |     +--ro actual-freq                  decimal64
        |     +--ro clock-precision              uint8
        |     +--ro clock-offset?                decimal64
        |     +--ro root-delay?                  decimal64
        |     +--ro root-dispersion?             decimal64
        |     +--ro reference-time?              yang:date-and-time
        |     +--ro sync-state                   ntp-sync-state
        +--rw unicast-configuration* [address type]
        |  +--rw address           inet:host
        |  +--rw type              unicast-configuration-type
        |  +--rw authentication
        |  |  +--rw (authentication-type)?
        |  |     +--:(symmetric-key)
        |  |        +--rw key-id?   leafref
        |  +--rw prefer?           boolean
        |  +--rw burst?            boolean
        |  +--rw iburst?           boolean
        |  +--rw source?           if:interface-ref
        |  +--rw minpoll?          ntp-minpoll
        |  +--rw maxpoll?          ntp-maxpoll
        |  +--rw port?             uint16 {ntp-port}?
        |  +--rw version?          ntp-version
        +--ro associations* [address local-mode isConfigured]
        |  +--ro address           inet:host
        |  +--ro local-mode        association-modes
        |  +--ro isConfigured      boolean
        |  +--ro stratum?          ntp-stratum
        |  +--ro refid?            union
        |  +--ro authentication?
        |  |       -> /ntp/authentication/authentication-keys/key-id
        |  +--ro prefer?           boolean
        |  +--ro peer-interface?   if:interface-ref
        |  +--ro minpoll?          ntp-minpoll
        |  +--ro maxpoll?          ntp-maxpoll
        |  +--ro port?             uint16 {ntp-port}?
        |  +--ro version?          ntp-version
        |  +--ro reach?            uint8
        |  +--ro unreach?          uint8
        |  +--ro poll?             uint8
        |  +--ro now?              uint32
        |  +--ro offset?           decimal64
        |  +--ro delay?            decimal64
        |  +--ro dispersion?       decimal64
        |  +--ro originate-time?   yang:date-and-time
        |  +--ro receive-time?     yang:date-and-time
        |  +--ro transmit-time?    yang:date-and-time
```

```
         |  +--ro input-time?       yang:date-and-time
         |  +--ro ntp-statistics
         |     +--ro packet-sent?        yang:counter32
         |     +--ro packet-sent-fail?   yang:counter32
         |     +--ro packet-received?    yang:counter32
         |     +--ro packet-dropped?     yang:counter32
         +--rw interfaces
         |  +--rw interface* [name]
         |     +--rw name                if:interface-ref
         |     +--rw broadcast-server!
         |     |  +--rw ttl?              uint8
         |     |  +--rw authentication
         |     |  |  +--rw (authentication-type)?
         |     |  |     +--:(symmetric-key)
         |     |  |        +--rw key-id?   leafref
         |     |  +--rw minpoll?          ntp-minpoll
         |     |  +--rw maxpoll?          ntp-maxpoll
         |     |  +--rw port?             uint16 {ntp-port}?
         |     |  +--rw version?          ntp-version
         |     +--rw broadcast-client!
         |     +--rw multicast-server* [address]
         |     |  +--rw address
         |     |  |       rt-types:ip-multicast-group-address
         |     |  +--rw ttl?              uint8
         |     |  +--rw authentication
         |     |  |  +--rw (authentication-type)?
         |     |  |     +--:(symmetric-key)
         |     |  |        +--rw key-id?   leafref
         |     |  +--rw minpoll?          ntp-minpoll
         |     |  +--rw maxpoll?          ntp-maxpoll
         |     |  +--rw port?             uint16 {ntp-port}?
         |     |  +--rw version?          ntp-version
         |     +--rw multicast-client* [address]
         |     |  +--rw address    rt-types:ip-multicast-group-address
         |     +--rw manycast-server* [address]
         |     |  +--rw address    rt-types:ip-multicast-group-address
         |     +--rw manycast-client* [address]
         |        +--rw address
         |        |       rt-types:ip-multicast-group-address
         |        +--rw authentication
         |        |  +--rw (authentication-type)?
         |        |     +--:(symmetric-key)
         |        |        +--rw key-id?   leafref
         |        +--rw ttl?              uint8
         |        +--rw minclock?         uint8
         |        +--rw maxclock?         uint8
         |        +--rw beacon?           uint8
         |        +--rw minpoll?          ntp-minpoll
```

```
        |        +--rw maxpoll?          ntp-maxpoll
        |        +--rw port?             uint16 {ntp-port}?
        |        +--rw version?          ntp-version
        +--ro ntp-statistics
           +--ro packet-sent?        yang:counter32
           +--ro packet-sent-fail?   yang:counter32
           +--ro packet-received?    yang:counter32
           +--ro packet-dropped?     yang:counter32

     grouping authentication-type-param
       +-- (authentication-type)?
          +--:(symmetric-key)
             +-- key-id?
                     -> /ntp/authentication/authentication-keys/key-id
     grouping comman-attributes
       +-- minpoll?   ntp-minpoll
       +-- maxpoll?   ntp-maxpoll
       +-- port?      uint16 {ntp-port}?
       +-- version?   ntp-version
     grouping association-ref
       +-- associations-address?        -> /ntp/associations/address
       +-- associations-local-mode?     -> /ntp/associations/local-mode
       +-- associations-isConfigured?
               -> /ntp/associations/isConfigured
     grouping statistics
       +-- packet-sent?        yang:counter32
       +-- packet-sent-fail?   yang:counter32
       +-- packet-received?    yang:counter32
       +-- packet-dropped?     yang:counter32
     grouping authentication-key
       +-- key-id?      uint32
       +-- algorithm?   identityref
       +-- password?    ianach:crypt-hash
```

This data model defines one top-level container which includes both the NTP configuration and the NTP running state including access rules, authentication, associations, unicast configurations, interfaces, system status and associations.

3. Relationship with NTPv4-MIB

If the device implements the NTPv4-MIB [RFC5907], data nodes from YANG module can be mapped to table entries in NTPv4-MIB.

The following tables list the YANG data nodes with corresponding objects in the NTPv4-MIB.

```
+--------------------------+--------------------------+
| YANG data nodes in /ntp/ |    NTPv4-MIB objects     |
+--------------------------+--------------------------+
|       ntp-enabled        | ntpEntStatusCurrentMode  |
+--------------------------+--------------------------+

+--------------------------------------+---------------------+
| YANG data nodes in /ntp/associations |  NTPv4-MIB objects  |
+--------------------------------------+---------------------+
|               address                | ntpAssocAddressType |
|                                      |   ntpAssocAddress   |
+--------------------------------------+---------------------+
```

YANG NTP Configuration Data Nodes and Related NTPv4-MIB Objects

```
+---------------------------------+---------------------------------+
|  YANG data nodes in /ntp/clock- |        NTPv4-MIB objects        |
|       state/system-status       |                                 |
+---------------------------------+---------------------------------+
|           clock-state           |     ntpEntStatusCurrentMode     |
|          clock-stratum          |       ntpEntStatusStratum       |
|           clock-refid           |  ntpEntStatusActiveRefSourceId  |
|                                 | ntpEntStatusActiveRefSourceName |
|         clock-precision         |       ntpEntTimePrecision       |
|           clock-offset          |     ntpEntStatusActiveOffset    |
|         root-dispersion         |      ntpEntStatusDispersion     |
+---------------------------------+---------------------------------+

+---------------------------------------+---------------------------+
| YANG data nodes in /ntp/associations/ |     NTPv4-MIB objects     |
+---------------------------------------+---------------------------+
|                address                |    ntpAssocAddressType    |
|                                       |      ntpAssocAddress      |
|                stratum                |      ntpAssocStratum      |
|                 refid                 |       ntpAssocRefId       |
|                 offset                |       ntpAssocOffset      |
|                 delay                 |    ntpAssocStatusDelay    |
|               dispersion              |  ntpAssocStatusDispersion |
|       ntp-statistics/packet-sent      |    ntpAssocStatOutPkts    |
|     ntp-statistics/packet-received    |     ntpAssocStatInPkts    |
|     ntp-statistics/packet-dropped     | ntpAssocStatProtocolError |
+---------------------------------------+---------------------------+
```

YANG NTP State Data Nodes and Related NTPv4-MIB Objects

4. Relationship with RFC 7317

This section describes the relationship with NTP definition in Section 3.2 System Time Management of [RFC7317] . YANG data nodes in /ntp/ also supports per-interface configurations which is not supported in /system/ntp

YANG data nodes in /ntp/	YANG data nodes in /system/ntp
ntp-enabled	enabled
unicast-configuration	server
	server/name
unicast-configuration/address	server/transport/udp/address
unicast-configuration/port	server/transport/udp/port
unicast-configuration/type	server/association-type
unicast-configuration/iburst	server/iburst
unicast-configuration/prefer	server/prefer

YANG NTP Configuration Data Nodes and counterparts in RFC 7317 Objects

5. NTP YANG Module

```
  <CODE BEGINS> file "ietf-ntp@2018-06-22.yang"
module ietf-ntp {

 yang-version 1.1;

 namespace "urn:ietf:params:xml:ns:yang:ietf-ntp";

 prefix "ntp";

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

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

 import ietf-interfaces {
   prefix "if";
   reference "RFC 8343";
 }
```

```
     import iana-crypt-hash {
       prefix "ianach";
       reference "RFC 7317";
     }

     import ietf-key-chain {
       prefix "key-chain";
       reference "RFC 8177";
     }

     import ietf-access-control-list {
       prefix "acl";
       reference "RFC XXXX";
     }
     import ietf-routing-types {
       prefix "rt-types";
       reference "RFC 8294";

     }

     organization
       "IETF NTP (Network Time Protocol) Working Group";

     contact
       "WG Web:  <http://tools.ietf.org/wg/ntp/>
        WG List:  <mailto: ntpwg@lists.ntp.org
        Editor:   Eric Wu
                  <mailto:eric.wu@huawei.com>
        Editor:   Anil Kumar S N
                  <mailto:anil.ietf@gmail.com>
        Editor:   Yi Zhao
                  <mailto:yi.z.zhao@ericsson.com>
        Editor:   Dhruv Dhody
                  <mailto:dhruv.ietf@gmail.com>
        Editor:   Ankit Kumar Sinha
                  <mailto:ankit.ietf@gmail.com>";
     description
       "This YANG module defines essential components for the
        management of a routing subsystem.

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

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

```
         (http://trustee.ietf.org/license-info).

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

      revision 2018-06-22 {
        description
          "Updated revision.";
        reference
          "RFC XXXX: A YANG Data Model for NTP Management";
      }

      /* Typedef Definitions */

      typedef ntp-stratum {
        type uint8 {
          range "1..16";
        }
        description
          "The level of each server in the hierarchy is defined by
          a stratum number. Primary servers are assigned stratum
          one; secondary servers at each lower level are assigned
          stratum numbers one greater than the preceding level";
      }

      typedef ntp-version {
        type uint8 {
          range "1..4";
        }
        default "3";
        description
          "The current NTP version supported by corresponding
          association.";
      }

      typedef ntp-minpoll {
        type uint8 {
          range "4..17";
        }
        default "6";
        description
          "The minimum poll exponent for this NTP association.";
      }

      typedef ntp-maxpoll {
        type uint8 {
          range "4..17";
        }
```

```
        default "10";
        description
          "The maximul poll exponent for this NTP association.";
      }

      typedef access-modes {
        type enumeration {
          enum peer {
            value "0";
            description
              "Sets the fully access authority. Both time
              request and control query can be performed
              on the local NTP service, and the local clock
              can be synchronized to the remote server.";
          }
          enum server {
            value "1";
            description
              "Enables the server access and query.
              Both time requests and control query can be
              performed on the local NTP service, but the
              local clock cannot be synchronized to the
              remote server.";
          }
          enum synchronization {
            value "2";
            description
              "Enables the server to access.
              Only time request can be performed on the
              local NTP service.";
          }
          enum query {
            value "3";
            description
              "Sets the maximum access limitation.
              Control query can be performed only on the
              local NTP service.";
          }
        }
        description
          "This defines NTP acess modes.";
      }

      typedef unicast-configuration-type {
        type enumeration {
          enum server {
            value "0";
            description
```

```
            "Use client association mode. This device
            will not provide synchronization to the
            configured NTP server.";
        }
        enum peer {
          value "1";
          description
            "Use symmetric active association mode.
            This device may provide synchronization
            to the configured NTP server.";
        }
      }
      description
        "This defines NTP unicast mode of operation.";
    }
    typedef association-modes {
      type enumeration {
        enum client {
          value "0";
          description
            "Use client association mode(mode 3).
            This device will not provide synchronization
            to the configured NTP server.";
        }
        enum active {
          value "1";
          description
            "Use symmetric active association mode(mode 1).
            This device may synchronize with its NTP peer,
            or provide synchronization to configured NTP peer.";
        }
        enum passive {
          value "2";
          description
            "Use symmetric passive association mode(mode 2).
            This device has learnt this asso dynamically.
            This device may synchronize with its NTP peer.";
        }
        enum broadcast {
          value "3";
          description
            "Use broadcast mode(mode 5).
            This mode defines that its eigther working
            as broadcast-server or multicast-server.";
        }
        enum broadcast-client {
          value "4";
          description
```

```
            "This mode defines that its eigther working
            as broadcast-client or multicast-client.";
        }
      }
      description
        "This defines NTP association modes.";
    }

    typedef ntp-clock-status {
      type enumeration {
        enum synchronized {
          value "0";
          description
            "Indicates that the local clock has been
            synchronized with an NTP server or
            the reference clock.";
        }
        enum unsynchronized {
          value "1";
          description
            "Indicates that the local clock has not been
            synchronized with any NTP server.";
        }
      }
      description
        "This defines NTP clock status.";
    }

    typedef ntp-sync-state {
      type enumeration {
        enum clock-not-set {
          value "0";
          description
            "Indicates the clock is not updated.";
        }
        enum freq-set-by-cfg {
          value "1";
          description
            "Indicates the clock frequency is set by
            NTP configuration.";
        }
        enum clock-set {
          value "2";
          description
            "Indicates the clock is set.";
        }
        enum freq-not-determined {
          value "3";
```

```
          description
            "Indicates the clock is set but the frequency
            is not determined.";
        }
        enum clock-synchronized {
          value "4";
          description
            "Indicates that the clock is synchronized";
        }
        enum spike {
          value "5";
          description
            "Indicates a time difference of more than 128
            milliseconds is detected between NTP server
            and client clock. The clock change will take
            effect in XXX seconds.";
        }
      }
      description
        "This defines NTP clock sync states.";
    }

   /* feature */
   feature ntp-port {
     description
       "Indicates that the device supports the configuration of
        the port for NTP.

        This is a 'feature', since many implementations do not
        support any port other than the default port.";
   }

    /* Groupings */
    grouping authentication-key {
      description
        "To define an authentication key for a Network Time
        Protocol (NTP) time source.";
      leaf key-id {
        type uint32 {
          range "1..max";
        }
        description
          "Authentication key identifier.";
      }
      leaf algorithm {
        type identityref {
          base key-chain:crypto-algorithm;
        }
```

```
        description
          "Authentication algorithm.";
      }
      leaf password {
        type ianach:crypt-hash;
        description "Clear or encrypted mode for password text.";
      }
    }

    grouping authentication-type-param {
      description
          "Authentication type.";
      choice authentication-type {
        description
          "Type of authentication.";
        case symmetric-key {
          leaf key-id {
            type leafref {
              path "/ntp:ntp/ntp:authentication/"
              + "ntp:authentication-keys/ntp:key-id";
            }
            description
              "Authentication key id referenced in this
              association.";
          }
        }
      }
    }

    grouping statistics {
      description
        "NTP packet statistic.";
      leaf packet-sent {
        type yang:counter32;
        description
          "Indicates the total number of packets sent.";
      }
      leaf packet-sent-fail {
        type yang:counter32;
        description
          "Indicates the number of times packet
          sending failed.";
      }
      leaf packet-received {
        type yang:counter32;
        description
          "Indicates the total number of packets received.";
      }
```

```
        leaf packet-dropped {
          type yang:counter32;
          description
            "Indicates the number of packets dropped.";
        }
      }

      grouping comman-attributes {
        description
          "NTP common attributes for configuration.";
        leaf minpoll {
          type ntp-minpoll;
          description
            "The minimum poll interval used in this association.";
        }
        leaf maxpoll {
          type ntp-maxpoll;
          description
            "The maximul poll interval used in this association.";
        }
        leaf port {
          if-feature ntp-port;
          type uint16 {
            range "123 | 1025..max";
          }
          default "123";
          description
            "Specify the port used to send NTP packets.";
        }
        leaf version {
          type ntp-version;
          description
            "NTP version.";
        }
      }

      grouping association-ref {
        description
          "Reference to NTP association mode";
        leaf associations-address {
            type leafref {
                path "/ntp:ntp/ntp:associations/ntp:address";
            }
            description
              "Indicates the association address
              which result in clock synchronization.";
        }
        leaf associations-local-mode {
```

```
            type leafref {
                path "/ntp:ntp/ntp:associations/ntp:local-mode";
            }
            description
              "Indicates the association local-mode
              which result in clock synchronization.";
        }
        leaf associations-isConfigured {
            type leafref {
                path "/ntp:ntp/ntp:associations/"
                     + "ntp:isConfigured";
            }
            description
              "Indicates the association was configured or
              dynamic which result in clock synchronization.";
        }
      }

      /* Configuration data nodes */
      container ntp {
        presence
          "NTP is enable";
        description
          "Configuration parameters for NTP.";
        leaf port {
          if-feature ntp-port;
          type uint16 {
            range "123 | 1025..max";
          }
          default "123";
          description
            "Specify the port used to send NTP packets.";
        }
        container refclock-master {
          presence
             "NTP master clock is enable";
          description
             "Configures the device as NTP server.";
          leaf master-stratum {
            type ntp-stratum;
            default "16";
            description
              "Stratum level from which NTP
              clients get their time synchronized.";
          }
        }
        container authentication {
         description
```

```
          "Configuration of authentication.";
          leaf auth-enabled {
            type boolean;
            default false;
            description
              "Controls whether NTP authentication is enabled
              or disabled on this device.";
          }
          list trusted-keys {
            key "key-id";
            description
              "List of keys trusted by NTP.";
            leaf key-id {
              type leafref {
                path "/ntp:ntp/ntp:authentication/"
                    + "ntp:authentication-keys/ntp:key-id";
              }
              description
              "The key trusted by NTP.";
            }
          }
          list authentication-keys {
            key "key-id";
            uses authentication-key;
            description
              "List of authentication key.";
          }
      }

      container access-rules {
        description
          "Configuration of access rules.";
          list access-rule {
            key "access-mode";
            description
              "List of access rules.";
            leaf access-mode {
              type access-modes;
              description
                "NTP access mode.";
             }
            leaf acl {
              type leafref {
               path "/acl:acls/acl:acl/acl:name";
              }
              description
                "NTP ACL.";
             }
```

```
            }
      }

      container clock-state {
        config "false";
        description
          "Operational state of the NTP.";

        container system-status {
          description
             "System status of NTP.";
          leaf clock-state {
            type ntp-clock-status;
            mandatory true;
            description "Indicates the state of system clock.";
          }
          leaf clock-stratum {
              type ntp-stratum;
              mandatory true;
              description
                "Indicates the stratum of the reference clock.";
          }
          leaf clock-refid {
              type union {
                  type inet:ipv4-address;
                  type binary {
                      length "4";
                  }
                  type string {
                      length "4";
                  }
              }
              mandatory true;
              description
                "IPv4 address or first 32 bits of the MD5 hash of
                the IPv6 address or reference clock of the peer to
                which clock is synchronized.";
          }

          uses association-ref {
            description
              "Reference to Association mode";
          }
          leaf nominal-freq {
              type decimal64 {
                  fraction-digits 4;
              }
              mandatory true;
```

```
            description
              "Indicates the nominal frequency of the
              local clock, in Hz.";
        }
        leaf actual-freq {
            type decimal64 {
                fraction-digits 4;
            }
            mandatory true;
            description
              "Indicates the actual frequency of the
              local clock, in Hz.";
        }
        leaf clock-precision {
            type uint8;
            mandatory true;
            description
              "Precision of the clock of this system
              in Hz.(prec=2^(-n))";
        }
        leaf clock-offset {
            type decimal64 {
                fraction-digits 4;
            }
            description
              "Offset of clock to synchronized peer,
              in milliseconds.";
        }
        leaf root-delay {
            type decimal64 {
                fraction-digits 2;
            }
            description
              "Total delay along path to root clock,
              in milliseconds.";
        }
        leaf root-dispersion {
            type decimal64 {
                fraction-digits 2;
            }
            description
              "Indicates the dispersion between the local clock
              and the master reference clock, in milliseconds.";
        }
        leaf reference-time {
            type yang:date-and-time;
            description
              "Indicates reference timestamp.";
```

```
          }
          leaf sync-state {
            type ntp-sync-state;
            mandatory true;
            description
              "Indicates the synchronization status of
              the local clock.";
          }
        }
      }
    list unicast-configuration {
      key "address type";
      description
        "list of unicast-configuration.";
      leaf address {
        type inet:host;
        description
          "The address of this association.";
      }
      leaf type {
        type unicast-configuration-type;
        description
          "Type for this NTP configuration";
      }
      container authentication{
        description
          "Authentication type.";
        uses authentication-type-param;
      }
      leaf prefer {
        type boolean;
        default "false";
        description
          "Whether this association is preferred.";
      }
      leaf burst {
        type boolean;
        default "false";
        description
          "Sends a series of packets instead of a single packet
          within each synchronization interval to achieve faster
          synchronization.";
      }
      leaf iburst {
        type boolean;
        default "false";
        description
          "Sends a series of packets instead of a single packet
```

```
          within the initial synchronization interval to achieve
          faster initial synchronization.";
      }
      leaf source {
        type if:interface-ref;
        description
          "The interface whose ip address this association used
          as source address.";
      }
      uses comman-attributes {
        description
          "Common attribute like port, version, min and max
          poll.";
      }
    }
    list associations {
      key "address local-mode isConfigured";
      config "false";
      description
        "list of NTP association.";
      leaf address {
        type inet:host;
        description
          "The address of this association.";
      }
      leaf local-mode {
        type association-modes;
        description
          "Local mode for this NTP association.";
      }
      leaf isConfigured {
        type boolean;
        description
          "Whether this association is configured or
           dynamically learnt.";
      }
      leaf stratum {
        type ntp-stratum;
        description
          "Indicates the stratum of the reference clock.";
      }
      leaf refid {
        type union {
          type inet:ipv4-address;
          type binary {
            length "4";
          }
          type string {
```

```
            length "4";
          }
        }
        description
          "Reference clock type or address for the peer.";
      }
      leaf authentication{
        type leafref {
          path "/ntp:ntp/ntp:authentication/"
              + "ntp:authentication-keys/ntp:key-id";
        }
        description
          "Authentication Key used for this association.";
      }
      leaf prefer {
        type boolean;
        default "false";
        description
          "Whether this association is preferred.";
      }
      leaf peer-interface {
        type if:interface-ref;
        description
          "The interface which is used for communication.";
      }
      uses comman-attributes {
        description
          "Common attribute like port, version, min and
           max poll.";
      }
      leaf reach {
        type uint8;
        description
           "Indicates the reachability of the configured
           server or peer.";
      }
      leaf unreach {
        type uint8;
        description
          "Indicates the unreachability of the configured
          server or peer.";
      }
      leaf poll {
        type uint8;
        description
          "Indicates the polling interval for current,
          in seconds.";
      }
```

```
      leaf now {
        type uint32;
        description
          "Indicates the time since the NTP packet was
          not received or last synchronized, in seconds.";
      }
      leaf offset {
        type decimal64 {
          fraction-digits 4;
        }
        description
          "Indicates the offset between the local clock
          and the superior reference clock.";
      }
      leaf delay {
        type decimal64 {
          fraction-digits 2;
        }
        description
          "Indicates the delay between the local clock
          and the superior reference clock.";
      }
      leaf dispersion {
        type decimal64 {
          fraction-digits 2;
        }
        description
          "Indicates the dispersion between the local
          clock and the superior reference clock.";
      }
      leaf originate-time {
        type yang:date-and-time;
        description
          "Indicates packet originate timestamp(T1).";
      }
      leaf receive-time {
        type yang:date-and-time;
        description
          "Indicates packet receive timestamp(T2).";
      }
      leaf transmit-time {
        type yang:date-and-time;
        description
          "Indicates packet transmit timestamp(T3).";
      }
      leaf input-time {
        type yang:date-and-time;
        description
```

```
          "Indicates packet input timestamp(T4).";
      }
      container ntp-statistics {
        description
          "Per Peer packet send and receive statistic.";
        uses statistics {
            description
              "NTP send and receive packet statistic.";
        }
      }
    }

    container interfaces {
      description
        "Configuration parameters for NTP interfaces.";
      list interface {
        key "name";
        description
          "List of interfaces.";
        leaf name {
              type if:interface-ref;
              description
                "The interface name.";
        }

        container broadcast-server {
          presence
            "NTP broadcast-server is configured";
          description
            "Configuration of broadcast server.";
          leaf ttl {
            type uint8;
            description
              "Specifies the time to live (TTL) of a
              broadcast packet.";
          }
          container authentication{
            description
              "Authentication type.";
              uses authentication-type-param;
          }
          uses comman-attributes {
            description
              "Common attribute like port, version, min and
               max poll.";
          }
        }
```

```
        container broadcast-client {
          presence
            "NTP broadcast-client is configured";
          description
            "Configuration of broadcast-client.";
        }

        list multicast-server {
          key "address";
          description
            "Configuration of multicast server.";
          leaf address {
             type rt-types:ip-multicast-group-address;
             description
               "The IP address to send NTP multicast packets.";
          }
          leaf ttl {
            type uint8;
            description
              "Specifies the time to live (TTL) of a
              multicast packet.";
          }
          container authentication{
            description
              "Authentication type.";
              uses authentication-type-param;
          }
          uses comman-attributes {
            description
              "Common attribute like port, version, min and
               max poll.";
          }
        }
        list multicast-client {
          key "address";
          description
            "Configuration of multicast-client.";
          leaf address {
            type rt-types:ip-multicast-group-address;
            description
              "The IP address of the multicast group to
               join.";
          }
        }
        list manycast-server {
          key "address";
          description
               "Configuration of manycast server.";
```

```
          leaf address {
            type rt-types:ip-multicast-group-address;
            description
              "The multicast group IP address to receive
              manycast client messages .";
          }
          reference
            "RFC 5905";
        }
        list manycast-client {
          key "address";
          description
            "Configuration of manycast-client.";
          leaf address {
            type rt-types:ip-multicast-group-address;
            description
              "The group IP address that the manycast client
              broadcasts the request message to.";
          }
          container authentication{
            description
              "Authentication type.";
              uses authentication-type-param;
          }
          leaf ttl {
            type uint8;
            description
              "Specifies the maximum time to live (TTL) for
              the expanding ring search.";
          }
          leaf minclock {
            type uint8;
            description
              "The minimum manycast survivors in this
              association.";
          }
          leaf maxclock {
            type uint8;
            description
              "The maximum manycast candidates in this
              association.";
          }
          leaf beacon {
            type uint8;
            description
              "The maximum interval between beacons in this
              association.";
          }
```

```
          uses comman-attributes {
            description
              "Common attribute like port, version, min and
               max poll.";
          }
          reference
            "RFC 5905";
        }
      }
    }
  container ntp-statistics {
    config "false";
    description
      "Total NTP packet statistic.";
    uses statistics {
        description
          "NTP send and receive packet statistic.";
    }
  }
}
}

<CODE ENDS>
```

6. Usage Example

6.1. Unicast association

Below is the example on how to configure a preferred unicast server present at 192.0.2.1 running at port 1025 with authentication-key 10 and version 4

```
<edit-config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <target>
    <running/>
  </target>
  <config>
    <ntp xmlns="urn:ietf:params:xml:ns:yang:ietf-ntp">
      <unicast-configuration>
        <address>192.0.2.1</address>
        <type>server</type>
        <prefer>true</prefer>
        <version>4</version>
        <port>1025</port>
        <authentication>
          <symmetric-key>
            <key-id>10</key-id>
          </symmetric-key>
        </authentication>
      </unicast-configuration>
    </ntp>
  </config>
</edit-config>
```

An example with IPv6 would used the an IPv6 address (say 2001:DB8::1) in the "address" leaf with no change in any other data tree.

Below is the example on how to get unicast configuration

```
<get>
  <filter type="subtree">
  <sys:ntp xmlns:sys="urn:ietf:params:xml:ns:yang:ietf-ntp">
      <sys:unicast-configuration>
      </sys:unicast-configuration>
  </sys:ntp>
  </filter>
</get>

<data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ntp xmlns="urn:ietf:params:xml:ns:yang:ietf-ntp">
    <unicast-configuration>
      <address>192.0.2.1</address>
      <type>server</type>
        <authentication>
          <symmetric-key>
            <key-id>10</key-id>
          </symmetric-key>
        </authentication>
      <prefer>true</prefer>
      <burst>false</burst>
```

```
        <iburst>true</iburst>
        <source/>
        <minpoll>6</minpoll>
        <maxpoll>10</maxpoll>
        <port>1025</port>
        <version>4</version>
        <stratum>9</stratum>
        <refid>20.1.1.1</refid>
        <reach>255</reach>
        <unreach>0</unreach>
        <poll>128</poll>
        <now>10</now>
        <offset>0.025</offset>
        <delay>0.5</delay>
        <dispersion>0.6</dispersion>
        <originate-time>10-10-2017 07:33:55.253 Z+05:30
        </originate-time>
        <receive-time>10-10-2017 07:33:55.258 Z+05:30
        </receive-time>
        <transmit-time>10-10-2017 07:33:55.300 Z+05:30
        </transmit-time>
        <input-time>10-10-2017 07:33:55.305 Z+05:30
        </input-time>
        <ntp-statistics>
          <packet-sent>20</packet-sent>
          <packet-sent-fail>0</packet-sent-fail>
          <packet-received>20</packet-received>
          <packet-dropped>0</packet-dropped>
        </ntp-statistics>
      </unicast-configuration>
    </ntp>
  </data>
```

6.2. Refclock master

Below is the example on how to configure reference clock with stratum 8

```
<edit-config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <target>
    <running/>
  </target>
  <config>
    <ntp xmlns="urn:ietf:params:xml:ns:yang:ietf-ntp">
      <refclock-master>
        <master-stratum>8</master-stratum>
      </refclock-master>
    </ntp>
  </config>
</edit-config>
```

Below is the example on how to get reference clock configuration

```
<get>
  <filter type="subtree">
    <sys:ntp xmlns:sys="urn:ietf:params:xml:ns:yang:ietf-ntp">
      <sys:refclock-master>
      </sys:refclock-master>
    </sys:ntp>
  </filter>
</get>

<data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ntp xmlns="urn:ietf:params:xml:ns:yang:ietf-ntp">
    <refclock-master>
      <master-stratum>8</master-stratum>
    </refclock-master>
  </ntp>
</data>
```

6.3. Authentication configuration

Below is the example on how to enable authentication and configure authentication key 10 with mode as md5 and password as abcd

```
<edit-config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <target>
    <running/>
  </target>
  <config>
    <ntp xmlns="urn:ietf:params:xml:ns:yang:ietf-ntp">
      <authentication>
        <auth-enabled>true</auth-enabled>
        <authentication-keys>
          <key-id>10</key-id>
          <algorithm>md5</algorithm>
          <password>abcd</password>
        </authentication-keys>
      </authentication>
    </ntp>
  </config>
</edit-config>
```

Below is the example on how to get authentication related configuration

```
<get>
  <filter type="subtree">
    <sys:ntp xmlns:sys="urn:ietf:params:xml:ns:yang:ietf-ntp">
      <sys:authentication>
      </sys:authentication>
    </sys:ntp>
  </filter>
</get>

<data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ntp xmlns="urn:ietf:params:xml:ns:yang:ietf-ntp">
    <authentication>
      <auth-enabled>false</auth-enabled>
      <trusted-keys/>
      <authentication-keys>
        <key-id>10</key-id>
        <algorithm>md5</algorithm>
        <password>abcd</password>
      </authentication-keys>
    </authentication>
  </ntp>
</data>
```

6.4. Access configuration

Below is the example on how to configure acess type peer associated with acl 2000

```
<edit-config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <target>
    <running/>
  </target>
  <config>
    <ntp xmlns="urn:ietf:params:xml:ns:yang:ietf-ntp">
      <access-rules>
        <access-rule>
          <access-mode>peer</access-mode>
          <acl>2000</acl>
        </access-rule>
      </access-rules>
    </ntp>
  </config>
</edit-config>
```

Below is the example on how to get access related configuration

```
<get>
  <filter type="subtree">
    <sys:ntp xmlns:sys="urn:ietf:params:xml:ns:yang:ietf-ntp">
      <sys:access-rules>
      </sys:access-rules>
    </sys:ntp>
  </filter>
</get>

<data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ntp xmlns="urn:ietf:params:xml:ns:yang:ietf-ntp">
    <access-rules>
      <access-rule>
        <access-mode>peer</access-mode>
        <acl>2000</acl>
      </access-rule>
    </access-rules>
  </ntp>
</data>
```

6.5. Multicast configuration

Below is the example on how to configure multicast-server with address as "224.1.1.1", port as 1025 and authentication keyid as 10

```
<edit-config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <target>
    <running/>
  </target>
  <config>
    <ntp xmlns="urn:ietf:params:xml:ns:yang:ietf-ntp">
      <interfaces>
        <interface>
          <name>Ethernet3/0/0</name>
          <multicast-server>
            <address>224.1.1.1</address>
            <authentication>
              <symmetric-key>
                <key-id>10</key-id>
              </symmetric-key>
            </authentication>
            <port>1025</port>
          </multicast-server>
        </interface>
      </interfaces>
    </ntp>
  </config>
</edit-config>
```

Below is the example on how to get multicast-server related configuration

```
<get>
  <filter type="subtree">
    <sys:ntp xmlns:sys="urn:ietf:params:xml:ns:yang:ietf-ntp">
      <sys:interfaces>
        <sys:interface>
          <sys:multicast-server>
          </sys:multicast-server>
        </sys:interface>
      </sys:interfaces>
    </sys:ntp>
  </filter>
</get>

<data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ntp xmlns="urn:ietf:params:xml:ns:yang:ietf-ntp">
    <interfaces>
      <interface>
        <name>Ethernet3/0/0</name>
        <multicast-server>
          <address>224.1.1.1</address>
          <ttl>224.1.1.1</ttl>
          <authentication>
            <symmetric-key>
              <key-id>10</key-id>
            </symmetric-key>
          </authentication>
          <minpoll>6</minpoll>
          <maxpoll>10</maxpoll>
          <port>1025</port>
          <version>3</version>
        </multicast-server>
      </interface>
    </interfaces>
  </ntp>
</data>
```

Below is the example on how to configure multicast-client with address as "224.1.1.1"

```
<edit-config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <target>
    <running/>
  </target>
  <config>
    <ntp xmlns="urn:ietf:params:xml:ns:yang:ietf-ntp">
      <interfaces>
        <interface>
          <name>Ethernet3/0/0</name>
          <multicast-client>
            <address>224.1.1.1</address>
          </multicast-client>
        </interface>
      </interfaces>
    </ntp>
  </config>
</edit-config>
```

Below is the example on how to get multicast-client related configuration

```
<get>
  <filter type="subtree">
    <sys:ntp xmlns:sys="urn:ietf:params:xml:ns:yang:ietf-ntp">
      <sys:interfaces>
        <sys:interface>
          <sys:multicast-client>
          </sys:multicast-client>
        </sys:interface>
      </sys:interfaces>
    </sys:ntp>
  </filter>
</get>
```

```
<data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ntp xmlns="urn:ietf:params:xml:ns:yang:ietf-ntp">
    <interfaces>
      <interface>
        <name>Ethernet3/0/0</name>
        <multicast-client>
          <address>224.1.1.1</address>
        </multicast-client>
      </interface>
    </interfaces>
  </ntp>
</data>
```

6.6. Manycast configuration

Below is the example on how to configure manycast-client with address as "224.1.1.1", port as 1025 and authentication keyid as 10

```
<edit-config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <target>
    <running/>
  </target>
  <config>
    <ntp xmlns="urn:ietf:params:xml:ns:yang:ietf-ntp">
      <interfaces>
        <interface>
          <name>Ethernet3/0/0</name>
          <manycast-client>
            <address>224.1.1.1</address>
            <authentication>
              <symmetric-key>
                <key-id>10</key-id>
              </symmetric-key>
            </authentication>
            <port>1025</port>
          </manycast-client>
        </interface>
      </interfaces>
    </ntp>
  </config>
</edit-config>
```

Below is the example on how to get manycast-client related configuration

```
<get>
  <filter type="subtree">
    <sys:ntp xmlns:sys="urn:ietf:params:xml:ns:yang:ietf-ntp">
      <sys:interfaces>
        <sys:interface>
          <sys:manycast-client>
          </sys:manycast-client>
        </sys:interface>
      </sys:interfaces>
    </sys:ntp>
  </filter>
</get>

<data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ntp xmlns="urn:ietf:params:xml:ns:yang:ietf-ntp">
    <interfaces>
      <interface>
        <name>Ethernet3/0/0</name>
        <manycast-client>
          <address>224.1.1.1</address>
          <authentication>
            <symmetric-key>
              <key-id>10</key-id>
            </symmetric-key>
          </authentication>
          <ttl>255</ttl>
          <minclock>3</minclock>
          <maxclock>10</maxclock>
          <beacon>6</beacon>
          <minpoll>6</minpoll>
          <maxpoll>10</maxpoll>
          <port>1025</port>
        </manycast-client>
      </interface>
    </interfaces>
  </ntp>
</data>
```

Below is the example on how to configure manycast-server with address as "224.1.1.1"

```
<edit-config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <target>
    <running/>
  </target>
  <config>
    <ntp xmlns="urn:ietf:params:xml:ns:yang:ietf-ntp">
      <interfaces>
        <interface>
          <name>Ethernet3/0/0</name>
          <manycast-server>
            <address>224.1.1.1</address>
          </manycast-server>
        </interface>
      </interfaces>
    </ntp>
  </config>
</edit-config>
```

Below is the example on how to get manycast-server related configuration

```
<get>
  <filter type="subtree">
    <sys:ntp xmlns:sys="urn:ietf:params:xml:ns:yang:ietf-ntp">
      <sys:interfaces>
        <sys:interface>
          <sys:manycast-server>
          </sys:manycast-server>
        </sys:interface>
      </sys:interfaces>
    </sys:ntp>
  </filter>
</get>
```

```
<data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ntp xmlns="urn:ietf:params:xml:ns:yang:ietf-ntp">
    <interfaces>
      <interface>
        <name>Ethernet3/0/0</name>
        <manycast-server>
          <address>224.1.1.1</address>
        </manycast-server>
      </interface>
    </interfaces>
  </ntp>
</data>
```

6.7. Clock state

Below is the example on how to get clock current state

```
<get>
  <filter type="subtree">
    <sys:ntp xmlns:sys="urn:ietf:params:xml:ns:yang:ietf-ntp">
      <sys:clock-state>
      </sys:clock-state>
    </sys:ntp>
  </filter>
</get>

<data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ntp xmlns="urn:ietf:params:xml:ns:yang:ietf-ntp">
    <clock-state>
      <system-status>
        <clock-state>synchronized</clock-state>
        <clock-stratum>7</clock-stratum>
        <clock-refid>192.0.2.1</clock-refid>
        <associations-address>192.0.2.1
        </associations-address>
        <associations-local-mode>client
        </associations-local-mode>
        <associations-isConfigured>yes
        </associations-isConfigured>
        <nominal-freq>100.0</nominal-freq>
        <actual-freq>100.0</actual-freq>
        <clock-precision>18</clock-precision>
        <clock-offset>0.025</clock-offset>
        <root-delay>0.5</root-delay>
        <root-dispersion>0.8</root-dispersion>
        <reference-time>10-10-2017 07:33:55.258 Z+05:30
        </reference-time>
        <sync-state>clock-synchronized</sync-state>
      </system-status>
    </clock-state>
  </ntp>
</data>
```

6.8. Get all association

Below is the example on how to get all association present

```
<get>
  <filter type="subtree">
    <sys:ntp xmlns:sys="urn:ietf:params:xml:ns:yang:ietf-ntp">
      <sys:associations>
      </sys:associations>
    </sys:ntp>
  </filter>
</get>

<data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ntp xmlns="urn:ietf:params:xml:ns:yang:ietf-ntp">
    <associations>
      <address>192.0.2.1</address>
      <stratum>9</stratum>
      <refid>20.1.1.1</refid>
      <local-mode>client</local-mode>
      <isConfigured>true</isConfigured>
      <authentication-key>10</authentication-key>
      <prefer>true</prefer>
      <peer-interface>Ethernet3/0/0</peer-interface>
      <minpoll>6</minpoll>
      <maxpoll>10</maxpoll>
      <port>1025</port>
      <version>4</version>
      <reach>255</reach>
      <unreach>0</unreach>
      <poll>128</poll>
      <now>10</now>
      <offset>0.025</offset>
      <delay>0.5</delay>
      <dispersion>0.6</dispersion>
      <originate-time>10-10-2017 07:33:55.253 Z+05:30
      </originate-time>
      <receive-time>10-10-2017 07:33:55.258 Z+05:30
      </receive-time>
      <transmit-time>10-10-2017 07:33:55.300 Z+05:30
      </transmit-time>
      <input-time>10-10-2017 07:33:55.305 Z+05:30
      </input-time>
      <ntp-statistics>
        <packet-sent>20</packet-sent>
        <packet-sent-fail>0</packet-sent-fail>
        <packet-received>20</packet-received>
        <packet-dropped>0</packet-dropped>
      </ntp-statistics>
    </associations>
  </ntp>
</data>
```

6.9. Global statistic

Below is the example on how to get clock current state

```
<get>
  <filter type="subtree">
    <sys:ntp xmlns:sys="urn:ietf:params:xml:ns:yang:ietf-ntp">
      <sys:ntp-statistics>
      </sys:ntp-statistics>
    </sys:ntp>
  </filter>
</get>

<data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ntp xmlns="urn:ietf:params:xml:ns:yang:ietf-ntp">
    <ntp-statistics>
      <packet-sent>30</packet-sent>
      <packet-sent-fail>5</packet-sent-fail>
      <packet-received>20</packet-received>
      <packet-dropped>2</packet-dropped>
    </ntp-statistics>
  </ntp>
</data>
```

7. IANA Considerations

This document registers a URI in the "IETF XML Registry" [RFC3688]. Following the format in RFC 3688, the following registration has been made.

URI: urn:ietf:params:xml:ns:yang:ietf-ntp

Registrant Contact: The NETMOD WG of the IETF.

XML: N/A; the requested URI is an XML namespace.

This document registers a YANG module in the "YANG Module Names" registry [RFC6020].

Name: ietf-ntp

Namespace: urn:ietf:params:xml:ns:yang:ietf-ntp

Prefix: ntp

Reference: RFC XXXX

8. Security Considerations

The YANG module specified in this document defines a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC5246].

The NETCONF access control model [RFC8341] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

There are a number of data nodes defined in this YANG module that are writable/creatable/deletable (i.e., config true, which is the default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g., edit-config) to these data nodes without proper protection can have a negative effect on network operations. These are the subtrees and data nodes and their sensitivity/vulnerability:

/ntp/port - This data node specify the port number to be used to send NTP packets. Unexpected changes could lead to disruption and/or network misbehavior.

/ntp/authentication and /ntp/access-rules - The entries in the list include the authentication and access control configurations. Car should be taken while setting these parameters.

/ntp/unicast-configuration - The entries in the list include all unicast configurations (server or peer mode), and indirectly creates or modify the NTP associations. Unexpected changes could lead to disruption and/or network misbehavior.

/ntp/interfaces/interface - The entries in the list inclide all per-interface configurations related to broadcast, multicast and manycast mode, and indirectly creates or modify the NTP associations. Unexpected changes could lead to disruption and/or network misbehavior.

Some of the readable data nodes in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get, get-config, or notification) to these data nodes. These are the subtrees and data nodes and their sensitivity/vulnerability:

/ntp/associations - The entries in the list includes all active NTP associations of all modes. Unauthorized access to this needs to be curtailed.

9. Acknowledgments

The authors would like to express their thanks to Sladjana Zoric, Danny Mayer, Harlan Stenn, Ulrich Windl, Miroslav Lichvar, and Maurice Angermann for their review and suggestions.

10. References

10.1. Normative References

[I-D.ietf-netmod-acl-model]
Jethanandani, M., Huang, L., Agarwal, S., and D. Blair, "Network Access Control List (ACL) YANG Data Model", draft-ietf-netmod-acl-model-19 (work in progress), April 2018.

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

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <https://www.rfc-editor.org/info/rfc3688>.

[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>.

[RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, DOI 10.17487/RFC5905, June 2010, <https://www.rfc-editor.org/info/rfc5905>.

[RFC5907] Gerstung, H., Elliott, C., and B. Haberman, Ed., "Definitions of Managed Objects for Network Time Protocol Version 4 (NTPv4)", RFC 5907, DOI 10.17487/RFC5907, June 2010, <https://www.rfc-editor.org/info/rfc5907>.

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

[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>.

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <https://www.rfc-editor.org/info/rfc6991>.

[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>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8177] Lindem, A., Ed., Qu, Y., Yeung, D., Chen, I., and J. Zhang, "YANG Data Model for Key Chains", RFC 8177, DOI 10.17487/RFC8177, June 2017, <https://www.rfc-editor.org/info/rfc8177>.

[RFC8294] Liu, X., Qu, Y., Lindem, A., Hopps, C., and L. Berger, "Common YANG Data Types for the Routing Area", RFC 8294, DOI 10.17487/RFC8294, December 2017, <https://www.rfc-editor.org/info/rfc8294>.

[RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <https://www.rfc-editor.org/info/rfc8340>.

[RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", STD 91, RFC 8341, DOI 10.17487/RFC8341, March 2018, <https://www.rfc-editor.org/info/rfc8341>.

[RFC8343] Bjorklund, M., "A YANG Data Model for Interface Management", RFC 8343, DOI 10.17487/RFC8343, March 2018, <https://www.rfc-editor.org/info/rfc8343>.

10.2. Informative References

[RFC7317] Bierman, A. and M. Bjorklund, "A YANG Data Model for System Management", RFC 7317, DOI 10.17487/RFC7317, August 2014, <https://www.rfc-editor.org/info/rfc7317>.

[RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture (NMDA)", RFC 8342, DOI 10.17487/RFC8342, March 2018, <https://www.rfc-editor.org/info/rfc8342>.

Authors' Addresses

Nan Wu
Huawei
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: eric.wu@huawei.com

Anil Kumar S N
RtBrick Inc.
Bangalore, Kanataka
India

Email: anil.ietf@gmail.com

Yi Zhao
Ericsson
China Digital Kingdom Bld., No.1 WangJing North Rd.
Beijing 100102
China

Email: yi.z.zhao@ericsson.com

Dhruv Dhody
Huawei
Divyashree Techno Park, Whitefield
Bangalore, Kanataka 560066
India

Email: dhruv.ietf@gmail.com

Ankit kumar Sinha
RtBrick Inc.
Bangalore, Kanataka
India

Email: ankit.ietf@gmail.com