Internet Engineering Task Force (IETF)
Request for Comments: 9249
Category: Standards Track
ISSN: 2070-1721

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

July 2022

A YANG Data Model for NTP

Abstract

This document defines a YANG data model that can be used to configure and manage Network Time Protocol (NTP) version 4. It can also be used to configure and manage version 3. The data model includes configuration data and state data.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc9249.

Table of Contents

1. Introduction

This document defines a YANG data model [RFC7950] that can be used to configure and manage Network Time Protocol version 4 [RFC5905]. Note that the model could also be used to configure and manage NTPv3 [RFC1305] (see Section 7).

The data model covers configuration of system parameters of NTP such as access rules, authentication and VPN Routing and Forwarding (VRF) binding, and various modes of NTP and per-interface parameters. It also provides access to 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 (NMDA)" [RFC8342]. NTP current state and statistics are also maintained in the operational state. The operational state also includes the NTP association state.

1.2. Terminology

The terminology used in this document is aligned with [RFC5905] and [RFC1305].

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]
sys	ietf-system	[RFC7317]
acl	ietf-access-control-list	[RFC8519]
rt-types	ietf-routing-types	[RFC8294]
nacm	ietf-netconf-acm	[RFC8341]

Table 1: Prefixes and Corresponding YANG Modules

1.5. References in the Model

The following documents are referenced in the model defined in this document.

Title	Reference
Network Time Protocol Version 4: Protocol and Algorithms Specification	[RFC5905]
Common YANG Data Types	[RFC6991]
A YANG Data Model for System Management	[RFC7317]
Common YANG Data Types for the Routing Area	[RFC8294]
Network Configuration Access Control Model	[RFC8341]
A YANG Data Model for Interface Management	[RFC8343]
YANG Data Model for Network Access Control Lists (ACLs)	[RFC8519]
Message Authentication Code for the Network Time Protocol	[RFC8573]
The AES-CMAC Algorithm	[RFC4493]
The MD5 Message-Digest Algorithm	[RFC1321]
US Secure Hash Algorithm 1 (SHA1)	[RFC3174]
FIPS 180-4: Secure Hash Standard (SHS)	[SHS]

Table 2: References in the YANG Module

1.6. 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.

2. NTP Data Model

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

```
module: ietf-ntp
  +--rw ntp!
     +--rw port?                    inet:port-number {ntp-port}?
     +--rw refclock-master!
     |  +--rw master-stratum?   ntp-stratum
     +--rw authentication {authentication}?
     |  +--rw auth-enabled?          boolean
     |  +--rw authentication-keys* [keyid]
     |     +--rw keyid       uint32
     |     +--...
     +--rw access-rules {access-rules}?
     |  +--rw access-rule* [access-mode]
     |     +--rw access-mode    identityref
     |     +--rw acl?           -> /acl:acls/acl/name
     +--ro clock-state
     |  +--ro system-status
     |     +--ro clock-state                 identityref
     |     +--ro clock-stratum               ntp-stratum
     |     +--ro clock-refid                 refid
     |     +--...
     +--rw unicast-configuration* [address type]
     |       {unicast-configuration}?
     |  +--rw address           inet:ip-address
     |  +--rw type              identityref
     |  +--...
     +--rw associations
     |  +--ro association* [address local-mode isconfigured]
     |     +--ro address           inet:ip-address
     |     +--ro local-mode        identityref
     |     +--ro isconfigured      boolean
     |     +--...
     |     +--ro ntp-statistics
     |        +--...
     +--rw interfaces
     |  +--rw interface* [name]
     |     +--rw name                 if:interface-ref
     |     +--rw broadcast-server! {broadcast-server}?
     |     |  +--...
     |     +--rw broadcast-client! {broadcast-client}?
     |     +--rw multicast-server* [address] {multicast-server}?
     |     |  +--rw address
     |     |  |       rt-types:ip-multicast-group-address
     |     |  +--...
     |     +--rw multicast-client* [address] {multicast-client}?
     |     |  +--rw address    rt-types:ip-multicast-group-address
     |     +--rw manycast-server* [address] {manycast-server}?
     |     |  +--rw address    rt-types:ip-multicast-group-address
     |     +--rw manycast-client* [address] {manycast-client}?
     |        +--rw address
     |        |       rt-types:ip-multicast-group-address
     |        +--...
     +--ro ntp-statistics
        +--...

  rpcs:
    +---x statistics-reset
       +---w input
          +---w (association-or-all)?
```

```
              +--:(association)
              |  +---w associations-address?
              |  |       -> /ntp/associations/association/address
              |  +---w associations-local-mode?
              |  |       -> /ntp/associations/association/local-mode
              |  +---w associations-isconfigured?
              |          -> /ntp/associations/association/isconfigured
              +--:(all)
```

The full data model tree for the YANG module "ietf-ntp" is in Appendix A.

This data model defines one top-level container that 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 the YANG module can be mapped to table entries in the NTPv4-MIB.

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

YANG Data Nodes in /ntp/ clock-state/system-status	NTPv4-MIB Objects
clock-state	ntpEntStatusCurrentMode
clock-stratum	ntpEntStatusStratum
clock-refid	ntpEntStatusActiveRefSourceId ntpEntStatusActiveRefSourceName
clock-precision	ntpEntTimePrecision
clock-offset	ntpEntStatusActiveOffset
root-dispersion	ntpEntStatusDispersion

Table 3: YANG NTP Data Nodes in /ntp/clock-state/system-status and Related NTPv4-MIB Objects

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

Table 4: YANG NTP Data Nodes in /ntp/associations/ and Related NTPv4-MIB Objects

4. Relationship with RFC 7317

This section describes the relationship with definition of NTP in Section 3.2 (System Time Management) of [RFC7317]. YANG data nodes in /ntp/ also support per-interface configuration, which is not supported in /system/ntp. If the YANG data model defined in this document is implemented, then /system/ntp SHOULD NOT be used and MUST be ignored.

YANG Data Nodes in /ntp/	YANG Data Nodes in /system/ntp
ntp!	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

Table 5: YANG NTP Configuration Data Nodes and Counterparts from RFC 7317

5. Access Rules

The access rules in this section refer to the on-the-wire access control to the NTP service and are completely independent of any management API access control, e.g., NETCONF Access Control Model (NACM) [RFC8341].

An Access Control List (ACL) is one of the basic elements used to configure device-forwarding behavior. An ACL is a user-ordered set of rules that is used to filter traffic on a networking device.

As per [RFC1305] (for NTPv3) and [RFC5905] (for NTPv4), NTP could include an access-control feature that prevents unauthorized access and that controls which peers are allowed to update the local clock. Further, it is useful to differentiate between the various kinds of access and attach a different acl-rule to each. For this, the YANG module allows such configuration via /ntp/access-rules. The access-rule itself is configured via [RFC8519].

The following access-modes are supported:

Peer: Permit others to synchronize their time with the NTP entity or vice versa. NTP control queries are also accepted.

Server: Permit others to synchronize their time with the NTP entity, but vice versa is not supported. NTP control queries are accepted.

Server-only: Permit others to synchronize their time with the NTP entity, but vice versa is not supported. NTP control queries are not accepted.

Query-only: Only control queries are accepted.

Query-only is the most restricted whereas the peer is the full access authority. The ability to give different ACL rules for different access-modes allows for a greater control by the operator.

6. Key Management

As per [RFC1305] (for NTPv3) and [RFC5905] (for NTPv4), when authentication is enabled, NTP employs a crypto-checksum, computed by the sender and checked by the receiver, together with a set of predistributed algorithms, and cryptographic keys indexed by a key identifier included in the NTP message. This keyid is a 32-bit unsigned integer that MUST be configured on the NTP peers before the authentication can be used. For this reason, this YANG module allows such configuration via /ntp/authentication/authentication-keys/. Further at the time of configuration of NTP association (for example, unicast server), the keyid is specified.

The 'nacm:default-deny-all' is used to prevent retrieval of the actual key information after it is set.

7. NTP Version

This YANG data model allows a version to be configured for the NTP association, i.e., an operator can control the use of NTPv3 [RFC1305] or NTPv4 [RFC5905] for each association it forms. This allows backward compatibility with a legacy system. Note that NTPv3 [RFC1305] is obsoleted by NTPv4 [RFC5905].

8. NTP YANG Module

<CODE BEGINS> file "ietf-ntp@2022-07-05.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: Common YANG Data Types";
  }
  import ietf-inet-types {
    prefix inet;
    reference
      "RFC 6991: Common YANG Data Types";
  }
  import ietf-interfaces {
    prefix if;
    reference
      "RFC 8343: A YANG Data Model for Interface Management";
  }
```

```
import ietf-system {
  prefix sys;
  reference
    "RFC 7317: A YANG Data Model for System Management";
}
import ietf-access-control-list {
  prefix acl;
  reference
    "RFC 8519: YANG Data Model for Network Access Control
     Lists (ACLs)";
}
import ietf-routing-types {
  prefix rt-types;
  reference
    "RFC 8294: Common YANG Data Types for the Routing Area";
}
import ietf-netconf-acm {
  prefix nacm;
  reference
    "RFC 8341: Network Configuration Access Control Model";
}

organization
  "IETF NTP (Network Time Protocol) Working Group";
contact
  "WG Web:  <https://datatracker.ietf.org/wg/ntp/>
   WG List:  <mailto: ntp@ietf.org
   Editor:   Dhruv Dhody
             <mailto:dhruv.ietf@gmail.com>
   Editor:   Ankit Kumar Sinha
             <mailto:ankit.ietf@gmail.com>";
description
  "This document defines a YANG data model that can be used
   to configure and manage  Network Time Protocol (NTP) version 4.
   It can also be used to configure and manage version 3.
   The data model includes configuration data and state data.

   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 (RFC 2119) (RFC 8174) when, and only when,
   they appear in all capitals, as shown here.

   Copyright (c) 2022 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 Revised BSD License
   set forth in Section 4.c of the IETF Trust's Legal Provisions
   Relating to IETF Documents
   (https://trustee.ietf.org/license-info).

   This version of this YANG module is part of RFC 9249; see the
   RFC itself for full legal notices.";

revision 2022-07-05 {
  description
    "Initial revision";
  reference
    "RFC 9249: A YANG Data Model for NTP";
}

/* Typedef Definitions */

typedef ntp-stratum {
```

```
  type uint8 {
    range "1..16";
  }
  description
    "The level of each server in the hierarchy is defined by
     a stratum.  Primary servers are assigned with stratum
     one; secondary servers at each lower level are assigned with
     one stratum greater than the preceding level.";
  reference
    "RFC 5905: Network Time Protocol Version 4: Protocol and
     Algorithms Specification, Section 3";
}

typedef ntp-version {
  type uint8 {
    range "3..max";
  }
  default "4";
  description
    "The current NTP version supported by the corresponding
     association";
  reference
    "RFC 5905: Network Time Protocol Version 4: Protocol and
     Algorithms Specification, Section 1";
}

typedef refid {
  type union {
    type inet:ipv4-address;
    type uint32;
    type string {
      length "4";
    }
  }
  description
    "A code identifying the particular server or reference
     clock.  The interpretation depends upon stratum.  It
     could be an IPv4 address, the first 32 bits of the MD5 hash
     of the IPv6 address, or a string for the Reference Identifier
     and kiss codes.  Some examples:

     -- a refclock ID like '127.127.1.0' for local clock sync

     -- uni/multi/broadcast associations for IPv4 will look like
        '203.0.113.1' and '0x4321FEDC' for IPv6

     -- sync with a primary source will look like 'DCN', 'NIST',
        'ATOM'

     -- kiss codes will look like 'AUTH', 'DROP', or 'RATE'

     Note that the use of an MD5 hash for IPv6 addresses is not
     for cryptographic purposes.";
  reference
    "RFC 5905: Network Time Protocol Version 4: Protocol and
     Algorithms Specification, Section 7.3";
}

typedef ntp-date-and-time {
  type union {
    type yang:date-and-time;
    type uint8;
  }
  description
    "Follows the date-and-time format when valid values exist.
     Otherwise, allows for setting a special value such as
```

```
     zero.";
  reference
    "RFC 6991: Common YANG Data Types";
}

typedef log2seconds {
  type int8;
  description
    "An 8-bit signed integer that represents signed log2
     seconds.";
}

/* features */

feature ntp-port {
  description
    "Support for NTP port configuration";
  reference
    "RFC 5905: Network Time Protocol Version 4: Protocol and
     Algorithms Specification, Section 7.2";
}

feature authentication {
  description
    "Support for NTP symmetric key authentication";
  reference
    "RFC 5905: Network Time Protocol Version 4: Protocol and
     Algorithms Specification, Section 7.3";
}

feature deprecated {
  description
    "Support deprecated MD5-based authentication (RFC 8573),
     SHA-1, or any other deprecated authentication mechanism.
     It is enabled to support legacy compatibility when secure
     cryptographic algorithms are not available to use.
     It is also used to configure keystrings in ASCII format.";
  reference
    "RFC 1321: The MD5 Message-Digest Algorithm,
     RFC 3174: US Secure Hash Algorithm 1 (SHA1),
     SHS: Secure Hash Standard (SHS) (FIPS PUB 180-4)";
}

feature hex-key-string {
  description
    "Support hexadecimal key string";
}

feature access-rules {
  description
    "Support for NTP access control";
  reference
    "RFC 5905: Network Time Protocol Version 4: Protocol and
     Algorithms Specification, Section 9.2";
}

feature unicast-configuration {
  description
    "Support for NTP client/server or active/passive
     in unicast";
  reference
    "RFC 5905: Network Time Protocol Version 4: Protocol and
     Algorithms Specification, Section 3";
}

feature broadcast-server {
```

```
  description
    "Support for broadcast server";
  reference
    "RFC 5905: Network Time Protocol Version 4: Protocol and
     Algorithms Specification, Section 3";
}

feature broadcast-client {
  description
    "Support for broadcast client";
  reference
    "RFC 5905: Network Time Protocol Version 4: Protocol and
     Algorithms Specification, Section 3";
}

feature multicast-server {
  description
    "Support for multicast server";
  reference
    "RFC 5905: Network Time Protocol Version 4: Protocol and
     Algorithms Specification, Section 3.1";
}
feature multicast-client {
  description
    "Support for multicast client";
  reference
    "RFC 5905: Network Time Protocol Version 4: Protocol and
     Algorithms Specification, Section 3.1";
}

feature manycast-server {
  description
    "Support for manycast server";
  reference
    "RFC 5905: Network Time Protocol Version 4: Protocol and
     Algorithms Specification, Section 3.1";
}

feature manycast-client {
  description
    "Support for manycast client";
  reference
    "RFC 5905: Network Time Protocol Version 4: Protocol and
     Algorithms Specification, Section 3.1";
}

/* Identity */
/* unicast-configurations types */

identity unicast-configuration-type {
  if-feature "unicast-configuration";
  description
    "This defines NTP unicast mode of operation as used
     for unicast-configurations.";
}

identity uc-server {
  if-feature "unicast-configuration";
  base unicast-configuration-type;
  description
    "Use client association mode where the unicast server
     address is configured.";
}

identity uc-peer {
  if-feature "unicast-configuration";
```

```
  base unicast-configuration-type;
  description
    "Use symmetric active association mode where the peer
     address is configured.";
}

/* association-modes */

identity association-mode {
  description
    "The NTP association modes";
  reference
    "RFC 5905: Network Time Protocol Version 4: Protocol and
     Algorithms Specification, Section 3";
}

identity active {
  base association-mode;
  description
    "Use symmetric active association mode (mode 1).
     This device may synchronize with its NTP peer
     or provide synchronization to a configured NTP peer.";
}

identity passive {
  base association-mode;
  description
    "Use symmetric passive association mode (mode 2).
     This device has learned this association dynamically.
     This device may synchronize with its NTP peer.";
}

identity client {
  base association-mode;
  description
    "Use client association mode (mode 3).
     This device will not provide synchronization
     to the configured NTP server.";
}

identity server {
  base association-mode;
  description
    "Use server association mode (mode 4).
     This device will provide synchronization to
     NTP clients.";
}

identity broadcast-server {
  base association-mode;
  description
    "Use broadcast server mode (mode 5).
     This mode defines that it's either working
     as a broadcast server or a multicast server.";
}

identity broadcast-client {
  base association-mode;
  description
    "This mode defines that it's either working
     as a broadcast client (mode 6) or a multicast client.";
}

/* access-mode */

identity access-mode {
```

```
  if-feature "access-rules";
  description
    "This defines NTP access-modes.  These identify
     how the ACL is applied with NTP.";
  reference
    "RFC 5905: Network Time Protocol Version 4: Protocol and
     Algorithms Specification, Section 9.2";
}

identity peer-access-mode {
  if-feature "access-rules";
  base access-mode;
  description
    "Permit others to synchronize their time with this NTP
     or vice versa.
     NTP control queries are also accepted.  This enables
     full access authority.";
}

identity server-access-mode {
  if-feature "access-rules";
  base access-mode;
  description
    "Permit others to synchronize their time with this NTP
     entity, but vice versa is not supported.  NTP control
     queries are accepted.";
}

identity server-only-access-mode {
  if-feature "access-rules";
  base access-mode;
  description
    "Permit others to synchronize their time with this NTP
     entity, but vice versa is not supported.  NTP control
     queries are not accepted.";
}

identity query-only-access-mode {
  if-feature "access-rules";
  base access-mode;
  description
    "Only control queries are accepted.";
}

/* clock-state */

identity clock-state {
  description
    "This defines NTP clock status at a high level.";
}

identity synchronized {
  base clock-state;
  description
    "Indicates that the local clock has been synchronized with
     an NTP server or the reference clock.";
}

identity unsynchronized {
  base clock-state;
  description
    "Indicates that the local clock has not been synchronized
     with any NTP server.";
}

/* ntp-sync-state */
```

```
identity ntp-sync-state {
  description
    "This defines NTP clock sync state at a more granular
     level.  Referred to as 'Clock state definitions' in
     RFC 5905.";
  reference
    "RFC 5905: Network Time Protocol Version 4: Protocol and
     Algorithms Specification, Appendix A.1.1";
}

identity clock-never-set {
  base ntp-sync-state;
  description
    "Indicates the clock was never set.";
}

identity freq-set-by-cfg {
  base ntp-sync-state;
  description
    "Indicates the clock frequency is set by
     NTP configuration or file.";
}

identity spike {
  base ntp-sync-state;
  description
    "Indicates a spike is detected.";
}

identity freq {
  base ntp-sync-state;
  description
    "Indicates the frequency mode.";
}

identity clock-synchronized {
  base ntp-sync-state;
  description
    "Indicates that the clock is synchronized.";
}

/* crypto-algorithm */

identity crypto-algorithm {
  description
    "Base identity of cryptographic algorithm options.";
}

identity md5 {
  if-feature "deprecated";
  base crypto-algorithm;
  description
    "The MD5 algorithm.  Note that RFC 8573
     deprecates the use of MD5-based authentication.";
  reference
    "RFC 1321: The MD5 Message-Digest Algorithm";
}

identity sha-1 {
  if-feature "deprecated";
  base crypto-algorithm;
  description
    "The SHA-1 algorithm";
  reference
    "RFC 3174: US Secure Hash Algorithm 1 (SHA1)";
```

```
}

identity hmac-sha-1 {
  if-feature "deprecated";
  base crypto-algorithm;
  description
    "HMAC-SHA-1 authentication algorithm";
  reference
    "SHS: Secure Hash Standard (SHS) (FIPS PUB 180-4)";
}

identity hmac-sha1-12 {
  if-feature "deprecated";
  base crypto-algorithm;
  description
    "The HMAC-SHA1-12 algorithm";
}

identity hmac-sha-256 {
  description
    "HMAC-SHA-256 authentication algorithm";
  reference
    "SHS: Secure Hash Standard (SHS) (FIPS PUB 180-4)";
}

identity hmac-sha-384 {
  description
    "HMAC-SHA-384 authentication algorithm";
  reference
    "SHS: Secure Hash Standard (SHS) (FIPS PUB 180-4)";
}

identity hmac-sha-512 {
  description
    "HMAC-SHA-512 authentication algorithm";
  reference
    "SHS: Secure Hash Standard (SHS) (FIPS PUB 180-4)";
}

identity aes-cmac {
  base crypto-algorithm;
  description
    "The AES-CMAC algorithm -- required by
     RFC 8573 for MAC for the NTP.";
  reference
    "RFC 4493: The AES-CMAC Algorithm,
     RFC 8573: Message Authentication Code for the Network
     Time Protocol";
}

/* Groupings */

grouping key {
  description
    "The key";
  nacm:default-deny-all;
  choice key-string-style {
    description
      "Key string styles";
    case keystring {
      leaf keystring {
        if-feature "deprecated";
        type string;
        description
          "Key string in ASCII format";
      }
```

```
    }
    case hexadecimal {
      if-feature "hex-key-string";
      leaf hexadecimal-string {
        type yang:hex-string;
        description
          "Key in hexadecimal string format.  When compared
           to ASCII, specification in hexadecimal affords
           greater key entropy with the same number of
           internal key-string octets.  Additionally, it
           discourages use of well-known words or
           numbers.";
      }
    }
  }
}

grouping authentication-key {
  description
    "To define an authentication key for an NTP
     time source.";
  leaf keyid {
    type uint32 {
      range "1..max";
    }
    description
      "Authentication key identifier";
  }
  leaf algorithm {
    type identityref {
      base crypto-algorithm;
    }
    description
      "Authentication algorithm.  Note that RFC 8573
       deprecates the use of MD5-based authentication
       and recommends AES-CMAC.";
  }
  container key {
    uses key;
    description
      "The key.  Note that RFC 8573 deprecates the use
       of MD5-based authentication.";
  }
  leaf istrusted {
    type boolean;
    description
      "Keyid is trusted or not";
  }
  reference
    "RFC 5905: Network Time Protocol Version 4: Protocol and
     Algorithms Specification, Sections 7.3 and 7.4";
}

grouping authentication {
  description
    "Authentication";
  choice authentication-type {
    description
      "Type of authentication";
    case symmetric-key {
      leaf keyid {
        type leafref {
          path "/ntp:ntp/ntp:authentication/"
             + "ntp:authentication-keys/ntp:keyid";
        }
        description
```

```
          "Authentication key id referenced in this
           association.";
      }
    }
  }
}

grouping statistics {
  description
    "NTP packet statistic";
  leaf discontinuity-time {
    type ntp-date-and-time;
    description
      "The time on the most recent occasion at which any one or
       more of these NTP counters suffered a discontinuity.  If
       no such discontinuities have occurred, then this node
       contains the time the NTP association was
       (re-)initialized.";
  }
  leaf packet-sent {
    type yang:counter32;
    description
      "The total number of NTP packets delivered to the
       transport service by this NTP entity for this
       association.
       Discontinuities in the value of this counter can occur
       upon cold start, reinitialization of the NTP entity or the
       management system, and at other times.";
  }
  leaf packet-sent-fail {
    type yang:counter32;
    description
      "The number of times NTP packet sending failed.";
  }
  leaf packet-received {
    type yang:counter32;
    description
      "The total number of NTP packets delivered to the
       NTP entity from this association.
       Discontinuities in the value of this counter can occur
       upon cold start, reinitialization of the NTP entity or the
       management system, and at other times.";
  }
  leaf packet-dropped {
    type yang:counter32;
    description
      "The total number of NTP packets that were delivered
       to this NTP entity from this association and that this
       entity was not able to process due to an NTP error.
       Discontinuities in the value of this counter can occur
       upon cold start, reinitialization of the NTP entity or the
       management system, and at other times.";
  }
}

grouping common-attributes {
  description
    "NTP common attributes for configuration";
  leaf minpoll {
    type log2seconds;
    default "6";
    description
      "The minimum poll interval used in this association";
    reference
      "RFC 5905: Network Time Protocol Version 4: Protocol and
       Algorithms Specification, Section 7.2";
```

```
  }
  leaf maxpoll {
    type log2seconds;
    default "10";
    description
      "The maximum poll interval used in this association";
    reference
      "RFC 5905: Network Time Protocol Version 4: Protocol and
       Algorithms Specification, Section 7.2";
  }
  leaf port {
    if-feature "ntp-port";
    type inet:port-number {
      range "123 | 1024..max";
    }
    default "123";
    description
      "Specify the port used to send NTP packets.";
    reference
      "RFC 5905: Network Time Protocol Version 4: Protocol and
       Algorithms Specification, Section 7.2";
  }
  leaf version {
    type ntp-version;
    description
      "NTP version";
  }
  reference
    "RFC 5905: Network Time Protocol Version 4: Protocol and
     Algorithms Specification";
}

grouping association-ref {
  description
    "Reference to NTP association mode";
  leaf associations-address {
    type leafref {
      path "/ntp:ntp/ntp:associations/ntp:association"
         + "/ntp:address";
    }
    description
      "Indicates the association's address
       that results in clock synchronization.";
  }
  leaf associations-local-mode {
    type leafref {
      path "/ntp:ntp/ntp:associations/ntp:association"
         + "/ntp:local-mode";
    }
    description
      "Indicates the association's local-mode
       that results in clock synchronization.";
  }
  leaf associations-isconfigured {
    type leafref {
      path "/ntp:ntp/ntp:associations/ntp:association/"
         + "ntp:isconfigured";
    }
    description
      "Indicates if the association (that resulted in the
       clock synchronization) is explicitly configured.";
  }
}

container ntp {
  when 'false() = boolean(/sys:system/sys:ntp)' {
```

```
  description
    "Applicable when the system /sys/ntp/ is not used.";
}
presence "NTP is enabled and system should attempt to
          synchronize the system clock with an NTP server
          from the 'ntp/associations' list.";
description
  "Configuration parameters for NTP";
leaf port {
  if-feature "ntp-port";
  type inet:port-number {
    range "123 | 1024..max";
  }
  default "123";
  description
    "Specify the port used to send and receive NTP packets.";
  reference
    "RFC 5905: Network Time Protocol Version 4: Protocol and
     Algorithms Specification, Section 7.2";
}
container refclock-master {
  presence "NTP master clock is enabled.";
  description
    "Configures the local clock of this 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 {
  if-feature "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 authentication-keys {
    key "keyid";
    uses authentication-key;
    description
      "List of authentication keys";
  }
}
container access-rules {
  if-feature "access-rules";
  description
    "Configuration to control access to NTP service
     by using the NTP access-group feature.
     The access-mode identifies how the ACL is
     applied with NTP.";
  list access-rule {
    key "access-mode";
    description
      "List of access rules";
    leaf access-mode {
      type identityref {
        base access-mode;
      }
      description
```

```
        "The NTP access-mode.  Some of the possible values
         include peer, server, synchronization, query,
         etc.";
    }
    leaf acl {
      type leafref {
        path "/acl:acls/acl:acl/acl:name";
      }
      description
        "Control access configuration to be used.";
    }
    reference
      "RFC 5905: Network Time Protocol Version 4: Protocol and
       Algorithms Specification, Section 9.2";
  }
}
container clock-state {
  config false;
  description
    "Clock operational state of the NTP";
  container system-status {
    description
      "System status of NTP";
    leaf clock-state {
      type identityref {
        base clock-state;
      }
      mandatory true;
      description
        "The state of the system clock.  Some of the possible
         values include synchronized and unsynchronized.";
    }
    leaf clock-stratum {
      type ntp-stratum;
      mandatory true;
      description
        "The NTP entity's own stratum value.  Should be one
         greater than the preceding level.
         16 if unsynchronized.";
      reference
        "RFC 5905: Network Time Protocol Version 4: Protocol and
         Algorithms Specification, Section 3";
    }
    leaf clock-refid {
      type refid;
      mandatory true;
      description
        "A code identifying the particular server or reference
         clock.  The interpretation depends upon stratum.  It
         could be an IPv4 address, the first 32 bits of the MD5
         hash of the IPv6 address, or a string for the Reference
         Identifier and kiss codes.  Some examples:

         -- a refclock ID like '127.127.1.0' for local clock sync

         -- uni/multi/broadcast associations for IPv4 will look
            like '203.0.113.1' and '0x4321FEDC' for IPv6

         -- sync with primary source will look like 'DCN',
            'NIST', 'ATOM'

         -- kiss codes will look like 'AUTH', 'DROP', 'RATE'

         Note that the use of MD5 hash for IPv6 address is not
         for cryptographic purposes.";
      reference
```

```
    "RFC 5905: Network Time Protocol Version 4: Protocol and
     Algorithms Specification, Section 7.3";
}
uses association-ref {
  description
    "Reference to association";
}
leaf nominal-freq {
  type decimal64 {
    fraction-digits 4;
  }
  units "Hz";
  mandatory true;
  description
    "The nominal frequency of the local clock.  An ideal
     frequency with zero uncertainty.";
}
leaf actual-freq {
  type decimal64 {
    fraction-digits 4;
  }
  units "Hz";
  mandatory true;
  description
    "The actual frequency of the local clock";
}
leaf clock-precision {
  type log2seconds;
  mandatory true;
  description
    "Clock precision of this system in signed integer format,
     in log 2 seconds -  (prec=2^(-n)).  A value of 5 would
     mean 2^-5 = 0.03125 seconds = 31.25 ms.";
  reference
    "RFC 5905: Network Time Protocol Version 4: Protocol and
     Algorithms Specification, Section 7.3";
}
leaf clock-offset {
  type decimal64 {
    fraction-digits 3;
  }
  units "milliseconds";
  description
    "The signed time offset to the current selected reference
     time source, e.g., '0.032ms' or '1.232ms'.  The negative
     value indicates that the local clock is behind the
     current selected reference time source.";
  reference
    "RFC 5905: Network Time Protocol Version 4: Protocol and
     Algorithms Specification, Section 9.1";
}
leaf root-delay {
  type decimal64 {
    fraction-digits 3;
  }
  units "milliseconds";
  description
    "Total delay along the path to the root clock";
  reference
    "RFC 5905: Network Time Protocol Version 4: Protocol and
     Algorithms Specification, Sections 4 and 7.3";
}
leaf root-dispersion {
  type decimal64 {
    fraction-digits 3;
  }
```

```
      units "milliseconds";
      description
        "The dispersion to the local clock
         and the root clock, e.g., '6.927ms'.";
      reference
        "RFC 5905: Network Time Protocol Version 4: Protocol and
         Algorithms Specification, Sections 4, 7.3, and 10";
    }
    leaf reference-time {
      type ntp-date-and-time;
      description
        "The reference timestamp.  Time when the system clock was
         last set or corrected.";
      reference
        "RFC 5905: Network Time Protocol Version 4: Protocol and
         Algorithms Specification, Section 7.3";
    }
    leaf sync-state {
      type identityref {
        base ntp-sync-state;
      }
      mandatory true;
      description
        "The synchronization status of the local clock.  Referred
         to as 'Clock state definitions' in RFC 5905.";
      reference
        "RFC 5905: Network Time Protocol Version 4: Protocol and
         Algorithms Specification, Appendix A.1.1";
    }
  }
}
list unicast-configuration {
  if-feature "unicast-configuration";
  key "address type";
  description
    "List of NTP unicast-configurations";
  leaf address {
    type inet:ip-address;
    description
      "Address of this association";
  }
  leaf type {
    type identityref {
      base unicast-configuration-type;
    }
    description
      "The unicast configuration type, for example,
       unicast-server";
  }
  container authentication {
    if-feature "authentication";
    description
      "Authentication used for this association";
    uses authentication;
  }
  leaf prefer {
    type boolean;
    default "false";
    description
      "Whether or not this association is preferred";
  }
  leaf burst {
    type boolean;
    default "false";
    description
      "If set, a series of packets are sent instead of a single
```

```
        packet within each synchronization interval to achieve
        faster synchronization.";
      reference
        "RFC 5905: Network Time Protocol Version 4: Protocol
        and Algorithms Specification, Section 13.1";
    }
    leaf iburst {
      type boolean;
      default "false";
      description
        "If set, a series of packets are sent instead of a single
        packet within the initial synchronization interval to
        achieve faster initial synchronization.";
      reference
        "RFC 5905: Network Time Protocol Version 4: Protocol
        and Algorithms Specification, Section 13.1";
    }
    leaf source {
      type if:interface-ref;
      description
        "The interface whose IP address is used by this association
        as the source address.";
    }
    uses common-attributes {
      description
        "Common attributes like port, version, and min and max
        poll.";
    }
  }
  container associations {
    description
      "Association parameters";
    list association {
      key "address local-mode isconfigured";
      config false;
      description
        "List of NTP associations.  Here address, local-mode,
        and isconfigured are required to uniquely identify
        a particular association.  Let's take the following
        examples:

        1) If RT1 is acting as broadcast server
           and RT2 is acting as broadcast client, then RT2
           will form a dynamic association with the address as
           RT1, local-mode as client, and isconfigured as false.

        2) When RT2 is configured with unicast server RT1,
           then RT2 will form an association with the address as
           RT1, local-mode as client, and isconfigured as true.

        Thus, all three leaves are needed as key to uniquely
        identify the association.";
      leaf address {
        type inet:ip-address;
        description
          "The remote address of this association.  Represents the
          IP address of a unicast/multicast/broadcast address.";
      }
      leaf local-mode {
        type identityref {
          base association-mode;
        }
        description
          "Local-mode of this NTP association";
      }
      leaf isconfigured {
```

```
  type boolean;
  description
    "Indicates if this association is configured (true) or
     dynamically learned (false).";
}
leaf stratum {
  type ntp-stratum;
  description
    "The association stratum value";
  reference
    "RFC 5905: Network Time Protocol Version 4: Protocol and
     Algorithms Specification, Section 3";
}
leaf refid {
  type refid;
  description
    "A code identifying the particular server or reference
     clock.  The interpretation depends upon stratum.  It
     could be an IPv4 address or first 32 bits of the MD5
     hash of the IPv6 address or a string for the Reference
     Identifier and kiss codes.  Some examples:

     -- a refclock ID like '127.127.1.0' for local clock sync

     -- uni/multi/broadcast associations for IPv4 will look
        like '203.0.113.1' and '0x4321FEDC' for IPv6

     -- sync with primary source will look like 'DCN',
        'NIST', or 'ATOM'

     -- kiss codes will look like 'AUTH', 'DROP', or 'RATE'

     Note that the use of an MD5 hash for IPv6 address is
     not for cryptographic purposes.";
  reference
    "RFC 5905: Network Time Protocol Version 4: Protocol and
     Algorithms Specification, Section 7.3";
}
leaf authentication {
  if-feature "authentication";
  type leafref {
    path "/ntp:ntp/ntp:authentication/"
       + "ntp:authentication-keys/ntp:keyid";
  }
  description
    "Authentication key used for this association";
}
leaf prefer {
  type boolean;
  default "false";
  description
    "Indicates if this association is preferred";
}
leaf peer-interface {
  type if:interface-ref;
  description
    "The interface that is used for communication";
}
uses common-attributes {
  description
    "Common attributes like port, version, and min and
     max poll";
}
leaf reach {
  type uint8;
  description
```

```
    "An 8-bit shift register that tracks packet
     generation and receipt.  It is used to determine
     whether the server is reachable and the data are
     fresh.";
  reference
    "RFC 5905: Network Time Protocol Version 4: Protocol and
     Algorithms Specification, Sections 9.2 and 13";
}
leaf unreach {
  type uint8;
  units "seconds";
  description
    "A count of how long in second the server has been
     unreachable, i.e., the reach value has been zero.";
  reference
    "RFC 5905: Network Time Protocol Version 4: Protocol and
     Algorithms Specification, Sections 9.2 and 13";
}
leaf poll {
  type log2seconds;
  description
    "The polling interval for current association in signed
     log2 seconds.";
  reference
    "RFC 5905: Network Time Protocol Version 4: Protocol and
     Algorithms Specification, Section 7.3";
}
leaf now {
  type uint32;
  units "seconds";
  description
    "The time since the last NTP packet was
     received or last synchronized.";
}
leaf offset {
  type decimal64 {
    fraction-digits 3;
  }
  units "milliseconds";
  description
    "The signed offset between the local clock
     and the peer clock, e.g., '0.032ms' or '1.232ms'.  The
     negative value indicates that the local clock is behind
     the peer.";
  reference
    "RFC 5905: Network Time Protocol Version 4: Protocol and
     Algorithms Specification, Section 8";
}
leaf delay {
  type decimal64 {
    fraction-digits 3;
  }
  units "milliseconds";
  description
    "The network delay between the local clock
     and the peer clock";
  reference
    "RFC 5905: Network Time Protocol Version 4: Protocol and
     Algorithms Specification, Section 8";
}
leaf dispersion {
  type decimal64 {
    fraction-digits 3;
  }
  units "milliseconds";
  description
```

```
        "The root dispersion between the local clock
         and the peer clock.";
      reference
        "RFC 5905: Network Time Protocol Version 4: Protocol and
         Algorithms Specification, Section 10";
    }
    leaf originate-time {
      type ntp-date-and-time;
      description
        "This is the local time, in timestamp format,
         when the latest NTP packet was sent to the peer
         (called T1).";
      reference
        "RFC 5905: Network Time Protocol Version 4: Protocol and
         Algorithms Specification, Section 8";
    }
    leaf receive-time {
      type ntp-date-and-time;
      description
        "This is the local time, in timestamp format,
         when the latest NTP packet arrived at the peer
         (called T2).  If the peer becomes unreachable,
         the value is set to zero.";
      reference
        "RFC 5905: Network Time Protocol Version 4: Protocol
         and Algorithms Specification, Section 8";
    }
    leaf transmit-time {
      type ntp-date-and-time;
      description
        "This is the local time, in timestamp format,
         at which the NTP packet departed the peer
         (called T3).  If the peer becomes unreachable,
         the value is set to zero.";
      reference
        "RFC 5905: Network Time Protocol Version 4: Protocol
         and Algorithms Specification, Section 8";
    }
    leaf input-time {
      type ntp-date-and-time;
      description
        "This is the local time, in timestamp format,
         when the latest NTP message from the peer arrived
         (called T4).  If the peer becomes unreachable,
         value is set to zero.";
      reference
        "RFC 5905: Network Time Protocol Version 4: Protocol
         and Algorithms Specification, Section 8";
    }
    container ntp-statistics {
      description
        "Per peer packet send and receive statistics";
      uses statistics {
        description
          "NTP send and receive packet statistics";
      }
    }
  }
}
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 {
  if-feature "broadcast-server";
  presence "NTP broadcast-server is configured on this
            interface.";
  description
    "Configuration of broadcast server";
  leaf ttl {
    type uint8;
    description
      "Specifies the time to live (TTL) for a
       broadcast packet";
    reference
      "RFC 5905: Network Time Protocol Version 4: Protocol
       and Algorithms Specification, Section 3.1";
  }
  container authentication {
    if-feature "authentication";
    description
      "Authentication used on this interface";
    uses authentication;
  }
  uses common-attributes {
    description
      "Common attributes such as port, version, and min and
       max poll";
  }
  reference
    "RFC 5905: Network Time Protocol Version 4: Protocol
     and Algorithms Specification, Section 3.1";
}
container broadcast-client {
  if-feature "broadcast-client";
  presence "NTP broadcast-client is configured on this
            interface.";
  description
    "Configuration of broadcast client";
  reference
    "RFC 5905: Network Time Protocol Version 4: Protocol
     and Algorithms Specification, Section 3.1";
}
list multicast-server {
  if-feature "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 TTL for a  multicast packet";
    reference
      "RFC 5905: Network Time Protocol Version 4: Protocol
       and Algorithms Specification, Section 3.1";
  }
  container authentication {
    if-feature "authentication";
    description
```

```
      "Authentication used on this interface";
    uses authentication;
  }
  uses common-attributes {
    description
      "Common attributes such as port, version, and min and
       max poll";
  }
  reference
    "RFC 5905: Network Time Protocol Version 4: Protocol
     and Algorithms Specification, Section 3.1";
}
list multicast-client {
  if-feature "multicast-client";
  key "address";
  description
    "Configuration of a multicast client";
  leaf address {
    type rt-types:ip-multicast-group-address;
    description
      "The IP address of the multicast group to
       join";
  }
  reference
    "RFC 5905: Network Time Protocol Version 4: Protocol
     and Algorithms Specification, Section 3.1";
}
list manycast-server {
  if-feature "manycast-server";
  key "address";
  description
    "Configuration of a 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: Network Time Protocol Version 4: Protocol
     and Algorithms Specification, Section 3.1";
}
list manycast-client {
  if-feature "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 {
    if-feature "authentication";
    description
      "Authentication used on this interface";
    uses authentication;
  }
  leaf ttl {
    type uint8;
    description
      "Specifies the maximum TTL for the expanding
       ring search";
    reference
      "RFC 5905: Network Time Protocol Version 4: Protocol
```

```
              and Algorithms Specification, Section 3.1";
          }
          leaf minclock {
            type uint8;
            description
              "The minimum manycast survivors in this
               association";
            reference
              "RFC 5905: Network Time Protocol Version 4: Protocol
               and Algorithms Specification, Section 13.2";
          }
          leaf maxclock {
            type uint8;
            description
              "The maximum manycast candidates in this
               association";
            reference
              "RFC 5905: Network Time Protocol Version 4: Protocol
               and Algorithms Specification, Section 13.2";
          }
          leaf beacon {
            type log2seconds;
            description
              "The beacon is the upper limit of the poll interval.
               When the TTL reaches its limit without finding the
               minimum number of manycast servers, the poll interval
               increases until reaching the beacon value, when it
               starts over from the beginning.";
            reference
              "RFC 5905: Network Time Protocol Version 4: Protocol
               and Algorithms Specification, Section 13.2";
          }
          uses common-attributes {
            description
              "Common attributes like port, version, and min and
               max poll";
          }
          reference
            "RFC 5905: Network Time Protocol Version 4: Protocol
             and Algorithms Specification, Section 3.1";
        }
      }
    }
    container ntp-statistics {
      config false;
      description
        "Total NTP packet statistics";
      uses statistics {
        description
          "NTP send and receive packet statistics";
      }
    }
  }

  rpc statistics-reset {
    description
      "Reset statistics collected.";
    input {
      choice association-or-all {
        description
          "Resets statistics for a particular association or
           all.";
        case association {
          uses association-ref;
          description
            "This resets all the statistics collected for
```

```
                the association.";
            }
            case all {
              description
                "This resets all the statistics collected.";
            }
          }
        }
      }
    }
    <CODE ENDS>
```

9. Usage Example

This section include examples for illustration purposes.

Note: '\' indicates line wrapping per [RFC8792].

9.1. Unicast Association

This example describes how to configure a preferred unicast server present at 192.0.2.1 running at port 1025 with authentication-key 10 and version 4 (default).

```
  <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>uc-server</type>
          <prefer>true</prefer>
          <port>1025</port>
          <authentication>
            <symmetric-key>
              <keyid>10</keyid>
            </symmetric-key>
          </authentication>
        </unicast-configuration>
      </ntp>
    </config>
  </edit-config>
```

An example with IPv6 would use an IPv6 address (say 2001:db8::1) in the "address" leaf with no change in any other data tree.

```
  <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>2001:db8::1</address>
          <type>uc-server</type>
          <prefer>true</prefer>
          <port>1025</port>
          <authentication>
            <symmetric-key>
              <keyid>10</keyid>
            </symmetric-key>
          </authentication>
        </unicast-configuration>
      </ntp>
```

```
      </config>
    </edit-config>
```

This example is for retrieving unicast configurations:

```
<get>
  <filter type="subtree">
  <ntp xmlns="urn:ietf:params:xml:ns:yang:ietf-ntp">
      <unicast-configuration>
      </unicast-configuration>
  </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>uc-server</type>
        <authentication>
          <symmetric-key>
            <keyid>10</keyid>
          </symmetric-key>
        </authentication>
      <prefer>true</prefer>
      <burst>false</burst>
      <iburst>true</iburst>
      <source/>
      <minpoll>6</minpoll>
      <maxpoll>10</maxpoll>
      <port>1025</port>
      <stratum>9</stratum>
      <refid>203.0.113.1</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>
```

9.2. Refclock Master

This example describes 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>
```

This example describes how to get reference clock configuration:

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

9.3. Authentication Configuration

This example describes how to enable authentication and configure trusted authentication key 10 with mode as AES-CMAC and a hexadecimal string key:

```
<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>
          <keyid>10</keyid>
          <algorithm>aes-cmac</algorithm>
          <key>
            <hexadecimal-string>
              bb1d6929e95937287fa37d129b756746
            </hexadecimal-string>
          </key>
          <istrusted>true</istrusted>
        </authentication-keys>
      </authentication>
    </ntp>
  </config>
</edit-config>
```

9.4. Access Configuration

This example describes how to configure "peer-access-mode" 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</access-mode>
          <acl>2000</acl>
        </access-rule>
      </access-rules>
    </ntp>
  </config>
</edit-config>
```

This example describes how to get access-related configuration:

```
<get>
  <filter type="subtree">
    <ntp xmlns="urn:ietf:params:xml:ns:yang:ietf-ntp">
      <access-rules>
      </access-rules>
    </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</access-mode>
        <acl>2000</acl>
      </access-rule>
    </access-rules>
  </ntp>
</data>
```

9.5. Multicast Configuration

This example describes how to configure a multicast server with an address of "224.0.1.1", port of 1025, version of 3, and authentication keyid of 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>224.0.1.1</address>
            <authentication>
              <symmetric-key>
                <keyid>10</keyid>
              </symmetric-key>
            </authentication>
            <port>1025</port>
            <version>3</version>
          </multicast-server>
        </interface>
      </interfaces>
    </ntp>
  </config>
</edit-config>
```

This example describes how to get multicast-server-related configuration:

```
<get>
  <filter type="subtree">
    <ntp xmlns="urn:ietf:params:xml:ns:yang:ietf-ntp">
      <interfaces>
        <interface>
          <multicast-server>
          </multicast-server>
        </interface>
      </interfaces>
    </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>224.0.1.1</address>
          <ttl>8</ttl>
          <authentication>
            <symmetric-key>
              <keyid>10</keyid>
            </symmetric-key>
          </authentication>
          <minpoll>6</minpoll>
          <maxpoll>10</maxpoll>
          <port>1025</port>
          <version>3</version>
        </multicast-server>
      </interface>
    </interfaces>
  </ntp>
</data>
```

This example describes how to configure a multicast client with an address of "224.0.1.1":

```
<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>224.0.1.1</address>
          </multicast-client>
        </interface>
      </interfaces>
    </ntp>
  </config>
</edit-config>
```

This example describes how to get multicast-client-related configuration:

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

```
      <interfaces>
        <interface>
          <multicast-client>
          </multicast-client>
        </interface>
      </interfaces>
    </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>224.0.1.1</address>
        </multicast-client>
      </interface>
    </interfaces>
  </ntp>
</data>
```

9.6. Manycast Configuration

This example describes how to configure a manycast-client with an address of "224.0.1.1", port of 1025, and authentication keyid of 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>224.0.1.1</address>
            <authentication>
              <symmetric-key>
                <keyid>10</keyid>
              </symmetric-key>
            </authentication>
            <port>1025</port>
          </manycast-client>
        </interface>
      </interfaces>
    </ntp>
  </config>
</edit-config>
```

This example describes how to get manycast-client-related configuration:

```
<get>
  <filter type="subtree">
    <ntp xmlns="urn:ietf:params:xml:ns:yang:ietf-ntp">
      <interfaces>
        <interface>
          <manycast-client>
          </manycast-client>
        </interface>
      </interfaces>
    </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>224.0.1.1</address>
          <authentication>
            <symmetric-key>
              <keyid>10</keyid>
            </symmetric-key>
          </authentication>
          <ttl>8</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>

This example describes how to configure a manycast-server with an
address of "224.0.1.1":

<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>224.0.1.1</address>
          </manycast-server>
        </interface>
      </interfaces>
    </ntp>
  </config>
</edit-config>

This example describes how to get manycast-server-related
configuration:

<get>
  <filter type="subtree">
    <ntp xmlns="urn:ietf:params:xml:ns:yang:ietf-ntp">
      <interfaces>
        <interface>
          <manycast-server>
          </manycast-server>
        </interface>
      </interfaces>
    </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>224.0.1.1</address>
          </manycast-server>
        </interface>
      </interfaces>
    </ntp>
  </data>
```

9.7. Clock State

This example describes how to get current clock state:

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

9.8. Get All Association

This example describes how to get all associations present in the system:

```
  <get>
    <filter type="subtree">
      <ntp xmlns="urn:ietf:params:xml:ns:yang:ietf-ntp">
        <associations>
        </associations>
      </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>
          <association>
            <address>192.0.2.1</address>
            <stratum>9</stratum>
            <refid>203.0.113.1</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>
          </association>
        </associations>
      </ntp>
    </data>
```

9.9. Global Statistic

This example describes how to get global statistics:

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

10. IANA Considerations

10.1. IETF XML Registry

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 IESG.

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

10.2. YANG Module Names

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 9249

11. 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 [RFC8446].

The Network Configuration Access Control Model (NACM) [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. The 'nacm:default-deny-all' is used to prevent retrieval of the key information.

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 specifies 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. Care 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 modifies the NTP associations. Unexpected changes could lead to disruption and/or network misbehavior.

/ntp/interfaces/interface: The entries in the list include all per-interface configurations related to broadcast, multicast, and manycast mode, and indirectly creates or modifies the NTP

associations. Unexpected changes could lead to disruption and/or network misbehavior. It could also lead to synchronization over an untrusted source over trusted ones.

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/authentication/authentication-keys: The entries in the list include all the NTP authentication keys. Unauthorized access to the keys can be easily exploited to permit unauthorized access to the NTP service. This information is sensitive; thus, unauthorized access to this needs to be curtailed.

/ntp/associations/association/: The entries in the list include all active NTP associations of all modes. Exposure of these nodes could reveal network topology or trust relationships. Unauthorized access to this also needs to be curtailed.

/ntp/authentication and /ntp/access-rules: The entries in the list include the authentication and access control configurations. Exposure of these nodes could reveal network topology or trust relationships.

Some of the RPC operations in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control access to these operations. These are the operations and their sensitivity/vulnerability:

statistics-reset: The RPC is used to reset statistics. Unauthorized reset could impact monitoring.

The leaf /ntp/authentication/authentication-keys/algorithm can be set to cryptographic algorithms that are no longer considered to be secure. As per [RFC8573], AES-CMAC is the recommended algorithm.

12. References

12.1. 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, <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>.

[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>.

[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>.

[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>.

[RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <https://www.rfc-editor.org/info/rfc7950>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/info/rfc8040>.

[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>.

[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>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[RFC8519] Jethanandani, M., Agarwal, S., Huang, L., and D. Blair, "YANG Data Model for Network Access Control Lists (ACLs)", RFC 8519, DOI 10.17487/RFC8519, March 2019, <https://www.rfc-editor.org/info/rfc8519>.

[RFC8573] Malhotra, A. and S. Goldberg, "Message Authentication Code for the Network Time Protocol", RFC 8573, DOI 10.17487/RFC8573, June 2019, <https://www.rfc-editor.org/info/rfc8573>.

12.2. Informative References

[RFC1305] Mills, D., "Network Time Protocol (Version 3) Specification, Implementation and Analysis", RFC 1305, DOI 10.17487/RFC1305, March 1992, <https://www.rfc-editor.org/info/rfc1305>.

[RFC1321] Rivest, R., "The MD5 Message-Digest Algorithm", RFC 1321, DOI 10.17487/RFC1321, April 1992, <https://www.rfc-editor.org/info/rfc1321>.

[RFC3174] Eastlake 3rd, D. and P. Jones, "US Secure Hash Algorithm 1 (SHA1)", RFC 3174, DOI 10.17487/RFC3174, September 2001, <https://www.rfc-editor.org/info/rfc3174>.

[RFC4493] Song, JH., Poovendran, R., Lee, J., and T. Iwata, "The AES-CMAC Algorithm", RFC 4493, DOI 10.17487/RFC4493, June 2006, <https://www.rfc-editor.org/info/rfc4493>.

[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>.

[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>.

[RFC8792] Watsen, K., Auerswald, E., Farrel, A., and Q. Wu, "Handling Long Lines in Content of Internet-Drafts and RFCs", RFC 8792, DOI 10.17487/RFC8792, June 2020, <https://www.rfc-editor.org/info/rfc8792>.

[SHS] National Institute of Standards and Technology (NIST), "Secure Hash Standard (SHS)", DOI 10.6028/NIST.FIPS.180-4, FIPS PUB 180-4, August 2015, <https://doi.org/10.6028/NIST.FIPS.180-4>.

Appendix A. Full YANG Tree

The full tree for the ietf-ntp YANG data model is as follows.

```
module: ietf-ntp
  +--rw ntp!
     +--rw port?                    inet:port-number {ntp-port}?
     +--rw refclock-master!
     |  +--rw master-stratum?   ntp-stratum
     +--rw authentication {authentication}?
     |  +--rw auth-enabled?          boolean
     |  +--rw authentication-keys* [keyid]
     |     +--rw keyid       uint32
     |     +--rw algorithm?   identityref
     |     +--rw key
     |     |  +--rw (key-string-style)?
     |     |     +--:(keystring)
     |     |     |  +--rw keystring?            string {deprecated}?
     |     |     +--:(hexadecimal) {hex-key-string}?
     |     |        +--rw hexadecimal-string?   yang:hex-string
     |     +--rw istrusted?   boolean
     +--rw access-rules {access-rules}?
     |  +--rw access-rule* [access-mode]
     |     +--rw access-mode    identityref
     |     +--rw acl?           -> /acl:acls/acl/name
     +--ro clock-state
     |  +--ro system-status
     |     +--ro clock-state                 identityref
     |     +--ro clock-stratum               ntp-stratum
     |     +--ro clock-refid                 refid
     |     +--ro associations-address?
     |     |       -> /ntp/associations/association/address
     |     +--ro associations-local-mode?
     |     |       -> /ntp/associations/association/local-mode
     |     +--ro associations-isconfigured?
     |     |       -> /ntp/associations/association/isconfigured
     |     +--ro nominal-freq                decimal64
```

```
|     +--ro actual-freq                  decimal64
|     +--ro clock-precision              log2seconds
|     +--ro clock-offset?                decimal64
|     +--ro root-delay?                  decimal64
|     +--ro root-dispersion?             decimal64
|     +--ro reference-time?              ntp-date-and-time
|     +--ro sync-state                   identityref
+--rw unicast-configuration* [address type]
|       {unicast-configuration}?
|  +--rw address           inet:ip-address
|  +--rw type              identityref
|  +--rw authentication {authentication}?
|  |  +--rw (authentication-type)?
|  |     +--:(symmetric-key)
|  |        +--rw keyid?   leafref
|  +--rw prefer?           boolean
|  +--rw burst?            boolean
|  +--rw iburst?           boolean
|  +--rw source?           if:interface-ref
|  +--rw minpoll?          log2seconds
|  +--rw maxpoll?          log2seconds
|  +--rw port?             inet:port-number {ntp-port}?
|  +--rw version?          ntp-version
+--rw associations
|  +--ro association* [address local-mode isconfigured]
|     +--ro address           inet:ip-address
|     +--ro local-mode        identityref
|     +--ro isconfigured      boolean
|     +--ro stratum?          ntp-stratum
|     +--ro refid?            refid
|     +--ro authentication?
|     |       -> /ntp/authentication/authentication-keys/keyid
|     |       {authentication}?
|     +--ro prefer?           boolean
|     +--ro peer-interface?   if:interface-ref
|     +--ro minpoll?          log2seconds
|     +--ro maxpoll?          log2seconds
|     +--ro port?             inet:port-number {ntp-port}?
|     +--ro version?          ntp-version
|     +--ro reach?            uint8
|     +--ro unreach?          uint8
|     +--ro poll?             log2seconds
|     +--ro now?              uint32
|     +--ro offset?           decimal64
|     +--ro delay?            decimal64
|     +--ro dispersion?       decimal64
|     +--ro originate-time?   ntp-date-and-time
|     +--ro receive-time?     ntp-date-and-time
|     +--ro transmit-time?    ntp-date-and-time
|     +--ro input-time?       ntp-date-and-time
|     +--ro ntp-statistics
|        +--ro discontinuity-time?   ntp-date-and-time
|        +--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! {broadcast-server}?
|     |  +--rw ttl?                uint8
|     |  +--rw authentication {authentication}?
|     |  |  +--rw (authentication-type)?
|     |  |     +--:(symmetric-key)
|     |  |        +--rw keyid?   leafref
|     |  +--rw minpoll?           log2seconds
```

```
      |     |  +--rw maxpoll?          log2seconds
      |     |  +--rw port?             inet:port-number {ntp-port}?
      |     |  +--rw version?          ntp-version
      |     +--rw broadcast-client! {broadcast-client}?
      |     +--rw multicast-server* [address] {multicast-server}?
      |     |  +--rw address
      |     |  |       rt-types:ip-multicast-group-address
      |     |  +--rw ttl?              uint8
      |     |  +--rw authentication {authentication}?
      |     |  |  +--rw (authentication-type)?
      |     |  |     +--:(symmetric-key)
      |     |  |        +--rw keyid?   leafref
      |     |  +--rw minpoll?          log2seconds
      |     |  +--rw maxpoll?          log2seconds
      |     |  +--rw port?             inet:port-number {ntp-port}?
      |     |  +--rw version?          ntp-version
      |     +--rw multicast-client* [address] {multicast-client}?
      |     |  +--rw address    rt-types:ip-multicast-group-address
      |     +--rw manycast-server* [address] {manycast-server}?
      |     |  +--rw address    rt-types:ip-multicast-group-address
      |     +--rw manycast-client* [address] {manycast-client}?
      |        +--rw address
      |        |       rt-types:ip-multicast-group-address
      |        +--rw authentication {authentication}?
      |        |  +--rw (authentication-type)?
      |        |     +--:(symmetric-key)
      |        |        +--rw keyid?   leafref
      |        +--rw ttl?              uint8
      |        +--rw minclock?         uint8
      |        +--rw maxclock?         uint8
      |        +--rw beacon?           log2seconds
      |        +--rw minpoll?          log2seconds
      |        +--rw maxpoll?          log2seconds
      |        +--rw port?             inet:port-number {ntp-port}?
      |        +--rw version?          ntp-version
      +--ro ntp-statistics
         +--ro discontinuity-time?   ntp-date-and-time
         +--ro packet-sent?          yang:counter32
         +--ro packet-sent-fail?     yang:counter32
         +--ro packet-received?      yang:counter32
         +--ro packet-dropped?       yang:counter32

  rpcs:
    +---x statistics-reset
       +---w input
          +---w (association-or-all)?
             +--:(association)
             |  +---w associations-address?
             |  |       -> /ntp/associations/association/address
             |  +---w associations-local-mode?
             |  |       -> /ntp/associations/association/local-mode
             |  +---w associations-isconfigured?
             |          -> /ntp/associations/association/isconfigured
             +--:(all)
```


Acknowledgments

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

Thanks to Andy Bierman for the YANG doctor review.

Thanks to Dieter Sibold for being the Document Shepherd and Erik Kline for being the Responsible AD.

Thanks to Takeshi Takahashi for SECDIR review. Thanks to Tim Evens for GENART review.

A special thanks to Tom Petch for a very detailed YANG review and providing great suggestions for improvements.

Thanks for the IESG review from Benjamin Kaduk, Francesca Palombini, Eric Vyncke, Murray Kucherawy, Robert Wilton, Roman Danyliw, and Zaheduzzaman Sarker.

Authors' Addresses

Nan Wu
Huawei
Huawei Bld., No.156 Beiqing Rd.
Beijing
100095
China
Email: eric.wu@huawei.com

Dhruv Dhody (editor)
Huawei
Divyashree Techno Park, Whitefield
Bangalore 560066
Kanataka
India
Email: dhruv.ietf@gmail.com

Ankit Kumar Sinha (editor)
RtBrick Inc.
Bangalore
Kanataka
India
Email: ankit.ietf@gmail.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