NTP Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 29, 2019

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

June 27, 2019

A YANG Data Model for NTP
draft-ietf-ntp-yang-data-model-07

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 29, 2019.

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
1.5. Refrences in the Model 4
2. NTP data model 4
3. Relationship with NTPv4-MIB 9
4. Relationship with RFC 7317 9
5. Access Rules 10
6. Key Management 10
7. NTP YANG Module 10
8. Usage Example 36
8.1. Unicast association 36
8.2. Refclock master 37
8.3. Authentication configuration 38
8.4. Access configuration 40
8.5. Multicast configuration 40
8.6. Manycast configuration 44
8.7. Clock state 47
8.8. Get all association 47
8.9. Global statistic 49
9. IANA Considerations 49
10. Security Considerations 50
11. Acknowledgments 51
12. References 51
12.1. Normative References 51
12.2. Informative References 53
Authors' Addresses 53

1. Introduction

This document defines a YANG [RFC7950] 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	[RFC8519]
rt-types	ietf-routing-types	[RFC8294]
nacm	ietf-netconf-acm	[RFC8341]

Table 1: Prefixes and corresponding YANG modules

1.5. Refrences in the Model

Following documents are refrenced 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]
YANG Data Model for Key Chains	[RFC8177]
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]

Table 2: Refrences in the YANG modules

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
        |  +--rw auth-enabled?          boolean
        |  +--rw authentication-keys* [key-id]
        |     +--rw key-id       uint32
        |     +--...
        +--rw access-rules
        |  +--rw access-rule* [access-mode]
        |     +--rw access-mode    access-mode
        |     +--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
        |     +--...
        +--rw unicast-configuration* [address type]
        |  +--rw address           inet:host
        |  +--rw type              unicast-configuration-type
        |  +--...
        +--ro associations* [address local-mode isconfigured]
        |  +--...
        |  +--ro ntp-statistics
        |     +--...
        +--rw interfaces
        |  +--rw interface* [name]
        |     +--rw name                if:interface-ref
        |     +--rw broadcast-server!
        |     |  +--...
        |     +--rw broadcast-client!
        |     +--rw multicast-server* [address]
        |     |  +--rw address
        |     |  |       rt-types:ip-multicast-group-address
        |     |  +--...
        |     +--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
        |        +--...
        +--ro ntp-statistics
           +--...
```

The full data model tree for the YANG module "ietf-ntp" is represented as -

```
module: ietf-ntp
  +--rw ntp!
     +--rw port?                    inet:port-number {ntp-port}?
     +--rw refclock-master!
     |  +--rw master-stratum?   ntp-stratum
     +--rw authentication
     |  +--rw auth-enabled?          boolean
     |  +--rw authentication-keys* [key-id]
     |     +--rw key-id       uint32
     |     +--rw algorithm?   identityref
     |     +--rw key?         ianach:crypt-hash
     |     +--rw istrusted?   boolean
     +--rw access-rules
     |  +--rw access-rule* [access-mode]
     |     +--rw access-mode    access-mode
     |     +--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?             inet:port-number {ntp-port}?
|  +--rw version?          ntp-version
+--ro associations* [address local-mode isconfigured]
|  +--ro address           inet:host
|  +--ro local-mode        association-mode
|  +--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?             inet:port-number {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?             inet:port-number {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?            inet:port-number {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?            inet:port-number {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
```

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 NTP Configuration Data Nodes and Related NTPv4-MIB Objects

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

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. If the yang 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

YANG NTP Configuration Data Nodes and counterparts in RFC 7317 Objects

5. Access Rules

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

6. Key Management

As per [RFC1305] and [RFC5905], 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 key-id is 32-bits unsigned integer that MUST be configured on the NTP peers before the authentication could be used. For this reason, this YANG modules allow such configuration via /ntp/authentication/authentication-keys/. Further at the time of configuration of NTP association (for example unicast-server), the key-id is specefied.

7. NTP YANG Module

```
<CODE BEGINS> file "ietf-ntp@2019-06-28.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 iana-crypt-hash {
      prefix "ianach";
      reference "RFC 7317: A YANG Data Model for System Management";
    }

    import ietf-key-chain {
      prefix "key-chain";
      reference "RFC 8177: YANG Data Model for Key Chains";
    }

    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 Protocol (NETCONF) Access
         Control Model";
    }

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

      Copyright (c) 2019 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
      (https://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 2019-06-28 {
      description
        "Initial revision.";
      reference
        "RFC XXXX: A YANG Data Model for NTP.";
    }

    /* Note: The RFC Editor will replace XXXX with the number assigned
    to this document once it becomes an RFC.*/

    /* 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";
    }

    typedef ntp-version {
      type uint8;
      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.";
      reference
        "RFC 5905: Network Time Protocol Version 4: Protocol and
        Algorithms Specification";
    }

    typedef ntp-maxpoll {
      type uint8 {
        range "4..17";
      }
      default "10";
      description
        "The maximum poll exponent for this NTP association.";
      reference
        "RFC 5905: Network Time Protocol Version 4: Protocol and
        Algorithms Specification";
    }

    typedef access-mode {
      type enumeration {
        enum peer {
          value "0";
          description
            "Enables the full access authority. Both time
            request and control query can be performed
```

```
            on the local NTP service, and the local clock
            can be synchronized with 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 with 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
            "Enables the maximum access limitation.
            Control query can be performed only on the
            local NTP service.";
        }
      }
      description
        "This defines NTP access 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-mode {
      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 learned this association dynamically.
            This device may synchronize with its NTP peer.";
        }
        enum broadcast {
          value "3";
          description
            "Use broadcast mode(mode 5).
            This mode defines that its either working
            as broadcast-server or multicast-server.";
        }
        enum broadcast-client {
          value "4";
          description
            "This mode defines that its either working
            as broadcast-client or multicast-client.";
        }
      }
      description
        "The 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.";
    }

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

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

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

    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";
    }

    feature broadcast-server {
      description
        "Support for broadcast server";
      reference
```

```
        "RFC 5905: Network Time Protocol Version 4: Protocol and
        Algorithms Specification";
    }

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

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

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

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

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

    }

    /* 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 key {
        nacm:default-deny-all;
        type ianach:crypt-hash;
        description
          "The key";
      }
      leaf istrusted {
        type boolean;
        description
          "Key-id is trusted or not";
      }
      reference
        "RFC 5905: Network Time Protocol Version 4: Protocol and
        Algorithms Specification";
    }

    grouping authentication {
      description
        "Authentication.";
      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
          "The total number of NTP packets delivered to the
          transport service by this NTP entity for this
          association.
          Discountinuities in the value of this counter can occur
          upon cold start or reinitialization of the NTP entity, the
          management system and at other times as indicated by
          discontinuities in the value of sysUpTime.";
      }
      leaf packet-sent-fail {
        type yang:counter32;
        description
          "The number of times NTP packets sending failed.";
      }
      leaf packet-received {
        type yang:counter32;
        description
          "The total number of NTP packets delivered to the
          NTP entity from this association.
          Discountinuities in the value of this counter can occur
          upon cold start or reinitialization of the NTP entity, the
          management system and at other times as indicated by
          discontinuities in the value of sysUpTime.";
      }
      leaf packet-dropped {
        type yang:counter32;
        description
          "The total number of NTP packets that were delivered
          to this NTP entity from this association and this entity
          was not able to process due to an NTP protocol error.
          Discountinuities in the value of this counter can occur
          upon cold start or reinitialization of the NTP entity, the
          management system and at other times as indicated by
          discontinuities in the value of sysUpTime.";
      }
    }

    grouping common-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 maximum poll interval used in this association.";
      }
      leaf port {
        if-feature ntp-port;
        type inet:port-number {
          range "123 | 1025..max";
        }
        default "123";
        description
          "Specify the port used to send NTP packets.";
      }
      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:address";
        }
        description
          "Indicates the association's address
          which result in clock synchronization.";
      }
      leaf associations-local-mode {
        type leafref {
          path "/ntp:ntp/ntp:associations/ntp:local-mode";
        }
        description
          "Indicates the association's local-mode
```

```
          which result in clock synchronization.";
      }
      leaf associations-isconfigured {
        type leafref {
          path "/ntp:ntp/ntp:associations/"
          + "ntp:isconfigured";
        }
        description
          "The association was configured or dynamic
          which result in clock synchronization.";
      }
    }

    /* Configuration data nodes */
    container ntp {
      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 | 1025..max";
        }
        default "123";
        description
          "Specify the port used to send and receive NTP packets.";
      }
      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 {
        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 "key-id";
          uses authentication-key;
          description
            "List of authentication keys.";
        }
      }

      container access-rules {
        description
          "Configuration to control access to NTP service
          by using 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 access-mode;
            description
              "NTP access mode. The defination of each possible values:
              peer(0): Both time request and control query can be
              performed.
              server(1): Enables the server access and query.
              synchronization(2): Enables the server access only.
              query(3): Enables control query only.";
          }
          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";
        }
      }

      container clock-state {
        config "false";
```

```
        description
          "Clock operational state of the NTP.";

        container system-status {
          description
            "System status of NTP.";
          leaf clock-state {
            type ntp-clock-status;
            mandatory true;
            description
              "The state of system clock. The definition of each
              possible value is:
              synchronized(0): Indicates local clock is synchronized.
              unsynchronized(1): Indicates local clock is not
              synchronized.";
          }
          leaf clock-stratum {
            type ntp-stratum;
            mandatory true;
            description
              "The NTP entity's own stratum value. Should be a stratum
              of syspeer + 1 (or 16 if no syspeer).";
            reference
              "RFC 5905: Network Time Protocol Version 4: Protocol and
              Algorithms Specification";
          }
          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.";
            reference
              "RFC 5905: Network Time Protocol Version 4: Protocol and
              Algorithms Specification";
          }

          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.";
            reference
              "RFC 5905: Network Time Protocol Version 4: Protocol and
              Algorithms Specification";
          }
          leaf actual-freq {
            type decimal64 {
              fraction-digits 4;
            }
            units Hz;
            mandatory true;
            description
              "The actual frequency of the
              local clock.";
            reference
              "RFC 5905: Network Time Protocol Version 4: Protocol and
              Algorithms Specification";
          }
          leaf clock-precision {
            type uint8;
            units Hz;
            mandatory true;
            description
              "Clock precision of this system in integer format
              (prec=2^(-n)). A value of 5 would mean 2^-5 = 31.25 ms.";
            reference
              "RFC 5905: Network Time Protocol Version 4: Protocol and
              Algorithms Specification";
          }
          leaf clock-offset {
            type decimal64 {
              fraction-digits 3;
            }
            units milliseconds;
            description
              "The time offset to the current selected reference time
              source e.g., '0.032' or '1.232'.";
```

```
            reference
              "RFC 5905: Network Time Protocol Version 4: Protocol and
              Algorithms Specification";
          }
          leaf root-delay {
            type decimal64 {
              fraction-digits 3;
            }
            units milliseconds;
            description
              "Total delay along the path to root clock.";
            reference
              "RFC 5905: Network Time Protocol Version 4: Protocol and
              Algorithms Specification";
          }
          leaf root-dispersion {
            type decimal64 {
              fraction-digits 3;
            }
            units milliseconds;
            description
              "The dispersion between the local clock
              and the root clock, e.g., '6.927'.";
            reference
              "RFC 5905: Network Time Protocol Version 4: Protocol and
              Algorithms Specification";
          }
          leaf reference-time {
            type yang:date-and-time;
            description
              "The reference timestamp.";
          }
          leaf sync-state {
            type ntp-sync-state;
            mandatory true;
            description
              "The synchronization status of
              the local clock.";
          }
        }
      }
      list unicast-configuration {
        key "address type";
        description
          "List of NTP unicast-configurations.";
        leaf address {
          type inet:host;
          description
```

```
            "Address of this association.";
        }
        leaf type {
          type unicast-configuration-type;
          description
            "Use client association mode.  This device
            will not provide synchronization to the
            configured NTP server.";
        }
        container authentication{
          description
            "Authentication used for this association.";
          uses authentication;
        }
        leaf prefer {
          type boolean;
          default "false";
          description
            "Whether this association is preferred or not.";
        }
        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";
        }
        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";
        }
        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, min and max
            poll.";
        }
      }
      list associations {
        key "address local-mode isconfigured";
        config "false";
        description
          "List of NTP associations. Here address, local-mode
          and isconfigured is required to uniquely identify
          a particular association. Lets take following examples -

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

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

          Thus all 3 leaves are needed as key to unique identify
          the association.";
        leaf address {
          type inet:host;
          description
            "The address of this association. Represents the IP
            address of a unicast/multicast/broadcast address.";
        }
        leaf local-mode {
          type association-mode;
          description
            "Local mode of this NTP association.";
        }
        leaf isconfigured {
          type boolean;
          description
            "Indicates if this association is configured or
            dynamically learned.";
        }
        leaf stratum {
          type ntp-stratum;
          description
            "The association stratum value.";
          reference
```

```
          "RFC 5905: Network Time Protocol Version 4: Protocol and
          Algorithms Specification";
        }
        leaf refid {
          type union {
            type inet:ipv4-address;
            type binary {
              length "4";
            }
            type string {
              length "4";
            }
          }
          description
            "The refclock driver ID, if available.
            -- a refclock driver ID like '127.127.1.0' for local clock
               sync
            -- uni/multi/broadcast associations will look like
               '20.1.1.1'
            -- sync with primary source will look like 'DCN', 'NIST',
               'ATOM'";
          reference
            "RFC 5905: Network Time Protocol Version 4: Protocol and
            Algorithms Specification";
        }
        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
            "Indicates if this association is preferred.";
        }
        leaf peer-interface {
          type if:interface-ref;
          description
            "The interface which is used for communication.";
        }
        uses common-attributes {
          description
            "Common attributes like port, version, min and
            max poll.";
```

```
        }
        leaf reach {
          type uint8;
          description
            "The reachability of the configured
            server or peer.";
          reference
            "RFC 5905: Network Time Protocol Version 4: Protocol and
            Algorithms Specification";
        }
        leaf unreach {
          type uint8;
          description
            "The unreachability of the configured
            server or peer.";
          reference
            "RFC 5905: Network Time Protocol Version 4: Protocol and
            Algorithms Specification";
        }
        leaf poll {
          type uint8;
          units seconds;
          description
            "The polling interval for current association";
          reference
            "RFC 5905: Network Time Protocol Version 4: Protocol and
            Algorithms Specification";
        }
        leaf now {
          type uint32;
          units seconds;
          description
            "The time since the NTP packet was
            not received or last synchronized.";
          reference
            "RFC 5905: Network Time Protocol Version 4: Protocol and
            Algorithms Specification";
        }
        leaf offset {
          type decimal64 {
            fraction-digits 3;
          }
          units milliseconds;
          description
            "The offset between the local clock
            and the peer clock, e.g., '0.032' or '1.232'";
          reference
            "RFC 5905: Network Time Protocol Version 4: Protocol and
```

```
          Algorithms Specification";
        }
        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";
        }
        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";
        }
        leaf originate-time {
          type yang:date-and-time;
          description
            "This is the local time, in timestamp format,
            when latest NTP packet was sent to peer(T1).";
          reference
            "RFC 5905: Network Time Protocol Version 4: Protocol and
            Algorithms Specification";
        }
        leaf receive-time {
          type yang:date-and-time;
          description
            "This is the local time, in timestamp format,
            when latest NTP packet arrived at peer(T2).
            If the peer becomes unreachable the value is set to zero.";
          reference
            "RFC 5905: Network Time Protocol Version 4: Protocol and
            Algorithms Specification";
        }
        leaf transmit-time {
          type yang:date-and-time;
          description
```

```
            "This is the local time, in timestamp format,
             at which the NTP packet departed the peer(T3).
             If the peer becomes unreachable the value is set to zero.";
           reference
             "RFC 5905: Network Time Protocol Version 4: Protocol and
             Algorithms Specification";
         }
         leaf input-time {
           type yang:date-and-time;
           description
             "This is the local time, in timestamp format,
             when the latest NTP message from the peer arrived(T4).
             If the peer becomes unreachable the value is set to zero.";
           reference
             "RFC 5905: Network Time Protocol Version 4: Protocol and
             Algorithms Specification";
         }
         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 {
             presence
               "NTP broadcast-server is configured";
             description
               "Configuration of broadcast server.";
             leaf ttl {
               type uint8;
               description
```

```
            "Specifies the time to live (TTL) for a
            broadcast packet.";
          }
          container authentication{
            description
              "Authentication used for this association.";
              uses authentication;
          }
          uses common-attributes {
            description
              "Common attribute like port, version, min and
              max poll.";
          }
          reference
            "RFC 5905: Network Time Protocol Version 4: Protocol and
            Algorithms Specification";
        }

        container broadcast-client {
          presence
            "NTP broadcast-client is configured.";
          description
            "Configuration of broadcast-client.";
          reference
            "RFC 5905: Network Time Protocol Version 4: Protocol and
            Algorithms Specification";
        }

        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) for a
              multicast packet.";
          }
          container authentication{
            description
              "Authentication used for this association.";
              uses authentication;
          }
```

```
            uses common-attributes {
              description
                "Common attributes like port, version, min and
                max poll.";
              }
              reference
                "RFC 5905: Network Time Protocol Version 4: Protocol and
                Algorithms Specification";
            }
            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: Network Time Protocol Version 4: Protocol and
                Algorithms Specification";
            }
            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 used for this association.";
                uses authentication;
```

```
            }
            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 common-attributes {
              description
                "Common attributes like port, version, min and
                max poll.";
            }
            reference
              "RFC 5905: Network Time Protocol Version 4: Protocol and
              Algorithms Specification";
          }
        }
      }
      container ntp-statistics {
        config "false";
        description
          "Total NTP packet statistics.";
        uses statistics {
          description
            "NTP send and receive packet statistics.";
        }
      }
    }
  }
  <CODE ENDS>
```

8. Usage Example

This section include examples for illustration purporses.

8.1. 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

```
  <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.

This example is for retriving unicast configurations -

```
<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>
```

8.2. 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">
    <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>
```

8.3. Authentication configuration

This example describes how to enable authentication and configure trusted authentication key 10 with mode as md5 and key 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>
          <key>abcd</key>
          <istrusted>true</istrusted>
        </authentication-keys>
      </authentication>
    </ntp>
  </config>
</edit-config>
```

This example describes 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>
        <key>abcd</key>
        <istrusted>true</istrusted>
      </authentication-keys>
    </authentication>
  </ntp>
</data>
```

8.4. Access configuration

This example describes how to configure access mode "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>
```

This example describes 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>
```

8.5. Multicast configuration

This example describes 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>
```

This example describes 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>
```

This example describes 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>
```

This example describes 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>
```

8.6. Manycast configuration

This example describes 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>
```

This example describes 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>

   This example describes 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>
```

This example describes 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>
```

8.7. Clock state

This example describes 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>
```

8.8. Get all association

This example describes how to get all association present in the system -

```
   <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>
```

8.9. Global statistic

This example describes 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>
```

9. 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 IESG.

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

Note: The RFC Editor will replace XXXX with the number assigned to this document once it becomes an RFC.

10. 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 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. 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 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/authentication/authentication-keys - The entries in the list includes all the NTP authentication keys. This information is sensitive and can be exploited and thus unauthorized access to this needs to be curtailed.

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

11. Acknowledgments

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

12. References

12.1. Normative 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>.

[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>.

[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>.

[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>.

[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>.

[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>.

12.2. 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

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

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