NETCONF Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 20, 2020

K. Watsen
Watsen Networks

October 18, 2019

NETCONF Client and Server Models
draft-ietf-netconf-netconf-client-server-15

Abstract

This document defines two YANG modules, one module to configure a NETCONF client and the other module to configure a NETCONF server. Both modules support both the SSH and TLS transport protocols, and support both standard NETCONF and NETCONF Call Home connections.

Editorial Note (To be removed by RFC Editor)

This draft contains many placeholder values that need to be replaced with finalized values at the time of publication. This note summarizes all of the substitutions that are needed. No other RFC Editor instructions are specified elsewhere in this document.

This document contains references to other drafts in progress, both in the Normative References section, as well as in body text throughout. Please update the following references to reflect their final RFC assignments:

- I-D.ietf-netconf-keystore
- I-D.ietf-netconf-tcp-client-server
- I-D.ietf-netconf-ssh-client-server
- I-D.ietf-netconf-tls-client-server

Artwork in this document contains shorthand references to drafts in progress. Please apply the following replacements:

- "XXXX" --> the assigned RFC value for this draft
- "AAAA" --> the assigned RFC value for I-D.ietf-netconf-tcp-client-server
- "YYYY" --> the assigned RFC value for I-D.ietf-netconf-ssh-client-server

- "ZZZZ" --> the assigned RFC value for I-D.ietf-netconf-tls-client-server

Artwork in this document contains placeholder values for the date of publication of this draft. Please apply the following replacement:

- "2019-10-18" --> the publication date of this draft

The following Appendix section is to be removed prior to publication:

- Appendix B. Change Log

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 April 20, 2020.

Copyright Notice

Table of Contents

1. Introduction

This document defines two YANG [RFC7950] modules, one module to configure a NETCONF [RFC6241] client and the other module to configure a NETCONF server. Both modules support both NETCONF over SSH [RFC6242] and NETCONF over TLS [RFC7589] and NETCONF Call Home connections [RFC8071].

2. Terminology

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.

3. The NETCONF Client Model

The NETCONF client model presented in this section supports both clients initiating connections to servers, as well as clients listening for connections from servers calling home, using either the SSH and TLS transport protocols.

YANG feature statements are used to enable implementations to advertise which potentially uncommon parts of the model the NETCONF client supports.

3.1. Tree Diagram

The following tree diagram [RFC8340] provides an overview of the data model for the "ietf-netconf-client" module.

This tree diagram only shows the nodes defined in this module; it does show the nodes defined by "grouping" statements used by this module.

Please see Appendix A.1 for a tree diagram that illustrates what the module looks like with all the "grouping" statements expanded.

```
module: ietf-netconf-client
  +--rw netconf-client
     +---u netconf-client-app-grouping

  grouping netconf-client-grouping
  grouping netconf-client-initiate-stack-grouping
    +-- (transport)
       +--:(ssh) {ssh-initiate}?
       |  +-- ssh
       |     +-- tcp-client-parameters
       |     |  +---u tcpc:tcp-client-grouping
       |     +-- ssh-client-parameters
       |     |  +---u sshc:ssh-client-grouping
       |     +-- netconf-client-parameters
       +--:(tls) {tls-initiate}?
          +-- tls
             +-- tcp-client-parameters
```

```
               |  +---u tcpc:tcp-client-grouping
               +-- tls-client-parameters
               |  +---u tlsc:tls-client-grouping
               +-- netconf-client-parameters
     grouping netconf-client-listen-stack-grouping
       +-- (transport)
          +--:(ssh) {ssh-listen}?
          |  +-- ssh
          |     +-- tcp-server-parameters
          |     |  +---u tcps:tcp-server-grouping
          |     +-- ssh-client-parameters
          |     |  +---u sshc:ssh-client-grouping
          |     +-- netconf-client-parameters
          +--:(tls) {tls-listen}?
             +-- tls
                +-- tcp-server-parameters
                |  +---u tcps:tcp-server-grouping
                +-- tls-client-parameters
                |  +---u tlsc:tls-client-grouping
                +-- netconf-client-parameters
     grouping netconf-client-app-grouping
       +-- initiate! {ssh-initiate or tls-initiate}?
       |  +-- netconf-server* [name]
       |     +-- name?                 string
       |     +-- endpoints
       |     |  +-- endpoint* [name]
       |     |     +-- name?                                        string
       |     |     +---u netconf-client-initiate-stack-grouping
       |     +-- connection-type
       |     |  +-- (connection-type)
       |     |     +--:(persistent-connection)
       |     |     |  +-- persistent!
       |     |     +--:(periodic-connection)
       |     |        +-- periodic!
       |     |           +-- period?         uint16
       |     |           +-- anchor-time?    yang:date-and-time
       |     |           +-- idle-timeout?   uint16
       |     +-- reconnect-strategy
       |        +-- start-with?     enumeration
       |        +-- max-attempts?   uint8
       +-- listen! {ssh-listen or tls-listen}?
          +-- idle-timeout?   uint16
          +-- endpoint* [name]
             +-- name?                                    string
             +---u netconf-client-listen-stack-grouping
```

3.2. Example Usage

The following example illustrates configuring a NETCONF client to initiate connections, using both the SSH and TLS transport protocols, as well as listening for call-home connections, again using both the SSH and TLS transport protocols.

This example is consistent with the examples presented in Section 2 of [I-D.ietf-netconf-trust-anchors] and Section 3.2 of [I-D.ietf-netconf-keystore].

```
========== NOTE: '\' line wrapping per BCP XXX (RFC XXXX) ===========

<netconf-client
  xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-client">

  <!-- NETCONF servers to initiate connections to -->
  <initiate>
    <netconf-server>
      <name>corp-fw1</name>
      <endpoints>
        <endpoint>
          <name>corp-fw1.example.com</name>
          <ssh>
            <tcp-client-parameters>
              <remote-address>corp-fw1.example.com</remote-address>
              <keepalives>
                <idle-time>15</idle-time>
                <max-probes>3</max-probes>
                <probe-interval>30</probe-interval>
              </keepalives>
            </tcp-client-parameters>
            <ssh-client-parameters>
              <client-identity>
                <username>foobar</username>
                <public-key>
                  <local-definition>
                    <algorithm>rsa2048</algorithm>
                    <private-key>base64encodedvalue==</private-key>
                    <public-key>base64encodedvalue==</public-key>
                  </local-definition>
                </public-key>
              </client-identity>
              <server-authentication>
                <ca-certs>
                  <truststore-reference>explicitly-trusted-server-ca\
-certs</truststore-reference>
                </ca-certs>
```

```
                  <server-certs>
                    <truststore-reference>explicitly-trusted-server-ce\
   rts</truststore-reference>
                  </server-certs>
                </server-authentication>
                <keepalives>
                  <max-wait>30</max-wait>
                  <max-attempts>3</max-attempts>
                </keepalives>
              </ssh-client-parameters>
              <netconf-client-parameters>
                <!-- nothing to configure -->
              </netconf-client-parameters>
            </ssh>
          </endpoint>
          <endpoint>
            <name>corp-fw2.example.com</name>
            <tls>
              <tcp-client-parameters>
                <remote-address>corp-fw2.example.com</remote-address>
                <keepalives>
                  <idle-time>15</idle-time>
                  <max-probes>3</max-probes>
                  <probe-interval>30</probe-interval>
                </keepalives>
              </tcp-client-parameters>
              <tls-client-parameters>
                <client-identity>
                  <local-definition>
                    <algorithm>rsa2048</algorithm>
                    <private-key>base64encodedvalue==</private-key>
                    <public-key>base64encodedvalue==</public-key>
                    <cert>base64encodedvalue==</cert>
                  </local-definition>
                </client-identity>
                <server-authentication>
                  <ca-certs>
                    <truststore-reference>explicitly-trusted-server-ca\
   -certs</truststore-reference>
                  </ca-certs>
                  <server-certs>
                    <truststore-reference>explicitly-trusted-server-ce\
   rts</truststore-reference>
                  </server-certs>
                </server-authentication>
                <keepalives>
                  <max-wait>30</max-wait>
                  <max-attempts>3</max-attempts>
```

```
                </keepalives>
              </tls-client-parameters>
              <netconf-client-parameters>
                <!-- nothing to configure -->
              </netconf-client-parameters>
            </tls>
          </endpoint>
        </endpoints>
        <connection-type>
          <persistent/>
        </connection-type>
        <reconnect-strategy>
          <start-with>last-connected</start-with>
        </reconnect-strategy>
      </netconf-server>
    </initiate>

    <!-- endpoints to listen for NETCONF Call Home connections on -->
    <listen>
      <endpoint>
        <name>Intranet-facing listener</name>
        <ssh>
          <tcp-server-parameters>
            <local-address>192.0.2.7</local-address>
          </tcp-server-parameters>
          <ssh-client-parameters>
            <client-identity>
              <username>foobar</username>
              <public-key>
                <local-definition>
                  <algorithm>rsa2048</algorithm>
                  <private-key>base64encodedvalue==</private-key>
                  <public-key>base64encodedvalue==</public-key>
                </local-definition>
              </public-key>
            </client-identity>
            <server-authentication>
              <ca-certs>
                <truststore-reference>explicitly-trusted-server-ca-cer\
  ts</truststore-reference>
              </ca-certs>
              <server-certs>
                <truststore-reference>explicitly-trusted-server-certs<\
  /truststore-reference>
              </server-certs>
              <ssh-host-keys>
                <truststore-reference>explicitly-trusted-ssh-host-keys\
  </truststore-reference>
```

```
              </ssh-host-keys>
            </server-authentication>
          </ssh-client-parameters>
          <netconf-client-parameters>
            <!-- nothing to configure -->
          </netconf-client-parameters>
        </ssh>
      </endpoint>
    </listen>
  </netconf-client>
```

3.3. YANG Module

This YANG module has normative references to [RFC6242], [RFC6991], [RFC7589], [RFC8071], [I-D.kwatsen-netconf-tcp-client-server], [I-D.ietf-netconf-ssh-client-server], and [I-D.ietf-netconf-tls-client-server].

<CODE BEGINS> file "ietf-netconf-client@2019-10-18.yang"

```
module ietf-netconf-client {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-netconf-client";
  prefix ncc;

  import ietf-yang-types {
    prefix yang;
    reference
      "RFC 6991: Common YANG Data Types";
  }

  import ietf-tcp-client {
    prefix tcpc;
    reference
      "RFC AAAA: YANG Groupings for TCP Clients and TCP Servers";
  }
  import ietf-tcp-server {
    prefix tcps;
    reference
      "RFC AAAA: YANG Groupings for TCP Clients and TCP Servers";
  }

  import ietf-ssh-client {
    prefix sshc;
    revision-date 2019-10-18; // stable grouping definitions
    reference
      "RFC BBBB: YANG Groupings for SSH Clients and SSH Servers";
  }
```

```
  import ietf-tls-client {
    prefix tlsc;
    revision-date 2019-10-18; // stable grouping definitions
    reference
      "RFC CCCC: YANG Groupings for TLS Clients and TLS Servers";
  }

  organization
    "IETF NETCONF (Network Configuration) Working Group";

  contact
    "WG Web:   <http://datatracker.ietf.org/wg/netconf/>
     WG List:  <mailto:netconf@ietf.org>
     Author:   Kent Watsen <mailto:kent+ietf@watsen.net>
     Author:   Gary Wu <mailto:garywu@cisco.com>";

  description
    "This module contains a collection of YANG definitions
     for configuring NETCONF clients.

     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
     (https://www.rfc-editor.org/info/rfcXXXX); see the RFC
     itself for full legal notices.;

     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.";

  revision 2019-10-18 {
    description
      "Initial version";
    reference
      "RFC XXXX: NETCONF Client and Server Models";
  }
```

```
  // Features

  feature ssh-initiate {
    description
      "The 'ssh-initiate' feature indicates that the NETCONF client
       supports initiating SSH connections to NETCONF servers.";
    reference
      "RFC 6242:
         Using the NETCONF Protocol over Secure Shell (SSH)";
  }

  feature tls-initiate {
    description
      "The 'tls-initiate' feature indicates that the NETCONF client
       supports initiating TLS connections to NETCONF servers.";
    reference
      "RFC 7589: Using the NETCONF Protocol over Transport
         Layer Security (TLS) with Mutual X.509 Authentication";
  }

  feature ssh-listen {
    description
      "The 'ssh-listen' feature indicates that the NETCONF client
       supports opening a port to listen for incoming NETCONF
       server call-home SSH connections.";
    reference
      "RFC 8071: NETCONF Call Home and RESTCONF Call Home";
  }

  feature tls-listen {
    description
      "The 'tls-listen' feature indicates that the NETCONF client
       supports opening a port to listen for incoming NETCONF
       server call-home TLS connections.";
    reference
      "RFC 8071: NETCONF Call Home and RESTCONF Call Home";
  }

  // Groupings

  grouping netconf-client-grouping {
    description
      "A reusable grouping for configuring a NETCONF client
       without any consideration for how underlying transport
       sessions are established.

       This grouping currently doesn't define any nodes.";
  }
```

```
     grouping netconf-client-initiate-stack-grouping {
       description
         "A reusable grouping for configuring a NETCONF client
          'initiate' protocol stack for a single connection.";
       choice transport {
         mandatory true;
         description
           "Selects between available transports.";
         case ssh {
           if-feature "ssh-initiate";
           container ssh {
             description
               "Specifies IP and SSH specific configuration
                for the connection.";
             container tcp-client-parameters {
               description
                 "A wrapper around the TCP client parameters
                  to avoid name collisions.";
               uses tcpc:tcp-client-grouping {
                 refine "remote-port" {
                   default "830";
                   description
                     "The NETCONF client will attempt to connect
                      to the IANA-assigned well-known port value
                      for 'netconf-ssh' (443) if no value is
                      specified.";
                 }
               }
             }
             container ssh-client-parameters {
               description
                 "A wrapper around the SSH client parameters to
                  avoid name collisions.";
               uses sshc:ssh-client-grouping;
             }
             container netconf-client-parameters {
               description
                 "A wrapper around the NETCONF client parameters
                  to avoid name collisions.";
               uses ncc:netconf-client-grouping;
             }
           }
         }
         case tls {
           if-feature "tls-initiate";
           container tls {
             description
               "Specifies IP and TLS specific configuration
```

```
               for the connection.";
            container tcp-client-parameters {
              description
                "A wrapper around the TCP client parameters
                 to avoid name collisions.";
              uses tcpc:tcp-client-grouping {
                refine "remote-port" {
                  default "6513";
                  description
                    "The NETCONF client will attempt to connect
                     to the IANA-assigned well-known port value
                     for 'netconf-tls' (6513) if no value is
                     specified.";
                }
              }
            }
            container tls-client-parameters {
              must "client-identity" {
                description
                  "NETCONF/TLS clients MUST pass some
                   authentication credentials.";
              }
              description
                "A wrapper around the TLS client parameters
                 to avoid name collisions.";
              uses tlsc:tls-client-grouping;
            }
            container netconf-client-parameters {
              description
                "A wrapper around the NETCONF client parameters
                 to avoid name collisions.";
              uses ncc:netconf-client-grouping;
            }
          }
        }
      }
    } // netconf-client-initiate-stack-grouping

    grouping netconf-client-listen-stack-grouping {
      description
        "A reusable grouping for configuring a NETCONF client
         'listen' protocol stack for a single connection.";
      choice transport {
        mandatory true;
        description
          "Selects between available transports.";
        case ssh {
          if-feature "ssh-listen";
```

```
          container ssh {
            description
              "SSH-specific listening configuration for inbound
               connections.";
            container tcp-server-parameters {
              description
                "A wrapper around the TCP server parameters
                 to avoid name collisions.";
              uses tcps:tcp-server-grouping {
                refine "local-port" {
                  default "4334";
                  description
                    "The NETCONF client will listen on the IANA-
                     assigned well-known port for 'netconf-ch-ssh'
                     (4334) if no value is specified.";
                }
              }
            }
            container ssh-client-parameters {
              description
                "A wrapper around the SSH client parameters
                 to avoid name collisions.";
              uses sshc:ssh-client-grouping;
            }
            container netconf-client-parameters {
              description
                "A wrapper around the NETCONF client parameters
                 to avoid name collisions.";
              uses ncc:netconf-client-grouping;
            }
          }
        }
        case tls {
          if-feature "tls-listen";
          container tls {
            description
              "TLS-specific listening configuration for inbound
               connections.";
            container tcp-server-parameters {
              description
                "A wrapper around the TCP server parameters
                 to avoid name collisions.";
              uses tcps:tcp-server-grouping {
                refine "local-port" {
                  default "4334";
                  description
                    "The NETCONF client will listen on the IANA-
                      assigned well-known port for 'netconf-ch-ssh'
```

```
                    (4334) if no value is specified.";
                }
              }
            }
            container tls-client-parameters {
              must "client-identity" {
                description
                  "NETCONF/TLS clients MUST pass some
                   authentication credentials.";
              }
              description
                "A wrapper around the TLS client parameters
                 to avoid name collisions.";
              uses tlsc:tls-client-grouping;
            }
            container netconf-client-parameters {
              description
                "A wrapper around the NETCONF client parameters
                 to avoid name collisions.";
              uses ncc:netconf-client-grouping;
            }
          }
        }
      }
    } // netconf-client-listen-stack-grouping

    grouping netconf-client-app-grouping {
      description
        "A reusable grouping for configuring a NETCONF client
         application that supports both 'initiate' and 'listen'
         protocol stacks for a multiplicity of connections.";
      container initiate {
        if-feature "ssh-initiate or tls-initiate";
        presence "Enables client to initiate TCP connections";
        description
          "Configures client initiating underlying TCP connections.";
        list netconf-server {
          key "name";
          min-elements 1;
          description
            "List of NETCONF servers the NETCONF client is to
             maintain simultaneous connections with.";
          leaf name {
            type string;
            description
              "An arbitrary name for the NETCONF server.";
          }
          container endpoints {
```

```
        description
          "Container for the list of endpoints.";
        list endpoint {
          key "name";
          min-elements 1;
          ordered-by user;
          description
            "A user-ordered list of endpoints that the NETCONF
             client will attempt to connect to in the specified
             sequence.  Defining more than one enables
             high-availability.";
          leaf name {
            type string;
            description
              "An arbitrary name for the endpoint.";
          }
          uses netconf-client-initiate-stack-grouping;
        } // list endpoint
      } // container endpoints

      container connection-type {
        description
          "Indicates the NETCONF client's preference for how the
           NETCONF connection is maintained.";
        choice connection-type {
          mandatory true;
          description
            "Selects between available connection types.";
          case persistent-connection {
            container persistent {
              presence "Indicates that a persistent connection is
                        to be maintained.";
              description
                "Maintain a persistent connection to the NETCONF
                 server.  If the connection goes down, immediately
                 start trying to reconnect to the NETCONF server,
                 using the reconnection strategy.

                 This connection type minimizes any NETCONF server
                 to NETCONF client data-transfer delay, albeit at
                 the expense of holding resources longer.";
            }
          }
          case periodic-connection {
            container periodic {
              presence "Indicates that a periodic connection is
                        to be maintained.";
              description
```

```
                "Periodically connect to the NETCONF server.

                 This connection type increases resource
                 utilization, albeit with increased delay in
                 NETCONF server to NETCONF client interactions.

                 The NETCONF client should close the underlying
                 TCP connection upon completing planned activities.

                 In the case that the previous connection is still
                 active, establishing a new connection is NOT
                 RECOMMENDED.";
              leaf period {
                type uint16;
                units "minutes";
                default "60";
                description
                  "Duration of time between periodic connections.";
              }
              leaf anchor-time {
                type yang:date-and-time {
                  // constrained to minute-level granularity
                  pattern '\d{4}-\d{2}-\d{2}T\d{2}:\d{2}'
                        + '(Z|[\+\-]\d{2}:\d{2})';
                }
                description
                  "Designates a timestamp before or after which a
                   series of periodic connections are determined.
                   The periodic connections occur at a whole
                   multiple interval from the anchor time.  For
                   example, for an anchor time is 15 minutes past
                   midnight and a period interval of 24 hours, then
                   a periodic connection will occur 15 minutes past
                   midnight everyday.";
              }
              leaf idle-timeout {
                type uint16;
                units "seconds";
                default 120; // two minutes
                description
                  "Specifies the maximum number of seconds that
                   a NETCONF session may remain idle. A NETCONF
                   session will be dropped if it is idle for an
                   interval longer then this number of seconds.
                   If set to zero, then the NETCONF client will
                   never drop a session because it is idle.";
              }
            }
```

```
            }
          }
        }
        container reconnect-strategy {
          description
            "The reconnection strategy directs how a NETCONF client
             reconnects to a NETCONF server, after discovering its
             connection to the server has dropped, even if due to a
             reboot.  The NETCONF client starts with the specified
             endpoint and tries to connect to it max-attempts times
             before trying the next endpoint in the list (round
             robin).";
          leaf start-with {
            type enumeration {
              enum first-listed {
                description
                  "Indicates that reconnections should start with
                   the first endpoint listed.";
              }
              enum last-connected {
                description
                  "Indicates that reconnections should start with
                   the endpoint last connected to.  If no previous
                   connection has ever been established, then the
                   first endpoint configured is used.   NETCONF
                   clients SHOULD be able to remember the last
                   endpoint connected to across reboots.";
              }
              enum random-selection {
                description
                  "Indicates that reconnections should start with
                   a random endpoint.";
              }
            }
            default "first-listed";
            description
              "Specifies which of the NETCONF server's endpoints
               the NETCONF client should start with when trying
               to connect to the NETCONF server.";
          }
          leaf max-attempts {
            type uint8 {
              range "1..max";
            }
            default "3";
            description
              "Specifies the number times the NETCONF client tries
               to connect to a specific endpoint before moving on
```

```
                to the next endpoint in the list (round robin).";
            }
          }
        } // netconf-server
      } // initiate

      container listen {
        if-feature "ssh-listen or tls-listen";
        presence "Enables client to accept call-home connections";
        description
          "Configures client accepting call-home TCP connections.";
        leaf idle-timeout {
          type uint16;
          units "seconds";
          default "3600"; // one hour
          description
            "Specifies the maximum number of seconds that a NETCONF
             session may remain idle. A NETCONF session will be
             dropped if it is idle for an interval longer than this
             number of seconds.  If set to zero, then the server
             will never drop a session because it is idle.  Sessions
             that have a notification subscription active are never
             dropped.";
        }
        list endpoint {
          key "name";
          min-elements 1;
          description
            "List of endpoints to listen for NETCONF connections.";
          leaf name {
            type string;
            description
              "An arbitrary name for the NETCONF listen endpoint.";
          }
          uses netconf-client-listen-stack-grouping;
        } // endpoint
      } // listen
    } // netconf-client-app-grouping

    // Protocol accessible node, for servers that implement this
    // module.

    container netconf-client {
      uses netconf-client-app-grouping;
      description
        "Top-level container for NETCONF client configuration.";
    }
  }
```

<CODE ENDS>

4. The NETCONF Server Model

The NETCONF server model presented in this section supports both listening for connections as well as initiating call-home connections, using either the SSH and TLS transport protocols.

YANG feature statements are used to enable implementations to advertise which potentially uncommon parts of the model the NETCONF server supports.

4.1. Tree Diagram

The following tree diagram [RFC8340] provides an overview of the data model for the "ietf-netconf-server" module.

This tree diagram only shows the nodes defined in this module; it does show the nodes defined by "grouping" statements used by this module.

Please see Appendix A.2 for a tree diagram that illustrates what the module looks like with all the "grouping" statements expanded.

```
module: ietf-netconf-server
  +--rw netconf-server
     +---u netconf-server-app-grouping

  grouping netconf-server-grouping
    +-- client-identification
       +-- cert-maps
          +---u x509c2n:cert-to-name
  grouping netconf-server-listen-stack-grouping
    +-- (transport)
       +--:(ssh) {ssh-listen}?
       |  +-- ssh
       |     +-- tcp-server-parameters
       |     |  +---u tcps:tcp-server-grouping
       |     +-- ssh-server-parameters
       |     |  +---u sshs:ssh-server-grouping
       |     +-- netconf-server-parameters
       |        +---u ncs:netconf-server-grouping
       +--:(tls) {tls-listen}?
          +-- tls
             +-- tcp-server-parameters
             |  +---u tcps:tcp-server-grouping
             +-- tls-server-parameters
             |  +---u tlss:tls-server-grouping
```

```
               +-- netconf-server-parameters
                  +---u ncs:netconf-server-grouping
     grouping netconf-server-callhome-stack-grouping
       +-- (transport)
          +--:(ssh) {ssh-call-home}?
          |  +-- ssh
          |     +-- tcp-client-parameters
          |     |  +---u tcpc:tcp-client-grouping
          |     +-- ssh-server-parameters
          |     |  +---u sshs:ssh-server-grouping
          |     +-- netconf-server-parameters
          |        +---u ncs:netconf-server-grouping
          +--:(tls) {tls-call-home}?
             +-- tls
                +-- tcp-client-parameters
                |  +---u tcpc:tcp-client-grouping
                +-- tls-server-parameters
                |  +---u tlss:tls-server-grouping
                +-- netconf-server-parameters
                   +---u ncs:netconf-server-grouping
     grouping netconf-server-app-grouping
       +-- listen! {ssh-listen or tls-listen}?
       |  +-- idle-timeout?   uint16
       |  +-- endpoint* [name]
       |     +-- name?                                    string
       |     +---u netconf-server-listen-stack-grouping
       +-- call-home! {ssh-call-home or tls-call-home}?
          +-- netconf-client* [name]
             +-- name?                string
             +-- endpoints
             |  +-- endpoint* [name]
             |     +-- name?                                        string
             |     +---u netconf-server-callhome-stack-grouping
             +-- connection-type
             |  +-- (connection-type)
             |     +--:(persistent-connection)
             |     |  +-- persistent!
             |     +--:(periodic-connection)
             |        +-- periodic!
             |           +-- period?         uint16
             |           +-- anchor-time?    yang:date-and-time
             |           +-- idle-timeout?   uint16
             +-- reconnect-strategy
                +-- start-with?     enumeration
                +-- max-attempts?   uint8
```

4.2. Example Usage

The following example illustrates configuring a NETCONF server to listen for NETCONF client connections using both the SSH and TLS transport protocols, as well as configuring call-home to two NETCONF clients, one using SSH and the other using TLS.

This example is consistent with the examples presented in Section 2 of [I-D.ietf-netconf-trust-anchors] and Section 3.2 of [I-D.ietf-netconf-keystore].

========== NOTE: '\' line wrapping per BCP XXX (RFC XXXX) ==========

```
<netconf-server
  xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-server"
  xmlns:x509c2n="urn:ietf:params:xml:ns:yang:ietf-x509-cert-to-name">

  <!-- endpoints to listen for NETCONF connections on -->
  <listen>
    <endpoint> <!-- listening for SSH connections -->
      <name>netconf/ssh</name>
      <ssh>
        <tcp-server-parameters>
          <local-address>192.0.2.7</local-address>
        </tcp-server-parameters>
        <ssh-server-parameters>
          <server-identity>
            <host-key>
              <name>deployment-specific-certificate</name>
              <public-key>
                <local-definition>
                  <algorithm>rsa2048</algorithm>
                  <private-key>base64encodedvalue==</private-key>
                  <public-key>base64encodedvalue==</public-key>
                </local-definition>
              </public-key>
            </host-key>
          </server-identity>
          <client-authentication>
            <supported-authentication-methods>
              <publickey/>
            </supported-authentication-methods>
            <client-auth-defined-elsewhere/>
          </client-authentication>
        </ssh-server-parameters>
        <netconf-server-parameters>
          <!-- nothing to configure -->
        </netconf-server-parameters>
```

```
        </ssh>
      </endpoint>
      <endpoint> <!-- listening for TLS sessions -->
        <name>netconf/tls</name>
        <tls>
          <tcp-server-parameters>
            <local-address>192.0.2.7</local-address>
          </tcp-server-parameters>
          <tls-server-parameters>
            <server-identity>
              <local-definition>
                <algorithm>rsa2048</algorithm>
                <private-key>base64encodedvalue==</private-key>
                <public-key>base64encodedvalue==</public-key>
                <cert>base64encodedvalue==</cert>
              </local-definition>
            </server-identity>
            <client-authentication>
              <required/>
              <ca-certs>
                <truststore-reference>explicitly-trusted-client-ca-cer\
  ts</truststore-reference>
              </ca-certs>
              <client-certs>
                <truststore-reference>explicitly-trusted-client-certs<\
  /truststore-reference>
              </client-certs>
            </client-authentication>
          </tls-server-parameters>
          <netconf-server-parameters>
            <client-identification>
              <cert-maps>
                <cert-to-name>
                  <id>1</id>
                  <fingerprint>11:0A:05:11:00</fingerprint>
                  <map-type>x509c2n:san-any</map-type>
                </cert-to-name>
                <cert-to-name>
                  <id>2</id>
                  <fingerprint>B3:4F:A1:8C:54</fingerprint>
                  <map-type>x509c2n:specified</map-type>
                  <name>scooby-doo</name>
                </cert-to-name>
              </cert-maps>
            </client-identification>
          </netconf-server-parameters>
        </tls>
      </endpoint>
```

```
    </listen>

    <!-- calling home to SSH and TLS based NETCONF clients -->
    <call-home>
      <netconf-client> <!-- SSH-based client -->
        <name>config-mgr</name>
        <endpoints>
          <endpoint>
            <name>east-data-center</name>
            <ssh>
              <tcp-client-parameters>
                <remote-address>east.config-mgr.example.com</remote-ad\
dress>
              </tcp-client-parameters>
              <ssh-server-parameters>
                <server-identity>
                  <host-key>
                    <name>deployment-specific-certificate</name>
                    <public-key>
                      <local-definition>
                        <algorithm>rsa2048</algorithm>
                        <private-key>base64encodedvalue==</private-key>
                        <public-key>base64encodedvalue==</public-key>
                      </local-definition>
                    </public-key>
                  </host-key>
                </server-identity>
                <client-authentication>
                  <supported-authentication-methods>
                    <publickey/>
                  </supported-authentication-methods>
                  <client-auth-defined-elsewhere/>
                </client-authentication>
              </ssh-server-parameters>
              <netconf-server-parameters>
                <!-- nothing to configure -->
              </netconf-server-parameters>
            </ssh>
          </endpoint>
          <endpoint>
            <name>west-data-center</name>
            <ssh>
              <tcp-client-parameters>
                <remote-address>west.config-mgr.example.com</remote-ad\
dress>
              </tcp-client-parameters>
              <ssh-server-parameters>
                <server-identity>
```

```
                  <host-key>
                    <name>deployment-specific-certificate</name>
                    <public-key>
                      <local-definition>
                        <algorithm>rsa2048</algorithm>
                        <private-key>base64encodedvalue==</private-key>
                        <public-key>base64encodedvalue==</public-key>
                      </local-definition>
                    </public-key>
                  </host-key>
                </server-identity>
                <client-authentication>
                  <supported-authentication-methods>
                    <publickey/>
                  </supported-authentication-methods>
                  <client-auth-defined-elsewhere/>
                </client-authentication>
              </ssh-server-parameters>
              <netconf-server-parameters>
                <!-- nothing to configure -->
              </netconf-server-parameters>
            </ssh>
          </endpoint>
        </endpoints>
        <connection-type>
          <periodic>
            <idle-timeout>300</idle-timeout>
            <period>60</period>
          </periodic>
        </connection-type>
        <reconnect-strategy>
          <start-with>last-connected</start-with>
          <max-attempts>3</max-attempts>
        </reconnect-strategy>
      </netconf-client>
      <netconf-client> <!-- TLS-based client -->
        <name>data-collector</name>
        <endpoints>
          <endpoint>
            <name>east-data-center</name>
            <tls>
              <tcp-client-parameters>
                <remote-address>east.analytics.example.com</remote-add\
   ress>
                <keepalives>
                  <idle-time>15</idle-time>
                  <max-probes>3</max-probes>
                  <probe-interval>30</probe-interval>
```

```
                </keepalives>
              </tcp-client-parameters>
              <tls-server-parameters>
                <server-identity>
                  <local-definition>
                    <algorithm>rsa2048</algorithm>
                    <private-key>base64encodedvalue==</private-key>
                    <public-key>base64encodedvalue==</public-key>
                    <cert>base64encodedvalue==</cert>
                  </local-definition>
                </server-identity>
                <client-authentication>
                  <required/>
                  <ca-certs>
                    <truststore-reference>explicitly-trusted-client-ca\
  -certs</truststore-reference>
                  </ca-certs>
                  <client-certs>
                    <truststore-reference>explicitly-trusted-client-ce\
  rts</truststore-reference>
                  </client-certs>
                </client-authentication>
                <keepalives>
                  <max-wait>30</max-wait>
                  <max-attempts>3</max-attempts>
                </keepalives>
              </tls-server-parameters>
              <netconf-server-parameters>
                <client-identification>
                  <cert-maps>
                    <cert-to-name>
                      <id>1</id>
                      <fingerprint>11:0A:05:11:00</fingerprint>
                      <map-type>x509c2n:san-any</map-type>
                    </cert-to-name>
                    <cert-to-name>
                      <id>2</id>
                      <fingerprint>B3:4F:A1:8C:54</fingerprint>
                      <map-type>x509c2n:specified</map-type>
                      <name>scooby-doo</name>
                    </cert-to-name>
                  </cert-maps>
                </client-identification>
              </netconf-server-parameters>
            </tls>
          </endpoint>
          <endpoint>
            <name>west-data-center</name>
```

```
            <tls>
              <tcp-client-parameters>
                <remote-address>west.analytics.example.com</remote-add\
   ress>
                <keepalives>
                  <idle-time>15</idle-time>
                  <max-probes>3</max-probes>
                  <probe-interval>30</probe-interval>
                </keepalives>
              </tcp-client-parameters>
              <tls-server-parameters>
                <server-identity>
                  <local-definition>
                    <algorithm>rsa2048</algorithm>
                    <private-key>base64encodedvalue==</private-key>
                    <public-key>base64encodedvalue==</public-key>
                    <cert>base64encodedvalue==</cert>
                  </local-definition>
                </server-identity>
                <client-authentication>
                  <required/>
                  <ca-certs>
                    <truststore-reference>explicitly-trusted-client-ca\
   -certs</truststore-reference>
                  </ca-certs>
                  <client-certs>
                    <truststore-reference>explicitly-trusted-client-ce\
   rts</truststore-reference>
                  </client-certs>
                </client-authentication>
                <keepalives>
                  <max-wait>30</max-wait>
                  <max-attempts>3</max-attempts>
                </keepalives>
              </tls-server-parameters>
              <netconf-server-parameters>
                <client-identification>
                  <cert-maps>
                    <cert-to-name>
                      <id>1</id>
                      <fingerprint>11:0A:05:11:00</fingerprint>
                      <map-type>x509c2n:san-any</map-type>
                    </cert-to-name>
                    <cert-to-name>
                      <id>2</id>
                      <fingerprint>B3:4F:A1:8C:54</fingerprint>
                      <map-type>x509c2n:specified</map-type>
                      <name>scooby-doo</name>
```

```
                  </cert-to-name>
                </cert-maps>
              </client-identification>
            </netconf-server-parameters>
          </tls>
        </endpoint>
      </endpoints>
      <connection-type>
        <persistent/>
      </connection-type>
      <reconnect-strategy>
        <start-with>first-listed</start-with>
        <max-attempts>3</max-attempts>
      </reconnect-strategy>
    </netconf-client>
  </call-home>
</netconf-server>
```

4.3. YANG Module

This YANG module has normative references to [RFC6242], [RFC6991], [RFC7407], [RFC7589], [RFC8071], [I-D.kwatsen-netconf-tcp-client-server], [I-D.ietf-netconf-ssh-client-server], and [I-D.ietf-netconf-tls-client-server].

<CODE BEGINS> file "ietf-netconf-server@2019-10-18.yang"

```
module ietf-netconf-server {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-netconf-server";
  prefix ncs;

  import ietf-yang-types {
    prefix yang;
    reference
      "RFC 6991: Common YANG Data Types";
  }

  import ietf-x509-cert-to-name {
    prefix x509c2n;
    reference
      "RFC 7407: A YANG Data Model for SNMP Configuration";
  }

  import ietf-tcp-client {
    prefix tcpc;
    reference
```

```
        "RFC AAAA: YANG Groupings for TCP Clients and TCP Servers";
    }

    import ietf-tcp-server {
      prefix tcps;
      reference
        "RFC AAAA: YANG Groupings for TCP Clients and TCP Servers";
    }

    import ietf-ssh-server {
      prefix sshs;
      revision-date 2019-10-18; // stable grouping definitions
      reference
        "RFC BBBB: YANG Groupings for SSH Clients and SSH Servers";
    }

    import ietf-tls-server {
      prefix tlss;
      revision-date 2019-10-18; // stable grouping definitions
      reference
        "RFC CCCC: YANG Groupings for TLS Clients and TLS Servers";
    }

    organization
      "IETF NETCONF (Network Configuration) Working Group";

    contact
      "WG Web:   <http://datatracker.ietf.org/wg/netconf/>
       WG List:  <mailto:netconf@ietf.org>
       Author:   Kent Watsen <mailto:kent+ietf@watsen.net>
       Author:   Gary Wu <mailto:garywu@cisco.com>
       Author:   Juergen Schoenwaelder
                 <mailto:j.schoenwaelder@jacobs-university.de>";
    description
      "This module contains a collection of YANG definitions
       for configuring NETCONF servers.

       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
```

```
      (https://www.rfc-editor.org/info/rfcXXXX); see the RFC
      itself for full legal notices.;

      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.";

    revision 2019-10-18 {
      description
        "Initial version";
      reference
        "RFC XXXX: NETCONF Client and Server Models";
    }

    // Features

    feature ssh-listen {
      description
        "The 'ssh-listen' feature indicates that the NETCONF server
         supports opening a port to accept NETCONF over SSH
         client connections.";
      reference
        "RFC 6242:
           Using the NETCONF Protocol over Secure Shell (SSH)";
    }

    feature tls-listen {
      description
        "The 'tls-listen' feature indicates that the NETCONF server
         supports opening a port to accept NETCONF over TLS
         client connections.";
      reference
        "RFC 7589: Using the NETCONF Protocol over Transport
                   Layer Security (TLS) with Mutual X.509
                   Authentication";
    }

    feature ssh-call-home {
      description
        "The 'ssh-call-home' feature indicates that the NETCONF
         server supports initiating a NETCONF over SSH call
         home connection to NETCONF clients.";
      reference
        "RFC 8071: NETCONF Call Home and RESTCONF Call Home";
    }
```

```
    feature tls-call-home {
      description
        "The 'tls-call-home' feature indicates that the NETCONF
         server supports initiating a NETCONF over TLS call
         home connection to NETCONF clients.";
      reference
        "RFC 8071: NETCONF Call Home and RESTCONF Call Home";
    }

    // Groupings

    grouping netconf-server-grouping {
      description
        "A reusable grouping for configuring a NETCONF server
         without any consideration for how underlying transport
         sessions are established.

         Note that this grouping uses a fairly typical descendent
         node name such that a stack of 'uses' statements will
         have name conflicts.  It is intended that the consuming
         data model will resolve the issue by wrapping the 'uses'
         statement in a container called, e.g.,
         'netconf-server-parameters'.  This model purposely does
         not do this itself so as to provide maximum flexibility
         to consuming models.";

      container client-identification {
        description
          "Specifies a mapping through which clients MAY be identified
           (i.e., the NETCONF username) from a supplied certificate.
           Note that a client MAY alternatively be identified via an
           HTTP-level authentication schema.  This configuration does
           not necessitate clients send a certificate (that can be
           controlled via the ietf-netconf-server module).";
        container cert-maps {
          when "../../../../tls";
          uses x509c2n:cert-to-name;
          description
            "The cert-maps container is used by TLS-based NETCONF
             servers (even if the TLS sessions are terminated
             externally) to map the NETCONF client's presented
             X.509 certificate to a NETCONF username.  If no
             matching and valid cert-to-name list entry can be
             found, then the NETCONF server MUST close the
             connection, and MUST NOT accept NETCONF messages
             over it.";
          reference
            "RFC 7407: A YANG Data Model for SNMP Configuration.";
```

```
        }
      }
    }

    grouping netconf-server-listen-stack-grouping {
      description
        "A reusable grouping for configuring a NETCONF server
         'listen' protocol stack for a single connection.";
      choice transport {
        mandatory true;
        description
          "Selects between available transports.";
        case ssh {
          if-feature "ssh-listen";
          container ssh {
            description
              "SSH-specific listening configuration for inbound
               connections.";
            container tcp-server-parameters {
              description
                "A wrapper around the TCP client parameters
                 to avoid name collisions.";
              uses tcps:tcp-server-grouping {
                refine "local-port" {
                  default "830";
                  description
                    "The NETCONF server will listen on the
                     IANA-assigned well-known port value
                     for 'netconf-ssh' (830) if no value
                     is specified.";
                }
              }
            }
            container ssh-server-parameters {
              description
                "A wrapper around the SSH server parameters
                 to avoid name collisions.";
              uses sshs:ssh-server-grouping;
            }
            container netconf-server-parameters {
              description
                "A wrapper around the NETCONF server parameters
                 to avoid name collisions.";
              uses ncs:netconf-server-grouping;
            }
          }
        }
        case tls {
```

```
            if-feature "tls-listen";
            container tls {
              description
                "TLS-specific listening configuration for inbound
                 connections.";
              container tcp-server-parameters {
                description
                  "A wrapper around the TCP client parameters
                   to avoid name collisions.";
                uses tcps:tcp-server-grouping {
                  refine "local-port" {
                    default "6513";
                    description
                      "The NETCONF server will listen on the
                       IANA-assigned well-known port value
                       for 'netconf-tls' (6513) if no value
                       is specified.";
                  }
                }
              }
              container tls-server-parameters {
                description
                  "A wrapper around the TLS server parameters to
                   avoid name collisions.";
                uses tlss:tls-server-grouping {
                  refine "client-authentication" {
                    //must 'ca-certs or client-certs';
                    description
                      "NETCONF/TLS servers MUST validate client
                       certificates.";
                  }
                }
              }
              container netconf-server-parameters {
                description
                  "A wrapper around the NETCONF server parameters
                   to avoid name collisions.";
                uses ncs:netconf-server-grouping;
              }
            }
          }
        }
      }

      grouping netconf-server-callhome-stack-grouping {
        description
          "A reusable grouping for configuring a NETCONF server
           'call-home' protocol stack, for a single connection.";
```

```
      choice transport {
        mandatory true;
        description
          "Selects between available transports.";
        case ssh {
          if-feature "ssh-call-home";
          container ssh {
            description
              "Specifies SSH-specific call-home transport
               configuration.";
            container tcp-client-parameters {
              description
                "A wrapper around the TCP client parameters
                 to avoid name collisions.";
              uses tcpc:tcp-client-grouping {
                refine "remote-port" {
                  default "4334";
                  description
                    "The NETCONF server will attempt to connect
                     to the IANA-assigned well-known port for
                     'netconf-ch-tls' (4334) if no value is
                     specified.";
                }
              }
            }
            container ssh-server-parameters {
              description
                "A wrapper around the SSH server parameters
                 to avoid name collisions.";
              uses sshs:ssh-server-grouping;
            }
            container netconf-server-parameters {
              description
                "A wrapper around the NETCONF server parameters
                 to avoid name collisions.";
              uses ncs:netconf-server-grouping;
            }
          }
        }
        case tls {
          if-feature "tls-call-home";
          container tls {
            description
              "Specifies TLS-specific call-home transport
               configuration.";
            container tcp-client-parameters {
              description
                "A wrapper around the TCP client parameters
```

```
          to avoid name collisions.";
        uses tcpc:tcp-client-grouping {
          refine "remote-port" {
            default "4335";
            description
              "The NETCONF server will attempt to connect
               to the IANA-assigned well-known port for
               'netconf-ch-tls' (4335) if no value is
               specified.";
          }
        }
      }
      container tls-server-parameters {
        description
          "A wrapper around the TLS server parameters
           to avoid name collisions.";
        uses tlss:tls-server-grouping {
          refine "client-authentication" {
            /* commented out since auth could be external
          must 'ca-certs or client-certs';
          */
            description
              "NETCONF/TLS servers MUST validate client
               certificates.";
          }
          augment "client-authentication" {
            description
              "Augments in the cert-to-name structure.";
            container cert-maps {
              uses x509c2n:cert-to-name;
              description
                "The cert-maps container is used by a
                 TLS-based NETCONF server to map the
                 NETCONF client's presented X.509
                 certificate to a NETCONF username.  If
                 no matching and valid cert-to-name list
                 entry can be found, then the NETCONF
                 server MUST close the connection, and
                 MUST NOT accept NETCONF messages over
                 it.";
              reference
                "RFC WWWW: NETCONF over TLS, Section 7";
            }
          }
        }
      }
      container netconf-server-parameters {
        description
```

```
                "A wrapper around the NETCONF server parameters
                 to avoid name collisions.";
              uses ncs:netconf-server-grouping;
            }
          }
        }
      }
    }

    grouping netconf-server-app-grouping {
      description
        "A reusable grouping for configuring a NETCONF server
         application that supports both 'listen' and 'call-home'
         protocol stacks for a multiplicity of connections.";
      container listen {
        if-feature "ssh-listen or tls-listen";
        presence
          "Enables server to listen for NETCONF client connections.";
        description
          "Configures listen behavior";
        leaf idle-timeout {
          type uint16;
          units "seconds";
          default 3600; // one hour
          description
            "Specifies the maximum number of seconds that a NETCONF
             session may remain idle. A NETCONF session will be
             dropped if it is idle for an interval longer than this
             number of seconds.  If set to zero, then the server
             will never drop a session because it is idle.  Sessions
             that have a notification subscription active are never
             dropped.";
        }
        list endpoint {
          key "name";
          min-elements 1;
          description
            "List of endpoints to listen for NETCONF connections.";
          leaf name {
            type string;
            description
              "An arbitrary name for the NETCONF listen endpoint.";
          }
          uses netconf-server-listen-stack-grouping;
        }
      }
      container call-home {
        if-feature "ssh-call-home or tls-call-home";
```

```
          presence
            "Enables the NETCONF server to initiate the underlying
             transport connection to NETCONF clients.";
          description "Configures call home behavior.";
          list netconf-client {
            key "name";
            min-elements 1;
            description
              "List of NETCONF clients the NETCONF server is to
               maintain simultaneous call-home connections with.";
            leaf name {
              type string;
              description
                "An arbitrary name for the remote NETCONF client.";
            }
            container endpoints {
              description
                "Container for the list of endpoints.";
              list endpoint {
                key "name";
                min-elements 1;
                ordered-by user;
                description
                  "A non-empty user-ordered list of endpoints for this
                   NETCONF server to try to connect to in sequence.
                   Defining more than one enables high-availability.";
                leaf name {
                  type string;
                  description
                    "An arbitrary name for this endpoint.";
                }
                uses netconf-server-callhome-stack-grouping;
              }
            }
            container connection-type {
              description
                "Indicates the NETCONF server's preference for how the
                 NETCONF connection is maintained.";
              choice connection-type {
                mandatory true;
                description
                  "Selects between available connection types.";
                case persistent-connection {
                  container persistent {
                    presence "Indicates that a persistent connection is
                              to be maintained.";
                    description
                      "Maintain a persistent connection to the NETCONF
```

```
                    client. If the connection goes down, immediately
                    start trying to reconnect to the NETCONF client,
                    using the reconnection strategy.

                    This connection type minimizes any NETCONF client
                    to NETCONF server data-transfer delay, albeit at
                    the expense of holding resources longer.";
                }
              }
              case periodic-connection {
                container periodic {
                  presence "Indicates that a periodic connection is
                            to be maintained.";
                  description
                    "Periodically connect to the NETCONF client.

                     This connection type increases resource
                     utilization, albeit with increased delay in
                     NETCONF client to NETCONF client interactions.

                     The NETCONF client SHOULD gracefully close the
                     connection using <close-session> upon completing
                     planned activities.  If the NETCONF session is
                     not closed gracefully, the NETCONF server MUST
                     immediately attempt to reestablish the connection.

                     In the case that the previous connection is still
                     active (i.e., the NETCONF client has not closed
                     it yet), establishing a new connection is NOT
                     RECOMMENDED.";
                  leaf period {
                    type uint16;
                    units "minutes";
                    default "60";
                    description
                      "Duration of time between periodic connections.";
                  }
                  leaf anchor-time {
                    type yang:date-and-time {
                      // constrained to minute-level granularity
                      pattern '\d{4}-\d{2}-\d{2}T\d{2}:\d{2}'
                            + '(Z|[\+\-]\d{2}:\d{2})';
                    }
                    description
                      "Designates a timestamp before or after which a
                       series of periodic connections are determined.
                       The periodic connections occur at a whole
                       multiple interval from the anchor time.  For
```

```
                    example, for an anchor time is 15 minutes past
                    midnight and a period interval of 24 hours, then
                    a periodic connection will occur 15 minutes past
                    midnight everyday.";
                }
                leaf idle-timeout {
                  type uint16;
                  units "seconds";
                  default 120; // two minutes
                  description
                    "Specifies the maximum number of seconds that
                     a NETCONF session may remain idle. A NETCONF
                     session will be dropped if it is idle for an
                     interval longer than this number of seconds.
                     If set to zero, then the server will never
                     drop a session because it is idle.";
                }
              }
            } // case periodic-connection
          } // choice connection-type
        } // container connection-type
        container reconnect-strategy {
          description
            "The reconnection strategy directs how a NETCONF server
             reconnects to a NETCONF client, after discovering its
             connection to the client has dropped, even if due to a
             reboot.  The NETCONF server starts with the specified
             endpoint and tries to connect to it max-attempts times
             before trying the next endpoint in the list (round
             robin).";
          leaf start-with {
            type enumeration {
              enum first-listed {
                description
                  "Indicates that reconnections should start with
                   the first endpoint listed.";
              }
              enum last-connected {
                description
                  "Indicates that reconnections should start with
                   the endpoint last connected to.  If no previous
                   connection has ever been established, then the
                   first endpoint configured is used.   NETCONF
                   servers SHOULD be able to remember the last
                   endpoint connected to across reboots.";
              }
              enum random-selection {
                description
```

```
                    "Indicates that reconnections should start with
                     a random endpoint.";
                  }
                }
                default "first-listed";
                description
                  "Specifies which of the NETCONF client's endpoints
                   the NETCONF server should start with when trying
                   to connect to the NETCONF client.";
              }
              leaf max-attempts {
                type uint8 {
                  range "1..max";
                }
                default "3";
                description
                  "Specifies the number times the NETCONF server tries
                   to connect to a specific endpoint before moving on
                   to the next endpoint in the list (round robin).";
              }
            } // container reconnect-strategy
          } // list netconf-client
        } // container call-home
      } // grouping netconf-server-app-grouping

      // Protocol accessible node, for servers that implement this
      // module.

      container netconf-server {
        uses netconf-server-app-grouping;
        description
          "Top-level container for NETCONF server configuration.";
      }
    }

    <CODE ENDS>
```

5. Security Considerations

The YANG module defined in this document uses groupings defined in [I-D.kwatsen-netconf-tcp-client-server], [I-D.ietf-netconf-ssh-client-server], and [I-D.ietf-netconf-tls-client-server]. Please see the Security Considerations section in those documents for concerns related those groupings.

The YANG modules defined in this document are designed to be accessed via YANG based management protocols, such as NETCONF [RFC6241] and

RESTCONF [RFC8040]. Both of these protocols have mandatory-to-implement secure transport layers (e.g., SSH, TLS) with mutual authentication.

The NETCONF access control model (NACM) [RFC8341] provides the means to restrict access for particular users to a pre-configured subset of all available protocol operations and content.

There are a number of data nodes defined in the YANG modules that are writable/creatable/deletable (i.e., config true, which is the default). Some of 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:

None of the subtrees or data nodes in the modules defined in this document need to be protected from write operations.

Some of the readable data nodes in the YANG modules 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:

None of the subtrees or data nodes in the modules defined in this document need to be protected from read operations.

Some of the RPC operations in the YANG modules 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:

The modules defined in this document do not define any 'RPC' or 'action' statements.

6. IANA Considerations

6.1. The IETF XML Registry

This document registers two URIs in the "ns" subregistry of the IETF XML Registry [RFC3688]. Following the format in [RFC3688], the following registrations are requested:

```
URI: urn:ietf:params:xml:ns:yang:ietf-netconf-client
Registrant Contact: The NETCONF WG of the IETF.
XML: N/A, the requested URI is an XML namespace.

URI: urn:ietf:params:xml:ns:yang:ietf-netconf-server
Registrant Contact: The NETCONF WG of the IETF.
XML: N/A, the requested URI is an XML namespace.
```

6.2. The YANG Module Names Registry

This document registers two YANG modules in the YANG Module Names registry [RFC6020]. Following the format in [RFC6020], the the following registrations are requested:

```
name:         ietf-netconf-client
namespace:    urn:ietf:params:xml:ns:yang:ietf-netconf-client
prefix:       ncc
reference:    RFC XXXX

name:         ietf-netconf-server
namespace:    urn:ietf:params:xml:ns:yang:ietf-netconf-server
prefix:       ncs
reference:    RFC XXXX
```

7. References

7.1. Normative References

[I-D.ietf-netconf-keystore]
Watsen, K., "A YANG Data Model for a Keystore", draft-ietf-netconf-keystore-12 (work in progress), July 2019.

[I-D.ietf-netconf-ssh-client-server]
Watsen, K., Wu, G., and L. Xia, "YANG Groupings for SSH Clients and SSH Servers", draft-ietf-netconf-ssh-client-server-14 (work in progress), June 2019.

[I-D.ietf-netconf-tls-client-server]
Watsen, K., Wu, G., and L. Xia, "YANG Groupings for TLS Clients and TLS Servers", draft-ietf-netconf-tls-client-server-14 (work in progress), July 2019.

[I-D.kwatsen-netconf-tcp-client-server]
Watsen, K. and M. Scharf, "YANG Groupings for TCP Clients and TCP Servers", draft-kwatsen-netconf-tcp-client-server-02 (work in progress), April 2019.

[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>.

[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>.

[RFC7407] Bjorklund, M. and J. Schoenwaelder, "A YANG Data Model for SNMP Configuration", RFC 7407, DOI 10.17487/RFC7407, December 2014, <https://www.rfc-editor.org/info/rfc7407>.

[RFC7589] Badra, M., Luchuk, A., and J. Schoenwaelder, "Using the NETCONF Protocol over Transport Layer Security (TLS) with Mutual X.509 Authentication", RFC 7589, DOI 10.17487/RFC7589, June 2015, <https://www.rfc-editor.org/info/rfc7589>.

[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>.

[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>.

7.2. Informative References

[I-D.ietf-netconf-trust-anchors]
Watsen, K., "A YANG Data Model for a Truststore", draft-ietf-netconf-trust-anchors-05 (work in progress), June 2019.

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

[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>.

[RFC8071] Watsen, K., "NETCONF Call Home and RESTCONF Call Home", RFC 8071, DOI 10.17487/RFC8071, February 2017, <https://www.rfc-editor.org/info/rfc8071>.

[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>.

Appendix A. Expanded Tree Diagrams

A.1. Expanded Tree Diagram for 'ietf-netconf-client'

The following tree diagram [RFC8340] provides an overview of the data model for the "ietf-netconf-client" module.

This tree diagram shows all the nodes defined in this module, including those defined by "grouping" statements used by this module.

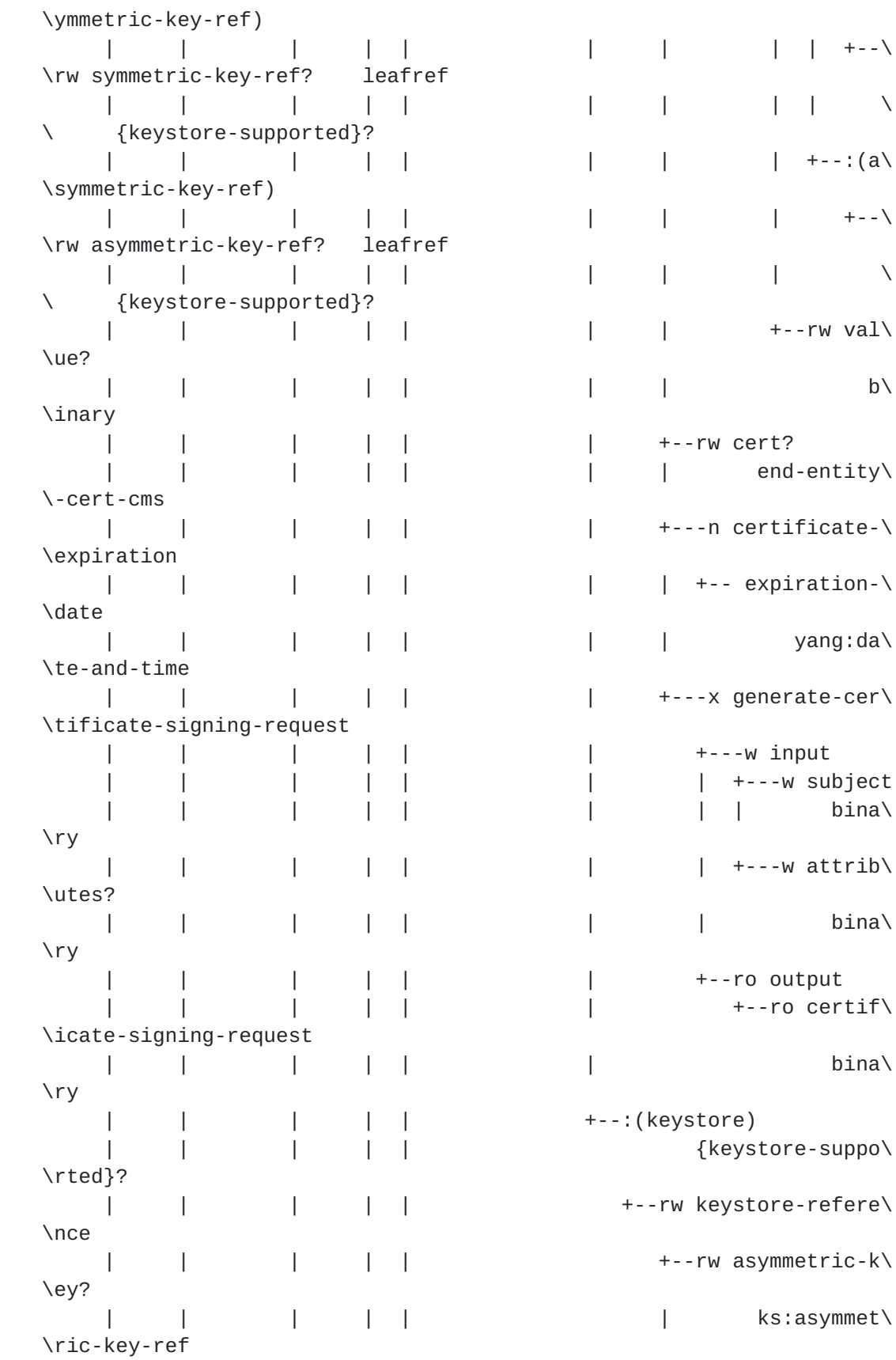
Please see Section 3.1 for a tree diagram that illustrates what the module looks like without all the "grouping" statements expanded.

```
========== NOTE: '\\' line wrapping per BCP XXX (RFC XXXX) ==========

module: ietf-netconf-client
  +--rw netconf-client
     +--rw initiate! {ssh-initiate or tls-initiate}?
     |  +--rw netconf-server* [name]
     |     +--rw name                  string
     |     +--rw endpoints
     |     |  +--rw endpoint* [name]
     |     |     +--rw name         string
     |     |     +--rw (transport)
     |     |        +--:(ssh) {ssh-initiate}?
     |     |        |  +--rw ssh
     |     |        |     +--rw tcp-client-parameters
     |     |        |     |  +--rw remote-address    inet:host
     |     |        |     |  +--rw remote-port?      inet:port-number
     |     |        |     |  +--rw local-address?    inet:ip-address
     |     |        |     |  |       {local-binding-supported}?
     |     |        |     |  +--rw local-port?       inet:port-number
     |     |        |     |  |       {local-binding-supported}?
     |     |        |     |  +--rw keepalives!
     |     |        |     |          {keepalives-supported}?
     |     |        |     |     +--rw idle-time         uint16
     |     |        |     |     +--rw max-probes        uint16
     |     |        |     |     +--rw probe-interval    uint16
     |     |        |     +--rw ssh-client-parameters
     |     |        |     |  +--rw client-identity
     |     |        |     |  |  +--rw username?            string
     |     |        |     |  |  +--rw (auth-type)
     |     |        |     |  |     +--:(password)
     |     |        |     |  |     |  +--rw password?      string
     |     |        |     |  |     +--:(public-key)
     |     |        |     |  |     |  +--rw public-key
     |     |        |     |  |     |     +--rw (local-or-keystore)
     |     |        |     |  |     |        +--:(local)
```

```
     |     |        |     |  |     |        |        {local-definiti\
\ons-supported}?
     |     |        |     |  |     |        |  +--rw local-definition
     |     |        |     |  |     |        |     +--rw algorithm
     |     |        |     |  |     |        |     |       asymmetric\
\-key-algorithm-t
     |     |        |     |  |     |        |     +--rw public-key-f\
\ormat?
     |     |        |     |  |     |        |     |       identityref
     |     |        |     |  |     |        |     +--rw public-key
     |     |        |     |  |     |        |     |       binary
     |     |        |     |  |     |        |     +--rw private-key-\
\format?
     |     |        |     |  |     |        |     |       identityref
     |     |        |     |  |     |        |     +--rw (private-key\
\-type)
     |     |        |     |  |     |        |        +--:(private-ke\
\y)
     |     |        |     |  |     |        |        |  +--rw privat\
\e-key?
     |     |        |     |  |     |        |        |          bina\
\ry
     |     |        |     |  |     |        |        +--:(hidden-pri\
\vate-key)
     |     |        |     |  |     |        |        |  +--rw hidden\
\-private-key?
     |     |        |     |  |     |        |        |          empty
     |     |        |     |  |     |        |        +--:(encrypted-\
\private-key)
     |     |        |     |  |     |        |           +--rw encryp\
\ted-private-key
     |     |        |     |  |     |        |              +--rw (ke\
\y-type)
     |     |        |     |  |     |        |              |  +--:(s\
\ymmetric-key-ref)
     |     |        |     |  |     |        |              |  |  +--\
\rw symmetric-key-ref?    leafref
     |     |        |     |  |     |        |              |  |     \
\     {keystore-supported}?
     |     |        |     |  |     |        |              |  +--:(a\
\symmetric-key-ref)
     |     |        |     |  |     |        |              |     +--\
\rw asymmetric-key-ref?   leafref
     |     |        |     |  |     |        |              |        \
\     {keystore-supported}?
     |     |        |     |  |     |        |              +--rw val\
\ue?
     |     |        |     |  |     |        |                    b\
```

```
   \inary
        |     |        |     |  |     |        +--:(keystore)
        |     |        |     |  |     |                 {keystore-suppo\
   \rted}?
        |     |        |     |  |     |           +--rw keystore-refere\
   \nce?
        |     |        |     |  |     |                   ks:asymmetric\
   \-key-ref
        |     |        |     |  |     +--:(certificate)
        |     |        |     |  |        +--rw certificate
        |     |        |     |  |                 {sshcmn:ssh-x509-certs\
   \}?
        |     |        |     |  |           +--rw (local-or-keystore)
        |     |        |     |  |              +--:(local)
        |     |        |     |  |              |        {local-definiti\
   \ons-supported}?
        |     |        |     |  |              |  +--rw local-definition
        |     |        |     |  |              |     +--rw algorithm
        |     |        |     |  |              |     |       asymmetric\
   \-key-algorithm-t
        |     |        |     |  |              |     +--rw public-key-f\
   \ormat?
        |     |        |     |  |              |     |       identityref
        |     |        |     |  |              |     +--rw public-key
        |     |        |     |  |              |     |       binary
        |     |        |     |  |              |     +--rw private-key-\
   \format?
        |     |        |     |  |              |     |       identityref
        |     |        |     |  |              |     +--rw (private-key\
   \-type)
        |     |        |     |  |              |     |  +--:(private-ke\
   \y)
        |     |        |     |  |              |     |  |  +--rw privat\
   \e-key?
        |     |        |     |  |              |     |  |          bina\
   \ry
        |     |        |     |  |              |     |  +--:(hidden-pri\
   \vate-key)
        |     |        |     |  |              |     |  |  +--rw hidden\
   \-private-key?
        |     |        |     |  |              |     |  |          empty
        |     |        |     |  |              |     |  +--:(encrypted-\
   \private-key)
        |     |        |     |  |              |     |     +--rw encryp\
   \ted-private-key
        |     |        |     |  |              |     |        +--rw (ke\
   \y-type)
        |     |        |     |  |              |     |        |  +--:(s\
```


\ymmetric-key-ref)
\rw symmetric-key-ref? leafref
\ {keystore-supported}?
\symmetric-key-ref)
\rw asymmetric-key-ref? leafref
\ {keystore-supported}?
\ue?
\inary
+--rw cert?
\-cert-cms
\expiration
\date
\te-and-time
\tificate-signing-request
+---w input
\ry
\utes?
\ry
+--ro output
\icate-signing-request
\ry
+--:(keystore)
{keystore-suppo\
\rted}?
\nce
\ey?
\ric-key-ref

```
       |     |        |     |  |                  +--rw certificate?\
   \      leafref
       |     |        |     |  +--rw server-authentication
       |     |        |     |  |  +--rw ssh-host-keys!
       |     |        |     |  |  |       {ts:ssh-host-keys}?
       |     |        |     |  |  |  +--rw (local-or-truststore)
       |     |        |     |  |  |     +--:(local)
       |     |        |     |  |  |     |        {local-definitions-su\
   \pported}?
       |     |        |     |  |  |     |  +--rw local-definition
       |     |        |     |  |  |     |     +--rw host-key*
       |     |        |     |  |  |     |             ct:ssh-host-key
       |     |        |     |  |  |     +--:(truststore)
       |     |        |     |  |  |              {truststore-supported\
   \,ssh-host-keys}?
       |     |        |     |  |  |        +--rw truststore-reference?
       |     |        |     |  |  |                ts:host-keys-ref
       |     |        |     |  |  +--rw ca-certs!
       |     |        |     |  |  |       {sshcmn:ssh-x509-certs,ts:x5\
   \09-certificates}?
       |     |        |     |  |  |  +--rw (local-or-truststore)
       |     |        |     |  |  |     +--:(local)
       |     |        |     |  |  |     |        {local-definitions-su\
   \pported}?
       |     |        |     |  |  |     |  +--rw local-definition
       |     |        |     |  |  |     |     +--rw cert*
       |     |        |     |  |  |     |     |       trust-anchor-cer\
   \t-cms
       |     |        |     |  |  |     |     +---n certificate-expira\
   \tion
       |     |        |     |  |  |     |        +-- expiration-date
       |     |        |     |  |  |     |                yang:date-and\
   \-time
       |     |        |     |  |  |     +--:(truststore)
       |     |        |     |  |  |              {truststore-supported\
   \,x509-certificates}?
       |     |        |     |  |  |        +--rw truststore-reference?
       |     |        |     |  |  |                ts:certificates-ref
       |     |        |     |  |  +--rw server-certs!
       |     |        |     |  |          {sshcmn:ssh-x509-certs,ts:x5\
   \09-certificates}?
       |     |        |     |  |     +--rw (local-or-truststore)
       |     |        |     |  |        +--:(local)
       |     |        |     |  |        |        {local-definitions-su\
   \pported}?
       |     |        |     |  |        |  +--rw local-definition
       |     |        |     |  |        |     +--rw cert*
       |     |        |     |  |        |     |       trust-anchor-cer\
```

```
   \t-cms
        |     |        |     |  |        |     +---n certificate-expira\
   \tion
        |     |        |     |  |        |        +-- expiration-date
        |     |        |     |  |        |                yang:date-and\
   \-time
        |     |        |     |  |        +--:(truststore)
        |     |        |     |  |                 {truststore-supported\
   \,x509-certificates}?
        |     |        |     |  |           +--rw truststore-reference?
        |     |        |     |  |                   ts:certificates-ref
        |     |        |     |  +--rw transport-params
        |     |        |     |  |       {ssh-client-transport-params-co\
   \nfig}?
        |     |        |     |  |  +--rw host-key
        |     |        |     |  |  |  +--rw host-key-alg*   identityref
        |     |        |     |  |  +--rw key-exchange
        |     |        |     |  |  |  +--rw key-exchange-alg*
        |     |        |     |  |  |          identityref
        |     |        |     |  |  +--rw encryption
        |     |        |     |  |  |  +--rw encryption-alg*
        |     |        |     |  |  |          identityref
        |     |        |     |  |  +--rw mac
        |     |        |     |  |     +--rw mac-alg*   identityref
        |     |        |     |  +--rw keepalives!
        |     |        |     |          {ssh-client-keepalives}?
        |     |        |     |     +--rw max-wait?       uint16
        |     |        |     |     +--rw max-attempts?   uint8
        |     |        |     +--rw netconf-client-parameters
        |     |        +--:(tls) {tls-initiate}?
        |     |           +--rw tls
        |     |              +--rw tcp-client-parameters
        |     |              |  +--rw remote-address    inet:host
        |     |              |  +--rw remote-port?      inet:port-number
        |     |              |  +--rw local-address?    inet:ip-address
        |     |              |  |       {local-binding-supported}?
        |     |              |  +--rw local-port?       inet:port-number
        |     |              |  |       {local-binding-supported}?
        |     |              |  +--rw keepalives!
        |     |              |          {keepalives-supported}?
        |     |              |     +--rw idle-time         uint16
        |     |              |     +--rw max-probes        uint16
        |     |              |     +--rw probe-interval    uint16
        |     |              +--rw tls-client-parameters
        |     |              |  +--rw client-identity
        |     |              |  |  +--rw (local-or-keystore)
        |     |              |  |     +--:(local)
        |     |              |  |     |        {local-definitions-suppo\
```

```
   \rted}?
        |     |              |  |     |  +--rw local-definition
        |     |              |  |     |     +--rw algorithm
        |     |              |  |     |     |       asymmetric-key-algo\
   \rithm-t
        |     |              |  |     |     +--rw public-key-format?
        |     |              |  |     |     |       identityref
        |     |              |  |     |     +--rw public-key
        |     |              |  |     |     |       binary
        |     |              |  |     |     +--rw private-key-format?
        |     |              |  |     |     |       identityref
        |     |              |  |     |     +--rw (private-key-type)
        |     |              |  |     |     |  +--:(private-key)
        |     |              |  |     |     |  |  +--rw private-key?
        |     |              |  |     |     |  |          binary
        |     |              |  |     |     |  +--:(hidden-private-key)
        |     |              |  |     |     |  |  +--rw hidden-private-\
   \key?
        |     |              |  |     |     |  |          empty
        |     |              |  |     |     |  +--:(encrypted-private-k\
   \ey)
        |     |              |  |     |     |     +--rw encrypted-priva\
   \te-key
        |     |              |  |     |     |        +--rw (key-type)
        |     |              |  |     |     |        |  +--:(symmetric-\
   \key-ref)
        |     |              |  |     |     |        |  |  +--rw symmet\
   \ric-key-ref?    leafref
        |     |              |  |     |     |        |  |          {key\
   \store-supported}?
        |     |              |  |     |     |        |  +--:(asymmetric\
   \-key-ref)
        |     |              |  |     |     |        |     +--rw asymme\
   \tric-key-ref?   leafref
        |     |              |  |     |     |        |             {key\
   \store-supported}?
        |     |              |  |     |     |        +--rw value?
        |     |              |  |     |     |                 binary
        |     |              |  |     |     +--rw cert?
        |     |              |  |     |     |       end-entity-cert-cms
        |     |              |  |     |     +---n certificate-expiration
        |     |              |  |     |     |  +-- expiration-date
        |     |              |  |     |     |          yang:date-and-ti\
   \me
        |     |              |  |     |     +---x generate-certificate-\
   \signing-request
        |     |              |  |     |        +---w input
        |     |              |  |     |        |  +---w subject
```

```
        |     |              |  |     |        |  |       binary
        |     |              |  |     |        |  +---w attributes?
        |     |              |  |     |        |          binary
        |     |              |  |     |        +--ro output
        |     |              |  |     |           +--ro certificate-sig\
   \ning-request
        |     |              |  |     |                  binary
        |     |              |  |     +--:(keystore)
        |     |              |  |              {keystore-supported}?
        |     |              |  |        +--rw keystore-reference
        |     |              |  |           +--rw asymmetric-key?
        |     |              |  |           |       ks:asymmetric-key-r\
   \ef
        |     |              |  |           +--rw certificate?      lea\
   \fref
        |     |              |  +--rw server-authentication
        |     |              |  |  +--rw ca-certs!
        |     |              |  |  |       {ts:x509-certificates}?
        |     |              |  |  |  +--rw (local-or-truststore)
        |     |              |  |  |     +--:(local)
        |     |              |  |  |     |        {local-definitions-su\
   \pported}?
        |     |              |  |  |     |  +--rw local-definition
        |     |              |  |  |     |     +--rw cert*
        |     |              |  |  |     |     |       trust-anchor-cer\
   \t-cms
        |     |              |  |  |     |     +---n certificate-expira\
   \tion
        |     |              |  |  |     |        +-- expiration-date
        |     |              |  |  |     |                 yang:date-and\
   \-time
        |     |              |  |  |     +--:(truststore)
        |     |              |  |  |              {truststore-supported\
   \,x509-certificates}?
        |     |              |  |  |        +--rw truststore-reference?
        |     |              |  |  |                 ts:certificates-ref
        |     |              |  |  +--rw server-certs!
        |     |              |  |           {ts:x509-certificates}?
        |     |              |  |     +--rw (local-or-truststore)
        |     |              |  |        +--:(local)
        |     |              |  |        |        {local-definitions-su\
   \pported}?
        |     |              |  |        |  +--rw local-definition
        |     |              |  |        |     +--rw cert*
        |     |              |  |        |     |       trust-anchor-cer\
   \t-cms
        |     |              |  |        |     +---n certificate-expira\
   \tion
```

```
      |     |              |  |        |        +-- expiration-date
      |     |              |  |        |                yang:date-and\
\-time
      |     |              |  |        +--:(truststore)
      |     |              |  |                 {truststore-supported\
\,x509-certificates}?
      |     |              |  |           +--rw truststore-reference?
      |     |              |  |                   ts:certificates-ref
      |     |              |  +--rw hello-params
      |     |              |  |       {tls-client-hello-params-config\
\}?
      |     |              |  |  +--rw tls-versions
      |     |              |  |  |  +--rw tls-version*   identityref
      |     |              |  |  +--rw cipher-suites
      |     |              |  |     +--rw cipher-suite*   identityref
      |     |              |  +--rw keepalives!
      |     |              |          {tls-client-keepalives}?
      |     |              |     +--rw max-wait?       uint16
      |     |              |     +--rw max-attempts?   uint8
      |     |             +--rw netconf-client-parameters
      |     +--rw connection-type
      |     |  +--rw (connection-type)
      |     |     +--:(persistent-connection)
      |     |     |  +--rw persistent!
      |     |     +--:(periodic-connection)
      |     |        +--rw periodic!
      |     |           +--rw period?         uint16
      |     |           +--rw anchor-time?    yang:date-and-time
      |     |           +--rw idle-timeout?   uint16
      |     +--rw reconnect-strategy
      |        +--rw start-with?     enumeration
      |        +--rw max-attempts?   uint8
      +--rw listen! {ssh-listen or tls-listen}?
         +--rw idle-timeout?   uint16
         +--rw endpoint* [name]
            +--rw name         string
            +--rw (transport)
               +--:(ssh) {ssh-listen}?
               |  +--rw ssh
               |     +--rw tcp-server-parameters
               |     |  +--rw local-address    inet:ip-address
               |     |  +--rw local-port?      inet:port-number
               |     |  +--rw keepalives! {keepalives-supported}?
               |     |     +--rw idle-time         uint16
               |     |     +--rw max-probes        uint16
               |     |     +--rw probe-interval    uint16
               |     +--rw ssh-client-parameters
               |     |  +--rw client-identity
```

```
                |     |  |  +--rw username?           string
                |     |  |  +--rw (auth-type)
                |     |  |     +--:(password)
                |     |  |     |  +--rw password?     string
                |     |  |     +--:(public-key)
                |     |  |     |  +--rw public-key
                |     |  |     |     +--rw (local-or-keystore)
                |     |  |     |        +--:(local)
                |     |  |     |        |        {local-definitions-su\
   \pported}?
                |     |  |     |        |  +--rw local-definition
                |     |  |     |        |     +--rw algorithm
                |     |  |     |        |     |       asymmetric-key-a\
   \lgorithm-t
                |     |  |     |        |     +--rw public-key-format?
                |     |  |     |        |     |       identityref
                |     |  |     |        |     +--rw public-key
                |     |  |     |        |     |       binary
                |     |  |     |        |     +--rw private-key-format?
                |     |  |     |        |     |       identityref
                |     |  |     |        |     +--rw (private-key-type)
                |     |  |     |        |        +--:(private-key)
                |     |  |     |        |        |  +--rw private-key?
                |     |  |     |        |        |          binary
                |     |  |     |        |        +--:(hidden-private-k\
   \ey)
                |     |  |     |        |        |  +--rw hidden-priva\
   \te-key?
                |     |  |     |        |        |          empty
                |     |  |     |        |        +--:(encrypted-privat\
   \e-key)
                |     |  |     |        |           +--rw encrypted-pr\
   \ivate-key
                |     |  |     |        |              +--rw (key-type)
                |     |  |     |        |              |  +--:(symmetr\
   \ic-key-ref)
                |     |  |     |        |              |  |  +--rw sym\
   \metric-key-ref?    leafref
                |     |  |     |        |              |  |          {\
   \keystore-supported}?
                |     |  |     |        |              |  +--:(asymmet\
   \ric-key-ref)
                |     |  |     |        |              |     +--rw asy\
   \mmetric-key-ref?   leafref
                |     |  |     |        |              |             {\
   \keystore-supported}?
                |     |  |     |        |              +--rw value?
                |     |  |     |        |                        binary
```

```
                |     |  |     |        +--:(keystore)
                |     |  |     |                 {keystore-supported}?
                |     |  |     |           +--rw keystore-reference?
                |     |  |     |                   ks:asymmetric-key-r\
   \ef
                |     |  |     +--:(certificate)
                |     |  |        +--rw certificate
                |     |  |                {sshcmn:ssh-x509-certs}?
                |     |  |           +--rw (local-or-keystore)
                |     |  |              +--:(local)
                |     |  |              |        {local-definitions-su\
   \pported}?
                |     |  |              |  +--rw local-definition
                |     |  |              |     +--rw algorithm
                |     |  |              |     |       asymmetric-key-a\
   \lgorithm-t
                |     |  |              |     +--rw public-key-format?
                |     |  |              |     |       identityref
                |     |  |              |     +--rw public-key
                |     |  |              |     |       binary
                |     |  |              |     +--rw private-key-format?
                |     |  |              |     |       identityref
                |     |  |              |     +--rw (private-key-type)
                |     |  |              |     |  +--:(private-key)
                |     |  |              |     |  |  +--rw private-key?
                |     |  |              |     |  |          binary
                |     |  |              |     |  +--:(hidden-private-k\
   \ey)
                |     |  |              |     |  |  +--rw hidden-priva\
   \te-key?
                |     |  |              |     |  |          empty
                |     |  |              |     |  +--:(encrypted-privat\
   \e-key)
                |     |  |              |     |     +--rw encrypted-pr\
   \ivate-key
                |     |  |              |     |        +--rw (key-type)
                |     |  |              |     |        |  +--:(symmetr\
   \ic-key-ref)
                |     |  |              |     |        |  |  +--rw sym\
   \metric-key-ref?    leafref
                |     |  |              |     |        |  |          {\
   \keystore-supported}?
                |     |  |              |     |        |  +--:(asymmet\
   \ric-key-ref)
                |     |  |              |     |        |     +--rw asy\
   \mmetric-key-ref?   leafref
                |     |  |              |     |        |             {\
   \keystore-supported}?
```

```
                |     |  |             |     |       +--rw value?
                |     |  |             |     |               binary
                |     |  |             |     +--rw cert?
                |     |  |             |     |       end-entity-cert-\
   \cms
                |     |  |             |     +---n certificate-expira\
   \tion
                |     |  |             |     |  +-- expiration-date
                |     |  |             |     |          yang:date-and\
   \-time
                |     |  |             |     +---x generate-certifica\
   \te-signing-request
                |     |  |             |        +---w input
                |     |  |             |        |  +---w subject
                |     |  |             |        |  |       binary
                |     |  |             |        |  +---w attributes?
                |     |  |             |        |          binary
                |     |  |             |        +--ro output
                |     |  |             |           +--ro certificate-\
   \signing-request
                |     |  |             |                   binary
                |     |  |             +--:(keystore)
                |     |  |                      {keystore-supported}?
                |     |  |                +--rw keystore-reference
                |     |  |                   +--rw asymmetric-key?
                |     |  |                   |       ks:asymmetric-ke\
   \y-ref
                |     |  |                   +--rw certificate?      \
   \leafref
                |     |  +--rw server-authentication
                |     |  |  +--rw ssh-host-keys! {ts:ssh-host-keys}?
                |     |  |  |  +--rw (local-or-truststore)
                |     |  |  |     +--:(local)
                |     |  |  |     |        {local-definitions-supporte\
   \d}?
                |     |  |  |     |  +--rw local-definition
                |     |  |  |     |     +--rw host-key*
                |     |  |  |     |             ct:ssh-host-key
                |     |  |  |     +--:(truststore)
                |     |  |  |              {truststore-supported,ssh-h\
   \ost-keys}?
                |     |  |  |        +--rw truststore-reference?
                |     |  |  |                 ts:host-keys-ref
                |     |  |  +--rw ca-certs!
                |     |  |  |       {sshcmn:ssh-x509-certs,ts:x509-cer\
   \tificates}?
                |     |  |  |  +--rw (local-or-truststore)
                |     |  |  |     +--:(local)
```

```
                |     |  |  |     |        {local-definitions-supporte\
   \d}?
                |     |  |  |     |  +--rw local-definition
                |     |  |  |     |     +--rw cert*
                |     |  |  |     |     |       trust-anchor-cert-cms
                |     |  |  |     |     +---n certificate-expiration
                |     |  |  |     |        +-- expiration-date
                |     |  |  |     |                yang:date-and-time
                |     |  |  |     +--:(truststore)
                |     |  |  |              {truststore-supported,x509-\
   \certificates}?
                |     |  |  |        +--rw truststore-reference?
                |     |  |  |                 ts:certificates-ref
                |     |  |  +--rw server-certs!
                |     |  |           {sshcmn:ssh-x509-certs,ts:x509-cer\
   \tificates}?
                |     |  |     +--rw (local-or-truststore)
                |     |  |        +--:(local)
                |     |  |        |        {local-definitions-supporte\
   \d}?
                |     |  |        |  +--rw local-definition
                |     |  |        |     +--rw cert*
                |     |  |        |     |       trust-anchor-cert-cms
                |     |  |        |     +---n certificate-expiration
                |     |  |        |        +-- expiration-date
                |     |  |        |                yang:date-and-time
                |     |  |        +--:(truststore)
                |     |  |                 {truststore-supported,x509-\
   \certificates}?
                |     |  |           +--rw truststore-reference?
                |     |  |                   ts:certificates-ref
                |     |  +--rw transport-params
                |     |  |       {ssh-client-transport-params-config}?
                |     |  |  +--rw host-key
                |     |  |  |  +--rw host-key-alg*   identityref
                |     |  |  +--rw key-exchange
                |     |  |  |  +--rw key-exchange-alg*   identityref
                |     |  |  +--rw encryption
                |     |  |  |  +--rw encryption-alg*   identityref
                |     |  |  +--rw mac
                |     |  |     +--rw mac-alg*   identityref
                |     |  +--rw keepalives! {ssh-client-keepalives}?
                |     |     +--rw max-wait?       uint16
                |     |     +--rw max-attempts?   uint8
                |     +--rw netconf-client-parameters
                +--:(tls) {tls-listen}?
                   +--rw tls
                      +--rw tcp-server-parameters
```

```
                        |  +--rw local-address    inet:ip-address
                        |  +--rw local-port?      inet:port-number
                        |  +--rw keepalives! {keepalives-supported}?
                        |     +--rw idle-time         uint16
                        |     +--rw max-probes        uint16
                        |     +--rw probe-interval    uint16
                        +--rw tls-client-parameters
                        |  +--rw client-identity
                        |  |  +--rw (local-or-keystore)
                        |  |     +--:(local)
                        |  |     |        {local-definitions-supported}?
                        |  |     |  +--rw local-definition
                        |  |     |     +--rw algorithm
                        |  |     |     |       asymmetric-key-algorithm-t
                        |  |     |     +--rw public-key-format?
                        |  |     |     |       identityref
                        |  |     |     +--rw public-key
                        |  |     |     |       binary
                        |  |     |     +--rw private-key-format?
                        |  |     |     |       identityref
                        |  |     |     +--rw (private-key-type)
                        |  |     |     |  +--:(private-key)
                        |  |     |     |  |  +--rw private-key?
                        |  |     |     |  |          binary
                        |  |     |     |  +--:(hidden-private-key)
                        |  |     |     |  |  +--rw hidden-private-key?
                        |  |     |     |  |          empty
                        |  |     |     |  +--:(encrypted-private-key)
                        |  |     |     |     +--rw encrypted-private-key
                        |  |     |     |        +--rw (key-type)
                        |  |     |     |        |  +--:(symmetric-key-re\
   \f)
                        |  |     |     |        |  |  +--rw symmetric-ke\
   \y-ref?    leafref
                        |  |     |     |        |  |          {keystore-\
   \supported}?
                        |  |     |     |        |  +--:(asymmetric-key-r\
   \ef)
                        |  |     |     |        |     +--rw asymmetric-k\
   \ey-ref?   leafref
                        |  |     |     |        |             {keystore-\
   \supported}?
                        |  |     |     |        +--rw value?
                        |  |     |     |                 binary
                        |  |     |     +--rw cert?
                        |  |     |     |       end-entity-cert-cms
                        |  |     |     +---n certificate-expiration
                        |  |     |     |  +-- expiration-date
```

```
                        |  |     |     |          yang:date-and-time
                        |  |     |     +---x generate-certificate-signin\
   \g-request
                        |  |     |        +---w input
                        |  |     |        |  +---w subject       binary
                        |  |     |        |  +---w attributes?   binary
                        |  |     |        +--ro output
                        |  |     |           +--ro certificate-signing-r\
   \equest
                        |  |     |                  binary
                        |  |     +--:(keystore) {keystore-supported}?
                        |  |        +--rw keystore-reference
                        |  |           +--rw asymmetric-key?
                        |  |           |       ks:asymmetric-key-ref
                        |  |           +--rw certificate?      leafref
                        |  +--rw server-authentication
                        |  |  +--rw ca-certs! {ts:x509-certificates}?
                        |  |  |  +--rw (local-or-truststore)
                        |  |  |     +--:(local)
                        |  |  |     |        {local-definitions-supporte\
   \d}?
                        |  |  |     |  +--rw local-definition
                        |  |  |     |     +--rw cert*
                        |  |  |     |     |       trust-anchor-cert-cms
                        |  |  |     |     +---n certificate-expiration
                        |  |  |     |        +-- expiration-date
                        |  |  |     |                yang:date-and-time
                        |  |  |     +--:(truststore)
                        |  |  |              {truststore-supported,x509-\
   \certificates}?
                        |  |  |        +--rw truststore-reference?
                        |  |  |                 ts:certificates-ref
                        |  |  +--rw server-certs! {ts:x509-certificates}?
                        |  |     +--rw (local-or-truststore)
                        |  |        +--:(local)
                        |  |        |        {local-definitions-supporte\
   \d}?
                        |  |        |  +--rw local-definition
                        |  |        |     +--rw cert*
                        |  |        |     |       trust-anchor-cert-cms
                        |  |        |     +---n certificate-expiration
                        |  |        |        +-- expiration-date
                        |  |        |                yang:date-and-time
                        |  |        +--:(truststore)
                        |  |                 {truststore-supported,x509-\
   \certificates}?
                        |  |           +--rw truststore-reference?
                        |  |                   ts:certificates-ref
```

```
                        |  +--rw hello-params
                        |  |       {tls-client-hello-params-config}?
                        |  |  +--rw tls-versions
                        |  |  |  +--rw tls-version*   identityref
                        |  |  +--rw cipher-suites
                        |  |     +--rw cipher-suite*   identityref
                        |  +--rw keepalives! {tls-client-keepalives}?
                        |     +--rw max-wait?       uint16
                        |     +--rw max-attempts?   uint8
                        +--rw netconf-client-parameters
```

A.2. Expanded Tree Diagram for 'ietf-netconf-server'

The following tree diagram [RFC8340] provides an overview of the data model for the "ietf-netconf-server" module.

This tree diagram shows all the nodes defined in this module, including those defined by "grouping" statements used by this module.

Please see Section 4.1 for a tree diagram that illustrates what the module looks like without all the "grouping" statements expanded.

========== NOTE: '\\' line wrapping per BCP XXX (RFC XXXX) ==========

```
module: ietf-netconf-server
  +--rw netconf-server
     +--rw listen! {ssh-listen or tls-listen}?
     |  +--rw idle-timeout?   uint16
     |  +--rw endpoint* [name]
     |     +--rw name         string
     |     +--rw (transport)
     |        +--:(ssh) {ssh-listen}?
     |        |  +--rw ssh
     |        |     +--rw tcp-server-parameters
     |        |     |  +--rw local-address    inet:ip-address
     |        |     |  +--rw local-port?      inet:port-number
     |        |     |  +--rw keepalives! {keepalives-supported}?
     |        |     |     +--rw idle-time         uint16
     |        |     |     +--rw max-probes        uint16
     |        |     |     +--rw probe-interval    uint16
     |        |     +--rw ssh-server-parameters
     |        |     |  +--rw server-identity
     |        |     |  |  +--rw host-key* [name]
     |        |     |  |     +--rw name                 string
     |        |     |  |     +--rw (host-key-type)
     |        |     |  |        +--:(public-key)
     |        |     |  |        |  +--rw public-key
     |        |     |  |        |     +--rw (local-or-keystore)
```

```
     |        |     |  |        |        +--:(local)
     |        |     |  |        |        |        {local-definitions\
\-supported}?
     |        |     |  |        |        |  +--rw local-definition
     |        |     |  |        |        |     +--rw algorithm
     |        |     |  |        |        |     |       asymmetric-ke\
\y-algorithm-t
     |        |     |  |        |        |     +--rw public-key-form\
\at?
     |        |     |  |        |        |     |       identityref
     |        |     |  |        |        |     +--rw public-key
     |        |     |  |        |        |     |       binary
     |        |     |  |        |        |     +--rw private-key-for\
\mat?
     |        |     |  |        |        |     |       identityref
     |        |     |  |        |        |     +--rw (private-key-ty\
\pe)
     |        |     |  |        |        |        +--:(private-key)
     |        |     |  |        |        |        |  +--rw private-k\
\ey?
     |        |     |  |        |        |        |          binary
     |        |     |  |        |        |        +--:(hidden-privat\
\e-key)
     |        |     |  |        |        |        |  +--rw hidden-pr\
\ivate-key?
     |        |     |  |        |        |        |          empty
     |        |     |  |        |        |        +--:(encrypted-pri\
\vate-key)
     |        |     |  |        |        |           +--rw encrypted\
\-private-key
     |        |     |  |        |        |              +--rw (key-t\
\ype)
     |        |     |  |        |        |              |  +--:(symm\
\etric-key-ref)
     |        |     |  |        |        |              |  |  +--rw \
\symmetric-key-ref?    leafref
     |        |     |  |        |        |              |  |       \
\  {keystore-supported}?
     |        |     |  |        |        |              |  +--:(asym\
\metric-key-ref)
     |        |     |  |        |        |              |     +--rw \
\asymmetric-key-ref?   leafref
     |        |     |  |        |        |              |           \
\  {keystore-supported}?
     |        |     |  |        |        |              +--rw value?
     |        |     |  |        |        |                       bina\
\ry
     |        |     |  |        |        +--:(keystore)
```

```
     |        |     |  |        |                {keystore-supporte\
   \d}?
     |        |     |  |        |          +--rw keystore-reference?
     |        |     |  |        |                 ks:asymmetric-ke\
   \y-ref
     |        |     |  |        +--:(certificate)
     |        |     |  |           +--rw certificate
     |        |     |  |                   {sshcmn:ssh-x509-certs}?
     |        |     |  |              +--rw (local-or-keystore)
     |        |     |  |                 +--:(local)
     |        |     |  |                 |        {local-definitions\
   \-supported}?
     |        |     |  |                 |  +--rw local-definition
     |        |     |  |                 |     +--rw algorithm
     |        |     |  |                 |     |       asymmetric-ke\
   \y-algorithm-t
     |        |     |  |                 |     +--rw public-key-form\
   \at?
     |        |     |  |                 |     |       identityref
     |        |     |  |                 |     +--rw public-key
     |        |     |  |                 |     |       binary
     |        |     |  |                 |     +--rw private-key-for\
   \mat?
     |        |     |  |                 |     |       identityref
     |        |     |  |                 |     +--rw (private-key-ty\
   \pe)
     |        |     |  |                 |     |  +--:(private-key)
     |        |     |  |                 |     |  |  +--rw private-k\
   \ey?
     |        |     |  |                 |     |  |          binary
     |        |     |  |                 |     |  +--:(hidden-privat\
   \e-key)
     |        |     |  |                 |     |  |  +--rw hidden-pr\
   \ivate-key?
     |        |     |  |                 |     |  |          empty
     |        |     |  |                 |     |  +--:(encrypted-pri\
   \vate-key)
     |        |     |  |                 |     |     +--rw encrypted\
   \-private-key
     |        |     |  |                 |     |        +--rw (key-t\
   \ype)
     |        |     |  |                 |     |        |  +--:(symm\
   \etric-key-ref)
     |        |     |  |                 |     |        |  |  +--rw \
   \symmetric-key-ref?    leafref
     |        |     |  |                 |     |        |  |       \
   \  {keystore-supported}?
     |        |     |  |                 |     |        |  +--:(asym\
```

```
   \metric-key-ref)
        |        |     |  |                 |     |        |     +--rw \
   \asymmetric-key-ref?   leafref
        |        |     |  |                 |     |        |           \
   \  {keystore-supported}?
        |        |     |  |                 |     |        +--rw value?
        |        |     |  |                 |     |                bina\
   \ry
        |        |     |  |                 |     +--rw cert?
        |        |     |  |                 |     |       end-entity-ce\
   \rt-cms
        |        |     |  |                 |     +---n certificate-exp\
   \iration
        |        |     |  |                 |     |  +-- expiration-date
        |        |     |  |                 |     |          yang:date-\
   \and-time
        |        |     |  |                 |     +---x generate-certif\
   \icate-signing-request
        |        |     |  |                 |        +---w input
        |        |     |  |                 |        |  +---w subject
        |        |     |  |                 |        |  |       binary
        |        |     |  |                 |        |  +---w attribute\
   \s?
        |        |     |  |                 |        |          binary
        |        |     |  |                 |        +--ro output
        |        |     |  |                 |           +--ro certifica\
   \te-signing-request
        |        |     |  |                 |                   binary
        |        |     |  |                 +--:(keystore)
        |        |     |  |                          {keystore-supporte\
   \d}?
        |        |     |  |                    +--rw keystore-reference
        |        |     |  |                       +--rw asymmetric-key?
        |        |     |  |                       |       ks:asymmetric\
   \-key-ref
        |        |     |  |                       +--rw certificate?   \
   \   leafref
        |        |     |  +--rw client-authentication
        |        |     |  |  +--rw supported-authentication-methods
        |        |     |  |  |  +--rw publickey?   empty
        |        |     |  |  |  +--rw passsword?   empty
        |        |     |  |  |  +--rw hostbased?   empty
        |        |     |  |  |  +--rw none?        empty
        |        |     |  |  |  +--rw other*       string
        |        |     |  |  +--rw (local-or-external)
        |        |     |  |     +--:(local)
        |        |     |  |     |        {local-client-auth-supported}?
        |        |     |  |     |  +--rw users
```

```
       |        |     |  |     |     +--rw user* [name]
       |        |     |  |     |        +--rw name            string
       |        |     |  |     |        +--rw password?
       |        |     |  |     |        |       ianach:crypt-hash
       |        |     |  |     |        +--rw host-keys!
       |        |     |  |     |        |       {ts:ssh-host-keys}?
       |        |     |  |     |        |  +--rw (local-or-truststore)
       |        |     |  |     |        |     +--:(local)
       |        |     |  |     |        |     |        {local-definiti\
  \ons-supported}?
       |        |     |  |     |        |     |  +--rw local-definition
       |        |     |  |     |        |     |     +--rw host-key*
       |        |     |  |     |        |     |             ct:ssh-hos\
  \t-key
       |        |     |  |     |        |     +--:(truststore)
       |        |     |  |     |        |              {truststore-sup\
  \ported,ssh-host-keys}?
       |        |     |  |     |        |        +--rw truststore-refe\
  \rence?
       |        |     |  |     |        |                ts:host-keys-\
  \ref
       |        |     |  |     |        +--rw ca-certs!
       |        |     |  |     |        |       {sshcmn:ssh-x509-certs\
  \,ts:x509-certificates}?
       |        |     |  |     |        |  +--rw (local-or-truststore)
       |        |     |  |     |        |     +--:(local)
       |        |     |  |     |        |     |        {local-definiti\
  \ons-supported}?
       |        |     |  |     |        |     |  +--rw local-definition
       |        |     |  |     |        |     |     +--rw cert*
       |        |     |  |     |        |     |     |       trust-anch\
  \or-cert-cms
       |        |     |  |     |        |     |     +---n certificate-\
  \expiration
       |        |     |  |     |        |     |        +-- expiration-\
  \date
       |        |     |  |     |        |     |                yang:da\
  \te-and-time
       |        |     |  |     |        |     +--:(truststore)
       |        |     |  |     |        |              {truststore-sup\
  \ported,x509-certificates}?
       |        |     |  |     |        |        +--rw truststore-refe\
  \rence?
       |        |     |  |     |        |                ts:certificat\
  \es-ref
       |        |     |  |     |        +--rw client-certs!
       |        |     |  |     |                {sshcmn:ssh-x509-certs\
  \,ts:x509-certificates}?
```

```
       |        |     |  |     |          +--rw (local-or-truststore)
       |        |     |  |     |             +--:(local)
       |        |     |  |     |             |       {local-definiti\
\ons-supported}?
       |        |     |  |     |             |  +--rw local-definition
       |        |     |  |     |             |     +--rw cert*
       |        |     |  |     |             |     |       trust-anch\
\or-cert-cms
       |        |     |  |     |             |     +---n certificate-\
\expiration
       |        |     |  |     |             |       +-- expiration-\
\date
       |        |     |  |     |             |               yang:da\
\te-and-time
       |        |     |  |     |             +--:(truststore)
       |        |     |  |     |                       {truststore-sup\
\ported,x509-certificates}?
       |        |     |  |     |                  +--rw truststore-refe\
\rence?
       |        |     |  |     |                        ts:certificat\
\es-ref
       |        |     |  |     +--:(external)
       |        |     |  |             {external-client-auth-supporte\
\d}?
       |        |     |  |       +--rw client-auth-defined-elsewhere?
       |        |     |  |               empty
       |        |     |  +--rw transport-params
       |        |     |  |      {ssh-server-transport-params-config}?
       |        |     |  |  +--rw host-key
       |        |     |  |  |  +--rw host-key-alg*   identityref
       |        |     |  |  +--rw key-exchange
       |        |     |  |  |  +--rw key-exchange-alg*   identityref
       |        |     |  |  +--rw encryption
       |        |     |  |  |  +--rw encryption-alg*   identityref
       |        |     |  |  +--rw mac
       |        |     |  |     +--rw mac-alg*   identityref
       |        |     |  +--rw keepalives! {ssh-server-keepalives}?
       |        |     |     +--rw max-wait?       uint16
       |        |     |     +--rw max-attempts?   uint8
       |        |     +--rw netconf-server-parameters
       |        |        +--rw client-identification
       |        |           +--rw cert-maps
       |        |              +--rw cert-to-name* [id]
       |        |                 +--rw id             uint32
       |        |                 +--rw fingerprint
       |        |                 |       x509c2n:tls-fingerprint
       |        |                 +--rw map-type       identityref
       |        |                 +--rw name           string
```

```
        |        +--:(tls) {tls-listen}?
        |           +--rw tls
        |              +--rw tcp-server-parameters
        |              |  +--rw local-address    inet:ip-address
        |              |  +--rw local-port?      inet:port-number
        |              |  +--rw keepalives! {keepalives-supported}?
        |              |     +--rw idle-time         uint16
        |              |     +--rw max-probes        uint16
        |              |     +--rw probe-interval    uint16
        |              +--rw tls-server-parameters
        |              |  +--rw server-identity
        |              |  |  +--rw (local-or-keystore)
        |              |  |     +--:(local)
        |              |  |     |        {local-definitions-supported}?
        |              |  |     |  +--rw local-definition
        |              |  |     |     +--rw algorithm
        |              |  |     |     |       asymmetric-key-algorithm-t
        |              |  |     |     +--rw public-key-format?
        |              |  |     |     |       identityref
        |              |  |     |     +--rw public-key
        |              |  |     |     |       binary
        |              |  |     |     +--rw private-key-format?
        |              |  |     |     |       identityref
        |              |  |     |     +--rw (private-key-type)
        |              |  |     |     |  +--:(private-key)
        |              |  |     |     |  |  +--rw private-key?
        |              |  |     |     |  |          binary
        |              |  |     |     |  +--:(hidden-private-key)
        |              |  |     |     |  |  +--rw hidden-private-key?
        |              |  |     |     |  |          empty
        |              |  |     |     |  +--:(encrypted-private-key)
        |              |  |     |     |     +--rw encrypted-private-key
        |              |  |     |     |        +--rw (key-type)
        |              |  |     |     |        |  +--:(symmetric-key-re\
\f)
        |              |  |     |     |        |  |  +--rw symmetric-ke\
\y-ref?    leafref
        |              |  |     |     |        |  |          {keystore-\
\supported}?
        |              |  |     |     |        |  +--:(asymmetric-key-r\
\ef)
        |              |  |     |     |        |     +--rw asymmetric-k\
\ey-ref?   leafref
        |              |  |     |     |        |             {keystore-\
\supported}?
        |              |  |     |     |        +--rw value?
        |              |  |     |     |                 binary
        |              |  |     |     +--rw cert?
```


| | | | | end-entity-cert-cms
| | | | +---n certificate-expiration
| | | | | +-- expiration-date
| | | | | yang:date-and-time
| | | | +---x generate-certificate-signin\
\g-request
| | | | +---w input
| | | | | +---w subject binary
| | | | | +---w attributes? binary
| | | | +--ro output
| | | | +--ro certificate-signing-r\
\equest
| | | | binary
| | | +--:(keystore) {keystore-supported}?
| | | +--rw keystore-reference
| | | +--rw asymmetric-key?
| | | | ks:asymmetric-key-ref
| | | +--rw certificate? leafref
| | +--rw client-authentication!
| | | +--rw (required-or-optional)
| | | | +--:(required)
| | | | | +--rw required?
| | | | | empty
| | | | +--:(optional)
| | | | +--rw optional?
| | | | empty
| | | +--rw (local-or-external)
| | | +--:(local)
| | | | {local-client-auth-supported}?
| | | | +--rw ca-certs!
| | | | | {ts:x509-certificates}?
| | | | | +--rw (local-or-truststore)
| | | | | +--:(local)
| | | | | | {local-definitions-su\
\pported}?
| | | | | | +--rw local-definition
| | | | | | +--rw cert*
| | | | | | | trust-anchor-cer\
\t-cms
| | | | | | +---n certificate-expira\
\tion
| | | | | | +-- expiration-date
| | | | | | yang:date-and\
\-time
| | | | | +--:(truststore)
| | | | | {truststore-supported\
\,x509-certificates}?
| | | | | +--rw truststore-reference?

```
       |              |  |     |  |               ts:certificates-ref
       |              |  |     |  +--rw client-certs!
       |              |  |     |           {ts:x509-certificates}?
       |              |  |     |     +--rw (local-or-truststore)
       |              |  |     |        +--:(local)
       |              |  |     |        |        {local-definitions-su\
  \pported}?
       |              |  |     |        |  +--rw local-definition
       |              |  |     |        |     +--rw cert*
       |              |  |     |        |     |       trust-anchor-cer\
  \t-cms
       |              |  |     |        |     +---n certificate-expira\
  \tion
       |              |  |     |        |        +-- expiration-date
       |              |  |     |        |                yang:date-and\
  \-time
       |              |  |     |        +--:(truststore)
       |              |  |     |                 {truststore-supported\
  \,x509-certificates}?
       |              |  |     |           +--rw truststore-reference?
       |              |  |     |                    ts:certificates-ref
       |              |  |     +--:(external)
       |              |  |              {external-client-auth-supporte\
  \d}?
       |              |  |        +--rw client-auth-defined-elsewhere?
       |              |  |                 empty
       |              |  +--rw hello-params
       |              |  |       {tls-server-hello-params-config}?
       |              |  |  +--rw tls-versions
       |              |  |  |  +--rw tls-version*   identityref
       |              |  |  +--rw cipher-suites
       |              |  |     +--rw cipher-suite*   identityref
       |              |  +--rw keepalives! {tls-server-keepalives}?
       |              |     +--rw max-wait?       uint16
       |              |     +--rw max-attempts?   uint8
       |              +--rw netconf-server-parameters
       |                 +--rw client-identification
       |                    +--rw cert-maps
       |                       +--rw cert-to-name* [id]
       |                          +--rw id             uint32
       |                          +--rw fingerprint
       |                          |       x509c2n:tls-fingerprint
       |                          +--rw map-type       identityref
       |                          +--rw name           string
       +--rw call-home! {ssh-call-home or tls-call-home}?
          +--rw netconf-client* [name]
             +--rw name                  string
             +--rw endpoints
```

```
             |  +--rw endpoint* [name]
             |     +--rw name         string
             |     +--rw (transport)
             |        +--:(ssh) {ssh-call-home}?
             |        |  +--rw ssh
             |        |     +--rw tcp-client-parameters
             |        |     |  +--rw remote-address    inet:host
             |        |     |  +--rw remote-port?      inet:port-number
             |        |     |  +--rw local-address?    inet:ip-address
             |        |     |  |       {local-binding-supported}?
             |        |     |  +--rw local-port?       inet:port-number
             |        |     |  |       {local-binding-supported}?
             |        |     |  +--rw keepalives!
             |        |     |          {keepalives-supported}?
             |        |     |     +--rw idle-time         uint16
             |        |     |     +--rw max-probes        uint16
             |        |     |     +--rw probe-interval    uint16
             |        |     +--rw ssh-server-parameters
             |        |     |  +--rw server-identity
             |        |     |  |  +--rw host-key* [name]
             |        |     |  |     +--rw name                 string
             |        |     |  |     +--rw (host-key-type)
             |        |     |  |        +--:(public-key)
             |        |     |  |        |  +--rw public-key
             |        |     |  |        |     +--rw (local-or-keystore)
             |        |     |  |        |        +--:(local)
             |        |     |  |        |        |        {local-defin\
\itions-supported}?
             |        |     |  |        |        |  +--rw local-defini\
\tion
             |        |     |  |        |        |     +--rw algorithm
             |        |     |  |        |        |     |       asymmet\
\ric-key-algorithm-t
             |        |     |  |        |        |     +--rw public-ke\
\y-format?
             |        |     |  |        |        |     |       identit\
\yref
             |        |     |  |        |        |     +--rw public-key
             |        |     |  |        |        |     |       binary
             |        |     |  |        |        |     +--rw private-k\
\ey-format?
             |        |     |  |        |        |     |       identit\
\yref
             |        |     |  |        |        |     +--rw (private-\
\key-type)
             |        |     |  |        |        |        +--:(private\
\-key)
             |        |     |  |        |        |        |  +--rw pri\
```

```
   \vate-key?
              |        |     |  |        |        |        |          b\
   \inary
              |        |     |  |        |        |        +--:(hidden-\
   \private-key)
              |        |     |  |        |        |        |  +--rw hid\
   \den-private-key?
              |        |     |  |        |        |        |          e\
   \mpty
              |        |     |  |        |        |        +--:(encrypt\
   \ed-private-key)
              |        |     |  |        |        |           +--rw enc\
   \rypted-private-key
              |        |     |  |        |        |              +--rw \
   \(key-type)
              |        |     |  |        |        |              |  +--\
   \:(symmetric-key-ref)
              |        |     |  |        |        |              |  |  \
   \+--rw symmetric-key-ref?    leafref
              |        |     |  |        |        |              |  |  \
   \        {keystore-supported}?
              |        |     |  |        |        |              |  +--\
   \:(asymmetric-key-ref)
              |        |     |  |        |        |              |     \
   \+--rw asymmetric-key-ref?   leafref
              |        |     |  |        |        |              |     \
   \        {keystore-supported}?
              |        |     |  |        |        |              +--rw \
   \value?
              |        |     |  |        |        |                    \
   \  binary
              |        |     |  |        |        +--:(keystore)
              |        |     |  |        |                 {keystore-su\
   \pported}?
              |        |     |  |        |           +--rw keystore-ref\
   \erence?
              |        |     |  |        |                   ks:asymmet\
   \ric-key-ref
              |        |     |  |       +--:(certificate)
              |        |     |  |          +--rw certificate
              |        |     |  |                  {sshcmn:ssh-x509-ce\
   \rts}?
              |        |     |  |              +--rw (local-or-keystore)
              |        |     |  |                 +--:(local)
              |        |     |  |                 |        {local-defin\
   \itions-supported}?
              |        |     |  |                 |  +--rw local-defini\
   \tion
```

```
           |        |     |  |                |     +--rw algorithm
           |        |     |  |                |     |       asymmet\
  \ric-key-algorithm-t
           |        |     |  |                |     +--rw public-ke\
  \y-format?
           |        |     |  |                |     |       identit\
  \yref
           |        |     |  |                |     +--rw public-key
           |        |     |  |                |     |       binary
           |        |     |  |                |     +--rw private-k\
  \ey-format?
           |        |     |  |                |     |       identit\
  \yref
           |        |     |  |                |     +--rw (private-\
  \key-type)
           |        |     |  |                |     |  +--:(private\
  \-key)
           |        |     |  |                |     |  |  +--rw pri\
  \vate-key?
           |        |     |  |                |     |  |          b\
  \inary
           |        |     |  |                |     |  +--:(hidden-\
  \private-key)
           |        |     |  |                |     |  |  +--rw hid\
  \den-private-key?
           |        |     |  |                |     |  |          e\
  \mpty
           |        |     |  |                |     |  +--:(encrypt\
  \ed-private-key)
           |        |     |  |                |     |     +--rw enc\
  \rypted-private-key
           |        |     |  |                |     |        +--rw \
  \(key-type)
           |        |     |  |                |     |        |  +--\
  \:(symmetric-key-ref)
           |        |     |  |                |     |        |  |  \
  \+--rw symmetric-key-ref?    leafref
           |        |     |  |                |     |        |  |  \
  \        {keystore-supported}?
           |        |     |  |                |     |        |  +--\
  \:(asymmetric-key-ref)
           |        |     |  |                |     |        |     \
  \+--rw asymmetric-key-ref?   leafref
           |        |     |  |                |     |        |     \
  \        {keystore-supported}?
           |        |     |  |                |     |        +--rw \
  \value?
           |        |     |  |                |     |             \
```

```
   \  binary
              |        |     |  |                 |     +--rw cert?
              |        |     |  |                 |     |       end-ent\
   \ity-cert-cms
              |        |     |  |                 |     +---n certifica\
   \te-expiration
              |        |     |  |                 |     |  +-- expirati\
   \on-date
              |        |     |  |                 |     |          yang\
   \:date-and-time
              |        |     |  |                 |     +---x generate-\
   \certificate-signing-request
              |        |     |  |                 |        +---w input
              |        |     |  |                 |        |  +---w sub\
   \ject
              |        |     |  |                 |        |  |       b\
   \inary
              |        |     |  |                 |        |  +---w att\
   \ributes?
              |        |     |  |                 |        |          b\
   \inary
              |        |     |  |                 |        +--ro output
              |        |     |  |                 |           +--ro cer\
   \tificate-signing-request
              |        |     |  |                 |                  b\
   \inary
              |        |     |  |                +--:(keystore)
              |        |     |  |                         {keystore-su\
   \pported}?
              |        |     |  |                   +--rw keystore-ref\
   \erence
              |        |     |  |                      +--rw asymmetri\
   \c-key?
              |        |     |  |                      |       ks:asym\
   \metric-key-ref
              |        |     |  |                      +--rw certifica\
   \te?      leafref
              |        |     |  +--rw client-authentication
              |        |     |  |  +--rw supported-authentication-metho\
   \ds
              |        |     |  |  |  +--rw publickey?   empty
              |        |     |  |  |  +--rw passsword?   empty
              |        |     |  |  |  +--rw hostbased?   empty
              |        |     |  |  |  +--rw none?        empty
              |        |     |  |  |  +--rw other*       string
              |        |     |  |  +--rw (local-or-external)
              |        |     |  |     +--:(local)
              |        |     |  |     |        {local-client-auth-suppo\
```

```
   \rted}?
              |        |     |  |     |  +--rw users
              |        |     |  |     |     +--rw user* [name]
              |        |     |  |     |        +--rw name
              |        |     |  |     |        |       string
              |        |     |  |     |        +--rw password?
              |        |     |  |     |        |       ianach:crypt-hash
              |        |     |  |     |        +--rw host-keys!
              |        |     |  |     |        |       {ts:ssh-host-key\
   \s}?
              |        |     |  |     |        |  +--rw (local-or-trust\
   \store)
              |        |     |  |     |        |     +--:(local)
              |        |     |  |     |        |     |        {local-de\
   \finitions-supported}?
              |        |     |  |     |        |     |  +--rw local-def\
   \inition
              |        |     |  |     |        |     |     +--rw host-k\
   \ey*
              |        |     |  |     |        |     |             ct:s\
   \sh-host-key
              |        |     |  |     |        |     +--:(truststore)
              |        |     |  |     |        |             {truststo\
   \re-supported,ssh-host-keys}?
              |        |     |  |     |        |        +--rw truststor\
   \e-reference?
              |        |     |  |     |        |               ts:host\
   \-keys-ref
              |        |     |  |     |        +--rw ca-certs!
              |        |     |  |     |        |       {sshcmn:ssh-x509\
   \-certs,ts:x509-certificates}?
              |        |     |  |     |        |  +--rw (local-or-trust\
   \store)
              |        |     |  |     |        |     +--:(local)
              |        |     |  |     |        |     |        {local-de\
   \finitions-supported}?
              |        |     |  |     |        |     |  +--rw local-def\
   \inition
              |        |     |  |     |        |     |     +--rw cert*
              |        |     |  |     |        |     |     |       trus\
   \t-anchor-cert-cms
              |        |     |  |     |        |     |     +---n certif\
   \icate-expiration
              |        |     |  |     |        |     |        +-- expir\
   \ation-date
              |        |     |  |     |        |     |               y\
   \ang:date-and-time
              |        |     |  |     |        |     +--:(truststore)
```

```
           |        |     |  |     |        |              {truststo\
  \re-supported,x509-certificates}?
           |        |     |  |     |        |        +--rw truststor\
  \e-reference?
           |        |     |  |     |        |                ts:cert\
  \ificates-ref
           |        |     |  |     |        +--rw client-certs!
           |        |     |  |     |                 {sshcmn:ssh-x509\
  \-certs,ts:x509-certificates}?
           |        |     |  |     |           +--rw (local-or-trust\
  \store)
           |        |     |  |     |              +--:(local)
           |        |     |  |     |              |        {local-de\
  \finitions-supported}?
           |        |     |  |     |              |  +--rw local-def\
  \inition
           |        |     |  |     |              |     +--rw cert*
           |        |     |  |     |              |     |       trus\
  \t-anchor-cert-cms
           |        |     |  |     |              |     +---n certif\
  \icate-expiration
           |        |     |  |     |              |        +-- expir\
  \ation-date
           |        |     |  |     |              |                y\
  \ang:date-and-time
           |        |     |  |     |              +--:(truststore)
           |        |     |  |     |                       {truststo\
  \re-supported,x509-certificates}?
           |        |     |  |     |                 +--rw truststor\
  \e-reference?
           |        |     |  |     |                       ts:cert\
  \ificates-ref
           |        |     |  |     +--:(external)
           |        |     |  |              {external-client-auth-su\
  \pported}?
           |        |     |  |        +--rw client-auth-defined-else\
  \where?
           |        |     |  |                empty
           |        |     |  +--rw transport-params
           |        |     |  |       {ssh-server-transport-params-co\
  \nfig}?
           |        |     |  |  +--rw host-key
           |        |     |  |  |  +--rw host-key-alg*   identityref
           |        |     |  |  +--rw key-exchange
           |        |     |  |  |  +--rw key-exchange-alg*
           |        |     |  |  |         identityref
           |        |     |  |  +--rw encryption
           |        |     |  |  |  +--rw encryption-alg*
```

```
               |        |     |  |  |          identityref
               |        |     |  |  +--rw mac
               |        |     |  |     +--rw mac-alg*   identityref
               |        |     |  +--rw keepalives!
               |        |     |          {ssh-server-keepalives}?
               |        |     |     +--rw max-wait?       uint16
               |        |     |     +--rw max-attempts?   uint8
               |        |     +--rw netconf-server-parameters
               |        |        +--rw client-identification
               |        |           +--rw cert-maps
               |        |              +--rw cert-to-name* [id]
               |        |                 +--rw id             uint32
               |        |                 +--rw fingerprint
               |        |                 |       x509c2n:tls-fingerprint
               |        |                 +--rw map-type
               |        |                 |       identityref
               |        |                 +--rw name           string
               |        +--:(tls) {tls-call-home}?
               |           +--rw tls
               |              +--rw tcp-client-parameters
               |              |  +--rw remote-address    inet:host
               |              |  +--rw remote-port?      inet:port-number
               |              |  +--rw local-address?    inet:ip-address
               |              |  |       {local-binding-supported}?
               |              |  +--rw local-port?       inet:port-number
               |              |  |       {local-binding-supported}?
               |              |  +--rw keepalives!
               |              |          {keepalives-supported}?
               |              |     +--rw idle-time         uint16
               |              |     +--rw max-probes        uint16
               |              |     +--rw probe-interval    uint16
               |              +--rw tls-server-parameters
               |              |  +--rw server-identity
               |              |  |  +--rw (local-or-keystore)
               |              |  |     +--:(local)
               |              |  |     |        {local-definitions-suppo\
   \rted}?
               |              |  |     |  +--rw local-definition
               |              |  |     |     +--rw algorithm
               |              |  |     |     |       asymmetric-key-algo\
   \rithm-t
               |              |  |     |     +--rw public-key-format?
               |              |  |     |     |       identityref
               |              |  |     |     +--rw public-key
               |              |  |     |     |       binary
               |              |  |     |     +--rw private-key-format?
               |              |  |     |     |       identityref
               |              |  |     |     +--rw (private-key-type)
```

```
            |              |  |     |     |  +--:(private-key)
            |              |  |     |     |  |  +--rw private-key?
            |              |  |     |     |  |          binary
            |              |  |     |     |  +--:(hidden-private-key)
            |              |  |     |     |  |  +--rw hidden-private-\
   \key?
            |              |  |     |     |  |          empty
            |              |  |     |     |  +--:(encrypted-private-k\
   \ey)
            |              |  |     |     |     +--rw encrypted-priva\
   \te-key
            |              |  |     |     |        +--rw (key-type)
            |              |  |     |     |        |  +--:(symmetric-\
   \key-ref)
            |              |  |     |     |        |  |  +--rw symmet\
   \ric-key-ref?    leafref
            |              |  |     |     |        |  |          {key\
   \store-supported}?
            |              |  |     |     |        |  +--:(asymmetric\
   \-key-ref)
            |              |  |     |     |        |     +--rw asymme\
   \tric-key-ref?   leafref
            |              |  |     |     |        |             {key\
   \store-supported}?
            |              |  |     |     |        +--rw value?
            |              |  |     |     |                 binary
            |              |  |     |     +--rw cert?
            |              |  |     |     |       end-entity-cert-cms
            |              |  |     |     +---n certificate-expiration
            |              |  |     |     |  +-- expiration-date
            |              |  |     |     |          yang:date-and-ti\
   \me
            |              |  |     |     +---x generate-certificate-\
   \signing-request
            |              |  |     |        +---w input
            |              |  |     |        |  +---w subject
            |              |  |     |        |  |       binary
            |              |  |     |        |  +---w attributes?
            |              |  |     |        |          binary
            |              |  |     |        +--ro output
            |              |  |     |           +--ro certificate-sig\
   \ning-request
            |              |  |     |                   binary
            |              |  |     +--:(keystore)
            |              |  |              {keystore-supported}?
            |              |  |        +--rw keystore-reference
            |              |  |           +--rw asymmetric-key?
            |              |  |           |       ks:asymmetric-key-r\
```

```
   \ef
              |              |  |          +--rw certificate?      lea\
   \fref
              |              |  +--rw client-authentication!
              |              |  |  +--rw (required-or-optional)
              |              |  |  |  +--:(required)
              |              |  |  |  |  +--rw required?
              |              |  |  |  |         empty
              |              |  |  |  +--:(optional)
              |              |  |  |     +--rw optional?
              |              |  |  |              empty
              |              |  |  +--rw (local-or-external)
              |              |  |  |  +--:(local)
              |              |  |  |  |        {local-client-auth-suppo\
   \rted}?
              |              |  |  |  |  +--rw ca-certs!
              |              |  |  |  |  |       {ts:x509-certificates}?
              |              |  |  |  |  |  +--rw (local-or-truststore)
              |              |  |  |  |  |     +--:(local)
              |              |  |  |  |  |     |        {local-definiti\
   \ons-supported}?
              |              |  |  |  |  |     |  +--rw local-definition
              |              |  |  |  |  |     |     +--rw cert*
              |              |  |  |  |  |     |     |       trust-anch\
   \or-cert-cms
              |              |  |  |  |  |     |     +---n certificate-\
   \expiration
              |              |  |  |  |  |     |        +-- expiration-\
   \date
              |              |  |  |  |  |     |                yang:da\
   \te-and-time
              |              |  |  |  |  |     +--:(truststore)
              |              |  |  |  |  |              {truststore-sup\
   \ported,x509-certificates}?
              |              |  |  |  |  |        +--rw truststore-refe\
   \rence?
              |              |  |  |  |  |                ts:certificat\
   \es-ref
              |              |  |  |  |  +--rw client-certs!
              |              |  |  |  |           {ts:x509-certificates}?
              |              |  |  |  |     +--rw (local-or-truststore)
              |              |  |  |  |        +--:(local)
              |              |  |  |  |        |        {local-definiti\
   \ons-supported}?
              |              |  |  |  |        |  +--rw local-definition
              |              |  |  |  |        |     +--rw cert*
              |              |  |  |  |        |     |       trust-anch\
   \or-cert-cms
```

```
            |              |  |  |  |        |     +---n certificate-\
   \expiration
            |              |  |  |  |        |        +-- expiration-\
   \date
            |              |  |  |  |        |               yang:da\
   \te-and-time
            |              |  |  |  |        +--:(truststore)
            |              |  |  |  |                 {truststore-sup\
   \ported,x509-certificates}?
            |              |  |  |  |           +--rw truststore-refe\
   \rence?
            |              |  |  |  |                   ts:certificat\
   \es-ref
            |              |  |  |  +--:(external)
            |              |  |  |           {external-client-auth-su\
   \pported}?
            |              |  |  |     +--rw client-auth-defined-else\
   \where?
            |              |  |  |            empty
            |              |  |  +--rw cert-maps
            |              |  |     +--rw cert-to-name* [id]
            |              |  |        +--rw id             uint32
            |              |  |        +--rw fingerprint
            |              |  |        |       x509c2n:tls-fingerprint
            |              |  |        +--rw map-type
            |              |  |        |       identityref
            |              |  |        +--rw name           string
            |              |  +--rw hello-params
            |              |  |       {tls-server-hello-params-config\
   \}?
            |              |  |  +--rw tls-versions
            |              |  |  |  +--rw tls-version*   identityref
            |              |  |  +--rw cipher-suites
            |              |  |     +--rw cipher-suite*   identityref
            |              |  +--rw keepalives!
            |              |          {tls-server-keepalives}?
            |              |     +--rw max-wait?       uint16
            |              |     +--rw max-attempts?   uint8
            |              +--rw netconf-server-parameters
            |                 +--rw client-identification
            |                    +--rw cert-maps
            |                       +--rw cert-to-name* [id]
            |                          +--rw id             uint32
            |                          +--rw fingerprint
            |                          |       x509c2n:tls-fingerprint
            |                          +--rw map-type
            |                          |       identityref
            |                          +--rw name           string
```

```
              +--rw connection-type
              |  +--rw (connection-type)
              |     +--:(persistent-connection)
              |     |  +--rw persistent!
              |     +--:(periodic-connection)
              |        +--rw periodic!
              |           +--rw period?         uint16
              |           +--rw anchor-time?    yang:date-and-time
              |           +--rw idle-timeout?   uint16
              +--rw reconnect-strategy
                 +--rw start-with?     enumeration
                 +--rw max-attempts?   uint8
```

Appendix B. Change Log

B.1. 00 to 01

- Renamed "keychain" to "keystore".

B.2. 01 to 02

- Added to ietf-netconf-client ability to connected to a cluster of endpoints, including a reconnection-strategy.

- Added to ietf-netconf-client the ability to configure connection-type and also keep-alive strategy.

- Updated both modules to accommodate new groupings in the ssh/tls drafts.

B.3. 02 to 03

- Refined use of tls-client-grouping to add a must statement indicating that the TLS client must specify a client-certificate.

- Changed 'netconf-client' to be a grouping (not a container).

B.4. 03 to 04

- Added RFC 8174 to Requirements Language Section.

- Replaced refine statement in ietf-netconf-client to add a mandatory true.

- Added refine statement in ietf-netconf-server to add a must statement.

- Now there are containers and groupings, for both the client and server models.

B.5. 04 to 05

- Now tree diagrams reference ietf-netmod-yang-tree-diagrams
- Updated examples to inline key and certificates (no longer a leafref to keystore)

B.6. 05 to 06

- Fixed change log missing section issue.
- Updated examples to match latest updates to the crypto-types, trust-anchors, and keystore drafts.
- Reduced line length of the YANG modules to fit within 69 columns.

B.7. 06 to 07

- Removed "idle-timeout" from "persistent" connection config.
- Added "random-selection" for reconnection-strategy's "starts-with" enum.
- Replaced "connection-type" choice default (persistent) with "mandatory true".
- Reduced the periodic-connection's "idle-timeout" from 5 to 2 minutes.
- Replaced reconnect-timeout with period/anchor-time combo.

B.8. 07 to 08

- Modified examples to be compatible with new crypto-types algs

B.9. 08 to 09

- Corrected use of "mandatory true" for "address" leafs.
- Updated examples to reflect update to groupings defined in the keystore draft.
- Updated to use groupings defined in new TCP and HTTP drafts.

- Updated copyright date, boilerplate template, affiliation, and folding algorithm.

B.10. 09 to 10

- Reformatted YANG modules.

B.11. 10 to 11

- Adjusted for the top-level "demux container" added to groupings imported from other modules.

- Added "must" expressions to ensure that keepalives are not configured for "periodic" connections.

- Updated the boilerplate text in module-level "description" statement to match copyeditor convention.

- Moved "expanded" tree diagrams to the Appendix.

B.12. 11 to 12

- Removed the "Design Considerations" section.

- Removed the 'must' statement limiting keepalives in periodic connections.

- Updated models and examples to reflect removal of the "demux" containers in the imported models.

- Updated the "periodic-connnection" description statements to be more like the RESTCONF draft, especially where it described dropping the underlying TCP connection.

- Updated text to better reference where certain examples come from (e.g., which Section in which draft).

- In the server model, commented out the "must 'pinned-ca-certs or pinned-client-certs'" statement to reflect change made in the TLS draft whereby the trust anchors MAY be defined externally.

- Replaced the 'listen', 'initiate', and 'call-home' features with boolean expressions.

B.13. 12 to 13

- Updated to reflect changes in trust-anchors drafts (e.g., s/trust-anchors/truststore/g + s/pinned.//)

B.14. 13 to 14

- Adjusting from change in TLS client model (removing the top-level 'certificate' container), by swapping refining-in a 'mandatory true' statement with a 'must' statement outside the 'uses' statement.

- Updated examples to reflect ietf-crypto-types change (e.g., identities --> enumerations)

B.15. 14 to 15

- Refactored both the client and server modules similar to how the ietf-restconf-server module was refactored in -13 of that draft, and the ietf-restconf-client grouping.

Acknowledgements

The authors would like to thank for following for lively discussions on list and in the halls (ordered by last name): Andy Bierman, Martin Bjorklund, Benoit Claise, Ramkumar Dhanapal, Mehmet Ersue, Balazs Kovacs, David Lamparter, Alan Luchuk, Ladislav Lhotka, Radek Krejci, Tom Petch, Juergen Schoenwaelder, Phil Shafer, Sean Turner, and Bert Wijnen.

Author's Address

Kent Watsen
Watsen Networks

EMail: kent+ietf@watsen.net