NETCONF Working Group
Internet-Draft
Intended status: Standards Track
Expires: November 21, 2020

K. Watsen
Watsen Networks

May 20, 2020

RESTCONF Client and Server Models
draft-ietf-netconf-restconf-client-server-19

Abstract

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

Editorial Note (To be removed by RFC Editor)

This draft contains 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.

Artwork in this document contains shorthand references to drafts in progress. Please apply the following replacements (note: not all may be present):

- "AAAA" --> the assigned RFC value for draft-ietf-netconf-crypto-types
- "BBBB" --> the assigned RFC value for draft-ietf-netconf-trust-anchors
- "CCCC" --> the assigned RFC value for draft-ietf-netconf-keystore
- "DDDD" --> the assigned RFC value for draft-ietf-netconf-tcp-client-server
- "EEEE" --> the assigned RFC value for draft-ietf-netconf-ssh-client-server
- "FFFF" --> the assigned RFC value for draft-ietf-netconf-tls-client-server
- "GGGG" --> the assigned RFC value for draft-ietf-netconf-http-client-server

- "HHHH" --> the assigned RFC value for draft-ietf-netconf-netconf-client-server

- "IIII" --> the assigned RFC value for this draft

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

- "2020-05-20" --> the publication date of this draft

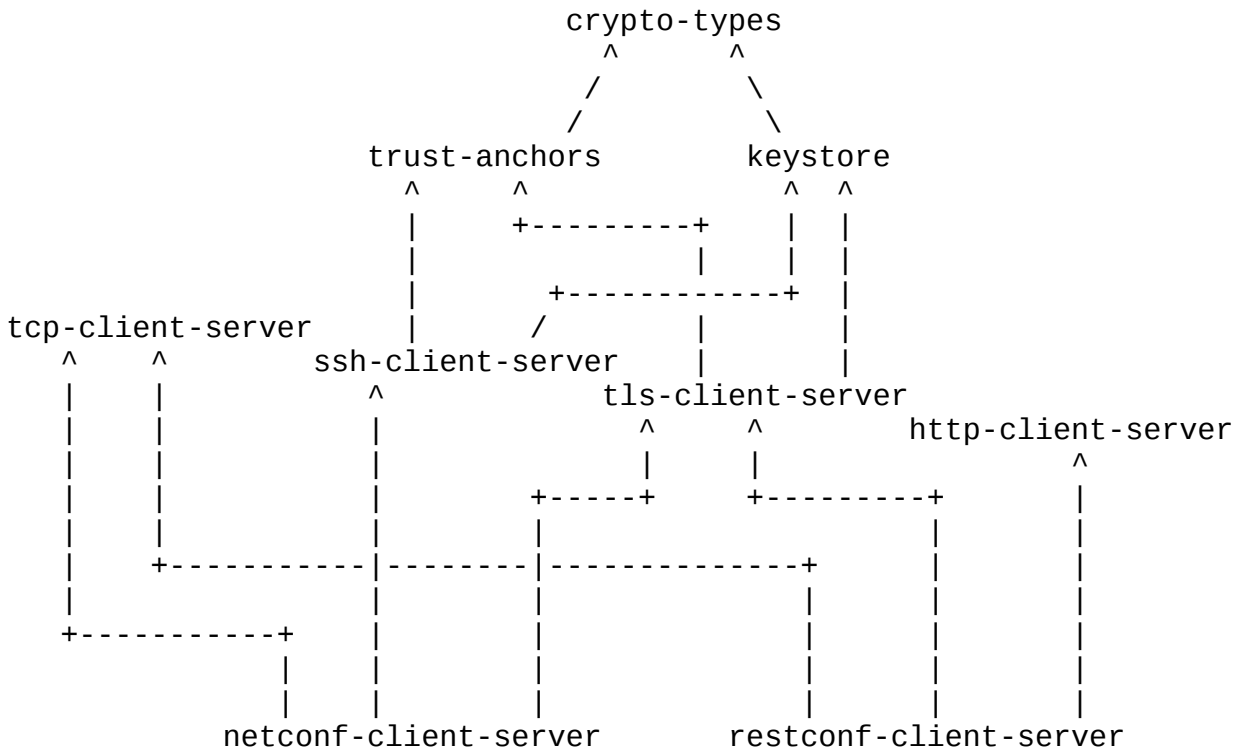
The following Appendix section is to be removed prior to publication:

- Appendix B. Change Log

Note to Reviewers (To be removed by RFC Editor)

This document presents a YANG module or modules that is/are part of a collection of drafts that work together to produce the ultimate goal of the NETCONF WG: to define configuration modules for NETCONF client and servers, and RESTCONF client and servers.

The relationship between the various drafts in the collection is presented in the below diagram.



Full draft names and link to drafts:

- draft-ietf-netconf-crypto-types (html [1])
- draft-ietf-netconf-trust-anchors (html [2])
- draft-ietf-netconf-keystore (html [3])
- draft-ietf-netconf-tcp-client-server (html [4])
- draft-ietf-netconf-ssh-client-server (html [5])
- draft-ietf-netconf-tls-client-server (html [6])
- draft-ietf-netconf-http-client-server (html [7])
- draft-ietf-netconf-netconf-client-server (html [8])
- draft-ietf-netconf-restconf-client-server (html [9])

Status of This Memo

Copyright Notice

Table of Contents

```
            +-- https
               +-- tcp-client-parameters
               |  +---u tcpc:tcp-client-grouping
               +-- tls-client-parameters
               |  +---u tlsc:tls-client-grouping
               +-- http-client-parameters
               |  +---u httpc:http-client-grouping
               +-- restconf-client-parameters
     grouping restconf-client-listen-stack-grouping
       +-- (transport)
          +--:(http) {http-listen}?
          |  +-- http
          |     +-- tcp-server-parameters
          |     |  +---u tcps:tcp-server-grouping
          |     +-- http-client-parameters
          |     |  +---u httpc:http-client-grouping
          |     +-- restconf-client-parameters
          +--:(https) {https-listen}?
             +-- https
                +-- tcp-server-parameters
                |  +---u tcps:tcp-server-grouping
                +-- tls-client-parameters
                |  +---u tlsc:tls-client-grouping
                +-- http-client-parameters
                |  +---u httpc:http-client-grouping
                +-- restconf-client-parameters
     grouping restconf-client-app-grouping
       +-- initiate! {https-initiate}?
       |  +-- restconf-server* [name]
       |     +-- name?                 string
       |     +-- endpoints
       |     |  +-- endpoint* [name]
       |     |     +-- name?                                          string
       |     |     +---u restconf-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! {http-listen or https-listen}?
          +-- idle-timeout?   uint16
```

```
          +-- endpoint* [name]
             +-- name?                                    string
             +---u restconf-client-listen-stack-grouping
```

2.2. Example Usage

The following example illustrates configuring a RESTCONF client to initiate connections, as well as listening for call-home connections.

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) ===========

<restconf-client
  xmlns="urn:ietf:params:xml:ns:yang:ietf-restconf-client"
  xmlns:ct="urn:ietf:params:xml:ns:yang:ietf-crypto-types">

  <!-- RESTCONF servers to initiate connections to -->
  <initiate>
    <restconf-server>
      <name>corp-fw1</name>
      <endpoints>
        <endpoint>
          <name>corp-fw1.example.com</name>
          <https>
            <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>
            <tls-client-parameters>
              <client-identity>
                <certificate>
                  <local-definition>
                    <public-key-format>ct:subject-public-key-info-fo\
rmat</public-key-format>
                    <public-key>base64encodedvalue==</public-key>
                    <private-key-format>ct:rsa-private-key-format</p\
rivate-key-format>
                    <private-key>base64encodedvalue==</private-key>
                    <cert>base64encodedvalue==</cert>
                  </local-definition>
                </certificate>
```

```
                </client-identity>
                <server-authentication>
                  <ca-certs>
                    <truststore-reference>trusted-server-ca-certs</tru\
   ststore-reference>
                  </ca-certs>
                  <ee-certs>
                    <truststore-reference>trusted-server-ee-certs</tru\
   ststore-reference>
                  </ee-certs>
                </server-authentication>
                <keepalives>
                  <test-peer-aliveness>
                    <max-wait>30</max-wait>
                    <max-attempts>3</max-attempts>
                  </test-peer-aliveness>
                </keepalives>
              </tls-client-parameters>
              <http-client-parameters>
                <client-identity>
                  <basic>
                    <user-id>bob</user-id>
                    <password>secret</password>
                  </basic>
                </client-identity>
              </http-client-parameters>
            </https>
          </endpoint>
          <endpoint>
            <name>corp-fw2.example.com</name>
            <https>
              <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>
                  <certificate>
                    <local-definition>
                      <public-key-format>ct:subject-public-key-info-fo\
   rmat</public-key-format>
                      <public-key>base64encodedvalue==</public-key>
                      <private-key-format>ct:rsa-private-key-format</p\
   rivate-key-format>
```

```
                      <private-key>base64encodedvalue==</private-key>
                      <cert>base64encodedvalue==</cert>
                    </local-definition>
                  </certificate>
                </client-identity>
                <server-authentication>
                  <ca-certs>
                    <truststore-reference>trusted-server-ca-certs</tru\
   ststore-reference>
                  </ca-certs>
                  <ee-certs>
                    <truststore-reference>trusted-server-ee-certs</tru\
   ststore-reference>
                  </ee-certs>
                </server-authentication>
                <keepalives>
                  <test-peer-aliveness>
                    <max-wait>30</max-wait>
                    <max-attempts>3</max-attempts>
                  </test-peer-aliveness>
                </keepalives>
              </tls-client-parameters>
              <http-client-parameters>
                <client-identity>
                  <basic>
                    <user-id>bob</user-id>
                    <password>secret</password>
                  </basic>
                </client-identity>
              </http-client-parameters>
            </https>
          </endpoint>
        </endpoints>
        <connection-type>
          <persistent/>
        </connection-type>
      </restconf-server>
    </initiate>

    <!-- endpoints to listen for RESTCONF Call Home connections on -->
    <listen>
      <endpoint>
        <name>Intranet-facing listener</name>
        <https>
          <tcp-server-parameters>
            <local-address>11.22.33.44</local-address>
          </tcp-server-parameters>
          <tls-client-parameters>
```

```
            <client-identity>
              <certificate>
                <local-definition>
                  <public-key-format>ct:subject-public-key-info-format\
</public-key-format>
                  <public-key>base64encodedvalue==</public-key>
                  <private-key-format>ct:rsa-private-key-format</priva\
te-key-format>
                  <private-key>base64encodedvalue==</private-key>
                  <cert>base64encodedvalue==</cert>
                </local-definition>
              </certificate>
            </client-identity>
            <server-authentication>
              <ca-certs>
                <truststore-reference>trusted-server-ca-certs</trustst\
ore-reference>
              </ca-certs>
              <ee-certs>
                <truststore-reference>trusted-server-ee-certs</trustst\
ore-reference>
              </ee-certs>
            </server-authentication>
            <keepalives>
              <peer-allowed-to-send/>
            </keepalives>
          </tls-client-parameters>
          <http-client-parameters>
            <client-identity>
              <basic>
                <user-id>bob</user-id>
                <password>secret</password>
              </basic>
            </client-identity>
          </http-client-parameters>
        </https>
      </endpoint>
    </listen>
  </restconf-client>
```

2.3. YANG Module

This YANG module has normative references to [RFC6991], [RFC8040], and [RFC8071], [I-D.kwatsen-netconf-tcp-client-server], [I-D.ietf-netconf-tls-client-server], and [I-D.kwatsen-netconf-http-client-server].

<CODE BEGINS> file "ietf-restconf-client@2020-05-20.yang"

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

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

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

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

     import ietf-tls-client {
       prefix tlsc;
       reference
         "RFC FFFF: YANG Groupings for TLS Clients and TLS Servers";
     }

     import ietf-http-client {
       prefix httpc;
       reference
         "RFC GGGG: YANG Groupings for HTTP Clients and HTTP 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 RESTCONF clients.
```

```
        Copyright (c) 2020 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 IIII
        (https://www.rfc-editor.org/info/rfcIIII); 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 2020-05-20 {
        description
          "Initial version";
        reference
          "RFC IIII: RESTCONF Client and Server Models";
      }

      // Features

      feature https-initiate {
        description
          "The 'https-initiate' feature indicates that the RESTCONF
           client supports initiating HTTPS connections to RESTCONF
           servers. This feature exists as HTTPS might not be a
           mandatory to implement transport in the future.";
        reference
          "RFC 8040: RESTCONF Protocol";
      }

      feature http-listen {
        description
          "The 'https-listen' feature indicates that the RESTCONF client
           supports opening a port to listen for incoming RESTCONF
           server call-home connections.  This feature exists as not
           all RESTCONF clients may support RESTCONF call home.";
        reference
          "RFC 8071: NETCONF Call Home and RESTCONF Call Home";
```

```
  }

  feature https-listen {
    description
      "The 'https-listen' feature indicates that the RESTCONF client
       supports opening a port to listen for incoming RESTCONF
       server call-home connections.  This feature exists as not
       all RESTCONF clients may support RESTCONF call home.";
    reference
      "RFC 8071: NETCONF Call Home and RESTCONF Call Home";
  }

  // Groupings

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

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

  grouping restconf-client-initiate-stack-grouping {
    description
      "A reusable grouping for configuring a RESTCONF client
       'initiate' protocol stack for a single connection.";

    choice transport {
      mandatory true;
      description
        "Selects between available transports. This is a
         'choice' statement so as to support additional
         transport options to be augmented in.";
      case https {
        if-feature "https-initiate";
        container https {
          description
            "Specifies HTTPS-specific 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 "443";
                description
```

```
                    "The RESTCONF client will attempt to
                     connect to the IANA-assigned well-known
                     port value for 'https' (443) 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 http-client-parameters {
              description
                "A wrapper around the HTTP client parameters
                 to avoid name collisions.";
              uses httpc:http-client-grouping;
            }
            container restconf-client-parameters {
              description
                "A wrapper around the HTTP client parameters
                 to avoid name collisions.";
              uses rcc:restconf-client-grouping;
            }
          }
        }
      }
    } // restconf-client-initiate-stack-grouping

    grouping restconf-client-listen-stack-grouping {
      description
        "A reusable grouping for configuring a RESTCONF client
         'listen' protocol stack for a single connection.  The
         'listen' stack supports call home connections, as
         described in RFC 8071";
      reference
        "RFC 8071: NETCONF Call Home and RESTCONF Call Home";
      choice transport {
        mandatory true;
        description
          "Selects between available transports. This is a
           'choice' statement so as to support additional
```

```
           transport options to be augmented in.";
         case http {
           if-feature "http-listen";
           container http {
             description
               "HTTP-specific listening configuration for inbound
                connections.

                This transport option is made available to support
                deployments where the TLS connections are terminated
                by another system (e.g., a load balanacer) fronting
                the client.";
             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 "4336";
                   description
                     "The RESTCONF client will listen on the IANA-
                      assigned well-known port for 'restconf-ch-tls'
                      (4336) if no value is specified.";
                 }
               }
             }
             container http-client-parameters {
               description
                 "A wrapper around the HTTP client parameters
                  to avoid name collisions.";
               uses httpc:http-client-grouping;
             }
             container restconf-client-parameters {
               description
                 "A wrapper around the RESTCONF client parameters
                  to avoid name collisions.";
               uses rcc:restconf-client-grouping;
             }
           }
         }
         case https {
           if-feature "https-listen";
           container https {
             description
               "HTTPS-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 "4336";
                description
                  "The RESTCONF client will listen on the IANA-
                   assigned well-known port for 'restconf-ch-tls'
                   (4336) 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 http-client-parameters {
            description
              "A wrapper around the HTTP client parameters
               to avoid name collisions.";
            uses httpc:http-client-grouping;
          }
          container restconf-client-parameters {
            description
              "A wrapper around the RESTCONF client parameters
               to avoid name collisions.";
            uses rcc:restconf-client-grouping;
          }
        }
      }
    }
  } // restconf-client-listen-stack-grouping

  grouping restconf-client-app-grouping {
    description
      "A reusable grouping for configuring a RESTCONF client
       application that supports both 'initiate' and 'listen'
       protocol stacks for a multiplicity of connections.";
    container initiate {
      if-feature "https-initiate";
      presence "Enables client to initiate TCP connections";
```

```
        description
          "Configures client initiating underlying TCP connections.";
        list restconf-server {
          key "name";
          min-elements 1;
          description
            "List of RESTCONF servers the RESTCONF client is to
             maintain simultaneous connections with.";
          leaf name {
            type string;
            description
              "An arbitrary name for the RESTCONF server.";
          }
          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
                 RESTCONF client 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 restconf-client-initiate-stack-grouping;
            }
          }
          container connection-type {
            description
              "Indicates the RESTCONF client's preference for how
               the RESTCONF 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
                     RESTCONF server. If the connection goes down,
                     immediately start trying to reconnect to the
```

```
                    RESTCONF server, using the reconnection strategy.

                    This connection type minimizes any RESTCONF server
                    to RESTCONF 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 RESTCONF server.

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

                     The RESTCONF client SHOULD gracefully close
                     the underlying TLS 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 the underlying TCP session may remain
                     idle. A TCP session will be dropped if it
                     is idle for an interval longer than this
                     number of seconds If set to zero, then the
                     RESTCONF client will never drop a session
                     because it is idle.";
                }
              }
            } // periodic-connection
          } // connection-type
        } // connection-type
        container reconnect-strategy {
          description
            "The reconnection strategy directs how a RESTCONF
             client reconnects to a RESTCONF server, after
             discovering its connection to the server has
             dropped, even if due to a reboot.  The RESTCONF
             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.   RESTCONF 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 RESTCONF server's
                   endpoints the RESTCONF client should start
                   with when trying to connect to the RESTCONF
                   server.";
              }
              leaf max-attempts {
                type uint8 {
                  range "1..max";
                }
                default "3";
                description
                  "Specifies the number times the RESTCONF client
                   tries to connect to a specific endpoint before
                   moving on to the next endpoint in the list
                   (round robin).";
              }
            }
          }
        } // initiate
        container listen {
          if-feature "http-listen or https-listen";
          presence "Enables client to accept call-home connections";
          description
            "Configures the client to accept call-home TCP connections.";
          leaf idle-timeout {
            type uint16;
            units "seconds";
            default 3600; // one hour
            description
              "Specifies the maximum number of seconds that an
               underlying TCP session may remain idle. A TCP session
               will be dropped if it is idle for an interval longer
               then 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 RESTCONF connections.";
```

```
          leaf name {
            type string;
            description
              "An arbitrary name for the RESTCONF listen endpoint.";
          }
          uses restconf-client-listen-stack-grouping;
        }
      }
    } // restconf-client-app-grouping

    // Protocol accessible node, for servers that implement
    // this module.
    container restconf-client {
      uses restconf-client-app-grouping;
      description
        "Top-level container for RESTCONF client configuration.";
    }
  }
```

<CODE ENDS>

3. The RESTCONF Server Model

The RESTCONF server model presented in this section supports both listening for connections as well as initiating call-home connections.

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

3.1. Tree Diagram

The following tree diagram [RFC8340] provides an overview of the data model for the "ietf-restconf-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-restconf-server
  +--rw restconf-server
     +---u restconf-server-app-grouping

  grouping restconf-server-grouping
```

```
      +-- client-identity-mappings
         +---u x509c2n:cert-to-name
    grouping restconf-server-listen-stack-grouping
      +-- (transport)
         +--:(http) {http-listen}?
         |  +-- http
         |     +-- external-endpoint!
         |     |  +-- address    inet:ip-address
         |     |  +-- port?      inet:port-number
         |     +-- tcp-server-parameters
         |     |  +---u tcps:tcp-server-grouping
         |     +-- http-server-parameters
         |     |  +---u https:http-server-grouping
         |     +-- restconf-server-parameters
         |        +---u rcs:restconf-server-grouping
         +--:(https) {https-listen}?
            +-- https
               +-- tcp-server-parameters
               |  +---u tcps:tcp-server-grouping
               +-- tls-server-parameters
               |  +---u tlss:tls-server-grouping
               +-- http-server-parameters
               |  +---u https:http-server-grouping
               +-- restconf-server-parameters
                  +---u rcs:restconf-server-grouping
    grouping restconf-server-callhome-stack-grouping
      +-- (transport)
         +--:(https) {https-listen}?
            +-- https
               +-- tcp-client-parameters
               |  +---u tcpc:tcp-client-grouping
               +-- tls-server-parameters
               |  +---u tlss:tls-server-grouping
               +-- http-server-parameters
               |  +---u https:http-server-grouping
               +-- restconf-server-parameters
                  +---u rcs:restconf-server-grouping
    grouping restconf-server-app-grouping
      +-- listen! {http-listen or https-listen}?
      |  +-- endpoint* [name]
      |     +-- name?                                      string
      |     +---u restconf-server-listen-stack-grouping
      +-- call-home! {https-call-home}?
         +-- restconf-client* [name]
            +-- name?                 string
            +-- endpoints
            |  +-- endpoint* [name]
            |     +-- name?                                    string
```

```
            |     +---u restconf-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
```

3.2. Example Usage

The following example illustrates configuring a RESTCONF server to listen for RESTCONF client connections, as well as configuring call-home to one RESTCONF client.

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) ===========

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

  <!-- endpoints to listen for RESTCONF connections on -->
  <listen>
    <endpoint>
      <name>netconf/tls</name>
      <https>
        <tcp-server-parameters>
          <local-address>11.22.33.44</local-address>
        </tcp-server-parameters>
        <tls-server-parameters>
          <server-identity>
            <certificate>
              <local-definition>
                <public-key-format>ct:subject-public-key-info-format\
</public-key-format>
                <public-key>base64encodedvalue==</public-key>
                <private-key-format>ct:rsa-private-key-format</priva\
te-key-format>
```

```
                  <private-key>base64encodedvalue==</private-key>
                  <cert>base64encodedvalue==</cert>
                </local-definition>
              </certificate>
            </server-identity>
            <client-authentication>
              <ca-certs>
                <truststore-reference>trusted-client-ca-certs</truststo\
re-reference>
              </ca-certs>
              <ee-certs>
                <truststore-reference>trusted-client-ee-certs</truststo\
re-reference>
              </ee-certs>
            </client-authentication>
            <keepalives>
              <peer-allowed-to-send/>
            </keepalives>
          </tls-server-parameters>
          <http-server-parameters>
            <server-name>foo.example.com</server-name>
          </http-server-parameters>
          <restconf-server-parameters>
            <client-identity-mappings>
              <cert-to-name>
                <id>1</id>
                <fingerprint>11:0A:05:11:00</fingerprint>
                <map-type>x509c2n:specified</map-type>
                <name>scooby-doo</name>
              </cert-to-name>
                <cert-to-name>
                <id>2</id>
                <map-type>x509c2n:san-any</map-type>
              </cert-to-name>
            </client-identity-mappings>
          </restconf-server-parameters>
        </https>
      </endpoint>
    </listen>

    <!-- call home to a RESTCONF client with two endpoints -->
    <call-home>
      <restconf-client>
        <name>config-manager</name>
        <endpoints>
          <endpoint>
            <name>east-data-center</name>
            <https>
```

```
              <tcp-client-parameters>
                <remote-address>east.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-server-parameters>
                <server-identity>
                  <certificate>
                    <local-definition>
                      <public-key-format>ct:subject-public-key-info-fo\
rmat</public-key-format>
                      <public-key>base64encodedvalue==</public-key>
                      <private-key-format>ct:rsa-private-key-format</p\
rivate-key-format>
                      <private-key>base64encodedvalue==</private-key>
                      <cert>base64encodedvalue==</cert>
                    </local-definition>
                  </certificate>
                </server-identity>
                <client-authentication>
                  <ca-certs>
                    <truststore-reference>trusted-client-ca-certs</tru\
ststore-reference>
                  </ca-certs>
                  <ee-certs>
                    <truststore-reference>trusted-client-ee-certs</tru\
ststore-reference>
                  </ee-certs>
                </client-authentication>
                <keepalives>
                  <test-peer-aliveness>
                    <max-wait>30</max-wait>
                    <max-attempts>3</max-attempts>
                  </test-peer-aliveness>
                </keepalives>
              </tls-server-parameters>
              <http-server-parameters>
                <server-name>foo.example.com</server-name>
              </http-server-parameters>
              <restconf-server-parameters>
                <client-identity-mappings>
                  <cert-to-name>
                    <id>1</id>
                    <fingerprint>11:0A:05:11:00</fingerprint>
                    <map-type>x509c2n:specified</map-type>
```

```
                    <name>scooby-doo</name>
                  </cert-to-name>
                  <cert-to-name>
                    <id>2</id>
                    <map-type>x509c2n:san-any</map-type>
                  </cert-to-name>
                </client-identity-mappings>
              </restconf-server-parameters>
            </https>
          </endpoint>
          <endpoint>
            <name>west-data-center</name>
            <https>
              <tcp-client-parameters>
                <remote-address>west.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-server-parameters>
                <server-identity>
                  <certificate>
                    <local-definition>
                      <public-key-format>ct:subject-public-key-info-fo\
rmat</public-key-format>
                      <public-key>base64encodedvalue==</public-key>
                      <private-key-format>ct:rsa-private-key-format</p\
rivate-key-format>
                      <private-key>base64encodedvalue==</private-key>
                      <cert>base64encodedvalue==</cert>
                    </local-definition>
                  </certificate>
                </server-identity>
                <client-authentication>
                  <ca-certs>
                    <truststore-reference>trusted-client-ca-certs</tru\
ststore-reference>
                  </ca-certs>
                  <ee-certs>
                    <truststore-reference>trusted-client-ee-certs</tru\
ststore-reference>
                  </ee-certs>
                </client-authentication>
                <keepalives>
                  <test-peer-aliveness>
                    <max-wait>30</max-wait>
```

```
                    <max-attempts>3</max-attempts>
                  </test-peer-aliveness>
                </keepalives>
              </tls-server-parameters>
              <http-server-parameters>
                <server-name>foo.example.com</server-name>
              </http-server-parameters>
              <restconf-server-parameters>
                <client-identity-mappings>
                  <cert-to-name>
                    <id>1</id>
                    <fingerprint>11:0A:05:11:00</fingerprint>
                    <map-type>x509c2n:specified</map-type>
                    <name>scooby-doo</name>
                  </cert-to-name>
                  <cert-to-name>
                    <id>2</id>
                    <map-type>x509c2n:san-any</map-type>
                  </cert-to-name>
                </client-identity-mappings>
              </restconf-server-parameters>
            </https>
          </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>
      </restconf-client>
    </call-home>
  </restconf-server>
```

3.3. YANG Module

This YANG module has normative references to [RFC6991], [RFC7407], [RFC8040], [RFC8071], [I-D.kwatsen-netconf-tcp-client-server], [I-D.ietf-netconf-tls-client-server], and [I-D.kwatsen-netconf-http-client-server].

```
<CODE BEGINS> file "ietf-restconf-server@2020-05-20.yang"

module ietf-restconf-server {
```

```
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-restconf-server";
  prefix rcs;

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

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

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

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

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

  import ietf-tls-server {
    prefix tlss;
    reference
      "RFC FFFF: YANG Groupings for TLS Clients and TLS Servers";
  }

  import ietf-http-server {
    prefix https;
    reference
      "RFC GGGG: YANG Groupings for HTTP Clients and HTTP 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 RESTCONF servers.

     Copyright (c) 2020 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 IIII
     (https://www.rfc-editor.org/info/rfcIIII); 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 2020-05-20 {
    description
      "Initial version";
    reference
      "RFC IIII: RESTCONF Client and Server Models";
  }

  // Features

  feature http-listen {
    description
      "The 'http-listen' feature indicates that the RESTCONF server
       supports opening a port to listen for incoming RESTCONF over
       TPC client connections, whereby the TLS connections are
       terminated by an external system.";
```

```
      reference
        "RFC 8040: RESTCONF Protocol";
    }

    feature https-listen {
      description
        "The 'https-listen' feature indicates that the RESTCONF server
         supports opening a port to listen for incoming RESTCONF over
         TLS client connections, whereby the TLS connections are
         terminated by the server itself.";
      reference
        "RFC 8040: RESTCONF Protocol";
    }

    feature https-call-home {
      description
        "The 'https-call-home' feature indicates that the RESTCONF
         server supports initiating connections to RESTCONF clients.";
      reference
        "RFC 8071: NETCONF Call Home and RESTCONF Call Home";
    }


    // Groupings

    grouping restconf-server-grouping {
      description
        "A reusable grouping for configuring a RESTCONF 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.,
         'restconf-server-parameters'.  This model purposely does
         not do this itself so as to provide maximum flexibility
         to consuming models.";

      container client-identity-mappings {
        description
          "Specifies mappings through which RESTCONF client X.509
           certificates are used to determine a RESTCONF username.
           If no matching and valid cert-to-name list entry can be
           found, then the RESTCONF server MUST close the connection,
           and MUST NOT accept RESTCONF messages over it.";
        reference
```

```
          "RFC 7407: A YANG Data Model for SNMP Configuration.";
        uses x509c2n:cert-to-name {
          refine "cert-to-name/fingerprint" {
            mandatory false;
            description
              "A 'fingerprint' value does not need to be specified
               when the 'cert-to-name' mapping is independent of
               fingerprint matching.  A 'cert-to-name' having no
               fingerprint value will match any client certificate
               and therefore should only be present at the end of
               the user-ordered 'cert-to-name' list.";
          }
        }
      }
    }

    grouping restconf-server-listen-stack-grouping {
      description
        "A reusable grouping for configuring a RESTCONF server
         'listen' protocol stack for a single connection.";
      choice transport {
        mandatory true;
        description
          "Selects between available transports. This is a
           'choice' statement so as to support additional
           transport options to be augmented in.";
        case http {
          if-feature "http-listen";
          container http {
            description
              "Configures RESTCONF server stack assuming that
               TLS-termination is handled externally.";
            container external-endpoint {
              presence
                "Specifies configuration for an external endpoint.";
              description
                "Identifies contact information for the external
                 system that terminates connections before passing
                 them thru to this server (e.g., a network address
                 translator or a load balancer).  These values have
                 no effect on the local operation of this server, but
                 may be used by the application when needing to
                 inform other systems how to contact this server.";
              leaf address {
                type inet:ip-address;
                mandatory true;
                description
                  "The IP address or hostname of the external system
```

```
                that terminates incoming RESTCONF client
                connections before forwarding them to this
                server.";
            }
            leaf port {
              type inet:port-number;
              default "443";
              description
                "The port number that the external system listens
                 on for incoming RESTCONF client connections that
                 are forwarded to this server.  The default HTTPS
                 port (443) is used, as expected for a RESTCONF
                 connection.";
            }
          }
          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 "80";
                description
                  "The RESTCONF server will listen on the IANA-
                   assigned well-known port value for 'http'
                   (80) if no value is specified.";
              }
            }
          }
          container http-server-parameters {
            description
              "A wrapper around the HTTP server parameters
               to avoid name collisions.";
            uses https:http-server-grouping;
          }
          container restconf-server-parameters {
            description
              "A wrapper around the RESTCONF server parameters
               to avoid name collisions.";
            uses rcs:restconf-server-grouping;
          }
        }
      }
      case https {
        if-feature "https-listen";
        container https {
          description
            "Configures RESTCONF server stack assuming that
```

```
              TLS-termination is handled internally.";
            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 "443";
                  description
                    "The RESTCONF server will listen on the IANA-
                     assigned well-known port value for 'https'
                     (443) 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;
            }
            container http-server-parameters {
              description
                "A wrapper around the HTTP server parameters
                 to avoid name collisions.";
              uses https:http-server-grouping;
            }
            container restconf-server-parameters {
              description
                "A wrapper around the RESTCONF server parameters
                 to avoid name collisions.";
              uses rcs:restconf-server-grouping;
            }
          }
        }
      }
    }

    grouping restconf-server-callhome-stack-grouping {
      description
        "A reusable grouping for configuring a RESTCONF server
         'call-home' protocol stack, for a single connection.";
      choice transport {
        mandatory true;
        description
          "Selects between available transports. This is a
           'choice' statement so as to support additional
           transport options to be augmented in.";
```

```
        case https {
          if-feature "https-listen";
          container https {
            description
              "Configures RESTCONF server stack assuming that
               TLS-termination is handled internally.";
            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 "4336";
                  description
                    "The RESTCONF server will attempt to
                     connect to the IANA-assigned well-known
                     port for 'restconf-ch-tls' (4336) 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;
            }
            container http-server-parameters {
              description
                "A wrapper around the HTTP server parameters
                 to avoid name collisions.";
              uses https:http-server-grouping;
            }
            container restconf-server-parameters {
              description
                "A wrapper around the RESTCONF server parameters
                 to avoid name collisions.";
              uses rcs:restconf-server-grouping;
            }
          }
        }
      }
    }

    grouping restconf-server-app-grouping {
      description
        "A reusable grouping for configuring a RESTCONF server
```

```
          application that supports both 'listen' and 'call-home'
          protocol stacks for a multiplicity of connections.";
        container listen {
          if-feature "http-listen or https-listen";
          presence
            "Enables the RESTCONF server to listen for RESTCONF
             client connections.";
          description "Configures listen behavior";
          list endpoint {
            key "name";
            min-elements 1;
            description
              "List of endpoints to listen for RESTCONF connections.";
            leaf name {
              type string;
              description
                "An arbitrary name for the RESTCONF listen endpoint.";
            }
            uses restconf-server-listen-stack-grouping;
          }
        }
        container call-home {
          if-feature "https-call-home";
          presence
            "Enables the RESTCONF server to initiate the underlying
             transport connection to RESTCONF clients.";
          description "Configures call-home behavior";
          list restconf-client {
            key "name";
            min-elements 1;
            description
              "List of RESTCONF clients the RESTCONF server is to
               maintain simultaneous call-home connections with.";
            leaf name {
              type string;
              description
                "An arbitrary name for the remote RESTCONF client.";
            }
            container endpoints {
              description
                "Container for the list of endpoints.";
              list endpoint {
                key "name";
                min-elements 1;
                ordered-by user;
                description
                  "User-ordered list of endpoints for this RESTCONF
                   client.  Defining more than one enables high-
```

```
               availability.";
            leaf name {
              type string;
              description
                "An arbitrary name for this endpoint.";
            }
            uses restconf-server-callhome-stack-grouping;
          }
        }
        container connection-type {
          description
            "Indicates the RESTCONF server's preference for how the
             RESTCONF 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 RESTCONF
                   client. If the connection goes down, immediately
                   start trying to reconnect to the RESTCONF server,
                   using the reconnection strategy.

                   This connection type minimizes any RESTCONF
                   client to RESTCONF 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 RESTCONF client.

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

                   The RESTCONF client SHOULD gracefully close
                   the underlying TLS connection upon completing
                   planned activities.  If the underlying TLS
                   connection is not closed gracefully, the
```

```
                    RESTCONF server MUST immediately attempt
                    to reestablish the connection.

                    In the case that the previous connection is
                    still active (i.e., the RESTCONF 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
                       the underlying TCP session may remain idle.
                       A TCP 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.";
                  }
                }
              }
            }
          }
          container reconnect-strategy {
```

```
            description
              "The reconnection strategy directs how a RESTCONF server
               reconnects to a RESTCONF client after discovering its
               connection to the client has dropped, even if due to a
               reboot.  The RESTCONF 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.   RESTCONF
                     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 RESTCONF client's endpoints
                 the RESTCONF server should start with when trying
                 to connect to the RESTCONF client.";
            }
            leaf max-attempts {
              type uint8 {
                range "1..max";
              }
              default "3";
              description
                "Specifies the number times the RESTCONF server tries
                 to connect to a specific endpoint before moving on to
                 the next endpoint in the list (round robin).";
            }
          }
        } // restconf-client
```

```
      } // call-home
    } // restconf-server-app-grouping



    // Protocol accessible node, for servers that implement
    // this module.
    container restconf-server {
      uses restconf-server-app-grouping;
      description
        "Top-level container for RESTCONF server configuration.";
    }

  }
```

<CODE ENDS>

4. Security Considerations

The YANG module defined in this document uses groupings defined in [I-D.kwatsen-netconf-tcp-client-server], [I-D.ietf-netconf-tls-client-server], and [I-D.kwatsen-netconf-http-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.

5. IANA Considerations

5.1. 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-restconf-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-restconf-server
Registrant Contact: The NETCONF WG of the IETF.
XML: N/A, the requested URI is an XML namespace.
```

5.2. 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-restconf-client
namespace:    urn:ietf:params:xml:ns:yang:ietf-restconf-client
prefix:       ncc
reference:    RFC IIII

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

6. References

6.1. Normative References

[I-D.ietf-netconf-keystore]
Watsen, K., "A YANG Data Model for a Keystore", draft-ietf-netconf-keystore-16 (work in progress), March 2020.

[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-18 (work in progress), March 2020.

[I-D.kwatsen-netconf-http-client-server]
Watsen, K., "YANG Groupings for HTTP Clients and HTTP Servers", draft-kwatsen-netconf-http-client-server-05 (work in progress), November 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>.

[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>.

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

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

[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>.

[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>.

6.2. Informative References

[I-D.ietf-netconf-trust-anchors] Watsen, K., "A YANG Data Model for a Truststore", draft-ietf-netconf-trust-anchors-09 (work in progress), March 2020.

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

[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>.

[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>.

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

6.3. URIs

[1] https://tools.ietf.org/html/draft-ietf-netconf-crypto-types

[2] https://tools.ietf.org/html/draft-ietf-netconf-trust-anchors

[3] https://tools.ietf.org/html/draft-ietf-netconf-keystore

[4] https://tools.ietf.org/html/draft-ietf-netconf-tcp-client-server

[5] https://tools.ietf.org/html/draft-ietf-netconf-ssh-client-server

[6] https://tools.ietf.org/html/draft-ietf-netconf-tls-client-server

[7] https://tools.ietf.org/html/draft-ietf-netconf-http-client-server

[8] https://tools.ietf.org/html/draft-ietf-netconf-netconf-client-server

[9] https://tools.ietf.org/html/draft-ietf-netconf-restconf-client-server

Appendix A. Expanded Tree Diagrams

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

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

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

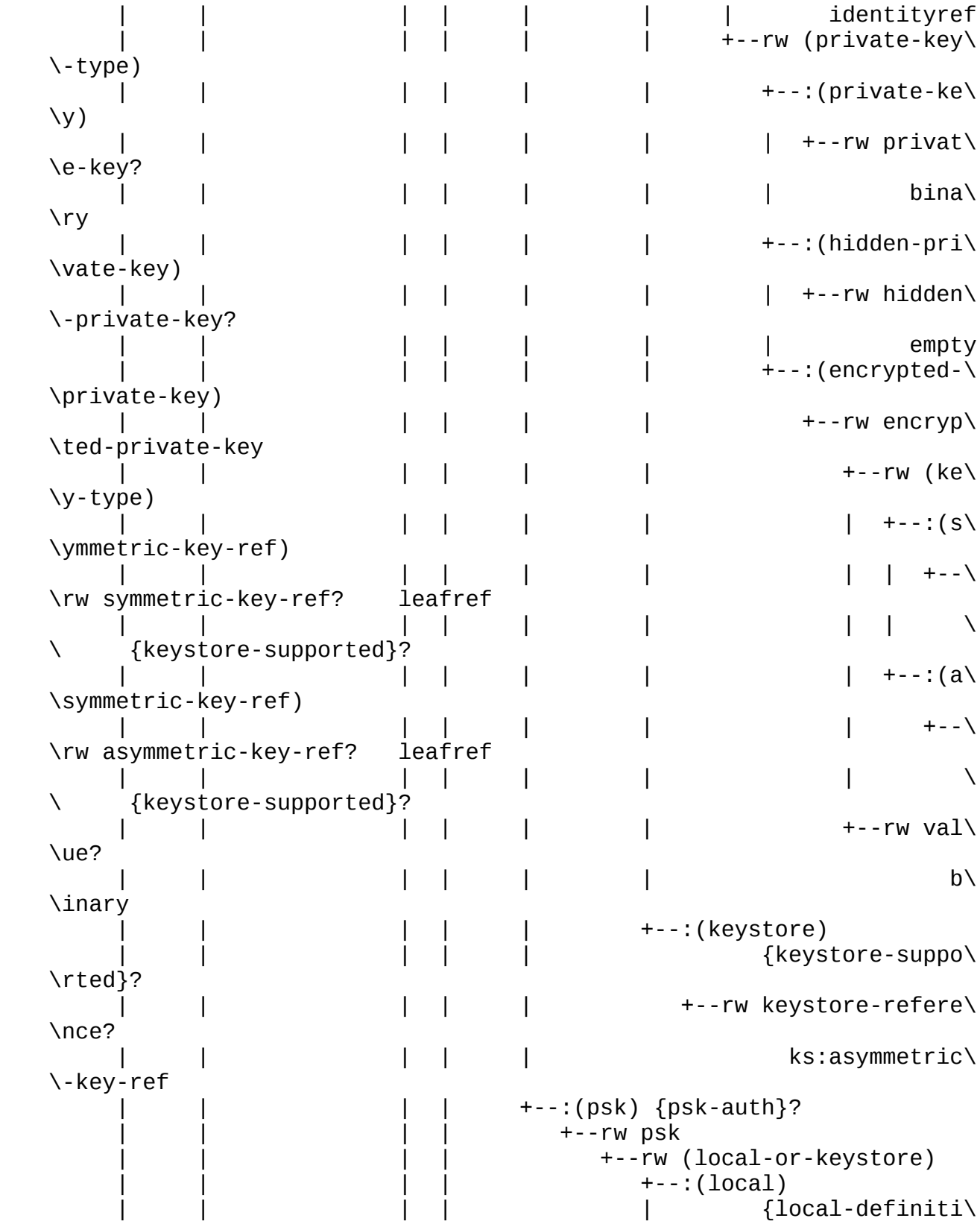
Please see Section 2.1 for a tree diagram that illustrates what the module looks like without all the "grouping" statements expanded.

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

module: ietf-restconf-client
  +--rw restconf-client
     +--rw initiate! {https-initiate}?
     |  +--rw restconf-server* [name]
     |     +--rw name                    string
     |     +--rw endpoints
     |     |  +--rw endpoint* [name]
     |     |     +--rw name           string
     |     |     +--rw (transport)
     |     |        +--:(https) {https-initiate}?
     |     |           +--rw https
     |     |              +--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 (auth-type)?
     |     |              |  |     +--:(certificate)
     |     |              |  |     |        {x509-certificate-auth}?
     |     |              |  |     |  +--rw certificate
     |     |              |  |     |     +--rw (local-or-keystore)
     |     |              |  |     |        +--:(local)
     |     |              |  |     |        |        {local-definiti\
\ons-supported}?
```

```
      |     |              |  |     |        |  +--rw local-definition
      |     |              |  |     |        |     +--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
      |     |              |  |     |        |     +--rw cert?
      |     |              |  |     |        |     |       end-entity\
\-cert-cms
      |     |              |  |     |        |     +---n certificate-\
```

```
   \expiration
        |     |              |  |     |        |     |  +-- expiration-\
   \date
        |     |              |  |     |        |     |          yang:da\
   \te-and-time
        |     |              |  |     |        |     +---x generate-cer\
   \tificate-signing-request
        |     |              |  |     |        |             {certifica\
   \te-signing-request-generation}?
        |     |              |  |     |        |        +---w input
        |     |              |  |     |        |        |  +---w subject
        |     |              |  |     |        |        |  |       bina\
   \ry
        |     |              |  |     |        |        |  +---w attrib\
   \utes?
        |     |              |  |     |        |        |          bina\
   \ry
        |     |              |  |     |        |        +--ro output
        |     |              |  |     |        |           +--ro certif\
   \icate-signing-request
        |     |              |  |     |        |                  ct:c\
   \sr
        |     |              |  |     |        +--:(keystore)
        |     |              |  |     |                 {keystore-suppo\
   \rted}?
        |     |              |  |     |           +--rw keystore-refere\
   \nce
        |     |              |  |     |              +--rw asymmetric-k\
   \ey?
        |     |              |  |     |              |       ks:asymmet\
   \ric-key-ref
        |     |              |  |     |              +--rw certificate?\
   \      leafref
        |     |              |  |     +--:(raw-public-key)
        |     |              |  |     |        {raw-public-key-auth}?
        |     |              |  |     |  +--rw raw-private-key
        |     |              |  |     |     +--rw (local-or-keystore)
        |     |              |  |     |        +--:(local)
        |     |              |  |     |        |        {local-definiti\
   \ons-supported}?
        |     |              |  |     |        |  +--rw local-definition
        |     |              |  |     |        |     +--rw public-key-f\
   \ormat
        |     |              |  |     |        |     |       identityref
        |     |              |  |     |        |     +--rw public-key
        |     |              |  |     |        |     |       binary
        |     |              |  |     |        |     +--rw private-key-\
   \format?
```

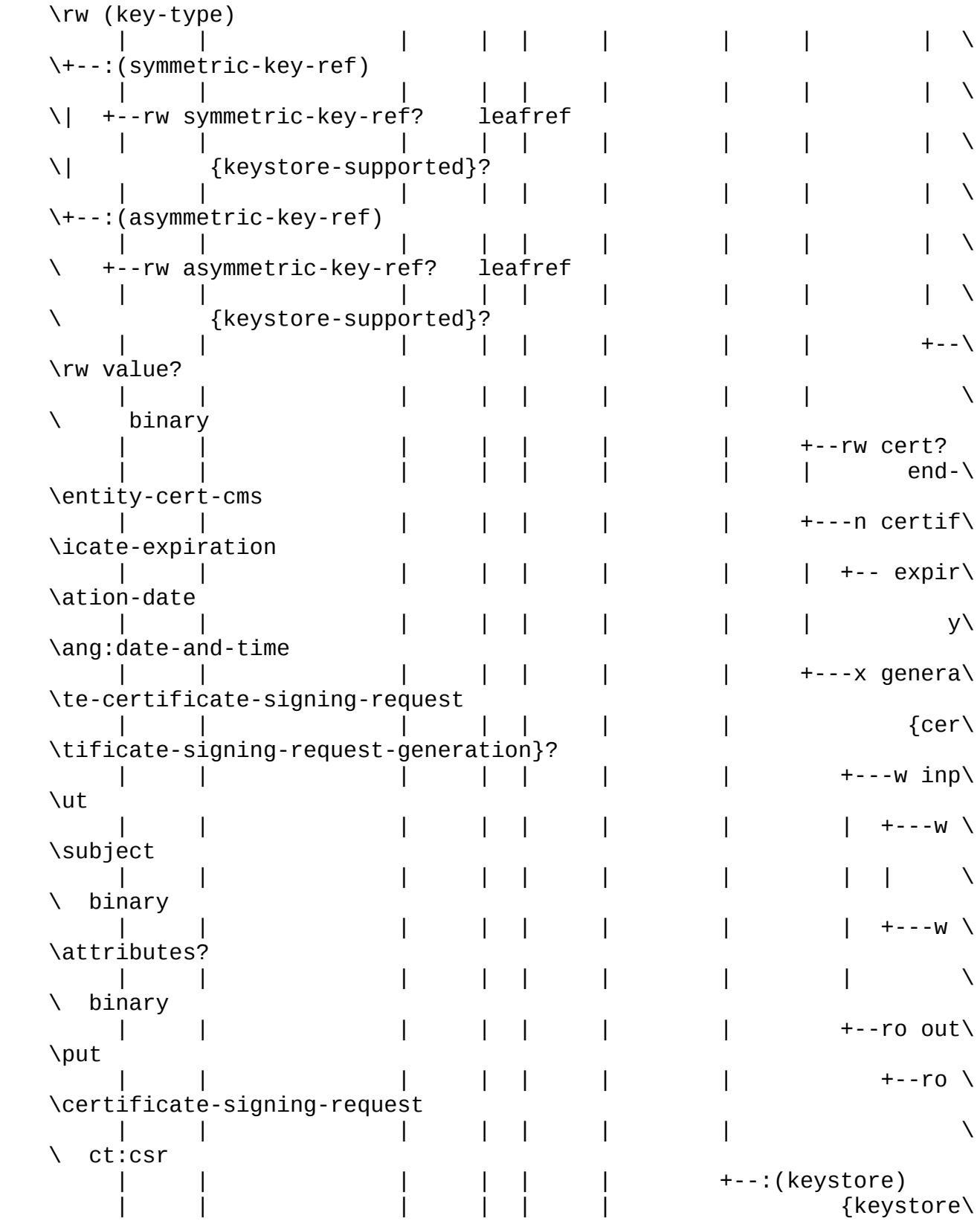

identityref
+--rw (private-key\
\-type)
+--:(private-ke\
\y)
+--rw privat\
\e-key?
bina\
\ry
+--:(hidden-pri\
\vate-key)
+--rw hidden\
\-private-key?
empty
+--:(encrypted-\
\private-key)
+--rw encryp\
\ted-private-key
+--rw (ke\
\y-type)
+--:(s\
\ymmetric-key-ref)
+--\
\rw symmetric-key-ref? leafref
\
\ {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
+--:(psk) {psk-auth}?
+--rw psk
+--rw (local-or-keystore)
+--:(local)
{local-definiti\

```
   \ons-supported}?
        |     |              |  |              |  +--rw local-definition
        |     |              |  |              |     +--rw key-format?
        |     |              |  |              |     |       identityref
        |     |              |  |              |     +--rw (key-type)
        |     |              |  |              |     |  +--:(key)
        |     |              |  |              |     |  |  +--rw key?
        |     |              |  |              |     |  |          bina\
   \ry
        |     |              |  |              |     |  +--:(hidden-key)
        |     |              |  |              |     |  |  +--rw hidden\
   \-key?
        |     |              |  |              |     |  |          empty
        |     |              |  |              |     |  +--:(encrypted-\
   \key)
        |     |              |  |              |     |     +--rw encryp\
   \ted-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
        |     |              |  |              |     +--rw id?
        |     |              |  |              |             string
        |     |              |  |              |             {ks:local-\
   \definitions-supported}?
        |     |              |  |              +--:(keystore)
        |     |              |  |                    {keystore-suppo\
   \rted}?
        |     |              |  |                 +--rw keystore-refere\
   \nce?
        |     |              |  |                       ks:symmetric-\
   \key-ref
        |     |              |  +--rw server-authentication
        |     |              |  |  +--rw ca-certs!
```

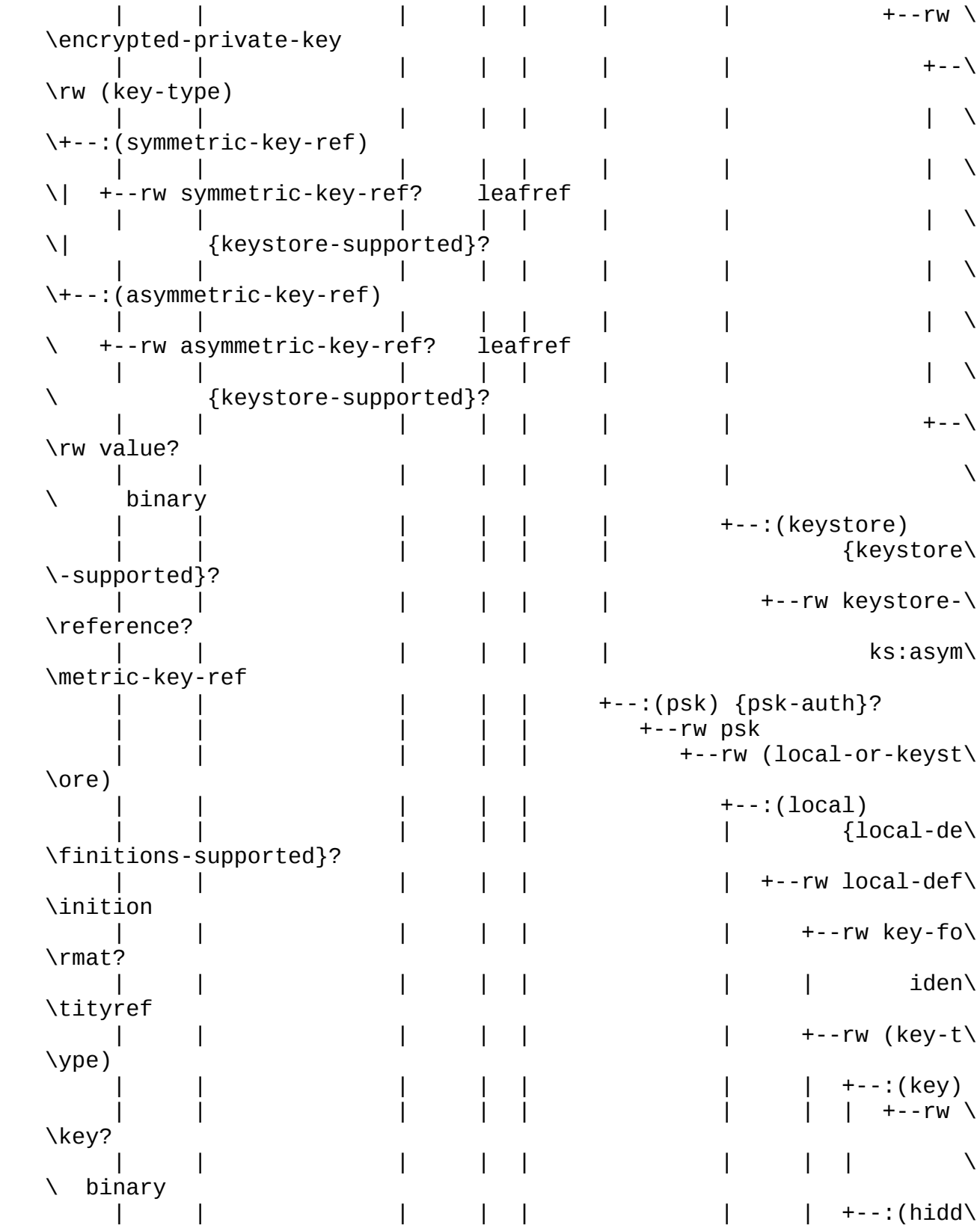
```
       |     |              |  |  |       {x509-certificate-auth}?
       |     |              |  |  |  +--rw (local-or-truststore)
       |     |              |  |  |     +--:(local)
       |     |              |  |  |     |        {local-definitions-su\
   \pported}?
       |     |              |  |  |     |  +--rw local-definition
       |     |              |  |  |     |     +--rw cert*
       |     |              |  |  |     |     |       trust-anchor-cer\
   \t-cms
       |     |              |  |  |     |     +---n certificate-expira\
   \tion
       |     |              |  |  |     |        +-- expiration-date
       |     |              |  |  |     |                yang:date-and\
   \-time
       |     |              |  |  |     +--:(truststore)
       |     |              |  |  |              {truststore-supported\
   \,certificates}?
       |     |              |  |  |        +--rw truststore-reference?
       |     |              |  |  |                ts:certificate-bag-\
   \ref
       |     |              |  |  +--rw ee-certs!
       |     |              |  |  |       {x509-certificate-auth}?
       |     |              |  |  |  +--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\
   \,certificates}?
       |     |              |  |  |        +--rw truststore-reference?
       |     |              |  |  |                ts:certificate-bag-\
   \ref
       |     |              |  |  +--rw raw-public-keys!
       |     |              |  |  |       {raw-public-key-auth}?
       |     |              |  |  |  +--rw (local-or-truststore)
       |     |              |  |  |     +--:(local)
       |     |              |  |  |     |        {local-definitions-su\
   \pported}?
       |     |              |  |  |     |  +--rw local-definition
```

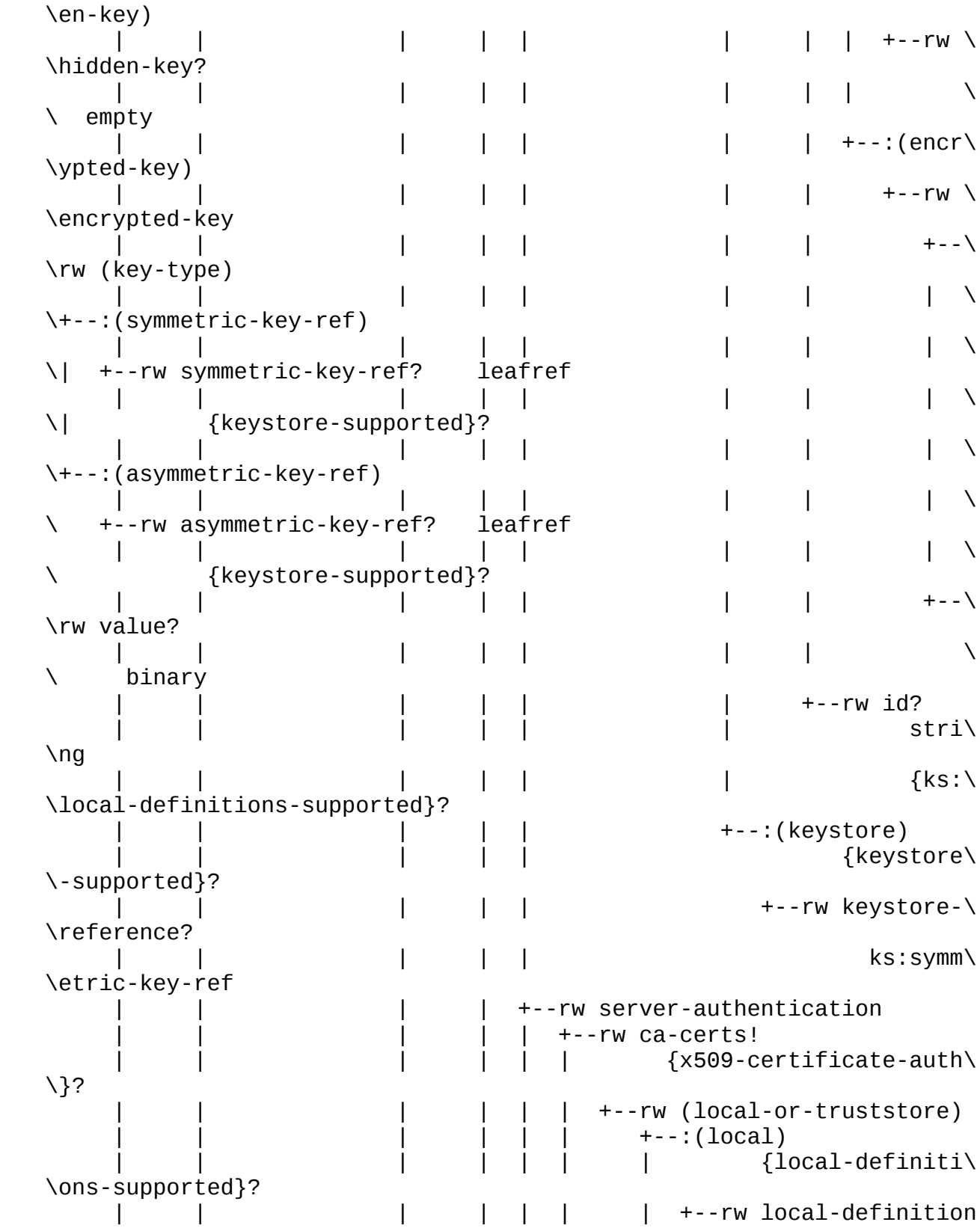
```
      |     |              |  |  |     |     +--rw public-key* [name]
      |     |              |  |  |     |        +--rw name
      |     |              |  |  |     |        |       string
      |     |              |  |  |     |        +--rw public-key-form\
 \at
      |     |              |  |  |     |        |       identityref
      |     |              |  |  |     |        +--rw public-key
      |     |              |  |  |     |                binary
      |     |              |  |  |     +--:(truststore)
      |     |              |  |  |              {truststore-supported\
 \,public-keys}?
      |     |              |  |  |        +--rw truststore-reference?
      |     |              |  |  |                 ts:public-key-bag-r\
 \ef
      |     |              |  |  +--rw psks! {psk-auth}?
      |     |              |  +--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 peer-allowed-to-send?   empty
      |     |              |     +--rw test-peer-aliveness!
      |     |              |        +--rw max-wait?       uint16
      |     |              |        +--rw max-attempts?   uint8
      |     |             +--rw http-client-parameters
      |     |             |  +--rw client-identity!
      |     |             |  |  +--rw (auth-type)?
      |     |             |  |     +--:(basic)
      |     |             |  |        +--rw basic {basic-auth}?
      |     |             |  |           +--rw user-id     string
      |     |             |  |           +--rw password    string
      |     |             |  +--rw proxy-server! {proxy-connect}?
      |     |             |     +--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 (auth-type)?
        |     |              |     |  |     +--:(certificate)
        |     |              |     |  |     |        {x509-certificate-\
   \auth}?
        |     |              |     |  |     |  +--rw certificate
        |     |              |     |  |     |     +--rw (local-or-keyst\
   \ore)
        |     |              |     |  |     |        +--:(local)
        |     |              |     |  |     |        |        {local-de\
   \finitions-supported}?
        |     |              |     |  |     |        |  +--rw local-def\
   \inition
        |     |              |     |  |     |        |     +--rw public\
   \-key-format
        |     |              |     |  |     |        |     |       iden\
   \tityref
        |     |              |     |  |     |        |     +--rw public\
   \-key
        |     |              |     |  |     |        |     |       bina\
   \ry
        |     |              |     |  |     |        |     +--rw privat\
   \e-key-format?
        |     |              |     |  |     |        |     |       iden\
   \tityref
        |     |              |     |  |     |        |     +--rw (priva\
   \te-key-type)
        |     |              |     |  |     |        |     |  +--:(priv\
   \ate-key)
        |     |              |     |  |     |        |     |  |  +--rw \
   \private-key?
        |     |              |     |  |     |        |     |  |       \
   \  binary
        |     |              |     |  |     |        |     |  +--:(hidd\
   \en-private-key)
        |     |              |     |  |     |        |     |  |  +--rw \
   \hidden-private-key?
        |     |              |     |  |     |        |     |  |       \
   \  empty
        |     |              |     |  |     |        |     |  +--:(encr\
   \ypted-private-key)
        |     |              |     |  |     |        |     |     +--rw \
   \encrypted-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?
\entity-cert-cms
+---n certif\
\icate-expiration
+-- expir\
\ation-date
y\
\ang:date-and-time
+---x genera\
\te-certificate-signing-request
{cer\
\tificate-signing-request-generation}?
+---w inp\
\ut
+---w \
\subject
\ binary
+---w \
\attributes?
\ binary
+--ro out\
\put
+--ro \
\certificate-signing-request
\ ct:csr
+--:(keystore)
{keystore\

```
   \-supported}?
        |     |              |     |  |     |          +--rw keystore-\
   \reference
        |     |              |     |  |     |             +--rw asymme\
   \tric-key?
        |     |              |     |  |     |             |       ks:a\
   \symmetric-key-ref
        |     |              |     |  |     |             +--rw certif\
   \icate?      leafref
        |     |              |     |  |     +--:(raw-public-key)
        |     |              |     |  |     |        {raw-public-key-au\
   \th}?
        |     |              |     |  |     |  +--rw raw-private-key
        |     |              |     |  |     |     +--rw (local-or-keyst\
   \ore)
        |     |              |     |  |     |        +--:(local)
        |     |              |     |  |     |        |        {local-de\
   \finitions-supported}?
        |     |              |     |  |     |        |  +--rw local-def\
   \inition
        |     |              |     |  |     |        |     +--rw public\
   \-key-format
        |     |              |     |  |     |        |     |       iden\
   \tityref
        |     |              |     |  |     |        |     +--rw public\
   \-key
        |     |              |     |  |     |        |     |       bina\
   \ry
        |     |              |     |  |     |        |     +--rw privat\
   \e-key-format?
        |     |              |     |  |     |        |     |       iden\
   \tityref
        |     |              |     |  |     |        |     +--rw (priva\
   \te-key-type)
        |     |              |     |  |     |        |        +--:(priv\
   \ate-key)
        |     |              |     |  |     |        |        |  +--rw \
   \private-key?
        |     |              |     |  |     |        |        |       \
   \  binary
        |     |              |     |  |     |        |        +--:(hidd\
   \en-private-key)
        |     |              |     |  |     |        |        |  +--rw \
   \hidden-private-key?
        |     |              |     |  |     |        |        |       \
   \  empty
        |     |              |     |  |     |        |        +--:(encr\
   \ypted-private-key)
```


+--rw \
\encrypted-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\
\-supported}?
+--rw keystore-\
\reference?
ks:asym\
\metric-key-ref
+--:(psk) {psk-auth}?
+--rw psk
+--rw (local-or-keyst\
\ore)
+--:(local)
{local-de\
\finitions-supported}?
+--rw local-def\
\inition
+--rw key-fo\
\rmat?
iden\
\tityref
+--rw (key-t\
\ype)
+--:(key)
+--rw \
\key?
\
\ binary
+--:(hidd\


\en-key)
\hidden-key?
\ empty
\ypted-key)
\encrypted-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 id?
stri\
\ng
{ks:\
\local-definitions-supported}?
+--:(keystore)
{keystore\
\-supported}?
+--rw keystore-\
\reference?
ks:symm\
\etric-key-ref
+--rw server-authentication
+--rw ca-certs!
{x509-certificate-auth\
\}?
+--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,certificates}?
      |     |              |     |  |  |        +--rw truststore-refe\
\rence?
      |     |              |     |  |  |                ts:certificat\
\e-bag-ref
      |     |              |     |  |  +--rw ee-certs!
      |     |              |     |  |  |       {x509-certificate-auth\
\}?
      |     |              |     |  |  |  +--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,certificates}?
      |     |              |     |  |  |        +--rw truststore-refe\
\rence?
      |     |              |     |  |  |                ts:certificat\
\e-bag-ref
      |     |              |     |  |  +--rw raw-public-keys!
      |     |              |     |  |  |       {raw-public-key-auth}?
      |     |              |     |  |  |  +--rw (local-or-truststore)
      |     |              |     |  |  |     +--:(local)
      |     |              |     |  |  |     |        {local-definiti\
\ons-supported}?
      |     |              |     |  |  |     |  +--rw local-definition
      |     |              |     |  |  |     |     +--rw public-key*
```

```
        |     |              |     |  |  |     |             [name]
        |     |              |     |  |  |     |       +--rw name
        |     |              |     |  |  |     |       |       string
        |     |              |     |  |  |     |       +--rw public-ke\
   \y-format
        |     |              |     |  |  |     |       |       identit\
   \yref
        |     |              |     |  |  |     |       +--rw public-key
        |     |              |     |  |  |     |               binary
        |     |              |     |  |  |     +--:(truststore)
        |     |              |     |  |  |             {truststore-sup\
   \ported,public-keys}?
        |     |              |     |  |  |       +--rw truststore-refe\
   \rence?
        |     |              |     |  |  |               ts:public-key\
   \-bag-ref
        |     |              |     |  |  +--rw psks! {psk-auth}?
        |     |              |     |  +--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 peer-allowed-to-send?
        |     |              |     |     |       empty
        |     |              |     |     +--rw test-peer-aliveness!
        |     |              |     |        +--rw max-wait?       uint16
        |     |              |     |        +--rw max-attempts?   uint8
        |     |              |     +--rw http-client-parameters
        |     |              |        +--rw client-identity!
        |     |              |           +--rw (auth-type)?
        |     |              |              +--:(basic)
        |     |              |                 +--rw basic {basic-auth}?
        |     |              |                    +--rw user-id
        |     |              |                    |       string
        |     |              |                    +--rw password
        |     |              |                            string
        |     |              +--rw restconf-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! {http-listen or https-listen}?
           +--rw idle-timeout?   uint16
           +--rw endpoint* [name]
              +--rw name           string
              +--rw (transport)
                 +--:(http) {http-listen}?
                 |  +--rw http
                 |     +--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 http-client-parameters
                 |     |  +--rw client-identity!
                 |     |  |  +--rw (auth-type)?
                 |     |  |     +--:(basic)
                 |     |  |        +--rw basic {basic-auth}?
                 |     |  |           +--rw user-id     string
                 |     |  |           +--rw password    string
                 |     |  +--rw proxy-server! {proxy-connect}?
                 |     |     +--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 (auth-type)?
                 |     |     |  |     +--:(certificate)
                 |     |     |  |     |        {x509-certificate-auth}?
                 |     |     |  |     |  +--rw certificate
                 |     |     |  |     |     +--rw (local-or-keystore)
```

```
                |     |     |  |     |        +--:(local)
                |     |     |  |     |        |        {local-definiti\
   \ons-supported}?
                |     |     |  |     |        |  +--rw local-definition
                |     |     |  |     |        |     +--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
                |     |     |  |     |        |     +--rw cert?
```

```
                |     |     |  |     |        |     |       end-entity\
   \-cert-cms
                |     |     |  |     |        |     +---n certificate-\
   \expiration
                |     |     |  |     |        |     |  +-- expiration-\
   \date
                |     |     |  |     |        |     |          yang:da\
   \te-and-time
                |     |     |  |     |        |     +---x generate-cer\
   \tificate-signing-request
                |     |     |  |     |        |             {certifica\
   \te-signing-request-generation}?
                |     |     |  |     |        |        +---w input
                |     |     |  |     |        |        |  +---w subject
                |     |     |  |     |        |        |  |       bina\
   \ry
                |     |     |  |     |        |        |  +---w attrib\
   \utes?
                |     |     |  |     |        |        |          bina\
   \ry
                |     |     |  |     |        |        +--ro output
                |     |     |  |     |        |           +--ro certif\
   \icate-signing-request
                |     |     |  |     |        |                  ct:c\
   \sr
                |     |     |  |     |        +--:(keystore)
                |     |     |  |     |                 {keystore-suppo\
   \rted}?
                |     |     |  |     |           +--rw keystore-refere\
   \nce
                |     |     |  |     |              +--rw asymmetric-k\
   \ey?
                |     |     |  |     |              |       ks:asymmet\
   \ric-key-ref
                |     |     |  |     |              +--rw certificate?\
   \      leafref
                |     |     |  |     +--:(raw-public-key)
                |     |     |  |     |        {raw-public-key-auth}?
                |     |     |  |     |  +--rw raw-private-key
                |     |     |  |     |     +--rw (local-or-keystore)
                |     |     |  |     |        +--:(local)
                |     |     |  |     |        |        {local-definiti\
   \ons-supported}?
                |     |     |  |     |        |  +--rw local-definition
                |     |     |  |     |        |     +--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
                |     |     |  |     +--:(psk) {psk-auth}?
                |     |     |  |        +--rw psk
```

```
                |     |     |  |           +--rw (local-or-keystore)
                |     |     |  |              +--:(local)
                |     |     |  |              |        {local-definiti\
   \ons-supported}?
                |     |     |  |              |  +--rw local-definition
                |     |     |  |              |     +--rw key-format?
                |     |     |  |              |     |       identityref
                |     |     |  |              |     +--rw (key-type)
                |     |     |  |              |     |  +--:(key)
                |     |     |  |              |     |  |  +--rw key?
                |     |     |  |              |     |  |          bina\
   \ry
                |     |     |  |              |     |  +--:(hidden-key)
                |     |     |  |              |     |  |  +--rw hidden\
   \-key?
                |     |     |  |              |     |  |          empty
                |     |     |  |              |     |  +--:(encrypted-\
   \key)
                |     |     |  |              |     |     +--rw encryp\
   \ted-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
                |     |     |  |              |     +--rw id?
                |     |     |  |              |             string
                |     |     |  |              |             {ks:local-\
   \definitions-supported}?
                |     |     |  |              +--:(keystore)
                |     |     |  |                       {keystore-suppo\
   \rted}?
                |     |     |  |                 +--rw keystore-refere\
   \nce?
                |     |     |  |                       ks:symmetric-\
```

```
   \key-ref
                |     |     |  +--rw server-authentication
                |     |     |  |  +--rw ca-certs!
                |     |     |  |  |       {x509-certificate-auth}?
                |     |     |  |  |  +--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\
   \,certificates}?
                |     |     |  |  |        +--rw truststore-reference?
                |     |     |  |  |                ts:certificate-bag-\
   \ref
                |     |     |  |  +--rw ee-certs!
                |     |     |  |  |       {x509-certificate-auth}?
                |     |     |  |  |  +--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\
   \,certificates}?
                |     |     |  |  |        +--rw truststore-reference?
                |     |     |  |  |                ts:certificate-bag-\
   \ref
                |     |     |  |  +--rw raw-public-keys!
                |     |     |  |  |       {raw-public-key-auth}?
                |     |     |  |  |  +--rw (local-or-truststore)
                |     |     |  |  |     +--:(local)
```

```
                |     |     |  |  |     |       {local-definitions-su\
   \pported}?
                |     |     |  |  |     |  +--rw local-definition
                |     |     |  |  |     |     +--rw public-key* [name]
                |     |     |  |  |     |        +--rw name
                |     |     |  |  |     |        |       string
                |     |     |  |  |     |        +--rw public-key-form\
   \at
                |     |     |  |  |     |        |       identityref
                |     |     |  |  |     |        +--rw public-key
                |     |     |  |  |     |                 binary
                |     |     |  |  |     +--:(truststore)
                |     |     |  |  |              {truststore-supported\
   \,public-keys}?
                |     |     |  |  |        +--rw truststore-reference?
                |     |     |  |  |                 ts:public-key-bag-r\
   \ef
                |     |     |  |  +--rw psks! {psk-auth}?
                |     |     |  +--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 peer-allowed-to-send?   empty
                |     |     |     +--rw test-peer-aliveness!
                |     |     |        +--rw max-wait?       uint16
                |     |     |        +--rw max-attempts?   uint8
                |     |     +--rw http-client-parameters
                |     |        +--rw client-identity!
                |     |           +--rw (auth-type)?
                |     |              +--:(basic)
                |     |                 +--rw basic {basic-auth}?
                |     |                    +--rw user-id     string
                |     |                    +--rw password    string
                |     +--rw restconf-client-parameters
                +--:(https) {https-listen}?
                   +--rw https
                      +--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 (auth-type)?
                      |  |     +--:(certificate)
                      |  |     |        {x509-certificate-auth}?
                      |  |     |  +--rw certificate
                      |  |     |     +--rw (local-or-keystore)
                      |  |     |        +--:(local)
                      |  |     |        |        {local-definitions-su\
   \pported}?
                      |  |     |        |  +--rw local-definition
                      |  |     |        |     +--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
```

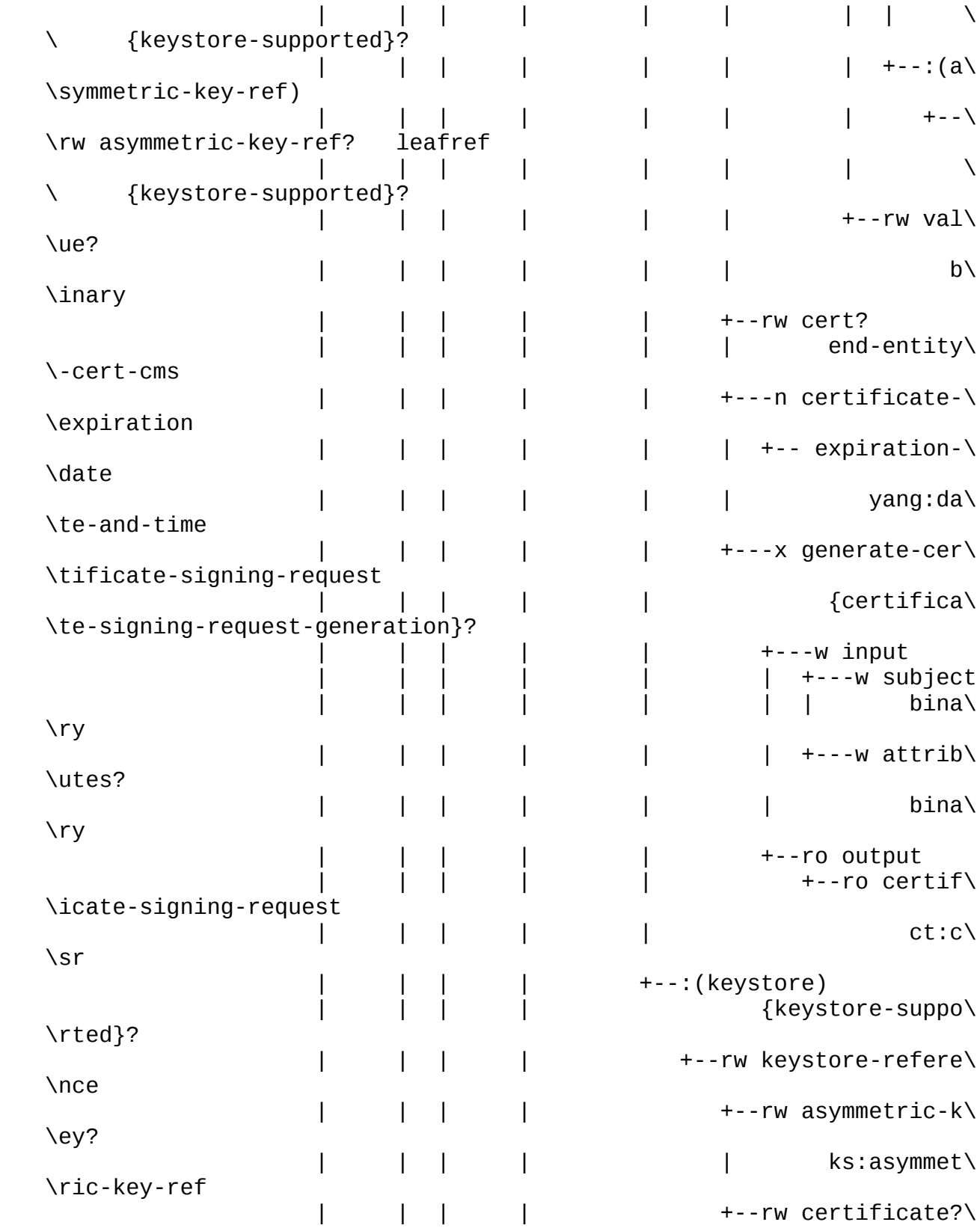
```
                        |  |     |        |     +---n certificate-expira\
   \tion
                        |  |     |        |     |  +-- expiration-date
                        |  |     |        |     |          yang:date-and\
   \-time
                        |  |     |        |     +---x generate-certifica\
   \te-signing-request
                        |  |     |        |              {certificate-sig\
   \ning-request-generation}?
                        |  |     |        |        +---w input
                        |  |     |        |        |  +---w subject
                        |  |     |        |        |  |       binary
                        |  |     |        |        |  +---w attributes?
                        |  |     |        |        |          binary
                        |  |     |        |        +--ro output
                        |  |     |        |           +--ro certificate-\
   \signing-request
                        |  |     |        |                    ct:csr
                        |  |     |        +--:(keystore)
                        |  |     |                    {keystore-supported}?
                        |  |     |           +--rw keystore-reference
                        |  |     |              +--rw asymmetric-key?
                        |  |     |              |       ks:asymmetric-ke\
   \y-ref
                        |  |     |              +--rw certificate?      \
   \leafref
                        |  |     +--:(raw-public-key)
                        |  |     |        {raw-public-key-auth}?
                        |  |     |  +--rw raw-private-key
                        |  |     |     +--rw (local-or-keystore)
                        |  |     |        +--:(local)
                        |  |     |        |        {local-definitions-su\
   \pported}?
                        |  |     |        |  +--rw local-definition
                        |  |     |        |     +--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
                        |  |     +--:(psk) {psk-auth}?
                        |  |        +--rw psk
                        |  |           +--rw (local-or-keystore)
                        |  |              +--:(local)
                        |  |              |        {local-definitions-su\
   \pported}?
                        |  |              |  +--rw local-definition
                        |  |              |     +--rw key-format?
                        |  |              |     |       identityref
                        |  |              |     +--rw (key-type)
                        |  |              |     |  +--:(key)
                        |  |              |     |  |  +--rw key?
                        |  |              |     |  |          binary
                        |  |              |     |  +--:(hidden-key)
                        |  |              |     |  |  +--rw hidden-key?
                        |  |              |     |  |          empty
                        |  |              |     |  +--:(encrypted-key)
                        |  |              |     |     +--rw encrypted-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 id?
                        |  |              |             string
                        |  |              |             {ks:local-defini\
   \tions-supported}?
                        |  |              +--:(keystore)
                        |  |                       {keystore-supported}?
                        |  |                    +--rw keystore-reference?
                        |  |                            ks:symmetric-key-ref
                        |  +--rw server-authentication
                        |  |  +--rw ca-certs! {x509-certificate-auth}?
                        |  |  |  +--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,certi\
   \ficates}?
                        |  |  |        +--rw truststore-reference?
                        |  |  |                 ts:certificate-bag-ref
                        |  |  +--rw ee-certs! {x509-certificate-auth}?
                        |  |  |  +--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,certi\
```

```
   \ficates}?
                        |  |  |        +--rw truststore-reference?
                        |  |  |                 ts:certificate-bag-ref
                        |  |  +--rw raw-public-keys!
                        |  |  |       {raw-public-key-auth}?
                        |  |  |  +--rw (local-or-truststore)
                        |  |  |     +--:(local)
                        |  |  |     |        {local-definitions-supporte\
   \d}?
                        |  |  |     |  +--rw local-definition
                        |  |  |     |     +--rw public-key* [name]
                        |  |  |     |        +--rw name
                        |  |  |     |        |       string
                        |  |  |     |        +--rw public-key-format
                        |  |  |     |        |       identityref
                        |  |  |     |        +--rw public-key
                        |  |  |     |                 binary
                        |  |  |     +--:(truststore)
                        |  |  |              {truststore-supported,publi\
   \c-keys}?
                        |  |  |        +--rw truststore-reference?
                        |  |  |                 ts:public-key-bag-ref
                        |  |  +--rw psks! {psk-auth}?
                        |  +--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 peer-allowed-to-send?   empty
                        |     +--rw test-peer-aliveness!
                        |        +--rw max-wait?       uint16
                        |        +--rw max-attempts?   uint8
                        +--rw http-client-parameters
                        |  +--rw client-identity!
                        |  |  +--rw (auth-type)?
                        |  |     +--:(basic)
                        |  |        +--rw basic {basic-auth}?
                        |  |           +--rw user-id     string
                        |  |           +--rw password    string
                        |  +--rw proxy-server! {proxy-connect}?
                        |     +--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 (auth-type)?
                        |     |  |     +--:(certificate)
                        |     |  |     |        {x509-certificate-auth}?
                        |     |  |     |  +--rw certificate
                        |     |  |     |     +--rw (local-or-keystore)
                        |     |  |     |        +--:(local)
                        |     |  |     |        |        {local-definiti\
   \ons-supported}?
                        |     |  |     |        |  +--rw local-definition
                        |     |  |     |        |     +--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}?
rw val\
ue?
b\
inary
+--rw cert?
end-entity\
-cert-cms
+---n certificate-\
expiration
+-- expiration-\
date
yang:da\
te-and-time
+---x generate-cer\
tificate-signing-request
{certifica\
te-signing-request-generation}?
+---w input
+---w subject
bina\
ry
+---w attrib\
utes?
bina\
ry
+--ro output
+--ro certif\
icate-signing-request
ct:c\
sr
+--:(keystore)
{keystore-suppo\
rted}?
+--rw keystore-refere\
nce
+--rw asymmetric-k\
ey?
ks:asymmet\
ric-key-ref
+--rw certificate?\

```
   \      leafref
                        |     |  |     +--:(raw-public-key)
                        |     |  |     |        {raw-public-key-auth}?
                        |     |  |     |  +--rw raw-private-key
                        |     |  |     |     +--rw (local-or-keystore)
                        |     |  |     |        +--:(local)
                        |     |  |     |        |        {local-definiti\
   \ons-supported}?
                        |     |  |     |        |  +--rw local-definition
                        |     |  |     |        |     +--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
                     |     |  |     +--:(psk) {psk-auth}?
                     |     |  |        +--rw psk
                     |     |  |           +--rw (local-or-keystore)
                     |     |  |              +--:(local)
                     |     |  |              |        {local-definiti\
   \ons-supported}?
                     |     |  |              |  +--rw local-definition
                     |     |  |              |     +--rw key-format?
                     |     |  |              |     |       identityref
                     |     |  |              |     +--rw (key-type)
                     |     |  |              |     |  +--:(key)
                     |     |  |              |     |  |  +--rw key?
                     |     |  |              |     |  |          bina\
   \ry
                     |     |  |              |     |  +--:(hidden-key)
                     |     |  |              |     |  |  +--rw hidden\
   \-key?
                     |     |  |              |     |  |          empty
                     |     |  |              |     |  +--:(encrypted-\
   \key)
                     |     |  |              |     |     +--rw encryp\
   \ted-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
                     |     |  |              |     +--rw id?
                     |     |  |              |             string
                     |     |  |              |             {ks:local-\
   \definitions-supported}?
                     |     |  |              +--:(keystore)
                     |     |  |                       {keystore-suppo\
   \rted}?
                     |     |  |                  +--rw keystore-refere\
   \nce?
                     |     |  |                       ks:symmetric-\
   \key-ref
                     |     |  +--rw server-authentication
                     |     |  |  +--rw ca-certs!
                     |     |  |  |       {x509-certificate-auth}?
                     |     |  |  |  +--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\
   \,certificates}?
                     |     |  |  |        +--rw truststore-reference?
                     |     |  |  |                 ts:certificate-bag-\
   \ref
                     |     |  |  +--rw ee-certs!
                     |     |  |  |       {x509-certificate-auth}?
                     |     |  |  |  +--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\
   \,certificates}?
                     |     |  |  |        +--rw truststore-reference?
                     |     |  |  |                 ts:certificate-bag-\
   \ref
                     |     |  |  +--rw raw-public-keys!
                     |     |  |  |       {raw-public-key-auth}?
                     |     |  |  |  +--rw (local-or-truststore)
                     |     |  |  |     +--:(local)
                     |     |  |  |     |        {local-definitions-su\
   \pported}?
                     |     |  |  |     |  +--rw local-definition
                     |     |  |  |     |     +--rw public-key* [name]
                     |     |  |  |     |        +--rw name
                     |     |  |  |     |        |       string
                     |     |  |  |     |        +--rw public-key-form\
   \at
                     |     |  |  |     |        |       identityref
                     |     |  |  |     |        +--rw public-key
                     |     |  |  |     |                 binary
                     |     |  |  |     +--:(truststore)
                     |     |  |  |              {truststore-supported\
   \,public-keys}?
                     |     |  |  |        +--rw truststore-reference?
                     |     |  |  |                 ts:public-key-bag-r\
   \ef
                     |     |  |  +--rw psks! {psk-auth}?
                     |     |  +--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 peer-allowed-to-send?   empty
                     |     |     +--rw test-peer-aliveness!
                     |     |        +--rw max-wait?       uint16
                     |     |        +--rw max-attempts?   uint8
                     |     +--rw http-client-parameters
                     |        +--rw client-identity!
                     |           +--rw (auth-type)?
                     |              +--:(basic)
```

```
                        |                 +--rw basic {basic-auth}?
                        |                    +--rw user-id     string
                        |                    +--rw password    string
                        +--rw restconf-client-parameters
```

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

The following tree diagram [RFC8340] provides an overview of the data model for the "ietf-restconf-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 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-restconf-server
  +--rw restconf-server
     +--rw listen! {http-listen or https-listen}?
     |  +--rw endpoint* [name]
     |     +--rw name              string
     |     +--rw (transport)
     |        +--:(http) {http-listen}?
     |        |  +--rw http
     |        |     +--rw external-endpoint!
     |        |     |  +--rw address    inet:ip-address
     |        |     |  +--rw port?      inet:port-number
     |        |     +--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 http-server-parameters
     |        |     |  +--rw server-name?             string
     |        |     |  +--rw client-authentication!
     |        |     |           {client-auth-config-supported}?
     |        |     |     +--rw users
     |        |     |        +--rw user* [user-id]
     |        |     |           +--rw user-id        string
     |        |     |           +--rw (auth-type)?
     |        |     |              +--:(basic)
     |        |     |                 +--rw basic {basic-auth}?
     |        |     |                    +--rw user-id?    string
     |        |     |                    +--rw password?
```

```
         |        |     |                              ianach:crypt-hash
         |        |     +--rw restconf-server-parameters
         |        |        +--rw client-identity-mappings
         |        |           +--rw cert-to-name* [id]
         |        |              +--rw id             uint32
         |        |              +--rw fingerprint?
         |        |              |       x509c2n:tls-fingerprint
         |        |              +--rw map-type       identityref
         |        |              +--rw name           string
         |        +--:(https) {https-listen}?
         |           +--rw https
         |              +--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 (auth-type)
         |              |  |     +--:(certificate)
         |              |  |     |        {x509-certificate-auth}?
         |              |  |     |  +--rw certificate
         |              |  |     |     +--rw (local-or-keystore)
         |              |  |     |        +--:(local)
         |              |  |     |        |        {local-definitions-su\
   \pported}?
         |              |  |     |        |  +--rw local-definition
         |              |  |     |        |     +--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
      |              |  |     |        |             {certificate-sig\
   \ning-request-generation}?
      |              |  |     |        |        +---w input
      |              |  |     |        |        |  +---w subject
      |              |  |     |        |        |  |       binary
      |              |  |     |        |        |  +---w attributes?
      |              |  |     |        |        |          binary
      |              |  |     |        |        +--ro output
      |              |  |     |        |           +--ro certificate-\
   \signing-request
      |              |  |     |        |                    ct:csr
      |              |  |     |        +--:(keystore)
      |              |  |     |                 {keystore-supported}?
      |              |  |     |           +--rw keystore-reference
      |              |  |     |              +--rw asymmetric-key?
      |              |  |     |              |       ks:asymmetric-ke\
   \y-ref
      |              |  |     |              +--rw certificate?      \
   \leafref
      |              |  |     +--:(raw-private-key)
      |              |  |     |        {raw-public-key-auth}?
      |              |  |     |  +--rw raw-private-key
      |              |  |     |     +--rw (local-or-keystore)
```

```
      |              |  |     |        +--:(local)
      |              |  |     |        |        {local-definitions-su\
   \pported}?
      |              |  |     |        |  +--rw local-definition
      |              |  |     |        |     +--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
      |              |  |     +--:(psk) {psk-auth}?
      |              |  |        +--rw psk
      |              |  |           +--rw (local-or-keystore)
      |              |  |              +--:(local)
      |              |  |              |        {local-definitions-su\
```

```
   \pported}?
        |              |  |              |  +--rw local-definition
        |              |  |              |     +--rw key-format?
        |              |  |              |     |       identityref
        |              |  |              |     +--rw (key-type)
        |              |  |              |     |  +--:(key)
        |              |  |              |     |  |  +--rw key?
        |              |  |              |     |  |          binary
        |              |  |              |     |  +--:(hidden-key)
        |              |  |              |     |  |  +--rw hidden-key?
        |              |  |              |     |  |          empty
        |              |  |              |     |  +--:(encrypted-key)
        |              |  |              |     |     +--rw encrypted-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 id?
        |              |  |              |             string
        |              |  |              |             {ks:local-defini\
   \tions-supported}?
        |              |  |              +--:(keystore)
        |              |  |                       {keystore-supported}?
        |              |  |                 +--rw keystore-reference?
        |              |  |                         ks:symmetric-key-ref
        |              |  +--rw client-authentication!
        |              |  |       {client-auth-config-supported}?
        |              |  |  +--rw ca-certs! {x509-certificate-auth}?
        |              |  |  |  +--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,certi\
   \ficates}?
      |              |  |  |        +--rw truststore-reference?
      |              |  |  |                 ts:certificate-bag-ref
      |              |  |  +--rw ee-certs! {x509-certificate-auth}?
      |              |  |  |  +--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,certi\
   \ficates}?
      |              |  |  |        +--rw truststore-reference?
      |              |  |  |                 ts:certificate-bag-ref
      |              |  |  +--rw raw-public-keys!
      |              |  |  |       {raw-public-key-auth}?
      |              |  |  |  +--rw (local-or-truststore)
      |              |  |  |     +--:(local)
      |              |  |  |     |        {local-definitions-supporte\
   \d}?
      |              |  |  |     |  +--rw local-definition
      |              |  |  |     |     +--rw public-key* [name]
      |              |  |  |     |        +--rw name
      |              |  |  |     |        |       string
      |              |  |  |     |        +--rw public-key-format
      |              |  |  |     |        |       identityref
      |              |  |  |     |        +--rw public-key
      |              |  |  |     |                binary
      |              |  |  |     +--:(truststore)
      |              |  |  |              {truststore-supported,publi\
   \c-keys}?
      |              |  |  |        +--rw truststore-reference?
      |              |  |  |                 ts:public-key-bag-ref
      |              |  |  +--rw psks! {psk-auth}?
      |              |  +--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 peer-allowed-to-send?   empty
        |              |     +--rw test-peer-aliveness!
        |              |        +--rw max-wait?       uint16
        |              |        +--rw max-attempts?   uint8
        |              +--rw http-server-parameters
        |              |  +--rw server-name?             string
        |              |  +--rw client-authentication!
        |              |           {client-auth-config-supported}?
        |              |     +--rw users
        |              |        +--rw user* [user-id]
        |              |           +--rw user-id        string
        |              |           +--rw (auth-type)?
        |              |              +--:(basic)
        |              |                 +--rw basic {basic-auth}?
        |              |                    +--rw user-id?    string
        |              |                    +--rw password?
        |              |                            ianach:crypt-hash
        |              +--rw restconf-server-parameters
        |                 +--rw client-identity-mappings
        |                    +--rw cert-to-name* [id]
        |                       +--rw id             uint32
        |                       +--rw fingerprint?
        |                       |       x509c2n:tls-fingerprint
        |                       +--rw map-type       identityref
        |                       +--rw name           string
        +--rw call-home! {https-call-home}?
           +--rw restconf-client* [name]
              +--rw name                   string
              +--rw endpoints
              |  +--rw endpoint* [name]
              |     +--rw name           string
              |     +--rw (transport)
              |        +--:(https) {https-listen}?
              |           +--rw https
              |              +--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 (auth-type)
               |              |  |     +--:(certificate)
               |              |  |     |        {x509-certificate-auth}?
               |              |  |     |  +--rw certificate
               |              |  |     |     +--rw (local-or-keystore)
               |              |  |     |        +--:(local)
               |              |  |     |        |        {local-definiti\
   \ons-supported}?
               |              |  |     |        |  +--rw local-definition
               |              |  |     |        |     +--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
              |               |  |     |        |     +--rw cert?
              |               |  |     |        |     |       end-entity\
   \-cert-cms
              |               |  |     |        |     +---n certificate-\
   \expiration
              |               |  |     |        |     |  +-- expiration-\
   \date
              |               |  |     |        |     |          yang:da\
   \te-and-time
              |               |  |     |        |     +---x generate-cer\
   \tificate-signing-request
              |               |  |     |        |             {certifica\
   \te-signing-request-generation}?
              |               |  |     |        |        +---w input
              |               |  |     |        |        |  +---w subject
              |               |  |     |        |        |  |       bina\
   \ry
              |               |  |     |        |        |  +---w attrib\
   \utes?
              |               |  |     |        |        |          bina\
   \ry
              |               |  |     |        |        +--ro output
              |               |  |     |        |           +--ro certif\
   \icate-signing-request
              |               |  |     |        |                   ct:c\
   \sr
              |               |  |     |        +--:(keystore)
              |               |  |     |                 {keystore-suppo\
   \rted}?
              |               |  |     |           +--rw keystore-refere\
   \nce
              |               |  |     |              +--rw asymmetric-k\
   \ey?
              |               |  |     |              |       ks:asymmet\
   \ric-key-ref
              |               |  |     |              +--rw certificate?\
   \      leafref
              |               |  |     +--:(raw-private-key)
              |               |  |     |        {raw-public-key-auth}?
              |               |  |     |  +--rw raw-private-key
              |               |  |     |     +--rw (local-or-keystore)
              |               |  |     |        +--:(local)
              |               |  |     |        |        {local-definiti\
```

```
   \ons-supported}?
              |                |  |     |        |  +--rw local-definition
              |                |  |     |        |     +--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
            |              |  |     +--:(psk) {psk-auth}?
            |              |  |        +--rw psk
            |              |  |           +--rw (local-or-keystore)
            |              |  |              +--:(local)
            |              |  |              |        {local-definiti\
   \ons-supported}?
            |              |  |              |  +--rw local-definition
            |              |  |              |     +--rw key-format?
            |              |  |              |     |       identityref
            |              |  |              |     +--rw (key-type)
            |              |  |              |     |  +--:(key)
            |              |  |              |     |  |  +--rw key?
            |              |  |              |     |  |          bina\
   \ry
            |              |  |              |     |  +--:(hidden-key)
            |              |  |              |     |  |  +--rw hidden\
   \-key?
            |              |  |              |     |  |          empty
            |              |  |              |     |  +--:(encrypted-\
   \key)
            |              |  |              |     |     +--rw encryp\
   \ted-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
            |              |  |              |     +--rw id?
            |              |  |              |             string
            |              |  |              |             {ks:local-\
   \definitions-supported}?
```

```
              |               |  |              +--:(keystore)
              |               |  |                       {keystore-suppo\
   \rted}?
              |               |  |                  +--rw keystore-refere\
   \nce?
              |               |  |                        ks:symmetric-\
   \key-ref
              |               |  +--rw client-authentication!
              |               |  |       {client-auth-config-supported}?
              |               |  |  +--rw ca-certs!
              |               |  |  |       {x509-certificate-auth}?
              |               |  |  |  +--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\
   \,certificates}?
              |               |  |  |        +--rw truststore-reference?
              |               |  |  |                ts:certificate-bag-\
   \ref
              |               |  |  +--rw ee-certs!
              |               |  |  |       {x509-certificate-auth}?
              |               |  |  |  +--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\
   \,certificates}?
```

```
              |              |  |  |       +--rw truststore-reference?
              |              |  |  |               ts:certificate-bag-\
   \ref
              |              |  |  +--rw raw-public-keys!
              |              |  |  |       {raw-public-key-auth}?
              |              |  |  |  +--rw (local-or-truststore)
              |              |  |  |     +--:(local)
              |              |  |  |     |        {local-definitions-su\
   \pported}?
              |              |  |  |     |  +--rw local-definition
              |              |  |  |     |     +--rw public-key* [name]
              |              |  |  |     |        +--rw name
              |              |  |  |     |        |       string
              |              |  |  |     |        +--rw public-key-form\
   \at
              |              |  |  |     |        |       identityref
              |              |  |  |     |        +--rw public-key
              |              |  |  |     |                binary
              |              |  |  |     +--:(truststore)
              |              |  |  |              {truststore-supported\
   \,public-keys}?
              |              |  |  |        +--rw truststore-reference?
              |              |  |  |                ts:public-key-bag-r\
   \ef
              |              |  |  +--rw psks! {psk-auth}?
              |              |  +--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 peer-allowed-to-send?   empty
              |              |     +--rw test-peer-aliveness!
              |              |        +--rw max-wait?       uint16
              |              |        +--rw max-attempts?   uint8
              |             +--rw http-server-parameters
              |             |  +--rw server-name?             string
              |             |  +--rw client-authentication!
              |             |           {client-auth-config-supported}?
              |             |     +--rw users
              |             |        +--rw user* [user-id]
              |             |           +--rw user-id        string
              |             |           +--rw (auth-type)?
              |             |              +--:(basic)
              |             |                 +--rw basic {basic-auth}?
```

```
               |              |                    +--rw user-id?
               |              |                    |       string
               |              |                    +--rw password?
               |              |                            ianach:crypt-\
   \hash
               |              +--rw restconf-server-parameters
               |                 +--rw client-identity-mappings
               |                    +--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

- Filled in previously missing 'ietf-restconf-client' module.
- Updated the ietf-restconf-server module to accommodate new grouping 'ietf-tls-server-grouping'.

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 restconf-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-restconf-client to add a mandatory true.
- Added refine statement in ietf-restconf-server to add a must statement.
- Now there are containers and groupings, for both the client and server models.
- Now tree diagrams reference ietf-netmod-yang-tree-diagrams
- Updated examples to inline key and certificates (no longer a leafref to keystore)

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 '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 better describe behavior when connections are not closed gracefully.
- 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.//)

- In ietf-restconf-server, Added 'http-listen' (not https-listen) choice, to support case when server is behind a TLS-terminator.

- Refactored server module to be more like other 'server' models. If folks like it, will also apply to the client model, as well as to both the netconf client/server models. Now the 'restconf-server-grouping' is just the RC-specific bits (i.e., the "demux" container minus the container), 'restconf-server-[listen|callhome]-stack-grouping' is the protocol stack for a single connection, and 'restconf-server-app-grouping' is effectively what was before (both listen+callhome for many inbound/outbound endpoints).

B.14. 13 to 14

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

- Adjusting from change in TLS client model (removing the top-level 'certificate' container).

- Added "external-endpoint" to the "http-listen" choice in ietf-restconf-server.

B.15. 14 to 15

- Added missing "or https-listen" clause in a "must" expression.

- Refactored the client module similar to how the server module was refactored in -13. Now the 'restconf-client-grouping' is just the RC-specific bits, the 'restconf-client-[initiate|listen]-stack-grouping' is the protocol stack for a single connection, and 'restconf-client-app-grouping' is effectively what was before (both listen+callhome for many inbound/outbound endpoints).

B.16. 15 to 16

- Added refinement to make "cert-to-name/fingerprint" be mandatory false.
- Commented out refinement to "tls-server-grouping/client-authentication" until a better "must" expression is defined.
- Updated restconf-client example to reflect that http-client-grouping no longer has a "protocol-version" leaf.

B.17. 16 to 17

- Updated examples to include the "*-key-format" nodes.
- Updated examples to remove the "required" nodes.

B.18. 17 to 18

- Updated examples to reflect new "bag" addition to truststore.

B.19. 18 to 19

- Updated examples to remove the 'algorithm' nodes.
- Updated examples to reflect the new TLS keepalives structure.
- Removed the 'protocol-versions' node from the restconf-server examples.
- Added a "Note to Reviewers" note to first page.

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, Mehmet Ersue, Ramkumar Dhanapal, Balazs Kovacs, Radek Krejci, David Lamparter, Ladislav Lhotka, Alan Luchuk, Tom Petch, Juergen Schoenwaelder, Phil Shafer, Sean Turner, Bert Wijnen.

Author's Address

Kent Watsen
Watsen Networks

EMail: kent+ietf@watsen.net