Stream: Internet Engineering Task Force (IETF)
RFC: 9643
Category: Standards Track
Published: August 2024
ISSN: 2070-1721
Authors: K. Watsen *Watsen Networks* M. Scharf *Hochschule Esslingen*

RFC 9643
YANG Groupings for TCP Clients and TCP Servers

Abstract

This document presents three YANG 1.1 modules to support the configuration of TCP clients and TCP servers. The modules include basic parameters of a TCP connection relevant for client or server applications, as well as client configuration required for traversing proxies. The modules can be used either standalone or in conjunction with configuration of other stack protocol layers.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

This document defines three YANG 1.1 [RFC7950] modules to support the configuration of TCP clients and TCP servers (TCP is defined in [RFC9293]), either as standalone or in conjunction with configuration of other stack protocol layers.

The modules focus on three different types of base TCP parameters that matter for TCP-based applications: First, the modules cover fundamental configuration of a TCP client or TCP server application, such as addresses and port numbers. Second, a reusable grouping enables modification of application-specific parameters for TCP connections, such as use of TCP keep-alives. And third, client configuration for traversing proxies is included as well. In each case, the modules have a very narrow scope and focus on a minimum set of required parameters.

Please be advised that while this document presents support for some TCP proxy techniques, there are other TCP proxy techniques that are not part of this document but could be added by augmenting the YANG module.

1.1. Relation to Other RFCs

This document presents three YANG modules [RFC7950] that are part of a collection of RFCs that work together to ultimately support the configuration of both the clients and servers of both the Network Configuration Protocol (NETCONF) [RFC6241] and RESTCONF [RFC8040].

The dependency relationship between the primary YANG groupings defined in the various RFCs is presented in the below diagram. In some cases, a document may define secondary groupings that introduce dependencies not illustrated in the diagram. The labels in the diagram are shorthand names for the defining RFCs. The citation references for shorthand names are provided below the diagram.

Please note that the arrows in the diagram point from referencer to referenced. For example, the "crypto-types" RFC does not have any dependencies, whilst the "keystore" RFC depends on the "crypto-types" RFC.

```
                               crypto-types
                                 ^      ^
                                /        \
                               /          \
                      truststore         keystore
                       ^     ^             ^  ^
                       |     +---------+   |  |
                       |               |   |  |
                       |      +------------+  |
tcp-client-server      |     /         |      |
   ^    ^        ssh-client-server     |      |
   |    |           ^            tls-client-server
   |    |           |              ^     ^        http-client-server
   |    |           |              |     |                 ^
   |    |           |        +-----+     +---------+       |
   |    |           |        |                     |       |
   |    +-----------|--------|--------------+      |       |
   |                |        |              |      |       |
   +-----------+    |        |              |      |       |
               |    |        |              |      |       |
               |    |        |              |      |       |
            netconf-client-server       restconf-client-server
```

Label in Diagram	Reference
crypto-types	[RFC9640]
truststore	[RFC9641]
keystore	[RFC9642]
tcp-client-server	RFC 9643
ssh-client-server	[RFC9644]
tls-client-server	[RFC9645]
http-client-server	[HTTP-CLIENT-SERVER]
netconf-client-server	[NETCONF-CLIENT-SERVER]
restconf-client-server	[RESTCONF-CLIENT-SERVER]

Table 1: Label in Diagram to RFC Mapping

1.2. Specification Language

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

1.3. Adherence to the NMDA

This document is compliant with the Network Management Datastore Architecture (NMDA) [RFC8342]. It does not define any protocol accessible nodes that are "config false".

2. The "ietf-tcp-common" Module

This section defines a YANG 1.1 module called "ietf-tcp-common". A high-level overview of the module is provided in Section 2.1. Examples illustrating the module's use are provided in Section 2.2 ("Example Usage"). The YANG module itself is defined in Section 2.3.

2.1. Data Model Overview

This section provides an overview of the "ietf-tcp-common" module in terms of its features and groupings.

2.1.1. Model Scope

This document presents a common "grouping" statement for basic TCP connection parameters that matter to applications. It is "common" in that this grouping is used by both the "ietf-tcp-client" and "ietf-tcp-server" modules. In some TCP stacks, such parameters can also directly be set by an application using system calls, such as the sockets API. The base YANG data model in this document focuses on modeling TCP keep-alives. This base model can be extended as needed.

2.1.2. Features

The following diagram lists all the "feature" statements defined in the "ietf-tcp-common" module:

```
Features:
  +-- keepalives-supported
```

The diagram above uses syntax that is similar to but not defined in [RFC8340].

2.1.3. Groupings

The "ietf-tcp-common" module defines the following "grouping" statement:

- tcp-common-grouping

This grouping is presented in the following subsection.

2.1.3.1. The "tcp-common-grouping" Grouping

The following tree diagram [RFC8340] illustrates the "tcp-common-grouping" grouping:

```
  grouping tcp-common-grouping:
    +-- keepalives! {keepalives-supported}?
       +-- idle-time?        uint16
       +-- max-probes?       uint16
       +-- probe-interval?   uint16
```

Comments:

- The "keepalives" node is a "presence" container so that the mandatory descendant nodes do not imply that keep-alives must be configured.
- The "idle-time", "max-probes", and "probe-interval" nodes have the common meanings. Please see the YANG module in Section 2.3 for details.

2.1.4. Protocol-Accessible Nodes

The "ietf-tcp-common" module defines only "grouping" statements that are used by other modules to instantiate protocol-accessible nodes. Thus, this module, when implemented, does not itself define any protocol-accessible nodes.

2.1.5. Guidelines for Configuring TCP Keep-Alives

Network stacks may include keep-alives in their TCP implementations, although this practice is not universally implemented. If keep-alives are included, [RFC9293] mandates that the application **MUST** be able to turn them on or off for each TCP connection and that they **MUST** default to off.

Keep-alive mechanisms exist in many protocols. Depending on the protocol stack, TCP keep-alives may only be one out of several alternatives. Which mechanism(s) to use depends on the use case and application requirements. If keep-alives are needed by an application, it is **RECOMMENDED** that the liveness check happens only at the protocol layers that are meaningful to the application.

A TCP keep-alive mechanism **SHOULD** only be invoked in server applications that might otherwise hang indefinitely and consume resources unnecessarily if a client crashes or aborts a connection during a network failure [RFC9293]. TCP keep-alives may consume significant resources both in the network and in endpoints (e.g., battery power). In addition, frequent keep-alives risk network congestion. The higher the frequency of keep-alives, the higher the overhead.

Given the cost of keep-alives, parameters have to be configured carefully:

- The default idle interval (leaf "idle-time") is two hours, i.e., 7200 seconds [RFC9293]. A lower value **MAY** be configured, but idle intervals **SHOULD NOT** be smaller than 15 seconds. Longer idle intervals **SHOULD** be used when possible.
- The maximum number of sequential keep-alive probes that can fail (leaf "max-probes") trades off responsiveness and robustness against packet loss. ACK segments that contain no data are not reliably transmitted by TCP. Consequently, if a keep-alive mechanism is implemented, it **MUST NOT** interpret failure to respond to any specific probe as a dead connection [RFC9293]. Typically, a single-digit number should suffice.

- TCP implementations may include a parameter for the number of seconds between TCP keep-alive probes (leaf "probe-interval"). In order to avoid congestion, the time interval between probes **MUST NOT** be smaller than one second. Significantly longer intervals **SHOULD** be used. It is important to note that keep-alive probes (or replies) can get dropped due to network congestion. Sending further probe messages into a congested path after a short interval, without backing off timers, could cause harm and result in a congestion collapse. Therefore, it is essential to pick a large, conservative value for this interval.

2.2. Example Usage

This section presents an example showing the "tcp-common-grouping" grouping populated with some data.

```
<!-- The outermost element below doesn't exist in the data model. -->
<!--  It simulates if the "grouping" were a "container" instead.  -->

<tcp-common xmlns="urn:ietf:params:xml:ns:yang:ietf-tcp-common">
  <keepalives>
    <idle-time>7200</idle-time>
    <max-probes>9</max-probes>
    <probe-interval>75</probe-interval>
  </keepalives>
</tcp-common>
```

2.3. YANG Module

The "ietf-tcp-common" YANG module references [RFC6991] and [RFC9293].

```
<CODE BEGINS> file "ietf-tcp-common@2024-04-04.yang"

module ietf-tcp-common {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-tcp-common";
  prefix tcpcmn;

  organization
    "IETF NETCONF (Network Configuration) Working Group and the
     IETF TCP Maintenance and Minor Extensions (TCPM) Working Group";

  contact
    "WG Web:   https://datatracker.ietf.org/wg/netconf
               https://datatracker.ietf.org/wg/tcpm
     WG List:  NETCONF WG list <mailto:netconf@ietf.org>
               TCPM WG list <mailto:tcpm@ietf.org>
     Authors:  Kent Watsen <mailto:kent+ietf@watsen.net>
               Michael Scharf
               <mailto:michael.scharf@hs-esslingen.de>";

  description
    "This module define a reusable 'grouping' that is common
     to both TCP clients and TCP servers.  This grouping statement
     is used by both the 'ietf-tcp-client' and 'ietf-tcp-server'
     modules.

     The key words 'MUST', 'MUST NOT', 'REQUIRED', 'SHALL',
     'SHALL NOT', 'SHOULD', 'SHOULD NOT', 'RECOMMENDED',
     'NOT RECOMMENDED', 'MAY', and 'OPTIONAL' in this document
     are to be interpreted as described in BCP 14 (RFC 2119)
     (RFC 8174) when, and only when, they appear in all
     capitals, as shown here.

     Copyright (c) 2024 IETF Trust and the persons identified
     as authors of the code.  All rights reserved.

     Redistribution and use in source and binary forms, with
     or without modification, is permitted pursuant to, and
     subject to the license terms contained in, the Revised
     BSD License set forth in Section 4.c of the IETF Trust's
     Legal Provisions Relating to IETF Documents
     (https://trustee.ietf.org/license-info).

     This version of this YANG module is part of RFC 9643
     (https://www.rfc-editor.org/info/rfc9643); see the RFC
     itself for full legal notices.";

  revision 2024-04-04 {
    description
      "Initial version.";
    reference
      "RFC 9643: YANG Groupings for TCP Clients and TCP Servers";
  }

  // Features

  feature keepalives-supported {
    description
      "Indicates that keepalives are supported.";
  }

  // Groupings

  grouping tcp-common-grouping {
    description
      "A reusable grouping for configuring TCP parameters common
       to TCP connections as well as the operating system as a
       whole.";
    container keepalives {
      if-feature "keepalives-supported";
      presence "Indicates that keepalives are enabled, aligning to
                the requirement in Section 3.8.4 of RFC 9293 that
                states keepalives are off by default.";
      description
        "Configures the keepalive policy to proactively test the
         aliveness of the TCP peer.  An unresponsive TCP peer is
         dropped after approximately (idle-time + max-probes *
         probe-interval) seconds.  Further guidance can be found
         in Section 2.1.5 of RFC 9643.";
      reference
        "RFC 9293: Transmission Control Protocol (TCP)";
      leaf idle-time {
        type uint16 {
          range "1..max";
        }
        units "seconds";
        default "7200";
        description
          "Sets the amount of time after which a TCP-level probe
           message will be sent to test the aliveness of the TCP
           peer if no data has been received from the TCP peer.
           Two hours (7200 seconds) is a safe value, per Section
           3.8.4 of RFC 9293.";
        reference
          "RFC 9293: Transmission Control Protocol (TCP)";
      }
      leaf max-probes {
        type uint16 {
          range "1..max";
        }
        default "9";
        description
          "Sets the maximum number of sequential keepalive probes
           that can fail to obtain a response from the TCP peer
           before assuming the TCP peer is no longer alive.";
      }
      leaf probe-interval {
        type uint16 {
          range "1..max";
        }
        units "seconds";
        default "75";
        description
          "Sets the time interval between failed probes.  The
           interval SHOULD be significantly longer than one second
           in order to avoid harm on a congested link.";
      }
    } // container keepalives
  } // grouping tcp-common-grouping

}

<CODE ENDS>
```

3. The "ietf-tcp-client" Module

This section defines a YANG 1.1 module called "ietf-tcp-client". A high-level overview of the module is provided in Section 3.1. Examples illustrating the module's use are provided in Section 3.2 ("Example Usage"). The YANG module itself is defined in Section 3.3.

3.1. Data Model Overview

This section provides an overview of the "ietf-tcp-client" module in terms of its features and groupings.

3.1.1. Features

The following diagram lists all the "feature" statements defined in the "ietf-tcp-client" module:

```
Features:
  +-- local-binding-supported
  +-- tcp-client-keepalives
  +-- proxy-connect
      +-- socks4-supported {proxy-connect}?
      +-- socks4a-supported {proxy-connect}?
      +-- socks5-supported {proxy-connect}?
          +-- socks5-gss-api {socks5-supported}?
          +-- socks5-username-password {socks5-supported}?
```

Comments:

- The "local-binding-supported" feature indicates that the server supports configuring local bindings (i.e., the local address and local port) for TCP clients.
- The "tcp-client-keepalives" feature indicates that per socket TCP keepalive parameters are configurable for TCP clients on the server implementing this feature.
- The "proxy-connect" feature indicates the TCP client supports connecting through TCP proxies.
- The "socks4-supported" feature indicates the TCP client supports Socks4-based proxies.
- The "socks4a-supported" feature indicates the TCP client supports Socks4a-based proxies. The difference between Socks4 and Socks4a is that Socks4a enables the "remote-address" to be specified using a hostname, in addition to an IP address.
- The "socks5-supported" feature indicates the TCP client supports Socks5-based proxies.
- The "socks5-gss-api" feature indicates that the server, when acting as a TCP client, supports authenticating to a SOCKS Version 5 proxy server using Generic Security Service Application Program Interface (GSS-API) credentials.
- The "socks5-username-password" feature indicates that the server, when acting as a TCP client, supports authenticating to a SOCKS Version 5 proxy server using "username" and "password" credentials."

The diagram above uses syntax that is similar to but not defined in [RFC8340].

3.1.2. Groupings

The "ietf-tcp-client" module defines the following "grouping" statement:

- tcp-client-grouping

This grouping is presented in the following subsection.

3.1.2.1. The "tcp-client-grouping" Grouping

The following tree diagram [RFC8340] illustrates the "tcp-client-grouping" grouping:

```
  grouping tcp-client-grouping:
    +-- remote-address                inet:host
    +-- remote-port?                  inet:port-number
    +-- local-address?                inet:ip-address
    |       {local-binding-supported}?
    +-- local-port?                   inet:port-number
    |       {local-binding-supported}?
    +-- proxy-server! {proxy-connect}?
    |  +-- (proxy-type)
    |     +--:(socks4) {socks4-supported}?
    |     |  +-- socks4-parameters
    |     |     +-- remote-address    inet:ip-address
    |     |     +-- remote-port?      inet:port-number
    |     +--:(socks4a) {socks4a-supported}?
    |     |  +-- socks4a-parameters
    |     |     +-- remote-address    inet:host
    |     |     +-- remote-port?      inet:port-number
    |     +--:(socks5) {socks5-supported}?
    |        +-- socks5-parameters
    |           +-- remote-address               inet:host
    |           +-- remote-port?                 inet:port-number
    |           +-- authentication-parameters!
    |              +-- (auth-type)
    |                 +--:(gss-api) {socks5-gss-api}?
    |                 |  +-- gss-api
    |                 +--:(username-password)
    |                          {socks5-username-password}?
    |                    +-- username-password
    |                       +-- username                string
    |                       +---u ct:password-grouping
    +---u tcpcmn:tcp-common-grouping
```

Comments:

- The "remote-address" node, which is mandatory, may be configured as an IPv4 address, an IPv6 address, or a hostname.
- The "remote-port" node is not mandatory, but its default value is the invalid value "0", thus forcing the consuming data model to refine it in order to provide it an appropriate default value.
- The "local-address" node, which is enabled by the "local-binding-supported" feature (Section 2.1.2), may be configured as an IPv4 address, an IPv6 address, or a wildcard value.
- The "local-port" node, which is enabled by the "local-binding-supported" feature (Section 2.1.2), is not mandatory. Its default value is "0", indicating that the operating system can pick an arbitrary port number.
- The "proxy-server" node is enabled by a "feature" statement and, for servers that enable it, is a "presence" container so that the descendant "mandatory true" choice node does not imply that the proxy-server node must be configured. The proxy-server node uses a "choice" statement to allow one of several types of proxies to be configured. The choices presented in this document include Socks4, Socks4a, and Socks5, each enabled by a YANG feature (see Section 3.1.1). Other proxy types may be added by future work.

- This grouping uses the "password-grouping" grouping discussed in [RFC9640].
- This grouping uses the "tcp-common-grouping" grouping discussed in Section 2.1.3.1.

3.1.3. Protocol-Accessible Nodes

The "ietf-tcp-client" module defines only "grouping" statements that are used by other modules to instantiate protocol-accessible nodes. Thus, this module, when implemented, does not itself define any protocol-accessible nodes.

3.2. Example Usage

This section presents two examples showing the "tcp-client-grouping" grouping populated with some data. This example shows a TCP client configured to not connect via a proxy:

```
<!-- The outermost element below doesn't exist in the data model. -->
<!--  It simulates if the "grouping" were a "container" instead.  -->

<tcp-client xmlns="urn:ietf:params:xml:ns:yang:ietf-tcp-client">
  <remote-address>www.example.com</remote-address>
  <remote-port>8443</remote-port>
  <local-address>192.0.2.2</local-address>
  <local-port>12345</local-port>
  <keepalives>
    <idle-time>7200</idle-time>
    <max-probes>9</max-probes>
    <probe-interval>75</probe-interval>
  </keepalives>
</tcp-client>
```

This example shows a TCP client configured to connect via a proxy:

```
<!-- The outermost element below doesn't exist in the data model. -->
<!--  It simulates if the "grouping" were a "container" instead.  -->

<tcp-client xmlns="urn:ietf:params:xml:ns:yang:ietf-tcp-client">
  <remote-address>www.example.com</remote-address>
  <remote-port>8443</remote-port>
  <local-address>192.0.2.2</local-address>
  <local-port>12345</local-port>
  <proxy-server>
    <socks5-parameters>
      <remote-address>proxy.example.com</remote-address>
      <remote-port>1080</remote-port>
      <authentication-parameters>
        <username-password>
          <username>foobar</username>
          <cleartext-password>secret</cleartext-password>
        </username-password>
      </authentication-parameters>
    </socks5-parameters>
  </proxy-server>
  <keepalives>
    <idle-time>7200</idle-time>
    <max-probes>9</max-probes>
    <probe-interval>75</probe-interval>
  </keepalives>
</tcp-client>
```

3.3. YANG Module

The "ietf-tcp-client" YANG module references [SOCKS_4A], [RFC1928], [RFC1929], [RFC2743], [RFC6991], [RFC9293], and [RFC9640].

```
<CODE BEGINS> file "ietf-tcp-client@2024-04-04.yang"

module ietf-tcp-client {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-tcp-client";
  prefix tcpc;

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

  import ietf-crypto-types {
    prefix ct;
    reference
      "RFC 9640: YANG Data Types and Groupings for Cryptography";
  }

  import ietf-tcp-common {
    prefix tcpcmn;
    reference
```

```
       "RFC 9643: YANG Groupings for TCP Clients and TCP Servers";
   }

   organization
     "IETF NETCONF (Network Configuration) Working Group and the
      IETF TCP Maintenance and Minor Extensions (TCPM) Working Group";

   contact
     "WG Web:   https://datatracker.ietf.org/wg/netconf
                https://datatracker.ietf.org/wg/tcpm
      WG List:  NETCONF WG list <mailto:netconf@ietf.org>
                TCPM WG list <mailto:tcpm@ietf.org>
      Authors:  Kent Watsen <mailto:kent+ietf@watsen.net>
                Michael Scharf
                <mailto:michael.scharf@hs-esslingen.de>";

   description
     "This module defines reusable groupings for TCP clients that
      can be used as a basis for specific TCP client instances.

      The key words 'MUST', 'MUST NOT', 'REQUIRED', 'SHALL',
      'SHALL NOT', 'SHOULD', 'SHOULD NOT', 'RECOMMENDED',
      'NOT RECOMMENDED', 'MAY', and 'OPTIONAL' in this document
      are to be interpreted as described in BCP 14 (RFC 2119)
      (RFC 8174) when, and only when, they appear in all
      capitals, as shown here.

      Copyright (c) 2024 IETF Trust and the persons identified
      as authors of the code.  All rights reserved.

      Redistribution and use in source and binary forms, with
      or without modification, is permitted pursuant to, and
      subject to the license terms contained in, the Revised
      BSD License set forth in Section 4.c of the IETF Trust's
      Legal Provisions Relating to IETF Documents
      (https://trustee.ietf.org/license-info).

      This version of this YANG module is part of RFC 9643
      (https://www.rfc-editor.org/info/rfc9643); see the RFC
      itself for full legal notices.";

   revision 2024-04-04 {
     description
       "Initial version.";
     reference
       "RFC 9643: YANG Groupings for TCP Clients and TCP Servers";
   }

   // Features

   feature local-binding-supported {
     description
       "Indicates that the server supports configuring local
        bindings (i.e., the local address and local port) for
        TCP clients.";
   }

   feature tcp-client-keepalives {
     description
       "Per socket TCP keepalive parameters are configurable for
        TCP clients on the server implementing this feature.";
     reference
       "RFC 9293: Transmission Control Protocol (TCP)";
   }

   feature proxy-connect {
     description
       "Indicates the TCP client supports connecting through
        TCP proxies.";
   }

   feature socks4-supported {
     if-feature "proxy-connect";
     description
       "Indicates the TCP client supports Socks4-based proxies.";
     reference
       "SOCKS Proceedings: 1992 Usenix Security Symposium";
   }

   feature socks4a-supported {
     if-feature "proxy-connect";
     description
       "Indicates the TCP client supports Socks4a-based proxies.";
     reference
       "OpenSSH message:
          SOCKS 4A: A Simple Extension to SOCKS 4 Protocol
          <https://www.openssh.com/txt/socks4a.protocol>";
   }

   feature socks5-supported {
     if-feature "proxy-connect";
     description
       "Indicates the TCP client supports Socks5-based proxies.";
     reference
       "RFC 1928: SOCKS Protocol Version 5";
   }

   feature socks5-gss-api {
     if-feature "socks5-supported";
     description
       "Indicates that the server, when acting as a TCP client,
        supports authenticating to a SOCKS Version 5 proxy server
        using GSS-API credentials.";
     reference
       "RFC 1928: SOCKS Protocol Version 5";
   }

   feature socks5-username-password {
     if-feature "socks5-supported";
     description
       "Indicates that the server, when acting as a TCP client,
        supports authenticating to a SOCKS Version 5 proxy server
        using 'username' and 'password' credentials.";
     reference
       "RFC 1928: SOCKS Protocol Version 5";
   }
```

```
  // Groupings

  grouping tcp-client-grouping {
    description
      "A reusable grouping for configuring a TCP client.

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

    leaf remote-address {
      type inet:host;
      mandatory true;
      description
        "The IP address or hostname of the remote peer to
         establish a connection with.  If a domain name is
         configured, then the DNS resolution should happen on
         each connection attempt.  If the DNS resolution
         results in multiple IP addresses, the IP addresses
         are tried according to local preference order until
         a connection has been established or until all IP
         addresses have failed.";
    }
    leaf remote-port {
      type inet:port-number;
      default "0";
      description
        "The IP port number for the remote peer to establish a
         connection with.  An invalid default value is used
         so that importing modules may 'refine' it with the
         appropriate default port number value.";
    }
    leaf local-address {
      if-feature "local-binding-supported";
      type inet:ip-address;
      description
        "The local IP address/interface to bind to for when
         connecting to the remote peer.  INADDR_ANY ('0.0.0.0') or
         INADDR6_ANY ('0:0:0:0:0:0:0:0' a.k.a. '::') MAY be used to
         explicitly indicate the implicit default, which the server
         can bind to any IPv4 or IPv6 address.";
    }
    leaf local-port {
      if-feature "local-binding-supported";
      type inet:port-number;
      default "0";
      description
        "The local IP port number to bind to for when connecting
         to the remote peer.  The port number '0', which is the
         default value, indicates that any available local port
         number may be used.";
    }
```

```
      container proxy-server {
        if-feature "proxy-connect";
        presence "Indicates that a proxy connection has been
                  configured. Present so that the mandatory
                  descendant nodes do not imply that this node
                  must be configured.";
        choice proxy-type {
          mandatory true;
          description
            "Selects a proxy connection protocol.";
          case socks4 {
            if-feature "socks4-supported";
            container socks4-parameters {
              leaf remote-address {
                type inet:ip-address;
                mandatory true;
                description
                  "The IP address of the proxy server.";
              }
              leaf remote-port {
                type inet:port-number;
                default "1080";
                description
                  "The IP port number for the proxy server.";
              }
              description
                "Parameters for connecting to a TCP-based proxy
                 server using the SOCKS4 protocol.";
              reference
                "SOCKS Proceedings: 1992 Usenix Security Symposium";
            }
          }
          case socks4a {
            if-feature "socks4a-supported";
            container socks4a-parameters {
              leaf remote-address {
                type inet:host;
                mandatory true;
                description
                  "The IP address or hostname of the proxy server.";
              }
              leaf remote-port {
                type inet:port-number;
                default "1080";
                description
                  "The IP port number for the proxy server.";
              }
              description
                "Parameters for connecting to a TCP-based proxy
                 server using the SOCKS4a protocol.";
              reference
                "SOCKS Proceedings:
                   1992 Usenix Security Symposium
                 OpenSSH message:
                   SOCKS 4A: A Simple Extension to SOCKS 4 Protocol
                   <https://www.openssh.com/txt/socks4a.protocol>";
            }
          }
```

```
         case socks5 {
           if-feature "socks5-supported";
           container socks5-parameters {
             leaf remote-address {
               type inet:host;
               mandatory true;
               description
                 "The IP address or hostname of the proxy server.";
             }
             leaf remote-port {
               type inet:port-number;
               default "1080";
               description
                 "The IP port number for the proxy server.";
             }
             container authentication-parameters {
               presence "Indicates that an authentication mechanism
                         has been configured.  Present so that the
                         mandatory descendant nodes do not imply that
                         this node must be configured.";
               description
                 "A container for SOCKS Version 5 authentication
                  mechanisms.

                  A complete list of methods is defined at:
                  <https://www.iana.org/assignments/socks-methods>.";
               reference
                 "RFC 1928: SOCKS Protocol Version 5";
               choice auth-type {
                 mandatory true;
                 description
                   "A choice amongst supported SOCKS Version 5
                    authentication mechanisms.";
                 case gss-api {
                   if-feature "socks5-gss-api";
                   container gss-api {
                     description
                       "Contains GSS-API configuration.  Defines
                        as an empty container to enable specific
                        GSS-API configuration to be augmented in
                        by future modules.";
                     reference
                       "RFC 1928: SOCKS Protocol Version 5
                        RFC 2743: Generic Security Service
                                  Application Program Interface
                                  Version 2, Update 1";
                   }
                 }
                 case username-password {
                   if-feature "socks5-username-password";
                   container username-password {
                     leaf username {
                       type string;
                       mandatory true;
                       description
                         "The 'username' value to use for client
                          identification.";
                     }
```

```
                      uses ct:password-grouping {
                        description
                          "The password to be used for client
                           authentication.";
                      }
                      description
                        "Contains username/password configuration.";
                      reference
                        "RFC 1929: Username/Password Authentication
                                   for SOCKS V5";
                    }
                  }
                }
              }
              description
                "Parameters for connecting to a TCP-based proxy server
                 using the SOCKS5 protocol.";
              reference
                "RFC 1928: SOCKS Protocol Version 5";
            }
          }
        }
        description
          "Proxy server settings.";
      }

      uses tcpcmn:tcp-common-grouping {
        refine "keepalives" {
          if-feature "tcp-client-keepalives";
          description
            "An 'if-feature' statement so that implementations
             can choose to support TCP client keepalives.";
        }
      }
    }
  }

<CODE ENDS>
```

4. The "ietf-tcp-server" Module

This section defines a YANG 1.1 module called "ietf-tcp-server". A high-level overview of the module is provided in Section 4.1. Examples illustrating the module's use are provided in Section 4.2 ("Example Usage"). The YANG module itself is defined in Section 4.3.

4.1. Data Model Overview

This section provides an overview of the "ietf-tcp-server" module in terms of its features and groupings.

4.1.1. Features

The following diagram lists all the "feature" statements defined in the "ietf-tcp-server" module:

```
Features:
  +-- tcp-server-keepalives
```

The diagram above uses syntax that is similar to but not defined in [RFC8340].

4.1.2. Groupings

The "ietf-tcp-server" module defines the following "grouping" statement:

- tcp-server-grouping

This grouping is presented in the following subsection.

4.1.2.1. The "tcp-server-grouping" Grouping

The following tree diagram [RFC8340] illustrates the "tcp-server-grouping" grouping:

```
  grouping tcp-server-grouping:
    +-- local-bind* [local-address]
    |  +-- local-address?   inet:ip-address
    |  +-- local-port?      inet:port-number
    +---u tcpcmn:tcp-common-grouping
```

Comments:

- The "local-address" node, which is mandatory, may be configured as an IPv4 address, an IPv6 address, or a wildcard value.
- The "local-port" node is not mandatory, but its default value is the invalid value "0", thus forcing the consuming data model to refine it in order to provide it an appropriate default value.
- This grouping uses the "tcp-common-grouping" grouping discussed in Section 2.1.3.1.

4.1.3. Protocol-Accessible Nodes

The "ietf-tcp-server" module defines only "grouping" statements that are used by other modules to instantiate protocol-accessible nodes. Thus, this module, when implemented, does not itself define any protocol-accessible nodes.

4.2. Example Usage

This section presents an example showing the "tcp-server-grouping" grouping populated with some data.

```
<!-- The outermost element below doesn't exist in the data model. -->
<!--  It simulates if the "grouping" were a "container" instead.  -->

<tcp-server xmlns="urn:ietf:params:xml:ns:yang:ietf-tcp-server">
  <local-bind>
    <local-address>192.0.2.2</local-address>
    <local-port>49152</local-port>
  </local-bind>
  <keepalives>
    <idle-time>7200</idle-time>
    <max-probes>9</max-probes>
    <probe-interval>75</probe-interval>
  </keepalives>
</tcp-server>
```

4.3. YANG Module

The "ietf-tcp-server" YANG module references [RFC6991] and [RFC9293].

```
<CODE BEGINS> file "ietf-tcp-server@2024-04-04.yang"

module ietf-tcp-server {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-tcp-server";
  prefix tcps;

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

  import ietf-tcp-common {
    prefix tcpcmn;
    reference
      "RFC 9643: YANG Groupings for TCP Clients and TCP Servers";
  }

  organization
    "IETF NETCONF (Network Configuration) Working Group and the
     IETF TCP Maintenance and Minor Extensions (TCPM) Working Group";

  contact
    "WG Web:   https://datatracker.ietf.org/wg/netconf
               https://datatracker.ietf.org/wg/tcpm
     WG List:  NETCONF WG list <mailto:netconf@ietf.org>
               TCPM WG list <mailto:tcpm@ietf.org>
     Authors:  Kent Watsen <mailto:kent+ietf@watsen.net>
               Michael Scharf
               <mailto:michael.scharf@hs-esslingen.de>";

  description
    "This module defines reusable groupings for TCP servers that
     can be used as a basis for specific TCP server instances.
     The key words 'MUST', 'MUST NOT', 'REQUIRED', 'SHALL',
     'SHALL NOT', 'SHOULD', 'SHOULD NOT', 'RECOMMENDED',
     'NOT RECOMMENDED', 'MAY', and 'OPTIONAL' in this document
     are to be interpreted as described in BCP 14 (RFC 2119)
     (RFC 8174) when, and only when, they appear in all
     capitals, as shown here.

     Copyright (c) 2024 IETF Trust and the persons identified
     as authors of the code.  All rights reserved.

     Redistribution and use in source and binary forms, with
     or without modification, is permitted pursuant to, and
     subject to the license terms contained in, the Revised
     BSD License set forth in Section 4.c of the IETF Trust's
     Legal Provisions Relating to IETF Documents
     (https://trustee.ietf.org/license-info).

     This version of this YANG module is part of RFC 9643
     (https://www.rfc-editor.org/info/rfc9643); see the RFC
     itself for full legal notices.";

  revision 2024-04-04 {
    description
      "Initial version.";
    reference
      "RFC 9643: YANG Groupings for TCP Clients and TCP Servers";
  }

  // Features

  feature tcp-server-keepalives {
    description
      "Per socket TCP keepalive parameters are configurable for
       TCP servers on the server implementing this feature.";
    reference
      "RFC 9293: Transmission Control Protocol (TCP)";
  }

  // Groupings

  grouping tcp-server-grouping {
    description
      "A reusable grouping for configuring a TCP server.

       Note that this grouping uses fairly typical descendant
       node names such that a stack of 'uses' statements will
       have name conflicts.  It is intended that the consuming
       data model will resolve the issue (e.g., by wrapping
       the 'uses' statement in a container called
       'tcp-server-parameters').  This model purposely does
       not do this itself so as to provide maximum flexibility
       to consuming models.";
    list local-bind {
      key "local-address";
      min-elements 1;
      description
        "A list of bind (listen) points for this server
         instance.  A server instance may have multiple
```

```
             bind points to support, e.g., the same port in
             different address families or different ports
             in the same address family.";
          leaf local-address {
            type inet:ip-address;
            description
              "The local IP address to listen on for incoming
               TCP client connections.  To configure listening
               on all IPv4 addresses, the value must be '0.0.0.0'
               (INADDR_ANY).  To configure listening on all IPv6
               addresses, the value must be '::' (INADDR6_ANY).";
          }
          leaf local-port {
            type inet:port-number;
            default "0";
            description
              "The local port number to listen on for incoming TCP
               client connections.  An invalid default value (0)
               is used (instead of 'mandatory true') so that an
               application-level data model may 'refine' it with
               an application-specific default port number value.";
          }
        }
        uses tcpcmn:tcp-common-grouping {
          refine "keepalives" {
            if-feature "tcp-server-keepalives";
            description
              "An 'if-feature' statement so that implementations
               can choose to support TCP server keepalives.";
          }
        }
      }
    }

<CODE ENDS>
```

5. Security Considerations

The three YANG modules in this document define groupings and will not be deployed as standalone modules. Their security implications may be context-dependent based on their use in other modules. The designers of modules that import these groupings must conduct their own analysis of the security considerations.

5.1. Considerations for the "ietf-tcp-common" YANG Module

This section follows the template defined in Section 3.7.1 of [RFC8407].

The "ietf-tcp-common" YANG module defines "grouping" statements that 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., Secure Shell (SSH), TLS) with mutual authentication.

The Network Configuration Access Control Model (NACM) [RFC8341] provides the means to restrict access for particular users to a preconfigured subset of all available protocol operations and content.

Please be aware that this YANG module uses groupings from other YANG modules that define nodes that may be considered sensitive or vulnerable in network environments. Please review the security considerations for dependent YANG modules for information as to which nodes may be considered sensitive or vulnerable in network environments.

None of the readable data nodes defined in this YANG module are considered sensitive or vulnerable in network environments. The NACM "default-deny-all" extension has not been set for any data nodes defined in this module.

None of the writable data nodes defined in this YANG module are considered sensitive or vulnerable in network environments. The NACM "default-deny-write" extension has not been set for any data nodes defined in this module.

This module does not define any RPCs, actions, or notifications, and thus, the security considerations for such are not provided here.

5.2. Considerations for the "ietf-tcp-client" YANG Module

This section follows the template defined in Section 3.7.1 of [RFC8407].

The "ietf-tcp-client" YANG module defines "grouping" statements that 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 Network Configuration Access Control Model (NACM) [RFC8341] provides the means to restrict access for particular users to a preconfigured subset of all available protocol operations and content.

Please be aware that this YANG module uses groupings from other YANG modules that define nodes that may be considered sensitive or vulnerable in network environments. Please review the security considerations for dependent YANG modules for information as to which nodes may be considered sensitive or vulnerable in network environments.

One readable data node defined in this YANG module may be considered sensitive or vulnerable in some network environments. This node is as follows:

- The "proxy-server/socks5-parameters/authentication-parameters/username-password/password" node:

  The "password" node defined in the "tcp-client-grouping" grouping is defined using the "password-grouping" grouping presented in [RFC9640]. This grouping enables both cleartext and encrypted passwords to be configured. As the referenced document states, configuration of cleartext passwords is **NOT RECOMMENDED**. However, in the case cleartext values are configured, this node is additionally sensitive to read operations such that, in normal use cases, it should never be returned to a client. For this reason, the NACM "default-deny-all" extension has been applied to it.

None of the writable data nodes defined in this YANG module are considered sensitive or vulnerable in network environments. The NACM "default-deny-write" extension has not been set for any data nodes defined in this module.

This module does not define any RPCs, actions, or notifications, and thus, the security considerations for such are not provided here.

Implementations are **RECOMMENDED** to implement the "local-binding-supported" feature for cryptographically secure protocols so as to enable more granular ingress/egress firewall rule bases. It is **NOT RECOMMENDED** to implement this feature for unsecure protocols, as per [RFC6056].

5.3. Considerations for the "ietf-tcp-server" YANG Module

This section follows the template defined in Section 3.7.1 of [RFC8407].

The "ietf-tcp-server" YANG module defines "grouping" statements that 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 Network Configuration Access Control Model (NACM) [RFC8341] provides the means to restrict access for particular users to a preconfigured subset of all available protocol operations and content.

Please be aware that this YANG module uses groupings from other YANG modules that define nodes that may be considered sensitive or vulnerable in network environments. Please review the security considerations for dependent YANG modules for information as to which nodes may be considered sensitive or vulnerable in network environments.

None of the readable data nodes defined in this YANG module are considered sensitive or vulnerable in network environments. The NACM "default-deny-all" extension has not been set for any data nodes defined in this module.

None of the writable data nodes defined in this YANG module are considered sensitive or vulnerable in network environments. The NACM "default-deny-write" extension has not been set for any data nodes defined in this module.

This module does not define any RPCs, actions, or notifications, and thus, the security considerations for such are not provided here.

6. IANA Considerations

6.1. The IETF XML Registry

IANA has registered the following URI in the "ns" registry of the "IETF XML Registry" [RFC3688].

URI: urn:ietf:params:xml:ns:yang:ietf-tcp-common
Registrant Contact: The IESG
XML: N/A; the requested URI is an XML namespace.

URI: urn:ietf:params:xml:ns:yang:ietf-tcp-client
Registrant Contact: The IESG
XML: N/A; the requested URI is an XML namespace.

URI: urn:ietf:params:xml:ns:yang:ietf-tcp-server
Registrant Contact: The IESG
XML: N/A; the requested URI is an XML namespace.

6.2. The YANG Module Names Registry

IANA has registered the following three YANG modules in the "YANG Module Names" registry [RFC6020].

name: ietf-tcp-common
namespace: urn:ietf:params:xml:ns:yang:ietf-tcp-common
prefix: tcpcmn
reference: RFC 9643

name: ietf-tcp-client
namespace: urn:ietf:params:xml:ns:yang:ietf-tcp-client
prefix: tcpc
reference: RFC 9643

name: ietf-tcp-server
namespace: urn:ietf:params:xml:ns:yang:ietf-tcp-server
prefix: tcps
reference: RFC 9643

7. References

7.1. Normative References

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

[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>.

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

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

[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>.

[RFC9293] Eddy, W., Ed., "Transmission Control Protocol (TCP)", STD 7, RFC 9293, DOI 10.17487/RFC9293, August 2022, <https://www.rfc-editor.org/info/rfc9293>.

[RFC9640] Watsen, K., "YANG Data Types and Groupings for Cryptography", RFC 9640, DOI 10.17487/RFC9640, August 2024, <https://www.rfc-editor.org/info/rfc9640>.

7.2. Informative References

[HTTP-CLIENT-SERVER] Watsen, K., "YANG Groupings for HTTP Clients and HTTP Servers", Work in Progress, Internet-Draft, draft-ietf-netconf-http-client-server-23, 15 August 2024, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-http-client-server-23>.

[NETCONF-CLIENT-SERVER] Watsen, K., "NETCONF Client and Server Models", Work in Progress, Internet-Draft, draft-ietf-netconf-netconf-client-server-37, 14 August 2024, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-netconf-client-server-37>.

[RESTCONF-CLIENT-SERVER] Watsen, K., "RESTCONF Client and Server Models", Work in Progress, Internet-Draft, draft-ietf-netconf-restconf-client-server-38, 14 August 2024, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-restconf-client-server-38>.

[RFC1928] Leech, M., Ganis, M., Lee, Y., Kuris, R., Koblas, D., and L. Jones, "SOCKS Protocol Version 5", RFC 1928, DOI 10.17487/RFC1928, March 1996, <https://www.rfc-editor.org/info/rfc1928>.

[RFC1929] Leech, M., "Username/Password Authentication for SOCKS V5", RFC 1929, DOI 10.17487/RFC1929, March 1996, <https://www.rfc-editor.org/info/rfc1929>.

[RFC2743] Linn, J., "Generic Security Service Application Program Interface Version 2, Update 1", RFC 2743, DOI 10.17487/RFC2743, January 2000, <https://www.rfc-editor.org/info/rfc2743>.

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

[RFC6056] Larsen, M. and F. Gont, "Recommendations for Transport-Protocol Port Randomization", BCP 156, RFC 6056, DOI 10.17487/RFC6056, January 2011, <https://www.rfc-editor.org/info/rfc6056>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <https://www.rfc-editor.org/info/rfc6242>.

[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>.

[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>.

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

[RFC8407] Bierman, A., "Guidelines for Authors and Reviewers of Documents Containing YANG Data Models", BCP 216, RFC 8407, DOI 10.17487/RFC8407, October 2018, <https://www.rfc-editor.org/info/rfc8407>.

[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>.

[RFC9641] Watsen, K., "A YANG Data Model for a Truststore", RFC 9641, DOI 10.17487/RFC9641, August 2024, <https://www.rfc-editor.org/info/rfc9641>.

[RFC9642] Watsen, K., "A YANG Data Model for a Keystore and Keystore Operations", RFC 9642, DOI 10.17487/RFC9642, August 2024, <https://www.rfc-editor.org/info/rfc9642>.

[RFC9644] Watsen, K., "YANG Groupings for SSH Clients and SSH Servers", RFC 9644, DOI 10.17487/RFC9644, August 2024, <https://www.rfc-editor.org/info/rfc9644>.

[RFC9645] Watsen, K., "YANG Groupings for TLS Clients and TLS Servers", RFC 9645, DOI 10.17487/RFC9645, August 2024, <https://www.rfc-editor.org/info/rfc9645>.

[SOCKS_4A] Lee, Y., "SOCKS 4A: A Simple Extension to SOCKS 4 Protocol", <https://www.openssh.com/txt/socks4a.protocol>.

Acknowledgements

The authors would like to thank the following for lively discussions on list and in the halls (ordered by first name): Éric Vyncke, Joe Clarke, Jürgen Schönwälder, Ladislav Lhotka, Mallory Knodel, Martin Duke, Michael Tüxen, Mohamed Boucadair, Nancy Cam-Winget, Nick Hancock, Per Andersson, Rob Wilton, Roman Danyliw, Tom Petch, and Wim Henderickx.

Authors' Addresses

Kent Watsen
Watsen Networks
Email: kent+ietf@watsen.net

Michael Scharf
Hochschule Esslingen - University of Applied Sciences
Email: michael.scharf@hs-esslingen.de