Workgroup: NETCONF Working Group
Internet-Draft:
draft-ietf-netconf-tcp-client-server-14
Published: 19 October 2022
Intended Status: Standards Track
Expires: 22 April 2023
Authors: K. Watsen
Watsen Networks
M. Scharf
Hochschule Esslingen

YANG Groupings for TCP Clients and TCP Servers

Abstract

This document defines 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.

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:

*AAAA --> the assigned RFC value for draft-ietf-netconf-crypto-types

*DDDD --> 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:

*2022-10-19 --> the publication date of this draft

The "Relation to other RFCs" section Section 1.1 contains the text "one or more YANG modules" and, later, "modules". This text is sourced from a file in a context where it is unknown how many modules a draft defines. The text is not wrong as is, but it may be improved by stating more directly how many modules are defined.

The "Relation to other RFCs" section Section 1.1 contains a self-reference to this draft, along with a corresponding Informative Reference in the Appendix.

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

*Appendix A. Change Log

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 22 April 2023.

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.

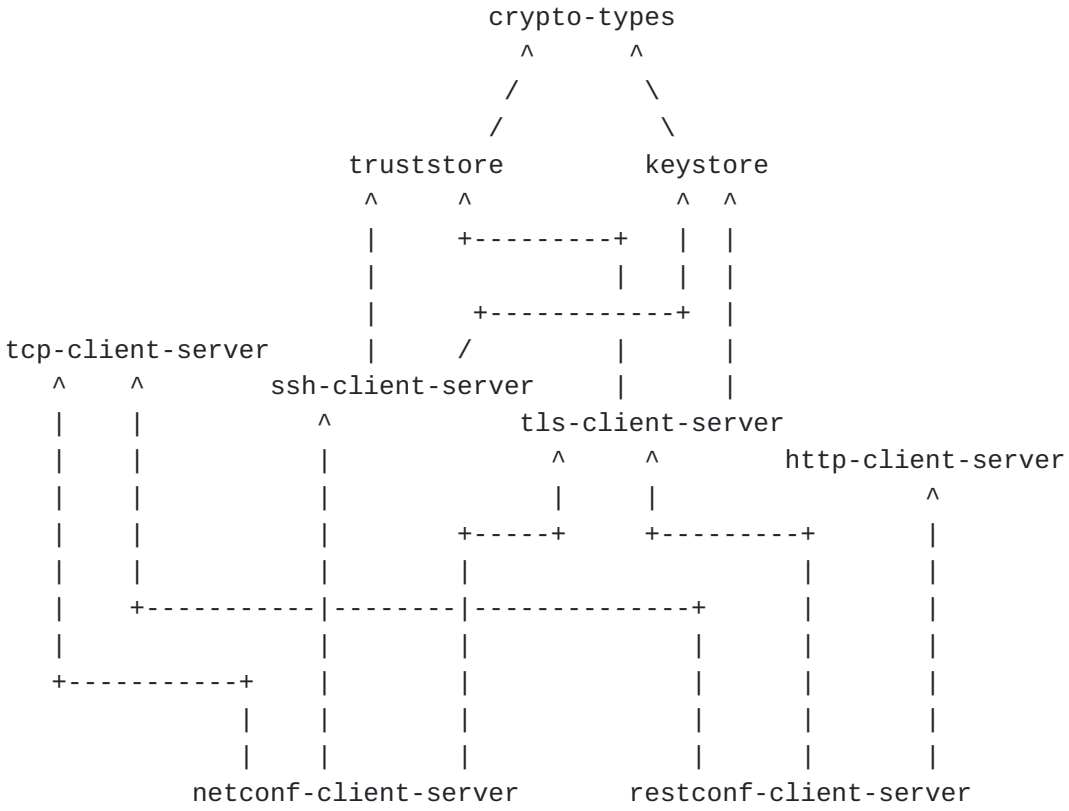
The modules focus on three different types of base TCP parameters that matter for TCP-based applications: First, the modules cover fundamental configuration of a TCP client or TCP server application, such as addresses and port numbers. Second, a reusable grouping enables modification of application-specific parameters for a 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.

1.1. Relation to other RFCs

This document presents one or more YANG modules [RFC7950] that are part of a collection of RFCs that work together to, ultimately, enable the configuration of the clients and servers of both the NETCONF [RFC6241] and RESTCONF [RFC8040] protocols.

These modules have been defined in a modular fashion to enable their use by other efforts, some of which are known to be in progress at the time of this writing, with many more expected to be defined in time.

The normative dependency relationship between the various RFCs in the collection is presented in the below diagram. The labels in the diagram represent the primary purpose provided by each RFC. Hyperlinks to each RFC are provided below the diagram.



Label in Diagram	Originating RFC
crypto-types	[I-D.ietf-netconf-crypto-types]
truststore	[I-D.ietf-netconf-trust-anchors]
keystore	[I-D.ietf-netconf-keystore]
tcp-client-server	[I-D.ietf-netconf-tcp-client-server]
ssh-client-server	[I-D.ietf-netconf-ssh-client-server]
tls-client-server	[I-D.ietf-netconf-tls-client-server]

http-client-server	[I-D.ietf-netconf-http-client-server]
netconf-client-server	[I-D.ietf-netconf-netconf-client-server]
restconf-client-server	[I-D.ietf-netconf-restconf-client-server]

Table 1: Label 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 Examples (Section 2.2). 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 defines a common "grouping" statement for basic TCP connection parameters that matter to applications. 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 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" node so that the mandatory descendant nodes do not imply that keepalives 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.

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 accepted. If keep-alives are included, [RFC1122] 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 aliveness 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 [RFC1122]. 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") MUST default to no less than two hours, i.e., 7200 seconds [RFC1122]. A lower value MAY be configured, but keep-alive messages SHOULD NOT be transmitted more frequently than once every 15 seconds. Longer 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 [RFC1122]. 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" 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>15</idle-time>
    <max-probes>3</max-probes>
    <probe-interval>30</probe-interval>
  </keepalives>
</tcp-common>
```

2.3. YANG Module

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

<CODE BEGINS> file "ietf-tcp-common@2022-10-19.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 defines reusable groupings for TCP commons that
     can be used as a basis for specific TCP common instances.

     Copyright (c) 2022 IETF Trust and the persons identified
     as authors of the code. All rights reserved.

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

     This version of this YANG module is part of RFC DDDD
     (https://www.rfc-editor.org/info/rfcDDDD); 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 2022-10-19 {
    description
      "Initial version";
    reference
      "RFC DDDD: 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.  This statement is
       present so the mandatory descendant nodes do not imply that
       this node must be configured.";
    description
      "Configures the keep-alive 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.";
    leaf idle-time {
      type uint16 {
        range "1..max";
      }
      units "seconds";
      mandatory true;
      description
        "Sets the amount of time after which if no data has been
         received from the TCP peer, a TCP-level probe message
         will be sent to test the aliveness of the TCP peer.
         Two hours (7200 seconds) is safe value, per RFC 1122.";
      reference
        "RFC 1122:
          Requirements for Internet Hosts -- Communication Layers";
    }
    leaf max-probes {
      type uint16 {
        range "1..max";
      }
      mandatory true;
      description
        "Sets the maximum number of sequential keep-alive 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";
        mandatory true;
        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 Examples (Section 3.2). 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
  +-- socks5-gss-api
  +-- socks5-username-password
```

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-parameters
  |     |     +-- remote-address    inet:ip-address
  |     |     +-- remote-port?      inet:port-number
  |     +--:(socks4a)
  |     |  +-- socks4a-parameters
  |     |     +-- remote-address    inet:host
  |     |     +-- remote-port?      inet:port-number
  |     +--:(socks5)
  |        +-- 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, 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.

* 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.

3.2. Example Usage

This section presents two examples showing the "tcp-client-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>443</remote-port>
  <local-address>0.0.0.0</local-address>
  <local-port>0</local-port>
  <keepalives>
    <idle-time>15</idle-time>
    <max-probes>3</max-probes>
    <probe-interval>30</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>443</remote-port>
  <local-address>0.0.0.0</local-address>
  <local-port>0</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>15</idle-time>
    <max-probes>3</max-probes>
    <probe-interval>30</probe-interval>
  </keepalives>
</tcp-client>
```

3.3. YANG Module

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

<CODE BEGINS> file "ietf-tcp-client@2022-10-19.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 AAAA: YANG Data Types and Groupings for Cryptography";
  }

  import ietf-tcp-common {
    prefix tcpcmn;
    reference
      "RFC DDDD: 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.

     Copyright (c) 2022 IETF Trust and the persons identified
     as authors of the code. All rights reserved.

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

```
   This version of this YANG module is part of RFC DDDD
   (https://www.rfc-editor.org/info/rfcDDDD); 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 2022-10-19 {
  description
    "Initial version";
  reference
    "RFC DDDD: 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.";
}

feature proxy-connect {
  description
    "Proxy connection configuration is configurable for
     TCP clients on the server implementing this feature.";
}

feature socks5-gss-api {
  description
    "Indicates that the server supports authenticating
     using GSSAPI when initiating TCP connections via
     and SOCKS Version 5 proxy server.";
  reference
    "RFC 1928: SOCKS Protocol Version 5";
}

feature socks5-username-password {
  description
```

```
    "Indicates that the server supports authenticating using
     username/password when initiating TCP connections via
     and SOCKS Version 5 proxy server.";
  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 (0) is used
       (instead of 'mandatory true') so that as application
       level data model may 'refine' it with an application
       specific default port number value.";
  }
  leaf local-address {
    if-feature "local-binding-supported";
    type inet:ip-address;
    description
```

```
    "The local IP address/interface (VRF?) 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, that the server
     can bind to any IPv4 or IPv6 addresses, respectively.";
}
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 {
      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 {
      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 {
  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
```

```
     /socks-methods.xhtml.";
    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 {
      augment "keepalives" {
        if-feature "tcp-client-keepalives";
        description
          "Add 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 Examples (Section 4.2). 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-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.

4.2. Example Usage

This section presents an example showing the "tcp-server-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-address>10.20.30.40</local-address>
  <local-port>7777</local-port>
  <keepalives>
    <idle-time>15</idle-time>
    <max-probes>3</max-probes>
    <probe-interval>30</probe-interval>
  </keepalives>
</tcp-server>
```

4.3. YANG Module

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

<CODE BEGINS> file "ietf-tcp-server@2022-10-19.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 DDDD: 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.

     Copyright (c) 2022 IETF Trust and the persons identified
     as authors of the code. All rights reserved.

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

     This version of this YANG module is part of RFC DDDD
     (https://www.rfc-editor.org/info/rfcDDDD); 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 2022-10-19 {
  description
    "Initial version";
  reference
    "RFC DDDD: 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.";
}

// 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.";
  leaf local-address {
    type inet:ip-address;
    mandatory true;
    description
      "The local IP address to listen on for incoming
       TCP client connections.  INADDR_ANY (0.0.0.0) or
       INADDR6_ANY (0:0:0:0:0:0:0:0 a.k.a. ::) MUST be
       used when the server is to listen on all IPv4 or
       IPv6 addresses, respectively.";
  }
  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 {
      augment "keepalives" {
        if-feature "tcp-server-keepalives";
        description
          "Add an if-feature statement so that implementations
           can choose to support TCP server keepalives.";
      }
    }
  }
}
```

<CODE ENDS>

5. Security Considerations

5.1. The "ietf-tcp-common" YANG Module

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., SSH, TLS) with mutual authentication.

The Network 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.

Since the module in this document only define groupings, these considerations are primarily for the designers of other modules that use these groupings.

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 consideration for such is not provided here.

5.2. The "ietf-tcp-client" YANG Module

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 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.

Since the module in this document only define groupings, these considerations are primarily for the designers of other modules that use these groupings.

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 cleartext "password" node defined in the "tcp-client-grouping" grouping 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 extension "default-deny-all" 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 consideration for such is 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 rulebases. It is NOT RECOMMENDED to implement this feature for unsecure protocols, as per [RFC6056].

5.3. The "ietf-tcp-server" YANG Module

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 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.

Since the module in this document only define groupings, these considerations are primarily for the designers of other modules that use these groupings.

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 consideration for such is not provided here.

6. IANA Considerations

6.1. The "IETF XML" Registry

This document registers three 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-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

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

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

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

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

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>.

7.2. Informative References

[I-D.ietf-netconf-crypto-types]

Watsen, K., "YANG Data Types and Groupings for Cryptography", Work in Progress, Internet-Draft, draft-ietf-netconf-crypto-types-24, 7 July 2022, <https://www.ietf.org/archive/id/draft-ietf-netconf-crypto-types-24.txt>.

[I-D.ietf-netconf-http-client-server]
Watsen, K., "YANG Groupings for HTTP Clients and HTTP Servers", Work in Progress, Internet-Draft, draft-ietf-netconf-http-client-server-10, 24 May 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-http-client-server-10>.

[I-D.ietf-netconf-keystore] Watsen, K., "A YANG Data Model for a Keystore", Work in Progress, Internet-Draft, draft-ietf-netconf-keystore-25, 24 May 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-keystore-25>.

[I-D.ietf-netconf-netconf-client-server]
Watsen, K., "NETCONF Client and Server Models", Work in Progress, Internet-Draft, draft-ietf-netconf-netconf-client-server-26, 24 May 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-netconf-client-server-26>.

[I-D.ietf-netconf-restconf-client-server]
Watsen, K., "RESTCONF Client and Server Models", Work in Progress, Internet-Draft, draft-ietf-netconf-restconf-client-server-26, 24 May 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-restconf-client-server-26>.

[I-D.ietf-netconf-ssh-client-server]
Watsen, K., "YANG Groupings for SSH Clients and SSH Servers", Work in Progress, Internet-Draft, draft-ietf-netconf-ssh-client-server-30, 30 August 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-ssh-client-server-30>.

[I-D.ietf-netconf-tcp-client-server] Watsen, K. and M. Scharf, "YANG Groupings for TCP Clients and TCP Servers", Work in Progress, Internet-Draft, draft-ietf-netconf-tcp-client-server-13, 24 May 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-tcp-client-server-13>.

[I-D.ietf-netconf-tls-client-server]
Watsen, K., "YANG Groupings for TLS Clients and TLS Servers", Work in Progress, Internet-Draft, draft-ietf-

netconf-tls-client-server-30, 30 August 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-tls-client-server-30>.

[I-D.ietf-netconf-trust-anchors]
Watsen, K., "A YANG Data Model for a Truststore", Work in Progress, Internet-Draft, draft-ietf-netconf-trust-anchors-18, 24 May 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-trust-anchors-18>.

[RFC1122] Braden, R., Ed., "Requirements for Internet Hosts - Communication Layers", STD 3, RFC 1122, DOI 10.17487/RFC1122, October 1989, <https://www.rfc-editor.org/info/rfc1122>.

[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>.

[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>.

[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>.

Appendix A. Change Log

This section is to be removed before publishing as an RFC.

A.1. 00 to 01

* Added 'local-binding-supported' feature to TCP-client model.

* Added 'keepalives-supported' feature to TCP-common model.

* Added 'external-endpoint-values' container and 'external-endpoints' feature to TCP-server model.

A.2. 01 to 02

* Removed the 'external-endpoint-values' container and 'external-endpoints' feature from the TCP-server model.

A.3. 02 to 03

* Moved the common model section to be before the client and server specific sections.

* Added sections "Model Scope" and "Usage Guidelines for Configuring TCP Keep-Alives" to the common model section.

A.4. 03 to 04

* Fixed a few typos.

A.5. 04 to 05

* Removed commented out "grouping tcp-system-grouping" statement kept for reviewers.

* Added a "Note to Reviewers" note to first page.

A.6. 05 to 06

* Added support for TCP proxies.

A.7. 06 to 07

* Expanded "Data Model Overview section(s) [remove "wall" of tree diagrams].

* Updated the Security Considerations section.

A.8. 07 to 08

* Added missing IANA registration for "ietf-tcp-common"

* Added "mandatory true" for the "username" and "password" leafs

* Added an example of a TCP-client configured to connect via a proxy

* Fixed issues found by the SecDir review of the "keystore" draft.

* Updated the "ietf-tcp-client" module to use the new "password-grouping" grouping from the "crypto-types" module.

A.9. 08 to 09

* Addressed comments raised by YANG Doctor in the ct/ts/ks drafts.

A.10. 09 to 10

* Updated Abstract and Intro to address comments by Tom Petch.

* Removed the "tcp-connection-grouping" grouping (now models use the "tcp-common-grouping" directly).

* Added XML-comment above examples explaining the reason for the unusual top-most element's presence.

* Added Securty Considerations section for the "local-binding-supported" feature.

* Replaced some hardcoded refs to <xref> elements.

* Fixed nits found by YANG Doctor reviews.

* Aligned modules with `pyang -f` formatting.

* Added an "Acknowledgements" secetion.

A.11. 10 to 11

* Replaced "base64encodedvalue==" with "BASE64VALUE=" in examples.

* Minor editorial nits

A.12. 11 to 12

*Fixed up the 'WG Web' and 'WG List' lines in YANG module(s)

*Fixed up copyright (i.e., s/Simplified/Revised/) in YANG module(s)

A.13. 12 to 13

*NO UPDATE.

A.14. 13 to 14

*Updated per Shepherd reviews.

Acknowledgements

The authors would like to thank for following for lively discussions on list and in the halls (ordered by first name): Juergen Schoenwaelder, Ladislav Lhotka, Nick Hancock, Per Andersson, and Tom Petch.

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