NETCONF Working Group
Internet-Draft
Intended status: Standards Track
Expires: 8 December 2025

K. Watsen
Watsen Networks

6 June 2025

# YANG Groupings for HTTP Clients and HTTP Servers
# draft-ietf-netconf-http-client-server-28

## Abstract

This document primarily presents two YANG modules: the first defines a minimal grouping for configuring an HTTP client, and the second defines a minimal grouping for configuring an HTTP server. It is intended that these groupings will be used to help define the configuration for simple HTTP-based protocols (not for complete web servers or browsers). This document additionally presents a third YANG module, to define a grouping for URIs.

## Editorial Note (To be removed by RFC Editor)

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

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

* GGGG --> the assigned RFC value for this draft
* HHHH --> the assigned RFC value for draft-ietf-netconf-netconf-client-server
* IIII --> the assigned RFC value for draft-ietf-netconf-restconf-client-server
* JJJJ --> the assigned RFC value for draft-ietf-netconf-udp-client-server

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

* 2025-06-06 --> 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 reference in the Appendix. Please replace the self-reference in this section with "This RFC" (or similar) and remove the self-reference in the "Normative/Informative References" section, whichever it is in.

Tree-diagrams in this draft may use the '\' line-folding mode defined in RFC 8792. However, nicer-to-the-eye is when the '\\' line-folding mode is used. The AD suggested suggested putting a request here for the RFC Editor to help convert "ugly" '\' folded examples to use the '\\' folding mode. "Help convert" may be interpreted as, identify what looks ugly and ask the authors to make the adjustment.

Due to a bug in the pyang tool used to create tree diagrams, some "key" nodes appear as optional (i.e., have a '?' postfix). Ideally the '?' character is removed in the tree diagrams for "key" nodes. Recipe: search for lists using the string "* [", then note the nodes appearing in the square brackets (e.g., "* [name]"), then look for matching child nodes and remove the '?' characters (e.g., "name?" becomes "name").

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 8 December 2025.

Table of Contents

## 1. Introduction

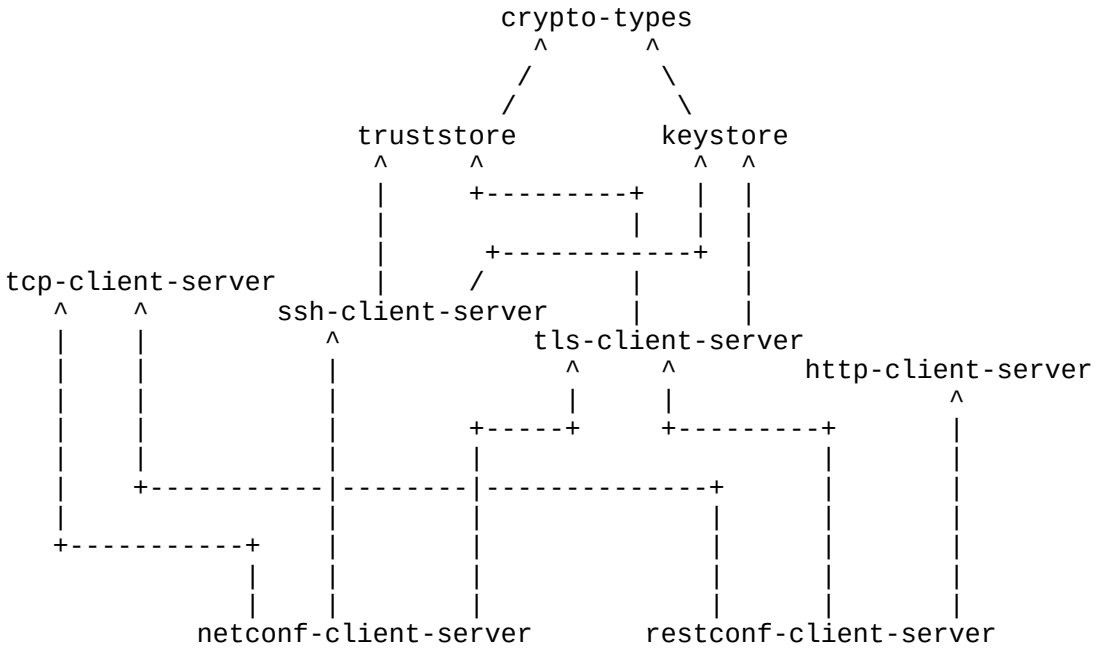
This document primarily presents two YANG 1.1 [RFC7950] modules: the first defines a minimal grouping for configuring an HTTP client, and the second defines a minimal grouping for configuring an HTTP server. It is intended that these groupings will be used to help define the configuration for simple HTTP-based protocols. This document additionally presents a third YANG module, to define a grouping for URIs.

### 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, support the configuration of both the clients and servers of both the NETCONF [RFC6241] and RESTCONF [RFC8040] protocols.

Primary dependency relationships between the YANG groupings defined in the various RFCs is presented in the below diagram. In some cases, a draft may define secondary groupings that introduce dependencies not illustrated in the diagram. The labels in the diagram are a shorthand name for the defining RFC. The citation reference for shorthand name is 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.



| Label in Diagram | Originating RFC |
|---|---|
| crypto-types | [RFC9640] |
| truststore | [RFC9641] |
| keystore | [RFC9642] |
| tcp-client-server | [RFC9643] |
| ssh-client-server | [RFC9644] |
| tls-client-server | [RFC9645] |
| 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 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]. For instance, as described in [RFC9641] and [RFC9642], trust anchors and keys installed during manufacturing are expected to appear in <operational> (Section 5.3 of [RFC8342]), and <system> [I-D.ietf-netmod-system-config], if implemented.

## 2. The "ietf-http-client" Module

This section defines a YANG 1.1 module called "ietf-http-client". 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. The YANG module itself is defined in Section 2.3.

### 2.1. Data Model Overview

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

#### 2.1.1. Features

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

```
Features:
  +-- tls-supported
  +-- http-11-supported
  +-- http-2-supported
  +-- http-3-supported
  +-- proxy-connect
```

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

#### 2.1.2. Groupings

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

* http-client-grouping

This grouping is presented in the following subsection.

##### 2.1.2.1. The "http-client-grouping" Grouping

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

```
  grouping http-client-grouping:
    +-- uri
    |  +---u uri:uri
    +-- protocol-versions!
    |  +-- min    min-max-typedef
    |  +-- max    min-max-typedef
    +-- tls-client-parameters! {tls-supported}?
    |  +---u tlsc:tls-client-grouping
    +-- proxy-connect! {proxy-connect}?
       +-- protocol    enumeration
       +-- host        inet:host
       +-- port        uint16
```

Comments:

* The ’uri’ node is mandatory due to the ’scheme’ and ’host’ descendant nodes being mandatory. The ’uri’ node is the only mandatory part of the "http-client-grouping" grouping.

* The optional ’protocol-versions’ node specifies a min-max range of protocol versions to use.

* The optional ’tls-client-parameters’ node specifies TLS-level client identity and server authentication credentials to use, if needed per the configured URI. When unspecified, TLS-based connections are not possible. The "tls-client-grouping" grouping is discussed in Section 3.1.2.1 of [RFC9645].

* The optional ’proxy-connect’ node specifies how to proxy the connection through a proxy server.

* Unlike other "client" groupings in the suite a "client-server" drafts mentioned in Section 1.1, this client grouping also encodes configuration for the lower protocol stack layers. Specifically, the "scheme" and "authority" parts of the URI encode information that portains to transport layers.

#### 2.1.3. Protocol-accessible Nodes

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

### 2.2. Example Usage

This section presents three examples showing the http-client-grouping populated with some data.

The first example illustrates the case where the HTTP client is configured to connect directly to an HTTP server without TLS.

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

<http-client xmlns="urn:ietf:params:xml:ns:yang:ietf-http-client">
  <uri>
    <scheme>http</scheme>
    <authority>
      <userinfo>user</userinfo>
      <host>example.com</host>
      <port>80</port>
    </authority>
    <path>/foo/bar</path>
    <query>query</query>
    <fragment>fragment</fragment>
  </uri>
</http-client>
```

The second example illustrates the case where the HTTP client is configured to connect directly to an HTTP server with TLS.

This example is consistent with examples presented in Section 2.2.1 of [RFC9641] and Section 2.2.1 of [RFC9642].

```
   =============== NOTE: '\' line wrapping per RFC 8792 ================

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

   <http-client xmlns="urn:ietf:params:xml:ns:yang:ietf-http-client">
     <uri>
       <scheme>https</scheme>
       <authority>
         <userinfo>user:pass</userinfo>
         <host>example.com</host>
         <port>443</port>
       </authority>
       <path>/foo/bar</path>
       <query>query</query>
       <fragment>fragment</fragment>
     </uri>
     <protocol-versions>
       <min>HTTP/1</min>
       <max>HTTP/3</max>
     </protocol-versions>
     <tls-client-parameters>
       <client-identity>
         <certificate>
           <central-keystore-reference>
             <asymmetric-key>rsa-asymmetric-key</asymmetric-key>
             <certificate>ex-rsa-cert</certificate>
           </central-keystore-reference>
         </certificate>
       </client-identity>
       <server-authentication>
         <ca-certs>
           <central-truststore-reference>trusted-server-ca-certs</centr\
   al-truststore-reference>
         </ca-certs>
         <ee-certs>
           <central-truststore-reference>trusted-server-ee-certs</centr\
   al-truststore-reference>
         </ee-certs>
       </server-authentication>
     </tls-client-parameters>
   </http-client>
```

The following example illustrates the case where the HTTP client is configured to connect through an HTTP proxy.

This example also is consistent with examples presented in Section 2.2.1 of [RFC9641] and Section 2.2.1 of [RFC9642].

```
   =============== NOTE: '\' line wrapping per RFC 8792 ================

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

   <http-client xmlns="urn:ietf:params:xml:ns:yang:ietf-http-client">
     <uri>
       <scheme>https</scheme>
       <authority>
         <userinfo>user:pass</userinfo>
         <host>example.com</host>
         <port>443</port>
       </authority>
       <path>/foo/bar</path>
       <query>query</query>
       <fragment>fragment</fragment>
     </uri>
     <tls-client-parameters>
       <client-identity>
         <certificate>
           <central-keystore-reference>
             <asymmetric-key>rsa-asymmetric-key</asymmetric-key>
             <certificate>ex-rsa-cert</certificate>
           </central-keystore-reference>
         </certificate>
       </client-identity>
       <server-authentication>
         <ca-certs>
           <central-truststore-reference>trusted-server-ca-certs</centr\
   al-truststore-reference>
         </ca-certs>
         <ee-certs>
           <central-truststore-reference>trusted-server-ee-certs</centr\
   al-truststore-reference>
         </ee-certs>
       </server-authentication>
     </tls-client-parameters>
     <proxy-connect>
       <protocol>CONNECT</protocol>
       <host>proxy.example.org</host>
       <port>4443</port>
     </proxy-connect>
   </http-client>
```

## 2.3. YANG Module

This YANG module has references to [RFC3986], [RFC6991], [RFC8341], [RFC9110], and [RFC9645].

```
<CODE BEGINS> file "ietf-http-client@2025-06-06.yang"

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

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

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

  import ietf-uri {
    prefix uri;
    reference
      "RFC GGGG: YANG Groupings for HTTP Clients and HTTP Servers";
  }

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

  contact
    "WG Web:   https://datatracker.ietf.org/wg/netconf
     WG List:  NETCONF WG list <mailto:netconf@ietf.org>
     Author:   Kent Watsen <mailto:kent+ietf@watsen.net>";

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

     Copyright (c) 2025 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 GGGG
```

```
        (https://www.rfc-editor.org/info/rfcGGGG); 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 2025-06-06 {
        description
          "Initial version";
        reference
          "RFC GGGG: YANG Groupings for HTTP Clients and HTTP Servers";
      }

      // Features

      feature tls-supported {
        description
          "Indicates that the server supports configuring
           HTTP client certificates.";
        reference
          "RFC 9110: HTTP Semantics";
      }

      feature http1-supported {
        description
          "Indicates that the server supports configuring
           clients to use the HTTP/1.1 protocol.";
        reference
          "RFC 9112: HTTP/1.1";
      }

      feature http2-supported {
        description
          "Indicates that the server supports configuring
           clients to use the HTTP/2 protocol.";
        reference
          "RFC 9113: HTTP/2";
      }

      feature http3-supported {
        if-feature tls-supported;
        description
          "Indicates that the server supports configuring
           clients to use the HTTP/3 protocol.";
```

```
      reference
        "RFC 9114: HTTP/3";
    }

    feature proxy-connect {
      description
        "Indicates that the server supports configuring HTTP
         clients to connect to a remote HTTP server via a
         proxy, per Section 9.3.6 of RFC 9110.";
      reference
        "RFC 9110: HTTP Semantics";
    }

    // Grouping

    grouping http-client-grouping {
      description
        "A grouping for HTTP client level configuration. This grouping
         is not expected to be used with groupings for lower protocol
         layer (e.g., the 'tcp-client-grouping' found in RFC 9643), as
         equivalent information is encoded into the URI.";
      reference
        "RFC 9643: YANG Groupings for TCP Clients and TCP Servers";

      container uri {
        must "scheme != 'https' or ../tls-client-parameters";
        description
          "The URI, described in Section 3 of RFC 3986, the HTTP client
           establishes a connection to.";
        reference
          "RFC 3986  URI Generic Syntax";
        uses uri:uri;
      }

      container protocol-versions {
        presence
          "If unconfigured, then all versions are supported.";
        description
          "HTTP protocol versions the client supports.";
        typedef min-max-typedef {
          type enumeration {
            enum "HTTP/1" {
              if-feature http1-supported;
              description
                "Indicates that 'HTTP/1' has been configured.";
            }
            enum "HTTP/2" {
              if-feature http2-supported;
```

```
            description
              "Indicates that 'HTTP/2' has been configured.";
          }
          enum "HTTP/3" {
            if-feature http3-supported;
            description
              "Indicates that 'HTTP/3' has been configured.";
          }
        }
        description
          "A typedef used by the 'min' and 'max' leafs.";
      }
      leaf min {
        type min-max-typedef;
        mandatory true;
        description
          "The minimum protocol version supported.";
      }
      leaf max {
        type min-max-typedef;
        mandatory true;
        description
          "The maximum protocol version supported.";
      }
    }

    container tls-client-parameters {
      if-feature tls-supported;
      presence
        "Indicates that TLS-client parameters have been configured.";
      description
        "TLS client parameters for 'https' connections.";
      uses tlsc:tls-client-grouping;
    }

    container proxy-connect {
      if-feature proxy-connect;
      presence
        "Indicates that the HTTP-client connects through the
         configured proxy.";
      description
        "Configures how to connect to the proxy.";
      leaf protocol {
        type enumeration {
          enum CONNECT {
            description
              "Use the 'CONNECT' method described in Section 9.3.6
               of RFC 9110.";
```

```
              reference
                "RFC 9110: HTTP Semantics";
            }
            enum CONNECT-UDP {
              description
                "Use the 'connect-udp' upgrade token described in
                 Section 3 of RFC 9298.";
              reference
                "RFC 9298: Proxying UDP in HTTP";
            }
          }
          mandatory true;
          description
            "The protocol to use when connecting to the proxy.";
        }
        leaf host {
          type inet:host;
          mandatory true;
          description
            "The 'Host' subcomponent, as described in Section 3.2.2 of
             RFC 3986, of the proxy.";
          reference
            "RFC 3986  URI Generic Syntax";
        }
        leaf port {
          type uint16;
          mandatory true;
          description
            "The 'Port' subcomponent, as described in Section 3.2.3 of
             RFC 3986, of the proxy.";
          reference
            "RFC 3986  URI Generic Syntax";
        }
      }
    }
  }
```

<CODE ENDS>

## 3. The "ietf-http-server" Module

This section defines a YANG 1.1 module called "ietf-http-server". 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. The YANG module itself is defined in Section 3.3.

### 3.1. Data Model Overview

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

#### 3.1.1. Features

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

```
Features:
  +-- client-auth-supported
  +-- local-users-supported
  +-- basic-auth
  +-- tcp-supported
  +-- tls-supported
  +-- quic-supported
```

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

#### 3.1.2. Groupings

The "ietf-http-server" module defines the following "grouping" statements:

* http-server-grouping
* http-server-listen-stack-grouping

Each of these groupings are presented in the following subsections.

##### 3.1.2.1. The "http-server-grouping" Grouping

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

```
grouping http-server-grouping:
  +-- server-name?             string
  +-- client-authentication! {client-auth-supported}?
     +-- users {local-users-supported}?
        +-- user* [user-id]
           +-- user-id?       string
           +-- (auth-type)
              +--:(basic)
                 +-- basic {basic-auth}?
                    +-- username?   string
                    +-- password
                       +-- hashed-password?   ianach:crypt-hash
                       +--ro last-modified?     yang:date-and-time
```

Comments:

* The "http-server-grouping" defines the configuration for just the "HTTP" part of a protocol stack. It does not, for instance, define any configuration for the "TCP" or "TLS" protocol layers (for that, see Section 3.1.2.2).

* The "server-name" node defines the HTTP server’s name, as presented to HTTP clients.

* The "client-authentication" node, which must by enabled by a feature, defines a very simple user-database. Only the "basic" authentication scheme is supported, albeit it must be enabled by a "feature". Other authentication schemes MAY be augmented in.

#### 3.1.2.2. The "http-server-listen-stack-grouping" Grouping

The following tree diagram [RFC8340] illustrates the "http-server-listen-stack-grouping" grouping:

```
     grouping http-server-listen-stack-grouping:
       +-- (transport)
       |  +--:(http-over-tcp) {tcp-supported}?
       |  |  +-- http-over-tcp
       |  |     +-- tcp-server-parameters
       |  |     |  +---u tcps:tcp-server-grouping
       |  |     +-- http-server-parameters
       |  |        +---u http-server-grouping
       |  +--:(http-over-tls) {tls-supported}?
       |  |  +-- http-over-tls
       |  |     +-- tcp-server-parameters
       |  |     |  +---u tcps:tcp-server-grouping
       |  |     +-- tls-server-parameters
       |  |     |  +---u tlss:tls-server-grouping
       |  |     +-- http-server-parameters
       |  |        +---u http-server-grouping
       |  +--:(http-over-quic) {quic-supported}?
       |     +-- http-over-quic
       |        +-- udp-server-parameters
       |        |  +---u udps:udp-server
       |        +-- tls-server-parameters
       |        |  +---u tlss:tls-server-grouping
       |        +-- http-server-parameters
       |           +---u http-server-grouping
       +-- protocol-versions!
          +-- min    min-max-typedef
          +-- max    min-max-typedef
```

Comments:

* The "http-server-listen-stack-grouping" is a convenience grouping for consuming modules. It defines protocol stacks for HTTP/1.1, HTTP/2, and HTTP/3, with each option enabled by a "feature" statement for application control. Other protocols may be added by future work using the YANG "augment" statement.

* For the referenced grouping statement(s):

  - The "udp-server-grouping" grouping is discussed in Section 3 of [I-D.ietf-netconf-udp-client-server].
  - The "tcp-server-grouping" grouping is discussed in Section 4.1.2.1 of [RFC9643].
  - The "tls-server-grouping" grouping is discussed in Section 4.1.2.1 of [RFC9645].
  - The "http-server-grouping" grouping is discussed in Section 3.1.2.1 in this document.

#### 3.1.3. Protocol-accessible Nodes

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

### 3.2. Example Usage

This section presents an example showing the http-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.  -->

<http-server xmlns="urn:ietf:params:xml:ns:yang:ietf-http-server">
  <server-name>foo.example.com</server-name>
</http-server>
```

### 3.3. YANG Module

This YANG module has references to [RFC6991], [RFC7317], [RFC7617], [RFC8341], [RFC9110], [RFC9643], [RFC9645], and [I-D.ietf-netconf-udp-client-server].

```
<CODE BEGINS> file "ietf-http-server@2025-06-06.yang"

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

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

  import iana-crypt-hash {
    prefix ianach;
    reference
      "RFC 7317: A YANG Data Model for System Management";
  }

  import ietf-netconf-acm {
    prefix nacm;
    reference
      "RFC 8341: Network Configuration Access Control Model";
```

```
  }

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

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

  import ietf-udp-server {
    prefix udps;
    reference
      "RFC JJJJ: YANG Groupings for UDP Clients and UDP Servers";
  }

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

  contact
    "WG Web:   https://datatracker.ietf.org/wg/netconf
     WG List:  NETCONF WG list <mailto:netconf@ietf.org>
     Author:   Kent Watsen <mailto:kent+ietf@watsen.net>";

  description
    "This module defines groupings for HTTP servers that can
     be used as a basis for specific HTTP server instances.

     Copyright (c) 2025 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 GGGG
     (https://www.rfc-editor.org/info/rfcGGGG); 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 2025-06-06 {
       description
         "Initial version";
       reference
         "RFC GGGG: YANG Groupings for HTTP Clients and HTTP Servers";
     }

     // Features

     feature client-auth-supported {
       description
         "Indicates that the server supports configuring HTTP
          servers to authenticate HTTP clients.  HTTP-level client
          authentication may not be needed when client authentication
          is expected to occur only at another protocol layer (e.g.,
          TLS).";
     }

     feature local-users-supported {
       if-feature "client-auth-supported";
       description
         "Indicates that the server supports configuring client
          authentication with its own database of local users, as
          opposed to in an application specific location.";
     }

     feature basic-auth {
       if-feature "local-users-supported";
       description
         "Indicates that the server supports configuring 'basic'
          authentication credentials in its local user database.";
       reference
         "RFC 7617: The 'Basic' HTTP Authentication Scheme";
     }

     feature tcp-supported {
       description
         "Indicates that the server supports configuring HTTP
          servers to listen for HTTP 1.1/2.0 connections over TCP.";
       reference
         "RFC 9110: HTTP Semantics";
     }
```

```
     feature tls-supported {
       description
         "Indicates that the server supports configuring HTTP
          servers to listen for HTTP 1.1/2.0 connections over TLS.";
       reference
         "RFC 9110: HTTP Semantics";
     }

     feature quic-supported {
       description
         "Indicates that the server supports configuring HTTP
          servers to listen for HTTP/3 connections over QUIC.";
       reference
         "RFC 9114: HTTP/3";
     }

     feature http1-supported {
       description
         "Indicates that the server supports configuring
          clients to use the HTTP/1.1 protocol.";
       reference
         "RFC 9112: HTTP/1.1";
     }

     feature http2-supported {
       description
         "Indicates that the server supports configuring
          clients to use the HTTP/2 protocol.";
       reference
         "RFC 9113: HTTP/2";
     }

     feature http3-supported {
       if-feature tls-supported;
       description
         "Indicates that the server supports configuring
          clients to use the HTTP/3 protocol.";
       reference
         "RFC 9114: HTTP/3";
     }

     // Groupings

     grouping http-server-grouping {
       description
         "A grouping for configuring HTTP server level parameters.

          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
          'http-server-parameters').  This model purposely does
          not do this itself so as to provide maximum flexibility
          to consuming models.";

        leaf server-name {
          nacm:default-deny-write;
          type string;
          description
            "The value of the 'Server' header field.  If not set, then
             underlying software's default value is used.  Set to the
             empty string to disable.";
        }

        container client-authentication {
          if-feature "client-auth-supported";
          nacm:default-deny-write;
          presence
            "Indicates that HTTP based client authentication is
             configured.  This statement is present so the mandatory
             descendant nodes do not imply that this node must be
             configured.";
          description
            "Configures how the HTTP server can authenticate HTTP
             clients.  The HTTP server will request that the HTTP
             client send authentication when needed.";
          container users {
            if-feature "local-users-supported";
            description
              "A list of locally configured users.";
            list user {
              key "user-id";
              description
                "The list of local users configured on this device.";
              leaf user-id {
                type string;
                description
                  "The user-id for the authenticating client.";
              }
              choice auth-type {
                mandatory true;
                description
                  "The authentication type.";
                case basic {
                  container basic {
```

```
                  if-feature "basic-auth";
                  leaf username {
                    type string;
                    description
                      "The username for the authenticating HTTP
                       client.";
                  }
                  container password {
                    description
                      "The hashed password the HTTP server uses to
                       authenticate this user.  A user is authenticated
                       if the hash of the supplied password matches
                       this value.";
                    leaf hashed-password {
                      type ianach:crypt-hash;
                      description
                        "The password for the authenticating client.";
                    }
                    leaf last-modified {
                      type yang:date-and-time;
                      config false;
                      description
                        "Identifies when the password was last set.";
                    }
                  }
                  description
                    "The 'basic' HTTP scheme credentials.";
                  reference
                    "RFC 7617:
                      The 'Basic' HTTP Authentication Scheme";
                }
              }
            }
          }
        }
      } // container client-authentication
    } // grouping http-server-grouping


    grouping http-server-listen-stack-grouping {
      description
        "A grouping that defines a single instance of an HTTP-based
         protocol stack to listen for HTTP connections.";
      choice transport {
        mandatory true;
        description
          "Choice amongst various transports type.";
        case http-over-tcp {
```

```
            if-feature "tcp-supported";
            container http-over-tcp {
              description
                "Container for TCP-based HTTP/1 or HTTP/2 protocols.";
              container tcp-server-parameters {
                description
                  "TCP-level server parameters to
                   listen for HTTP connections.";
                uses tcps:tcp-server-grouping {
                  refine "local-bind/local-port" {
                    default "80";
                    description
                      "The HTTP server will attempt to connect
                       to the IANA-assigned well-known port for
                       'http' (80) if no value is specified.";
                  }
                }
              }
              container http-server-parameters {
                description
                  "HTTP-level server parameters to
                   listen for HTTP connections.";
                uses http-server-grouping;
              }
            }
          }
          case http-over-tls {
            if-feature "tls-supported";
            container http-over-tls {
              description
                "Container for TLS-based HTTP/1 or HTTP/2 protocols.";
              container tcp-server-parameters {
                description
                  "TCP-level server parameters to
                   listen for HTTPS connections.";
                uses tcps:tcp-server-grouping {
                  refine "local-bind/local-port" {
                    default "443";
                    description
                      "The HTTP server will attempt to connect
                       to the IANA-assigned well-known port for
                       'https' (443) if no value is specified.";
                  }
                }
              }
              container tls-server-parameters {
                description
                  "TLS-level server parameters to
```

```
                listen for HTTPS connections.";
            uses tlss:tls-server-grouping;
          }
          container http-server-parameters {
            description
              "HTTP-level server parameters to
               listen for HTTPS connections.";
            uses http-server-grouping;
          }
        }
      }
      case http-over-quic {
        if-feature "quic-supported";
        container http-over-quic {
          description
            "Container for the QUIC-based HTTP/3 protocol.";
          container udp-server-parameters {
            description
              "UDP-level server parameters.";
            uses udps:udp-server;
          }
          container tls-server-parameters {
            description
              "TLS-level server parameters.";
            uses tlss:tls-server-grouping;
          }
          container http-server-parameters {
            description
              "HTTP-level server parameters.";
            uses http-server-grouping;
          }
        }
      }
    }
    container protocol-versions {
      presence
        "If unconfigured, then all versions are supported.";
      description
        "HTTP protocol versions the client supports.";
      typedef min-max-typedef {
        type enumeration {
          enum "HTTP/1" {
            if-feature http1-supported;
            description
              "Indicates that 'HTTP/1' has been configured.";
          }
          enum "HTTP/2" {
            if-feature http2-supported;
```

```
              description
                "Indicates that 'HTTP/2' has been configured.";
            }
            enum "HTTP/3" {
              if-feature http3-supported;
              description
                "Indicates that 'HTTP/3' has been configured.";
            }
          }
          description
            "A typedef used by the 'min' and 'max' leafs.";
        }
        leaf min {
          type min-max-typedef;
          mandatory true;
          description
            "The minimum protocol version supported.";
        }
        leaf max {
          type min-max-typedef;
          mandatory true;
          description
            "The maximum protocol version supported.";
        }
      }
    } // http-server-listen-stack-grouping

  }

  <CODE ENDS>
```

# 4. The "ietf-uri" Module

This section defines a YANG 1.1 module called "ietf-uri". 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. The YANG module itself is defined in Section 4.3.

## 4.1. Data Model Overview

This section provides an overview of the "ietf-uri" module, which defines a single grouping called "uri" presenting a data model for the URI defined in Section 3 of [RFC3986].

### 4.1.1. The "uri" Grouping

The following tree diagram [RFC8340] illustrates the "uri" grouping:

```
grouping uri:
  +-- scheme       string
  +-- authority!
  |  +-- userinfo?   string
  |  +-- host        inet:host
  |  +-- port?       inet:port-number
  +-- path?        string
  +-- query?       string
  +-- fragment?    string
```

#### 4.1.2. Protocol-accessible Nodes

The "ietf-uri" module defines only a "grouping" statement that is used by other modules to instantiate protocol-accessible nodes. This module, when implemented, does not define any protocol-accessible nodes.

### 4.2. Example Usage

This section presents two examples showing the uri grouping populated with some data.

The first example illustrates the case where the URI represents the string "https://example.com".

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

<uri xmlns="urn:ietf:params:xml:ns:yang:ietf-uri">
  <scheme>https</scheme>
  <authority>
    <userinfo>user:pass</userinfo>
    <host>example.com</host>
    <port>443</port>
  </authority>
  <path>/foo/bar</path>
  <query>query</query>
  <fragment>fragment</fragment>
</uri>
```

The second example illustrates the case where the URI represents the string "https://user:pass@example.com:443/foo/bar?query#fragment".

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

   <uri xmlns="urn:ietf:params:xml:ns:yang:ietf-uri">
     <scheme>https</scheme>
     <authority>
       <host>example.com</host>
     </authority>
   </uri>
```

### 4.3. YANG Module

This YANG module has references to [RFC3986], [RFC6991] and [RFC8341].

```
<CODE BEGINS> file "ietf-uri@2025-06-06.yang"

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

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

  import ietf-netconf-acm {
    prefix nacm;
    reference
      "RFC 8341: Network Configuration Access Control Model";
  }

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

  contact
    "WG Web:   https://datatracker.ietf.org/wg/netconf
     WG List:  NETCONF WG list <mailto:netconf@ietf.org>
     Author:   Kent Watsen <mailto:kent+ietf@watsen.net>";

  description
    "This module defines a grouping for the URI described
     in Section 3 of RFC 3986.

     Copyright (c) 2025 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 GGGG
        (https://www.rfc-editor.org/info/rfcGGGG); 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 2025-06-06 {
        description
          "Initial version";
        reference
          "RFC GGGG: YANG Groupings for HTTP Clients and HTTP Servers";
      }

      // Grouping

      grouping uri {
        description
          "A grouping that defines leafs for each component of the
           URI described in Section 3 of RFC 3986.  This grouping
           may be preferred, to the 'uri' typedef in Section 4 of
           RFC 6991, when validation is important.

           This grouping does not define an outer container, e.g.,
           called 'uri', thus enabling consuming YANG modules to
           chose the appropriate name for the data model.";
        reference
          "RFC 3986: URI Generic Syntax
           RFC 6991: Common YANG Data Types";
        leaf scheme {
          type string;
          mandatory true;
          description
            "The 'Scheme' as described in Section 3.1 of RFC 3986.";
        }
        container authority {
          presence
```

```
          "Indicates that 'authority' has been configured.";
        description
          "The 'Authority' as described in Section 3.2 of RFC 3986.";
        leaf userinfo {
          nacm:default-deny-all;
          type string;
          description
            "The 'User Information' as described in Section 3.2.1
             of RFC 3986.";
        }
        leaf host {
          type inet:host;
          mandatory true;
          description
            "The 'Host' as described in Section 3.2.2 of RFC 3986.";
        }
        leaf port {
          type inet:port-number;
          description
            "The 'Port' as described in Section 3.2.3 of RFC 3986.";
        }
      }
      leaf path {
        type string;
        description
          "The 'Path' as described in Section 3.3 of RFC 3986.";
      }
      leaf query {
        type string;
        description
          "The 'Query' as described in Section 3.4 of RFC 3986.";
      }
      leaf fragment {
        type string;
        description
          "The 'Fragment' as described in Section 3.5 of RFC 3986.";
      }
    }
  }

  <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 which import these grouping must conduct their own analysis of the security considerations.

### 5.1. Considerations for the "ietf-http-client" YANG Module

This section is modeled after the template defined in Section 3.7.1 of [RFC8407].

The "ietf-http-client" YANG module defines data nodes that are designed to be accessed via YANG-based management protocols, such as NETCONF [RFC6241] and RESTCONF [RFC8040]. The YANG-based management protocols have to use a secure transport layer such as SSH [RFC4252], TLS [RFC8446], or QUIC [RFC9000]. The YANG-based management protocols also have to use 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.

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, i.e., excluding imported modules, are considered sensitive or vulnerable in some network environments.

The following writable data nodes defined in this YANG module, i.e., excluding imported modules, are considered sensitive or vulnerable in network environments:

* The "proxy-connect" node in the "http-client-grouping" grouping may be considered sensitive or vulnerable in some network environments. For this reason, the NACM extension "default-deny-all" has been applied to it. A misconfigured "proxy-connect" node may result in loss of connectivity or, perhaps, unexpectedly elevated authorization.

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

### 5.2. Considerations for the "ietf-http-server" YANG Module

This section is modeled after the template defined in Section 3.7.1 of [RFC8407].

The "ietf-http-server" YANG module defines data nodes that are designed to be accessed via YANG-based management protocols, such as NETCONF [RFC6241] and RESTCONF [RFC8040]. The YANG-based management protocols have to use a secure transport layer such as SSH [RFC4252], TLS [RFC8446], or QUIC [RFC9000]. The YANG-based management protocols also have to use 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.

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, i.e., excluding imported modules, 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.

The following writable data nodes defined in this YANG module, i.e., excluding imported modules, are considered sensitive or vulnerable in network environments:

* The "server-name" node in the "http-server-grouping" grouping may be considered sensitive or vulnerable in some network environments. For this reason, the NACM extension "default-deny-write" has been applied to it. A misconfigured "server-name" may mislead clients into not knowing how to interoperate with the server (e.g., "foo v1.0" vs "foo 2.0").

* The "client-authentication" node in the "http-server-grouping" grouping may be considered sensitive or vulnerable in some network environments. For this reason, the NACM extension "default-deny-write" has been applied to it. The feature "tls-supported" is not required, in order to support cases where an external device is used to terminate TLS connections, but such arrangements leave the client- credentials to be unprotected by the transport. Misconfigured "client-authentication" may lead the server to authenticate invalid client credentials.

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

### 5.3. Considerations for the "ietf-http-client" YANG Module

This section is modeled after the template defined in Section 3.7.1 of [RFC8407].

The "ietf-http-client" YANG module defines data nodes that are designed to be accessed via YANG-based management protocols, such as NETCONF [RFC6241] and RESTCONF [RFC8040]. The YANG-based management protocols have to use a secure transport layer such as SSH [RFC4252], TLS [RFC8446], or QUIC [RFC9000]. The YANG-based management protocols also have to use 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.

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.

Some of the readable data nodes defined in this YANG module, i.e., excluding imported modules, may be considered sensitive or vulnerable in some network environments. It is thus important to control read access to these data nodes. The following data nodes have particular sensitivity/vulnerability:

* The "userinfo" node:

  The "userinfo" node may encode a cleartext password (basic or digest authentication). For this reason, the NACM extension "default-deny-all" was applied to it.

None of the writable data nodes defined in this YANG module, i.e., excluding imported modules, are considered sensitive or vulnerable in network environments.

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

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

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

URI: urn:ietf:params:xml:ns:yang:ietf-uri
Registrant Contact: The IESG
XML: N/A, the requested URI is an XML namespace.
```

## 6.2. The "YANG Module Names" Registry

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

```
Name:         ietf-http-client
Namespace:    urn:ietf:params:xml:ns:yang:ietf-http-client
Prefix:       httpc
Reference:    RFC GGGG
Maintained by IANA:  N

Name:         ietf-http-server
Namespace:    urn:ietf:params:xml:ns:yang:ietf-http-server
Prefix:       https
Reference:    RFC GGGG
Maintained by IANA:  N

Name:         ietf-uri
Namespace:    urn:ietf:params:xml:ns:yang:ietf-uri
Prefix:       uri
Reference:    RFC GGGG
Maintained by IANA:  N
```

# 7. References

## 7.1. Normative References

[I-D.ietf-netconf-udp-client-server]
Feng, A. H., Francois, P., and K. Watsen, "YANG Groupings for UDP Clients and UDP Servers", Work in Progress, Internet-Draft, draft-ietf-netconf-udp-client-server-07, 14 May 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-udp-client-server-07>.

[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>.

[RFC7317] Bierman, A. and M. Bjorklund, "A YANG Data Model for System Management", RFC 7317, DOI 10.17487/RFC7317, August 2014, <https://www.rfc-editor.org/info/rfc7317>.

[RFC7617] Reschke, J., "The 'Basic' HTTP Authentication Scheme", RFC 7617, DOI 10.17487/RFC7617, September 2015, <https://www.rfc-editor.org/info/rfc7617>.

[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>.

[RFC9110] Fielding, R., Ed., Nottingham, M., Ed., and J. Reschke, Ed., "HTTP Semantics", STD 97, RFC 9110, DOI 10.17487/RFC9110, June 2022, <https://www.rfc-editor.org/info/rfc9110>.

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

[RFC9643] Watsen, K. and M. Scharf, "YANG Groupings for TCP Clients and TCP Servers", RFC 9643, DOI 10.17487/RFC9643, October 2024, <https://www.rfc-editor.org/info/rfc9643>.

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

## 7.2. Informative References

[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-26, 24 April 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-http-client-server-26>.

[I-D.ietf-netconf-netconf-client-server]
Watsen, K., "A YANG Data Model for NETCONF Clients and Servers", Work in Progress, Internet-Draft, draft-ietf-netconf-netconf-client-server-39, 24 April 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-netconf-client-server-39>.

[I-D.ietf-netconf-restconf-client-server]
Watsen, K., "A YANG Data Model for RESTCONF Clients and Servers", Work in Progress, Internet-Draft, draft-ietf-netconf-restconf-client-server-43, 24 April 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-restconf-client-server-43>.

[I-D.ietf-netmod-system-config]
Ma, Q., Wu, Q., and C. Feng, "System-defined Configuration", Work in Progress, Internet-Draft, draft-ietf-netmod-system-config-12, 12 February 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-netmod-system-config-12>.

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

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <https://www.rfc-editor.org/info/rfc3986>.

[RFC4252] Ylonen, T. and C. Lonvick, Ed., "The Secure Shell (SSH) Authentication Protocol", RFC 4252, DOI 10.17487/RFC4252, January 2006, <https://www.rfc-editor.org/info/rfc4252>.

[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>.

[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>.

[RFC9000] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <https://www.rfc-editor.org/info/rfc9000>.

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

[RFC9642] Watsen, K., "A YANG Data Model for a Keystore", RFC 9642, DOI 10.17487/RFC9642, October 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, October 2024, <https://www.rfc-editor.org/info/rfc9644>.

## Appendix A. Change Log

### A.1. 00 to 01

* Modified Abstract and Intro to be more accurate wrt intended applicability.

* In ietf-http-client, removed "protocol-version" and all auth schemes except "basic".

* In ietf-http-client, factored out "client-identity-grouping" for proxy connections.

* In ietf-http-server, removed "choice required-or-optional" and "choice inline-or-external".

* In ietf-http-server, moved the basic auth under a "choice auth-type" limited by new "feature basic-auth".

### A.2. 01 to 02

* Removed the unused "external-client-auth-supported" feature from ietf-http-server.

### A.3. 02 to 03

* Removed "protocol-versions" from ietf-http-server based on HTTP WG feedback.

* Slightly restructured the "proxy-server" definition in ietf-http-client.

* Added http-client example show proxy server use.

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

### A.4. 03 to 04

* Added a parent "container" to "client-identity-grouping" so that it could be better used by the proxy model.

* Added a "choice" to the proxy model enabling selection of proxy types.

* Added ’http-client-stack-grouping’ and ’http-server-listen-stack-grouping’ convenience groupings.

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

* Updated the Security Considerations section.

### A.5. 04 to 05

* Fixed titles and a ref in the IANA Considerations section

* Cleaned up examples (e.g., removed FIXMEs)

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

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

### A.6. 05 to 06

* Removed note questioning if okay for app to augment-in a ’path’ node when needed, discussed during the 108 session.

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

### A.7. 06 to 07

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

* Renamed ’client-auth-config-supported’ to ’client-auth-supported’ consistent with other drafts.

* Wrapped ’container basic’ choice inside a ’case basic’ per best practice.

* Aligned modules with ‘pyang -f‘ formatting.

* Fixed nits found by YANG Doctor reviews.

### A.8. 07 to 08

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

* Minor editorial nits

### A.9. 08 to 09

* 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.10. 09 to 10

* NO UPDATE.

### A.11. 10 to 11

* Updated per Shepherd reviews impacting the suite of drafts.

### A.12. 11 to 12

* Updated per Shepherd reviews impacting the suite of drafts.

### A.13. 12 to 13

* Updated per Tom Petch reviews.
* Renamed draft title to limit to HTTP 1.1 and 2.0.
* Added refs to RFCs 7317, 7617, and 9110.
* Added "if-feature local-users-supported" to "feature basic-auth".

### A.14. 13 to 14

* Addresses AD review comments.
* Added note to Editor to fix line foldings.
* Removed "Conventions" section as there are no "BASE64VALUE=" values used in draft.
* Clarified that the modules, when implemented, do not define any protocol-accessible nodes.
* Added Security Considerations text to also look a SC-section from imported modules.
* Removed "A wrapper around the foobar parameters to avoid name collisions" text.
* Removed "public-key-format" and "public-key" nodes from examples.

### A.15. 14 to 15

* Addresses AD review by Rob Wilton.

### A.16. 15 to 16

* Addresses 1st-round of IESG reviews.

### A.17. 16 to 18

* Addresses issues found in OpsDir review of the ssh-client-server draft.

* s/defines/presents/ in a few places.

* Add refs to where the ’operational’ and ’system’ datastores are defined.

* Renamed Security Considerations section s/Template for/ Considerations for/.

* Updated "http-client-stack-grouping" and "http-server-listen-stack-grouping" for HTTP/3.

### A.18. 18 to 19

* Address IESG review comments.

### A.19. 19 to 20

* Updated to reflect comments from Paul Wouters.

### A.20. 23 to 24

* Replaced ’uri’ as string with a container with components.

* Added ’if-feature full-uri-supported’ statement to the ’path’, ’query’ and ’fragment’ components

* Replaced ’protocol-versions’ as ’bits’ with a min/max range

* Fixed ’proxy-connect’ to reflect RFC 9110 (not legacy the HTTP/S Proxy mechanism)

* Added support for CONNECT-UDP per RFC 9298

* Eliminated the ’http-client-common-grouping’ grouping (not needed after updating the proxy solution)

* Renamed 'http-server-stack-grouping' to 'http-server-listen-stack-grouping', to make way for a possible future 'http-server-callhome-stack-grouping'

### A.21. 24 to 25

* Updated 2nd paragraph in Security Considerations section to match RFC Editor's version for other drafts in the suite of client-server drafts.

* Removed the 'full-uri-supported' feature added in -24, after discussing some with Mahesh.

### A.22. 25 to 26

* Updated to reflect some of Med's AD-review comments

### A.23. 26 to 27

* Added ability for HTTP server's to also limit what HTTP versions are supported.

### A.24. 27 to 28

* Factored the "uri" node into a grouping defined in another module.

## Acknowledgements

The authors would like to thank the following for lively discussions on list and in the halls (ordered by first name): Ben Schwartz, Darrel Miller, テ詠ic Vyncke, Francesca Palombini, Mark Nottingham, Mahesh Jethanandani, Med Boucladair, Mike Bishop, Murray Kucherawy, Orie Steele, Rob Wilton, Robert Varga, Roman Danyliw, Shivan Sahib, Willy Tarreau, and Zaheduzzaman Sarker.

## Author's Address

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