Workgroup: NETCONF Working Group
Internet-Draft:
draft-ietf-netconf-http-client-server-16
Published: 4 February 2024
Intended Status: Standards Track
Expires: 7 August 2024
Authors: K. Watsen
Watsen Networks

YANG Groupings for HTTP 1.1/2.0 Clients and HTTP Servers

Abstract

This document defines 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).

Editorial Note (To be removed by RFC Editor)

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

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

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

* DDDD --> the assigned RFC value for draft-ietf-netconf-tcp-client-server

* FFFF --> the assigned RFC value for draft-ietf-netconf-tls-client-server

* GGGG --> 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:

* 2024-02-04 --> 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.

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 7 August 2024.

Copyright Notice

Table of Contents

1. Introduction

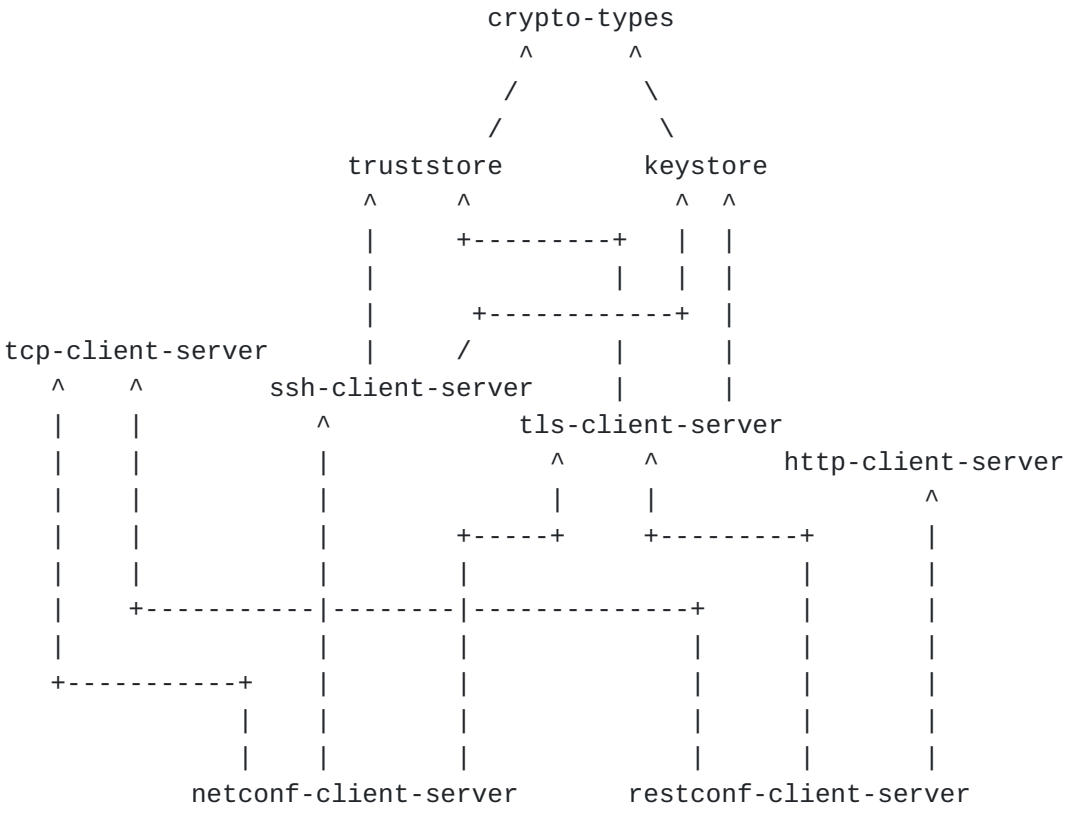
This document defines two YANG 1.1 [RFC7950] modules: the first defines a minimal grouping for configuring an HTTP client, and the second defines a minimal grouping for configuring an HTTP server. It is intended that these groupings will be used to help define the configuration for simple HTTP-based protocols (not for complete web servers or browsers).

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 both the clients and servers of both the NETCONF [RFC6241] and RESTCONF [RFC8040] protocols.

The dependency relationship between the primary 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	[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 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

[I-D.ietf-netconf-trust-anchors] and [I-D.ietf-netconf-keystore], trust anchors and keys installed during manufacturing are expected to appear in <operational>.

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 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-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:
  +-- proxy-connect
  +-- basic-auth
  +-- tcp-supported
  +-- tls-supported
```

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" statements:

- *http-client-identity-grouping
- *http-client-grouping
- *http-client-stack-grouping

Each of these groupings are presented in the following subsections.

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

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

```
grouping http-client-identity-grouping:
  +-- client-identity!
     +-- (auth-type)
        +--:(basic)
           +-- basic {basic-auth}?
              +-- user-id                string
              +---u ct:password-grouping
```

Comments:

*This grouping exists because it is used three times by the "http-client-grouping" discussed in Section 2.1.2.2.

*The "client-identity" node is a "presence" container so the mandatory descendant nodes do not imply that this node must be configured, as a client identity may be configured at protocol layers.

*The "basic" authentication scheme is the only scheme defined by this module, albeit it must be enabled via the "basic-auth" feature (see Section 2.1.1).

*Other authentication schemes MAY be augmented in as needed by the application.

2.1.2.2. The "http-client-grouping" Grouping

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

```
grouping http-client-grouping:
  +---u http-client-identity-grouping
  +-- proxy-connect! {proxy-connect}?
     +-- (proxy-type)
        +--:(http)
        |  +-- http-proxy
        |     +-- tcp-client-parameters
        |     |  +---u tcpc:tcp-client-grouping
        |     +-- http-client-parameters
        |        +---u http-client-identity-grouping
        +--:(https)
           +-- https-proxy
              +-- tcp-client-parameters
              |  +---u tcpc:tcp-client-grouping
              +-- tls-client-parameters
              |  +---u tlsc:tls-client-grouping
              +-- http-client-parameters
                 +---u http-client-identity-grouping
```

Comments:

*The "http-client-grouping" primarily (not including the proxy configuration) defines the configuration for just "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 2.1.2.3).

*Beyond configuring the client's identity, via the "http-client-identity-grouping" grouping discussed in Section 2.1.2.1, this grouping defines support for HTTP-proxies, albeit it must be enabled via a "feature" statement.

*The "proxy-connect" node is a "presence" container so the mandatory descendant nodes do not imply that this node must be configured, assuming the server supports the "proxy-connect" feature.

*For the referenced grouping statement(s):

-The "http-client-identity-grouping" grouping is discussed in Section 2.1.2.1.
-The "tcp-client-grouping" grouping is discussed in Section 3.1.2.1 of [I-D.ietf-netconf-tcp-client-server].
-The "tls-client-grouping" grouping is discussed in Section 3.1.2.1 of [I-D.ietf-netconf-tls-client-server].

2.1.2.3. The "http-client-stack-grouping" Grouping

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

```
grouping http-client-stack-grouping:
  +-- (transport)
     +--:(tcp) {tcp-supported}?
     |  +-- tcp
     |     +-- tcp-client-parameters
     |     |  +---u tcpc:tcp-client-grouping
     |     +-- http-client-parameters
     |        +---u http-client-grouping
     +--:(tls) {tls-supported}?
        +-- tls
           +-- tcp-client-parameters
           |  +---u tcpc:tcp-client-grouping
           +-- tls-client-parameters
           |  +---u tlsc:tls-client-grouping
           +-- http-client-parameters
              +---u http-client-grouping
```

Comments:

*The "http-client-stack-grouping" is a convenience grouping for consuming modules. It defines both the "HTTP" and "HTTPS" protocol stacks, with each option enabled by a "feature" statement for application control.

*For the referenced grouping statement(s):

-The "tcp-client-grouping" grouping is discussed in Section 3.1.2.1 of [I-D.ietf-netconf-tcp-client-server].
-The "tls-client-grouping" grouping is discussed in Section 3.1.2.1 of [I-D.ietf-netconf-tls-client-server].
-The "http-client-grouping" grouping is discussed in Section 2.1.2.2 in this document.

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 two examples showing the http-client-grouping populated with some data.

The following example illustrates the case where the HTTP client connects directly to an HTTP server. Note, the information identifying the remote server (e.g., its hostname) would be configured in the "tcp-client-grouping" (not shown).

```
<!-- 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">
  <client-identity>
    <basic>
      <user-id>bob</user-id>
      <cleartext-password>secret</cleartext-password>
    </basic>
  </client-identity>
</http-client>
```

The following example illustrates the same client connecting through an HTTP proxy. This example is consistent with examples presented in Section 2.2.1 of [I-D.ietf-netconf-trust-anchors] and Section 2.2.1 of [I-D.ietf-netconf-keystore].

```
=============== 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">
  <client-identity>
    <basic>
      <user-id>bob</user-id>
      <cleartext-password>secret</cleartext-password>
    </basic>
  </client-identity>
  <proxy-connect>
    <https-proxy>
      <tcp-client-parameters>
        <remote-address>corp-fw2.example.com</remote-address>
        <keepalives>
          <idle-time>7200</idle-time>
          <max-probes>9</max-probes>
          <probe-interval>75</probe-interval>
        </keepalives>
      </tcp-client-parameters>
      <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</c\
entral-truststore-reference>
          </ca-certs>
          <ee-certs>
            <central-truststore-reference>trusted-server-ee-certs</c\
entral-truststore-reference>
          </ee-certs>
        </server-authentication>
      </tls-client-parameters>
      <http-client-parameters>
        <client-identity>
          <basic>
            <user-id>local-app-1</user-id>
            <cleartext-password>secret</cleartext-password>
          </basic>
        </client-identity>
```

```
      </http-client-parameters>
    </https-proxy>
  </proxy-connect>
</http-client>
```

2.3. YANG Module

This YANG module has normative references to [RFC6991] [RFC7617], [RFC9110], [I-D.ietf-netconf-crypto-types], [I-D.ietf-netconf-tcp-client-server], and [I-D.ietf-netconf-tls-client-server].

<CODE BEGINS> file "ietf-http-client@2024-02-04.yang"

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

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

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

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

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

  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 reusable groupings for HTTP clients that
     can be used as a basis for specific HTTP client instances.

     Copyright (c) 2023 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 2024-02-04 {
  description
    "Initial version";
  reference
    "RFC GGGG: YANG Groupings for HTTP Clients and HTTP Servers";
}

// Features

feature proxy-connect {
  description
    "Indicates that the server supports configuring HTTP
     clients to connect to a remote HTTP server via an
     HTTP proxy.";
}

feature basic-auth {
  description
    "Indicates that the server supports configuring HTTP
     clients to authenticate themselves to an HTTP server
     using the 'basic' HTTP authentication scheme.";
  reference
    "RFC 7617: The 'Basic' HTTP Authentication Scheme";
}

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

feature tls-supported {
  description
    "Indicates that the server supports configuring
     HTTP 1.1/2.0 clients to initiate HTTP 1.1/2.0
```

```
     connections over TLS.";
  reference
    "RFC 9110: HTTP Semantics";
}

// Groupings

grouping http-client-identity-grouping {
  description
    "A grouping to provide HTTP credentials used by the
     client to authenticate itself to the HTTP server.";
  container client-identity {
    nacm:default-deny-write;
    presence
      "Indicates that a client identity has been configured.
       This statement is present so the mandatory descendant
       nodes do not imply that this node must be configured.";
    description
      "The identity the HTTP client should use when
       authenticating itself to the HTTP server.";
    choice auth-type {
      mandatory true;
      description
        "A choice amongst available authentication types.";
      case basic {
        container basic {
          if-feature "basic-auth";
          leaf user-id {
            type string;
            mandatory true;
            description
              "The user-id for the authenticating client.";
          }
          uses ct:password-grouping {
            description
              "The password for the authenticating client.";
          }
          description
            "The 'basic' HTTP scheme credentials.";
          reference
            "RFC 7617: The 'Basic' HTTP Authentication Scheme";
        }
      }
    }
  }
} // grouping http-client-identity-grouping

grouping http-client-grouping {
  description
```

```
    "A reusable grouping for configuring a HTTP client.

     This grouping is expected to be used in conjunction with
     other configurations providing, e.g., the hostname or IP
     address and port number the client initiates connections
     to.

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

  uses http-client-identity-grouping;

  container proxy-connect {
    nacm:default-deny-write;
    if-feature "proxy-connect";
    presence
      "Indicates that a proxy server connections have been
       configured.  This statement is present so the mandatory
       descendant nodes do not imply that this node must be
       configured.";
    description
      "Configures the proxy server the HTTP-client is to
       connect through.";
    choice proxy-type {
      mandatory true;
      description
        "Choice amongst proxy server types.";
      case http {
        container http-proxy {
          description
            "Container for HTTP Proxy (Web Proxy) server
             configuration parameters.";
          container tcp-client-parameters {
            description
              "TCP client parameters.";
            uses tcpc:tcp-client-grouping;
          }
          container http-client-parameters {
            description
              "HTTP client parameters.";
            uses http-client-identity-grouping;
          }
        }
```

```
      }
      case https {
        container https-proxy {
          description
            "Container for HTTPS Proxy (Secure Web Proxy) server
             configuration parameters.";
          container tcp-client-parameters {
            description
              "TCP client parameters.";
            uses tcpc:tcp-client-grouping;
          }
          container tls-client-parameters {
            description
              "TLS client parameters.";
            uses tlsc:tls-client-grouping;
          }
          container http-client-parameters {
            description
              "HTTP client parameters.";
            uses http-client-identity-grouping;
          }
        }
      }
    }
  }
} // grouping http-client-grouping

grouping http-client-stack-grouping {
  description
    "A grouping that defines common HTTP-based protocol stacks.";
  choice transport {
    mandatory true;
    description
      "Choice amongst various transports type.  TCP, with and
       without TLS are defined here, with 'feature' statements
       so that they may be disabled.  Other transports MAY be
       augmented in as 'case' statements by future efforts.";
    case tcp {
      if-feature "tcp-supported";
      container tcp {
        description
          "Container for TCP-based HTTP protocols.";
        container tcp-client-parameters {
          description
            "TCP client parameters.";
          uses tcpc:tcp-client-grouping;
        }
        container http-client-parameters {
          description
```

```
              "HTTP client parameters.";
            uses http-client-grouping;
          }
        }
      }
      case tls {
        if-feature "tls-supported";
        container tls {
          description
            "Container for TLS-based HTTP protocols.";
          container tcp-client-parameters {
            description
              "TCP client parameters.";
            uses tcpc:tcp-client-grouping;
          }
          container tls-client-parameters {
            description
              "TLS client parameters.";
            uses tlsc:tls-client-grouping;
          }
          container http-client-parameters {
            description
              "HTTP client parameters.";
            uses http-client-grouping;
          }
        }
      }
    }
  } // http-client-stack-grouping

}
```

<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 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-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
```

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-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 "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-stack-grouping" Grouping

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

```
grouping http-server-stack-grouping:
  +-- (transport)
     +--:(tcp) {tcp-supported}?
     |  +-- tcp
     |     +-- tcp-server-parameters
     |     |  +---u tcps:tcp-server-grouping
     |     +-- http-server-parameters
     |        +---u http-server-grouping
     +--:(tls) {tls-supported}?
        +-- tls
           +-- tcp-server-parameters
           |  +---u tcps:tcp-server-grouping
           +-- tls-server-parameters
           |  +---u tlss:tls-server-grouping
           +-- http-server-parameters
              +---u http-server-grouping
```

Comments:

*The "http-server-stack-grouping" is a convenience grouping for consuming modules. It defines both the "HTTP" and "HTTPS" protocol stacks, with each option enabled by a "feature" statement for application control.

*For the referenced grouping statement(s):

-The "tcp-server-grouping" grouping is discussed in Section 4.1.2.1 of [I-D.ietf-netconf-tcp-client-server].
-The "tls-server-grouping" grouping is discussed in Section 4.1.2.1 of [I-D.ietf-netconf-tls-client-server].
-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 normative references to [RFC6991], [RFC7317], [RFC7617], [RFC9110], [I-D.ietf-netconf-tcp-client-server], and [I-D.ietf-netconf-tls-client-server].

<CODE BEGINS> file "ietf-http-server@2024-02-04.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";
  }

  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 reusable groupings for HTTP servers that
     can be used as a basis for specific HTTP server instances.

     Copyright (c) 2023 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 2024-02-04 {
   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";
}

// Groupings

grouping http-server-grouping {
  description
    "A reusable grouping for configuring an HTTP 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
     '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-stack-grouping {
  description
    "A grouping that defines common HTTP-based protocol stacks.";
  choice transport {
    mandatory true;
    description
      "Choice amongst various transports type.  TCP, with and
       without TLS are defined here, with 'feature' statements
       so that they may be disabled.  Other transports MAY be
       augmented in as 'case' statements by future efforts.";
    case tcp {
      if-feature "tcp-supported";
      container tcp {
        description
          "Container for TCP-based HTTP protocols.";
        container tcp-server-parameters {
          description
            "TCP-level server parameters to
             listen for HTTP connections.";
          uses tcps:tcp-server-grouping;
        }
        container http-server-parameters {
          description
            "HTTP-level server parameters to
             listen for HTTP connections.";
          uses http-server-grouping;
        }
      }
    }
    case tls {
      if-feature "tls-supported";
      container tls {
        description
          "Container for TLS-based HTTP protocols.";
```

```
          container tcp-server-parameters {
            description
              "TCP-level server parameters to
               listen for HTTPS connections.";
            uses tcps:tcp-server-grouping;
          }
          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;
          }
        }
      }
    }
  } // http-server-stack-grouping

}
```

<CODE ENDS>

4. Security Considerations

4.1. Template for the "ietf-http-client" YANG Module

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

The "ietf-http-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 this module only defines groupings, these considerations are primarily for the designers of other modules that use these groupings.

Please be aware that this YANG module uses groupings from other YANG modules that define nodes that may be considered sensitive or

vulnerable in network environments. Please review the Security Considerations for dependent YANG modules for information as to which nodes may be considered sensitive or vulnerable in network environments.

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

The follwing writable data nodes defined in this YANG module are considered sensitive or vulnerable in network environments:

* The "client-identity" node in the "http-client-identity-grouping" grouping may be considered sensitive or vulnerable in some network environments. For this reason, it NACM extension "default-deny-write" has been applied to it.

* The "proxy-connect" node in the "http-client-grouping" grouping may be considered sensitive or vulnerable in some network environments. For this reason, it NACM extension "default-deny-write" has been applied to it.

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

4.2. Template for the "ietf-http-server" YANG Module

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

The "ietf-http-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 this module only defines groupings, these considerations are primarily for the designers of other modules that use these groupings.

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

which nodes may be considered sensitive or vulnerable in network environments.

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

The follwing writable data nodes defined in this YANG module 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, it NACM extension "default-deny-write" has been applied to it.

* The "client-authentication" node in the "http-server-grouping" grouping may be considered sensitive or vulnerable in some network environments. For this reason, it NACM extension "default-deny-write" has been applied to it.

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

5. IANA Considerations

5.1. The "IETF XML" Registry

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

URI: urn:ietf:params:xml:ns:yang:ietf-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.

5.2. The "YANG Module Names" Registry

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

```
name:         ietf-http-client
namespace:    urn:ietf:params:xml:ns:yang:ietf-http-client
prefix:       httpc
reference:    RFC GGGG

name:         ietf-http-server
namespace:    urn:ietf:params:xml:ns:yang:ietf-http-server
prefix:       https
reference:    RFC GGGG
```

6. References

6.1. Normative 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-29, 26 January 2024, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-crypto-types-29>.

[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-19, 29 January 2024, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-tcp-client-server-19>.

[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-36, 29 January 2024, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-tls-client-server-36>.

[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>.

6.2. Informative References

[I-D.ietf-netconf-http-client-server]
Watsen, K., "YANG Groupings for HTTP 1.1/2.0 Clients and HTTP Servers", Work in Progress, Internet-Draft, draft-ietf-netconf-http-client-server-15, 26 January 2024, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-http-client-server-15>.

[I-D.ietf-netconf-keystore] Watsen, K., "A YANG Data Model for a Keystore", Work in Progress, Internet-Draft, draft-ietf-netconf-keystore-30, 26 January 2024, <https://

datatracker.ietf.org/doc/html/draft-ietf-netconf-keystore-30>.

[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-31, 26 January 2024, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-netconf-client-server-31>.

[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-31, 26 January 2024, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-restconf-client-server-31>.

[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-35, 26 January 2024, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-ssh-client-server-35>.

[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-23, 26 January 2024, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-trust-anchors-23>.

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

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

[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>.

Appendix A. Change Log

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

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-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.

Acknowledgements

The authors would like to thank for following for lively discussions on list and in the halls (ordered by first name): Ben Schwartz, Mark Nottingham, Mahesh Jethanandani, Rob Wilton, and Willy Tarreau.

Author's Address

Kent Watsen
Watsen Networks

Email: kent+ietf@watsen.net