NETCONF Working Group
Internet-Draft
Intended status: Standards Track
Expires: May 3, 2018

K. Watsen
Juniper Networks
G. Wu
Cisco Systems

October 30, 2017

YANG Groupings for TLS Clients and TLS Servers
draft-ietf-netconf-tls-client-server-05

Abstract

This document defines three YANG modules: the first defines groupings for a generic TLS client, the second defines groupings for a generic TLS server, and the third defines common identities and groupings used by both the client and the server. It is intended that these groupings will be used by applications using the TLS protocol.

Editorial Note (To be removed by RFC Editor)

This draft contains many 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.

This document contains references to other drafts in progress, both in the Normative References section, as well as in body text throughout. Please update the following references to reflect their final RFC assignments:

- I-D.ietf-netconf-keystore

Artwork in this document contains shorthand references to drafts in progress. Please apply the following replacements:

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

- "YYYY" --> the assigned RFC value for I-D.ietf-netconf-keystore

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

- "2017-10-30" --> the publication date of this draft

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

- Appendix A. Change Log

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

This document defines three YANG [RFC7950] modules: the first defines a grouping for a generic TLS client, the second defines a grouping for a generic TLS server, and the third defines identities and groupings common to both the client and the server (TLS is defined in [RFC5246]). It is intended that these groupings will be used by applications using the TLS protocol. For instance, these groupings could be used to help define the data model for an HTTPS [RFC2818] server or a NETCONF over TLS [RFC7589] based server.

The client and server YANG modules in this document each define one grouping, which is focused on just TLS-specific configuration, and specifically avoids any transport-level configuration, such as what ports to listen-on or connect-to. This enables applications the opportunity to define their own strategy for how the underlying TCP connection is established. For instance, applications supporting NETCONF Call Home [RFC8071] could use the grouping for the TLS parts it provides, while adding data nodes for the TCP-level call-home configuration.

2. Terminology

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.

3. The TLS Client Model

The TLS client model presented in this section contains one YANG grouping, to just configure the TLS client, omitting, for instance, any configuration for which IP address or port the client should connect to.

This grouping references data nodes defined by the keystore model [I-D.ietf-netconf-keystore]. For instance, a reference to the keystore model is made to indicate which trusted CA certificate a client should use to authenticate the server's certificate.

3.1. Tree Diagram

The following tree diagram [I-D.ietf-netmod-yang-tree-diagrams] provides an overview of the data model for the "ietf-tls-client" module.

```
   module: ietf-tls-client

     grouping tls-client-grouping
       +---- client-identity
       |  +---- (auth-type)?
       |     +--:(certificate)
       |        +---- certificate
       |           +---- algorithm?
       |           |       identityref
       |           +---- private-key?                              union
       |           +---- public-key?                               binary
       |           +---x generate-private-key
       |           |  +---w input
       |           |     +---w algorithm    identityref
       |           +---- certificates
       |           |  +---- certificate* [name]
       |           |     +---- name?    string
       |           |     +---- value?   binary
       |           +---x generate-certificate-signing-request
       |              +---w input
       |              |  +---w subject       binary
       |              |  +---w attributes?   binary
       |              +--ro output
       |                 +--ro certificate-signing-request    binary
       +---- server-auth
       |  +---- pinned-ca-certs?       ks:pinned-certificates
       |  +---- pinned-server-certs?   ks:pinned-certificates
       +---- hello-params {tls-client-hello-params-config}?
          +---- tls-versions
          |  +---- tls-version*   identityref
          +---- cipher-suites
             +---- cipher-suite*   identityref
```

3.2. Example Usage

This section shows how it would appear if the tls-client-grouping were populated with some data. This example is consistent with the examples presented in Section 2.2 of [I-D.ietf-netconf-keystore].

```
  [ note: '\' line wrapping for formatting only]

  <!-- hypothetical example, as groupings don't have instance data -->
  <tls-client xmlns="urn:ietf:params:xml:ns:yang:ietf-tls-client">

    <!-- how this client will authenticate itself to the server -->
    <client-identity>
      <certificate>
        <algorithm xmlns:ks="urn:ietf:params:xml:ns:yang:ietf-keystore"\
  >ks:secp521r1</algorithm>
        <private-key>base64encodedvalue==</private-key>
        <public-key>base64encodedvalue==</public-key>
        <certificates>
          <certificate>
            <name>domain certificate</name>
            <value>base64encodedvalue==</value>
          </certificate>
        </certificates>
      </certificate>
    </client-identity>

    <!-- which certificates will this client trust -->
    <server-auth>
      <pinned-ca-certs>deployment-specific-ca-certs</pinned-ca-certs>
      <pinned-server-certs>explicitly-trusted-client-certs</pinned-serv\
  er-certs>
    </server-auth>

  </tls-client>
```

3.3. YANG Module

This YANG module has a normative references to [RFC6991] and [I-D.ietf-netconf-keystore].

```
<CODE BEGINS> file "ietf-tls-client@2017-10-30.yang"
module ietf-tls-client {
  yang-version 1.1;

  namespace "urn:ietf:params:xml:ns:yang:ietf-tls-client";
  prefix "tlsc";

  import ietf-tls-common {
    prefix tlscmn;
    revision-date 2017-10-30; // stable grouping definitions
    reference
      "RFC XXXX: YANG Groupings for TLS Clients and TLS Servers";
  }
```

```
     import ietf-keystore {
       prefix ks;
       reference
         "RFC YYYY: Keystore Model";
     }

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

     contact
      "WG Web:   <http://tools.ietf.org/wg/netconf/>
       WG List:  <mailto:netconf@ietf.org>

       Author:   Kent Watsen
                 <mailto:kwatsen@juniper.net>

       Author:   Gary Wu
                 <mailto:garywu@cisco.com>";


     description
      "This module defines a reusable grouping for a TLS client that
       can be used as a basis for specific TLS client instances.

       Copyright (c) 2017 IETF Trust and the persons identified as
       authors of the code. All rights reserved.

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

       This version of this YANG module is part of RFC XXXX; see
       the RFC itself for full legal notices.";

     revision "2017-10-30" {
       description
        "Initial version";
       reference
        "RFC XXXX: YANG Groupings for TLS Clients and TLS Servers";
     }

     // features

     feature tls-client-hello-params-config {
       description
```

```
          "TLS hello message parameters are configurable on a TLS
           client.";
      }

      // groupings

      grouping tls-client-grouping {
        description
          "A reusable grouping for configuring a TLS client without
           any consideration for how an underlying TCP session is
           established.";

        container client-identity {
          description
            "The credentials used by the client to authenticate to
             the TLS server.";

          choice auth-type {
            description
              "The authentication type.";
            container certificate {
              uses ks:private-key-grouping;
              uses ks:certificate-grouping;
              description
                "Choice statement for future augmentations.";
            }
          }
        }

        container server-auth {
          must 'pinned-ca-certs or pinned-server-certs';
          description
            "Trusted server identities.";
          leaf pinned-ca-certs {
            type ks:pinned-certificates;
            description
              "A reference to a list of certificate authority (CA)
               certificates used by the TLS client to authenticate
               TLS server certificates.  A server certificate is
               authenticated if it has a valid chain of trust to
               a configured pinned CA certificate.";
          }

          leaf pinned-server-certs {
            type ks:pinned-certificates;
            description
              "A reference to a list of server certificates used by
               the TLS client to authenticate TLS server certificates.
```

```
              A server certificate is authenticated if it is an
              exact match to a configured pinned server certificate.";
          }
        }

        container hello-params {
          if-feature tls-client-hello-params-config;
          uses tlscmn:hello-params-grouping;
          description
            "Configurable parameters for the TLS hello message.";
        }

      } // end tls-client-grouping

    }
    <CODE ENDS>
```

4. The TLS Server Model

The TLS server model presented in this section contains one YANG grouping, for just the TLS-level configuration, omitting, for instance, configuration for which ports to open to listen for connections on.

This grouping references data nodes defined by the keystore model [I-D.ietf-netconf-keystore]. For instance, a reference to the keystore model is made to indicate which certificate a server should present.

4.1. Tree Diagram

The following tree diagram [I-D.ietf-netmod-yang-tree-diagrams] provides an overview of the data model for the "ietf-tls-server" module.

```
   module: ietf-tls-server

     grouping tls-server-grouping
       +---- server-identity
       |  +---- algorithm?                                identityref
       |  +---- private-key?                              union
       |  +---- public-key?                               binary
       |  +---x generate-private-key
       |  |  +---w input
       |  |     +---w algorithm    identityref
       |  +---- certificates
       |  |  +---- certificate* [name]
       |  |     +---- name?    string
       |  |     +---- value?   binary
       |  +---x generate-certificate-signing-request
       |     +---w input
       |     |  +---w subject       binary
       |     |  +---w attributes?   binary
       |     +--ro output
       |        +--ro certificate-signing-request    binary
       +---- client-auth
       |  +---- pinned-ca-certs?       ks:pinned-certificates
       |  +---- pinned-client-certs?   ks:pinned-certificates
       +---- hello-params {tls-server-hello-params-config}?
          +---- tls-versions
          |  +---- tls-version*   identityref
          +---- cipher-suites
             +---- cipher-suite*   identityref
```

4.2. Example Usage

This section shows how it would appear if the tls-server-grouping were populated with some data. This example is consistent with the examples presented in Section 2.2 of [I-D.ietf-netconf-keystore].

```
  [ note: '\' line wrapping for formatting only]

  <tls-server xmlns="urn:ietf:params:xml:ns:yang:ietf-tls-server">

    <!-- how this server will authenticate itself to the client -->
    <server-identity>
      <algorithm xmlns:ks="urn:ietf:params:xml:ns:yang:ietf-keystore">k\
  s:secp521r1</algorithm>
      <private-key>base64encodedvalue==</private-key>
      <public-key>base64encodedvalue==</public-key>
      <certificates>
        <certificate>
          <name>domain certificate</name>
          <value>base64encodedvalue==</value>
        </certificate>
      </certificates>
    </server-identity>

    <!-- which certificates will this server trust -->
    <client-auth>
      <pinned-ca-certs>deployment-specific-ca-certs</pinned-ca-certs>
      <pinned-client-certs>explicitly-trusted-client-certs</pinned-clie\
  nt-certs>
    </client-auth>

  </tls-server>
```

4.3. YANG Module

This YANG module has a normative references to [RFC6991], and [I-D.ietf-netconf-keystore].

```
<CODE BEGINS> file "ietf-tls-server@2017-10-30.yang"
module ietf-tls-server {
  yang-version 1.1;

  namespace "urn:ietf:params:xml:ns:yang:ietf-tls-server";
  prefix "tlss";

  import ietf-tls-common {
    prefix tlscmn;
    revision-date 2017-10-30; // stable grouping definitions
    reference
      "RFC XXXX: YANG Groupings for TLS Clients and TLS Servers";
  }

  import ietf-keystore {
    prefix ks;
```

```
      reference
        "RFC YYYY: Keystore Model";
    }

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

    contact
     "WG Web:   <http://tools.ietf.org/wg/netconf/>
      WG List:  <mailto:netconf@ietf.org>

      Author:   Kent Watsen
                <mailto:kwatsen@juniper.net>

      Author:   Gary Wu
                <mailto:garywu@cisco.com>";


    description
     "This module defines a reusable grouping for a TLS server that
      can be used as a basis for specific TLS server instances.

      Copyright (c) 2017 IETF Trust and the persons identified as
      authors of the code. All rights reserved.

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

      This version of this YANG module is part of RFC XXXX; see
      the RFC itself for full legal notices.";

    revision "2017-10-30" {
      description
       "Initial version";
      reference
       "RFC XXXX: YANG Groupings for TLS Clients and TLS Servers";
    }

    // features

    feature tls-server-hello-params-config {
      description
        "TLS hello message parameters are configurable on a TLS
         server.";
```

```
    }


    // groupings

    grouping tls-server-grouping {
      description
        "A reusable grouping for configuring a TLS server without
         any consideration for how underlying TCP sessions are
         established.";

      container server-identity {
        description
          "The list of certificates the TLS server will present when
           establishing a TLS connection in its Certificate message,
           as defined in Section 7.4.2 in RFC 5246.";
        reference
          "RFC 5246:
             The Transport Layer Security (TLS) Protocol Version 1.2";
        uses ks:private-key-grouping;
        uses ks:certificate-grouping;
      }

      container client-auth {
        description
          "A reference to a list of pinned certificate authority (CA)
           certificates and a reference to a list of pinned client
           certificates.";
        leaf pinned-ca-certs {
          type ks:pinned-certificates;
          description
            "A reference to a list of certificate authority (CA)
             certificates used by the TLS server to authenticate
             TLS client certificates.  A client certificate is
             authenticated if it has a valid chain of trust to
             a configured pinned CA certificate.";
        }
        leaf pinned-client-certs {
          type ks:pinned-certificates;
          description
            "A reference to a list of client certificates used by
             the TLS server to authenticate TLS client certificates.
             A clients certificate is authenticated if it is an
             exact match to a configured pinned client certificate.";
        }
      }

      container hello-params {
```

```
        if-feature tls-server-hello-params-config;
        uses tlscmn:hello-params-grouping;
        description
          "Configurable parameters for the TLS hello message.";
      }

    } // end tls-server-grouping

  }
  <CODE ENDS>
```

5. The TLS Common Model

The TLS common model presented in this section contains identities and groupings common to both TLS clients and TLS servers. The hello-params-grouping can be used to configure the list of TLS algorithms permitted by the TLS client or TLS server. The lists of algorithms are ordered such that, if multiple algorithms are permitted by the client, the algorithm that appears first in its list that is also permitted by the server is used for the TLS transport layer connection. The ability to restrict the the algorithms allowed is provided in this grouping for TLS clients and TLS servers that are capable of doing so and may serve to make TLS clients and TLS servers compliant with security policies.

Features are defined for algorithms that are OPTIONAL or are not widely supported by popular implementations. Note that the list of algorithms is not exhaustive.

5.1. Tree Diagram

The following tree diagram [I-D.ietf-netmod-yang-tree-diagrams] provides an overview of the data model for the "ietf-tls-common" module.

```
module: ietf-tls-common

  grouping hello-params-grouping
    +---- tls-versions
    |  +---- tls-version*   identityref
    +---- cipher-suites
       +---- cipher-suite*   identityref
```

5.2. Example Usage

This section shows how it would appear if the transport-params-grouping were populated with some data.

```
   <!-- hypothetical example, as groupings don't have instance data -->
   <hello-params
      xmlns="urn:ietf:params:xml:ns:yang:ietf-tls-common"
      xmlns:tlscmn="urn:ietf:params:xml:ns:yang:ietf-tls-common">
     <tls-versions>
       <tls-version>tlscmn:tls-1.1</tls-version>
       <tls-version>tlscmn:tls-1.2</tls-version>
     </tls-versions>
     <cipher-suites>
       <cipher-suite>tlscmn:dhe-rsa-with-aes-128-cbc-sha</cipher-suite>
       <cipher-suite>tlscmn:rsa-with-aes-128-cbc-sha</cipher-suite>
       <cipher-suite>tlscmn:rsa-with-3des-ede-cbc-sha</cipher-suite>
     </cipher-suites>
   </hello-params>
```

5.3. YANG Module

This YANG module has a normative references to [RFC2246], [RFC4346], [RFC4492], [RFC5246], [RFC5288], and [RFC5289].

```
   <CODE BEGINS> file "ietf-tls-common@2017-10-30.yang"
   module ietf-tls-common {
     yang-version 1.1;

     namespace "urn:ietf:params:xml:ns:yang:ietf-tls-common";
     prefix "tlscmn";

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

     contact
      "WG Web:   <http://tools.ietf.org/wg/netconf/>
       WG List:  <mailto:netconf@ietf.org>

       Author:   Kent Watsen
                 <mailto:kwatsen@juniper.net>

       Author:   Gary Wu
                 <mailto:garywu@cisco.com>";


     description
      "This module defines a common features, identities, and groupings
       for Transport Layer Security (TLS).

       Copyright (c) 2017 IETF Trust and the persons identified as
       authors of the code. All rights reserved.
```

This version of this YANG module is part of RFC XXXX; see
the RFC itself for full legal notices.";

```
  revision "2017-10-30" {
    description
     "Initial version";
    reference
     "RFC XXXX: YANG Groupings for TLS Clients and TLS Servers";
  }

  // features

  feature tls-1_0 {
    description
      "TLS Protocol Version 1.0 is supported.";
    reference
      "RFC 2246: The TLS Protocol Version 1.0";
  }

  feature tls-1_1 {
    description
      "TLS Protocol Version 1.1 is supported.";
    reference
      "RFC 4346: The Transport Layer Security (TLS) Protocol
                 Version 1.1";
  }

  feature tls-1_2 {
    description
      "TLS Protocol Version 1.2 is supported.";
    reference
      "RFC 5246: The Transport Layer Security (TLS) Protocol
                 Version 1.2";
  }

  feature tls-ecc {
    description
      "Elliptic Curve Cryptography (ECC) is supported for TLS.";
    reference
      "RFC 4492: Elliptic Curve Cryptography (ECC) Cipher Suites
                 for Transport Layer Security (TLS)";
```

```
    }

    feature tls-dhe {
      description
        "Ephemeral Diffie-Hellman key exchange is supported for TLS.";
      reference
        "RFC 5246: The Transport Layer Security (TLS) Protocol
                   Version 1.2";
    }

    feature tls-3des {
      description
        "The Triple-DES block cipher is supported for TLS.";
      reference
        "RFC 5246: The Transport Layer Security (TLS) Protocol
                   Version 1.2";
    }

    feature tls-gcm {
      description
        "The Galois/Counter Mode authenticated encryption mode is
         supported for TLS.";
      reference
        "RFC 5288: AES Galois Counter Mode (GCM) Cipher Suites for
                   TLS";
    }

    feature tls-sha2 {
      description
        "The SHA2 family of cryptographic hash functions is supported
         for TLS.";
      reference
        "FIPS PUB 180-4: Secure Hash Standard (SHS)";
    }

    // identities

    identity tls-version-base {
      description
        "Base identity used to identify TLS protocol versions.";
    }

    identity tls-1.0 {
      base tls-version-base;
      if-feature tls-1_0;
      description
        "TLS Protocol Version 1.0.";
      reference
```

```
        "RFC 2246: The TLS Protocol Version 1.0";
    }

    identity tls-1.1 {
      base tls-version-base;
      if-feature tls-1_1;
      description
        "TLS Protocol Version 1.1.";
      reference
        "RFC 4346: The Transport Layer Security (TLS) Protocol
                   Version 1.1";
    }

    identity tls-1.2 {
      base tls-version-base;
      if-feature tls-1_2;
      description
        "TLS Protocol Version 1.2.";
      reference
        "RFC 5246: The Transport Layer Security (TLS) Protocol
                   Version 1.2";
    }

    identity cipher-suite-base {
      description
        "Base identity used to identify TLS cipher suites.";
    }

    identity rsa-with-aes-128-cbc-sha {
      base cipher-suite-base;
      description
        "Cipher suite TLS_RSA_WITH_AES_128_CBC_SHA.";
      reference
        "RFC 5246: The Transport Layer Security (TLS) Protocol
                   Version 1.2";
    }

    identity rsa-with-aes-256-cbc-sha {
      base cipher-suite-base;
      description
        "Cipher suite TLS_RSA_WITH_AES_256_CBC_SHA.";
      reference
        "RFC 5246: The Transport Layer Security (TLS) Protocol
                   Version 1.2";
    }

    identity rsa-with-aes-128-cbc-sha256 {
      base cipher-suite-base;
```

```
      if-feature tls-sha2;
      description
        "Cipher suite TLS_RSA_WITH_AES_128_CBC_SHA256.";
      reference
        "RFC 5246: The Transport Layer Security (TLS) Protocol
                   Version 1.2";
    }

    identity rsa-with-aes-256-cbc-sha256 {
      base cipher-suite-base;
      if-feature tls-sha2;
      description
        "Cipher suite TLS_RSA_WITH_AES_256_CBC_SHA256.";
      reference
        "RFC 5246: The Transport Layer Security (TLS) Protocol
                   Version 1.2";
    }

    identity dhe-rsa-with-aes-128-cbc-sha {
      base cipher-suite-base;
      if-feature tls-dhe;
      description
        "Cipher suite TLS_DHE_RSA_WITH_AES_128_CBC_SHA.";
      reference
        "RFC 5246: The Transport Layer Security (TLS) Protocol
                   Version 1.2";
    }

    identity dhe-rsa-with-aes-256-cbc-sha {
      base cipher-suite-base;
      if-feature tls-dhe;
      description
        "Cipher suite TLS_DHE_RSA_WITH_AES_256_CBC_SHA.";
      reference
        "RFC 5246: The Transport Layer Security (TLS) Protocol
                   Version 1.2";
    }

    identity dhe-rsa-with-aes-128-cbc-sha256 {
      base cipher-suite-base;
      if-feature "tls-dhe and tls-sha2";
      description
        "Cipher suite TLS_DHE_RSA_WITH_AES_128_CBC_SHA256.";
      reference
        "RFC 5246: The Transport Layer Security (TLS) Protocol
                   Version 1.2";
    }
```

```
    identity dhe-rsa-with-aes-256-cbc-sha256 {
      base cipher-suite-base;
      if-feature "tls-dhe and tls-sha2";
      description
        "Cipher suite TLS_DHE_RSA_WITH_AES_256_CBC_SHA256.";
      reference
        "RFC 5246: The Transport Layer Security (TLS) Protocol
                   Version 1.2";
    }

    identity ecdhe-ecdsa-with-aes-128-cbc-sha256 {
      base cipher-suite-base;
      if-feature "tls-ecc and tls-sha2";
      description
        "Cipher suite TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256.";
      reference
        "RFC 5289: TLS Elliptic Curve Cipher Suites with
                   SHA-256/384 and AES Galois Counter Mode (GCM)";
    }

    identity ecdhe-ecdsa-with-aes-256-cbc-sha384 {
      base cipher-suite-base;
      if-feature "tls-ecc and tls-sha2";
      description
        "Cipher suite TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384.";
      reference
        "RFC 5289: TLS Elliptic Curve Cipher Suites with
                   SHA-256/384 and AES Galois Counter Mode (GCM)";
    }

    identity ecdhe-rsa-with-aes-128-cbc-sha256 {
      base cipher-suite-base;
      if-feature "tls-ecc and tls-sha2";
      description
        "Cipher suite TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256.";
      reference
        "RFC 5289: TLS Elliptic Curve Cipher Suites with
                   SHA-256/384 and AES Galois Counter Mode (GCM)";
    }

    identity ecdhe-rsa-with-aes-256-cbc-sha384 {
      base cipher-suite-base;
      if-feature "tls-ecc and tls-sha2";
      description
        "Cipher suite TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384.";
      reference
        "RFC 5289: TLS Elliptic Curve Cipher Suites with
                   SHA-256/384 and AES Galois Counter Mode (GCM)";
```

```
    }

    identity ecdhe-ecdsa-with-aes-128-gcm-sha256 {
      base cipher-suite-base;
      if-feature "tls-ecc and tls-gcm and tls-sha2";
      description
        "Cipher suite TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256.";
      reference
        "RFC 5289: TLS Elliptic Curve Cipher Suites with
                   SHA-256/384 and AES Galois Counter Mode (GCM)";
    }

    identity ecdhe-ecdsa-with-aes-256-gcm-sha384 {
      base cipher-suite-base;
      if-feature "tls-ecc and tls-gcm and tls-sha2";
      description
        "Cipher suite TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384.";
      reference
        "RFC 5289: TLS Elliptic Curve Cipher Suites with
                   SHA-256/384 and AES Galois Counter Mode (GCM)";
    }

    identity ecdhe-rsa-with-aes-128-gcm-sha256 {
      base cipher-suite-base;
      if-feature "tls-ecc and tls-gcm and tls-sha2";
      description
        "Cipher suite TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256.";
      reference
        "RFC 5289: TLS Elliptic Curve Cipher Suites with
                   SHA-256/384 and AES Galois Counter Mode (GCM)";
    }

    identity ecdhe-rsa-with-aes-256-gcm-sha384 {
      base cipher-suite-base;
      if-feature "tls-ecc and tls-gcm and tls-sha2";
      description
        "Cipher suite TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384.";
      reference
        "RFC 5289: TLS Elliptic Curve Cipher Suites with
                   SHA-256/384 and AES Galois Counter Mode (GCM)";
    }

    identity rsa-with-3des-ede-cbc-sha {
      base cipher-suite-base;
      if-feature tls-3des;
      description
        "Cipher suite TLS_RSA_WITH_3DES_EDE_CBC_SHA.";
      reference
```

```
          "RFC 5246: The Transport Layer Security (TLS) Protocol
                     Version 1.2";
      }

      identity ecdhe-rsa-with-3des-ede-cbc-sha {
        base cipher-suite-base;
        if-feature "tls-ecc and tls-3des";
        description
          "Cipher suite TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA.";
        reference
          "RFC 4492: Elliptic Curve Cryptography (ECC) Cipher Suites
                     for Transport Layer Security (TLS)";
      }

      identity ecdhe-rsa-with-aes-128-cbc-sha {
        base cipher-suite-base;
        if-feature "tls-ecc";
        description
          "Cipher suite TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA.";
        reference
          "RFC 4492: Elliptic Curve Cryptography (ECC) Cipher Suites
                     for Transport Layer Security (TLS)";
      }

      identity ecdhe-rsa-with-aes-256-cbc-sha {
        base cipher-suite-base;
        if-feature "tls-ecc";
        description
          "Cipher suite TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA.";
        reference
          "RFC 4492: Elliptic Curve Cryptography (ECC) Cipher Suites
                     for Transport Layer Security (TLS)";
      }

      // groupings

      grouping hello-params-grouping {
        description
          "A reusable grouping for TLS hello message parameters.";
        reference
          "RFC 5246: The Transport Layer Security (TLS) Protocol
                     Version 1.2";

        container tls-versions {
          description
            "Parameters regarding TLS versions.";
          leaf-list tls-version {
            type identityref {
```

```
            base tls-version-base;
          }
          description
            "Acceptable TLS protocol versions.

             If this leaf-list is not configured (has zero elements)
             the acceptable TLS protocol versions are implementation-
             defined.";
        }
      }
      container cipher-suites {
        description
          "Parameters regarding cipher suites.";
        leaf-list cipher-suite {
          type identityref {
            base cipher-suite-base;
          }
          ordered-by user;
          description
            "Acceptable cipher suites in order of descending
             preference.

             If this leaf-list is not configured (has zero elements)
             the acceptable cipher suites are implementation-
             defined.";
        }
      }

    } // end hello-params-grouping

}
<CODE ENDS>
```

6. Security Considerations

The YANG modules defined in this document are designed to be accessed via YANG based management protocols, such as NETCONF [RFC6241] and RESTCONF [RFC8040]. Both of these protocols have mandatory-to-implement secure transport layers (e.g., SSH, TLS) with mutual authentication.

The NETCONF access control model (NACM) [RFC6536] provides the means to restrict access for particular users to a pre-configured subset of all available protocol operations and content.

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

There are a number of data nodes defined in the YANG modules that are writable/creatable/deletable (i.e., config true, which is the default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g., edit-config) to these data nodes without proper protection can have a negative effect on network operations. These are the subtrees and data nodes and their sensitivity/vulnerability:

/: The entire data tree of all the groupings defined in this draft is sensitive to write operations. For instance, the addition or removal of references to keys, certificates, trusted anchors, etc., can dramatically alter the implemented security policy. However, no NACM annotations are applied as the data SHOULD be editable by users other than a designated 'recovery session'.

Some of the readable data nodes in the YANG modules may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get, get-config, or notification) to these data nodes. These are the subtrees and data nodes and their sensitivity/vulnerability:

NONE

Some of the RPC operations in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control access to these operations. These are the operations and their sensitivity/vulnerability:

NONE

7. IANA Considerations

7.1. The IETF XML Registry

This document registers three URIs in the IETF XML registry [RFC3688]. Following the format in [RFC3688], the following registrations are requested:

```
URI: urn:ietf:params:xml:ns:yang:ietf-tls-client
Registrant Contact: The NETCONF WG of the IETF.
XML: N/A, the requested URI is an XML namespace.

URI: urn:ietf:params:xml:ns:yang:ietf-tls-server
Registrant Contact: The NETCONF WG of the IETF.
XML: N/A, the requested URI is an XML namespace.

URI: urn:ietf:params:xml:ns:yang:ietf-tls-common
Registrant Contact: The NETCONF WG of the IETF.
XML: N/A, the requested URI is an XML namespace.
```

7.2. The YANG Module Names Registry

This document registers three YANG modules in the YANG Module Names registry [RFC7950]. Following the format in [RFC7950], the the following registrations are requested:

```
name:         ietf-tls-client
namespace:    urn:ietf:params:xml:ns:yang:ietf-tls-client
prefix:       tlsc
reference:    RFC XXXX

name:         ietf-tls-server
namespace:    urn:ietf:params:xml:ns:yang:ietf-tls-server
prefix:       tlss
reference:    RFC XXXX

name:         ietf-tls-common
namespace:    urn:ietf:params:xml:ns:yang:ietf-tls-common
prefix:       tlscmn
reference:    RFC XXXX
```

8. Acknowledgements

The authors would like to thank for following for lively discussions on list and in the halls (ordered by last name): Andy Bierman, Martin Bjorklund, Benoit Claise, Mehmet Ersue, Balazs Kovacs, David Lamparter, Alan Luchuk, Ladislav Lhotka, Radek Krejci, Tom Petch, Juergen Schoenwaelder, Phil Shafer, Sean Turner, and Bert Wijnen.

9. References

9.1. Normative References

[I-D.ietf-netconf-keystore]
Watsen, K., "Keystore Model", draft-ietf-netconf-keystore-02 (work in progress), June 2017.

[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>.

[RFC2246] Dierks, T. and C. Allen, "The TLS Protocol Version 1.0", RFC 2246, DOI 10.17487/RFC2246, January 1999, <https://www.rfc-editor.org/info/rfc2246>.

[RFC4346] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.1", RFC 4346, DOI 10.17487/RFC4346, April 2006, <https://www.rfc-editor.org/info/rfc4346>.

[RFC4492] Blake-Wilson, S., Bolyard, N., Gupta, V., Hawk, C., and B. Moeller, "Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS)", RFC 4492, DOI 10.17487/RFC4492, May 2006, <https://www.rfc-editor.org/info/rfc4492>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <https://www.rfc-editor.org/info/rfc5246>.

[RFC5288] Salowey, J., Choudhury, A., and D. McGrew, "AES Galois Counter Mode (GCM) Cipher Suites for TLS", RFC 5288, DOI 10.17487/RFC5288, August 2008, <https://www.rfc-editor.org/info/rfc5288>.

[RFC5289] Rescorla, E., "TLS Elliptic Curve Cipher Suites with SHA-256/384 and AES Galois Counter Mode (GCM)", RFC 5289, DOI 10.17487/RFC5289, August 2008, <https://www.rfc-editor.org/info/rfc5289>.

[RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, DOI 10.17487/RFC6536, March 2012, <https://www.rfc-editor.org/info/rfc6536>.

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <https://www.rfc-editor.org/info/rfc6991>.

[RFC7589] Badra, M., Luchuk, A., and J. Schoenwaelder, "Using the NETCONF Protocol over Transport Layer Security (TLS) with Mutual X.509 Authentication", RFC 7589, DOI 10.17487/RFC7589, June 2015, <https://www.rfc-editor.org/info/rfc7589>.

[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>.

9.2. Informative References

[I-D.ietf-netmod-yang-tree-diagrams] Bjorklund, M. and L. Berger, "YANG Tree Diagrams", draft-ietf-netmod-yang-tree-diagrams-02 (work in progress), October 2017.

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, DOI 10.17487/RFC2818, May 2000, <https://www.rfc-editor.org/info/rfc2818>.

[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>.

[RFC8071] Watsen, K., "NETCONF Call Home and RESTCONF Call Home", RFC 8071, DOI 10.17487/RFC8071, February 2017, <https://www.rfc-editor.org/info/rfc8071>.

Appendix A. Change Log

A.1. 00 to 01

- Noted that '0.0.0.0' and '::' might have special meanings.
- Renamed "keychain" to "keystore".

A.2. 01 to 02

- Removed the groupings containing transport-level configuration. Now modules contain only the transport-independent groupings.
- Filled in previously incomplete 'ietf-tls-client' module.
- Added cipher suites for various algorithms into new 'ietf-tls-common' module.

A.3. 02 to 03

- Added a 'must' statement to container 'server-auth' asserting that at least one of the various auth mechanisms must be specified.
- Fixed description statement for leaf 'trusted-ca-certs'.

A.4. 03 to 04

- Updated title to "YANG Groupings for TLS Clients and TLS Servers"
- Updated leafref paths to point to new keystore path
- Changed the YANG prefix for ietf-tls-common from 'tlscom' to 'tlscmn'.
- Added TLS protocol verions 1.0 and 1.1.
- Made author lists consistent
- Now tree diagrams reference ietf-netmod-yang-tree-diagrams
- Updated YANG to use typedefs around leafrefs to common keystore paths
- Now inlines key and certificates (no longer a leafref to keystore)

Authors' Addresses

Kent Watsen
Juniper Networks

EMail: kwatsen@juniper.net

Gary Wu
Cisco Systems

EMail: garywu@cisco.com