TLS Client and Server Models
draft-ietf-netconf-tls-client-server-02

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:

   o I-D.ietf-netconf-keystore

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

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

   o "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:

   o "2017-03-13" --> the publication date of this draft

   The following two Appendix sections are to be removed prior to
   publication:

   o Appendix A. Change Log

   o [Appendix B](#). Open Issues

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 [http://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 September 14, 2017.

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.

1.1. Terminology

   The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
   "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this
   document are to be interpreted as described in RFC 2119 [RFC2119].

1.2. **Tree Diagrams**

   A simplified graphical representation of the data models is used in
   this document. The meaning of the symbols in these diagrams is as
   follows:

   o Brackets "[" and "]" enclose list keys.

   o Braces "{" and "}" enclose feature names, and indicate that the
      named feature must be present for the subtree to be present.

   o Abbreviations before data node names: "rw" means configuration
      (read-write) and "ro" state data (read-only).

   o Symbols after data node names: "?" means an optional node, "!"
      means a presence container, and "*" denotes a list and leaf-list.

   o Parentheses enclose choice and case nodes, and case nodes are also
      marked with a colon (":").

   o Ellipsis ("...") stands for contents of subtrees that are not
      shown.

2. **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.

2.1. **Tree Diagram**

   The following tree diagram presents the data model for the grouping
   defined in the ietf-tls-client module. Please see Section 1.2 for
   tree diagram notation.

```
module: ietf-tls-client
  groupings:
  tls-client-grouping
      +---- server-auth
      |  +---- trusted-ca-certs?
      |  |        -> /ks:keystore/trusted-certificates/name
      |  +---- trusted-server-certs?
      |           -> /ks:keystore/trusted-certificates/name
      +---- client-auth
      |  +---- (auth-type)?
      |     +--:(certificate)
      |        +---- certificate?   leafref
      +---- hello-params {tls-client-hello-params-config}?
         +---- tls-versions
         |  +---- tls-version*   identityref
         +---- cipher-suites
            +---- cipher-suite*   identityref
```

2.2. 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].

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

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

  <!-- how this client will authenticate itself to the server -->
  <client-auth>
    <certificate>builtin-idevid-cert</certificate>
  </client-auth>

</tls-client>
```

2.3. YANG Model

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

   <CODE BEGINS> file "ietf-tls-client@2017-03-13.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 tlscom;
    revision-date 2017-03-13; // stable grouping definitions
    reference
      "RFC XXXX: TLS Client and Server Models";
  }

  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>";


  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) 2014 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 "2017-03-13" {
```

```
      description
       "Initial version";
      reference
       "RFC XXXX: TLS Client and Server Models";
    }

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

    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 server-auth {
        description
          "Trusted server identities.";

        leaf trusted-ca-certs {
          type leafref {
            path "/ks:keystore/ks:trusted-certificates/ks:name";
          }
          description
            "A reference to a list of certificate authority (CA)
             certificates used by the TLS client to authenticate
             TLS server certificates.";
        }

        leaf trusted-server-certs {
          type leafref {
            path "/ks:keystore/ks:trusted-certificates/ks:name";
          }
          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 trusted server certificate.";
        }
      }

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

```
        choice auth-type {
          description
            "The authentication type.";
          leaf certificate {
            type leafref {
              path "/ks:keystore/ks:keys/ks:key/ks:certificates"
                    + "/ks:certificate/ks:name";
            }
            description
              "A certificates to be used for user authentication.";
          }
        }
      }

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

    } // end tls-client-grouping

  }



  <CODE ENDS>
```

3. 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.

3.1. Tree Diagram

The following tree diagram presents the data model for the grouping
defined in the ietf-tls-server module. Please see Section 1.2 for
tree diagram notation.

```
   module: ietf-tls-server
     groupings:
     tls-server-grouping
        +---- certificates
        |  +---- certificate* [name]
        |     +---- name?   leafref
        +---- client-auth
        |  +---- trusted-ca-certs?
        |  |        -> /ks:keystore/trusted-certificates/name
        |  +---- trusted-client-certs?
        |           -> /ks:keystore/trusted-certificates/name
        +---- hello-params {tls-server-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-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].

```
<!-- hypothetical example, groupings don't have instance data -->

<tls-server
  xmlns="urn:ietf:params:xml:ns:yang:ietf-tls-server">
  <certificates>
    <certificate>
      <name>tls-ec-cert</name>
    </certificate>
  </certificates>
  <client-auth>
    <trusted-ca-certs>deployment-specific-ca-certs</trusted-ca-certs>
    <trusted-client-certs>explicitly-trusted-client-certs</trusted-client-
certs>
  </client-auth>
</tls-server>
```

3.3. YANG Model

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

   <CODE BEGINS> file "ietf-tls-server@2017-03-13.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 tlscom;
  revision-date 2017-03-13; // stable grouping definitions
  reference
    "RFC XXXX: TLS Client and Server Models";
}

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>";


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) 2014 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 "2017-03-13" {
  description
```

```
           "Initial version";
         reference
          "RFC XXXX: TLS Client and Server Models";
      }

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

      // grouping
      grouping tls-server-grouping {
        description
          "A reusable grouping for configuring a TLS server without
           any consideration for how underlying TCP sessions are
           established.";
        container certificates {
          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 RRC 5246.";
          reference
            "RFC 5246:
               The Transport Layer Security (TLS) Protocol Version 1.2";
          list certificate {
            key name;
            min-elements 1;
            description
              "An unordered list of certificates the TLS server can pick
               from when sending its Server Certificate message.";
            reference
              "RFC 5246: The TLS Protocol, Section 7.4.2";
            leaf name {
              type leafref {
                path "/ks:keystore/ks:keys/ks:key/ks:certificates/"
                     + "ks:certificate/ks:name";
              }
              description
                "The name of the certificate in the keystore.";
            }
          }
        }

        container client-auth {
          description
            "A reference to a list of trusted certificate authority (CA)
             certificates and a reference to a list of trusted client
```

```
           certificates.";
         leaf trusted-ca-certs {
           type leafref {
             path "/ks:keystore/ks:trusted-certificates/ks:name";
           }
           description
             "A reference to a list of certificate authority (CA)
              certificates used by the TLS server to authenticate
              TLS client certificates.";
         }

         leaf trusted-client-certs {
           type leafref {
             path "/ks:keystore/ks:trusted-certificates/ks:name";
           }
           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 trusted client certificate.";
         }
       }

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

     } // end tls-server-grouping

   }



   <CODE ENDS>
```

4. 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.

4.1. Tree Diagram

The following tree diagram presents the data model for the grouping
defined in the ietf-tls-common module. Please see Section 1.2 for
tree diagram notation.

```
module: ietf-tls-common
  groupings:
  hello-params-grouping
      +---- tls-versions
      |  +---- tls-version*   identityref
      +---- cipher-suites
         +---- cipher-suite*   identityref
```

4.2. 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">

  <tls-versions>
    <tls-version>tls-1.2</tls-version>
  </tls-versions>
  <cipher-suites>
    <cipher-suite>ecdhe-rsa-with-3des-ede-cbc-sha</cipher-suite>
    <cipher-suite>dhe-rsa-with-aes-128-cbc-sha</cipher-suite>
    <cipher-suite>rsa-with-aes-128-cbc-sha</cipher-suite>
    <cipher-suite>rsa-with-3des-ede-cbc-sha</cipher-suite>
  </cipher-suites>

</hello-params>
```

4.3. YANG Model

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

```
<CODE BEGINS> file "ietf-tls-common@2017-03-13.yang"

module ietf-tls-common {
  yang-version 1.1;

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

  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.

    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 "2017-03-13" {
    description
     "Initial version";
    reference
     "RFC XXXX: TLS Client and Server Models";
  }

  // features
  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-1.2 {
      base tls-version-base;
      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.  For
           configurable parameters, a zero-element leaf-list indicates the
           system default configuration for that parameter.";
        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
              "Allowed TLS protocol versions.";
          }
        }
        container cipher-suites {
          description
            "Parameters regarding cipher suites.";
          leaf-list cipher-suite {
            type identityref {
              base cipher-suite-base;
            }
            ordered-by user;
            description
              "Cipher suites in order of descending preference.";
          }
        }
```

```
    }
  }
```

  <CODE ENDS>

5. Security Considerations

   The YANG module defined in this document is 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.

   There are a number of data nodes defined in this YANG module 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 defined by this module 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 this YANG module 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

6. IANA Considerations

6.1. 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.

6.2. 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: tlss
      reference: RFC XXXX

7. 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, David Lamparter, Alan Luchuk,
   Ladislav Lhotka, Radek Krejci, Tom Petch, Juergen Schoenwaelder, Phil
   Shafer, Sean Turner, and Bert Wijnen.

8. References

8.1. Normative References

[I-D.ietf-netconf-keystore]
          Watsen, K. and G. Wu, "Keystore Model", draft-ietf-
          netconf-keystore-00 (work in progress), October 2016.

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

[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,
          <http://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,
          <http://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,
          <http://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,
          <http://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,
          <http://www.rfc-editor.org/info/rfc6536>.

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types",
          RFC 6991, DOI 10.17487/RFC6991, July 2013,
          <http://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,
          <http://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,
              <http://www.rfc-editor.org/info/rfc7950>.

8.2. Informative References

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

   [RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688,
              DOI 10.17487/RFC3688, January 2004,
              <http://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,
              <http://www.rfc-editor.org/info/rfc6241>.

   [RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF
              Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017,
              <http://www.rfc-editor.org/info/rfc8040>.

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

Appendix A. Change Log

A.1. server-model-09 to 00

o This draft was split out from draft-ietf-netconf-server-model-09.

o Noted that '0.0.0.0' and '::' might have special meanings.

A.2. 00 to 01

o Renamed "keychain" to "keystore".

A.3. 01 to 02

o Removed the groupings containing transport-level configuration.
   Now modules contain only the transport-independent groupings.

o Filled in previously incomplete 'ietf-tls-client' module.

o Added cipher suites for various algorithms into new 'ietf-tls-
   common' module.

Appendix B. Open Issues

Please see: https://github.com/netconf-wg/tls-client-server/issues.

Authors' Addresses

Kent Watsen
Juniper Networks

EMail: kwatsen@juniper.net

Gary Wu
Cisco Systems

EMail: garywu@cisco.com