Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 13, 2015

Y. Nir
Check Point
S. Josefsson
SJD
June 11, 2015

Using Curve25519 for IKEv2 Key Agreement
draft-nir-ipsecme-curve25519-00

Abstract

This document describes the use of Curve25519 for ephemeral key exchange in the Internet Key Exchange (IKEv2) protocol.

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 December 13, 2015.

Copyright Notice

Table of Contents

1. Introduction

[CFRG-Curves] specifies a new elliptic curve function for use in cryptographic applications. Curve25519 is a Diffie-Hellman function designed with performance and security in mind.

Almost ten years ago [RFC4753] specified the first elliptic curve Diffie-Hellman groups for the Internet Key Exchange protocol (IKEv2 - [RFC7296]). These were the so-called NIST curves. The state of the art has advanced since then. More modern curves allow faster implementations while making it much easier to write constant-time implementations free from side-channel attacks. This document defines such a curve for use in IKE. See [Curve25519] for details about the speed and security of this curve.

1.1. Conventions Used in This Document

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

2. Curve25519

All cryptographic computations are done using the Curve25519 function defined in [CFRG-Curves]. In this document, this function is considered a black box that takes for input a (secret key, public key) pair and outputs a public key. Public keys are defined as strings of 32 octets. Secret keys are defined as 255-bit numbers such that high-order bit (bit 254) is set, and the three lowest-order bits are unset. In addition, a common public key, denoted by G, is shared by all users.

An ephemeral Diffie-Hellman key exchange using Curve25519 goes as follows: Each party picks a secret key d uniformly at random and computes the corresponding public key:

```
x_mine = Curve25519(d, G)
```

Parties exchange their public keys (see Section 3.1) and compute a shared secret:

```
SHARED_SECRET = Curve25519(d, x_peer).
```

This shared secret is used directly as the value denoted g^ir in section 2.14 of RFC 7296. It is always exactly 32 octets when Curve25519 is used.

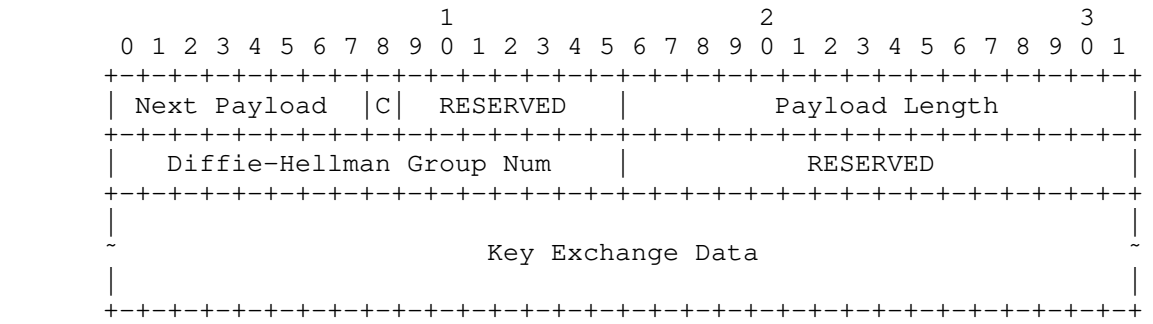
A complete description of the Curve25519 function, as well as a few implementation notes, are provided in Appendix A.

3. Use and Negotiation in IKEv2

The use of Curve25519 in IKEv2 is negotiated using a Transform Type 4 (Diffie-Hellman group) in the SA payload of either an IKE_SA_INIT or a CREATE_CHILD_SA exchange.

3.1. Key Exchange Payload

The diagram for the Key Exchange Payload from section 3.4 of RFC 7296 is copied below for convenience:



- Payload Length - Since a Curve25519 public key is 32 octets, the Payload Length is always 40.
- The Diffie-Hellman Group Num is xx for Curve25519 (TBA by IANA)
- The Key Exchange Data is 32 octets encoded as an array of bytes in little-endian order as described in section 8 of [CFRG-Curves]

3.2. Recipient Tests

This section describes the checks that a recipient of a public key needs to perform. It is the equivalent of the tests described in [RFC6989] for other Diffie-Hellman groups.

Curve25519 was designed in a way that the result of Curve25519(x, d) will never reveal information about d, provided is was chosen as prescribed, for any value of x.

Define legitimate values of x as the values that can be obtained as x = Curve25519(G, d') for some d, and call the other values illegitimate. The definition of the Curve25519 function shows that legitimate values all share the following property: the high-order bit of the last byte is not set.

Since there are some implementation of the Curve25519 function that impose this restriction on their input and others that don't, implementations of Curve25519 in IKE SHOULD reject public keys when the high-order bit of the last byte is set (in other words, when the value of the leftmost byte is greater than 0x7F) in order to prevent implementation fingerprinting.

Other than this recommended check, implementations do not need to ensure that the public keys they receive are legitimate: this is not necessary for security with Curve25519.

4. Security Considerations

Curve25519 is designed to facilitate the production of high-performance constant-time implementations of the Curve25519 function. Implementors are encouraged to use a constant-time implementation of the Curve25519 function. This point is of crucial importance if the implementation chooses to reuse its supposedly ephemeral key pair for many key exchanges, which some implementations do in order to improve performance.

Curve25519 is believed to be at least as secure as the 256-bit random ECP group (group 19) defined in RFC 4753, also known as NIST P-256. While the NIST curves are advertised as being chosen verifiably at random, there is no explanation for the seeds used to generate them. In contrast, the process used to pick Curve25519 is fully documented and rigid enough so that independent verification has been done. This is widely seen as a security advantage for Curve25519, since it prevents the generating party from maliciously manipulating the parameters.

Another family of curves available in IKE, generated in a fully verifiable way, is the Brainpool curves [RFC6954]. Specifically, brainpoolP256 (group 28) is expected to provide a level of security comparable to Curve25519 and NIST P-256. However, due to the use of pseudo-random prime, it is significantly slower than NIST P-256, which is itself slower than Curve25519.

5. IANA Considerations

IANA is requested to assign one value from the IKEv2 "Transform Type 4 - Diffie-Hellman Group Transform IDs" registry, with name Curve25519, and this document as reference. The Recipient Tests field should also point to this document.

6. Acknowledgements

Curve25519 was designed by D. J. Bernstein and Tanja Lange. The specification of wire format is by Sean Turner, Rich Salz, and Watson Ladd, with Adam Langley editing the current document. Much of the text in this document is copied from Simon’s draft for the TLS working group.

7. References

7.1. Normative References

[CFRG-Curves]
Langley, A., "Elliptic Curves for Security", draft-agl-cfrgcurve-00 (work in progress), January 2015.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC7296] Kivinen, T., Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, "Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 7296, October 2014.

7.2. Informative References

[Curve25519]
Bernstein, J., "Curve25519: New Diffie-Hellman Speed Records", LNCS 3958, February 2006, <http://dx.doi.org/10.1007/11745853_14>.

[EFD] Bernstein, D. and T. Lange, "Explicit-Formulas Database: XZ coordinates for Montgomery curves", January 2014, <http://www.hyperelliptic.org/EFD/g1p/auto-montgom-xz.html>.

[NaCl] Bernstein, D., "Cryptography in NaCL", March 2013, <http://cr.yp.to/highspeed/naclcrypto-20090310.pdf>.

[RFC4753] Fu, D. and J. Solinas, "ECP Groups For IKE and IKEv2", RFC 4753, January 2007.

[RFC6954] Merkle, J. and M. Lochter, "Using the Elliptic Curve Cryptography (ECC) Brainpool Curves for the Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 6954, July 2013.

[RFC6989] Sheffer, Y. and S. Fluhrer, "Additional Diffie-Hellman Tests for the Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 6989, July 2013.

Appendix A. The curve25519 function

A.1. Formulas

This section completes Section 2 by defining the Curve25519 function and the common public key G. It is meant as an alternative, self-contained specification for the Curve25519 function, possibly easier to follow than the original paper for most implementors.

A.1.1. Field Arithmetic

Throughout this section, P denotes the integer 2^255-19 = 0x7FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFED. The letters X and Z, and their numbered variants such as x1, z2, etc. denote integers modulo P, that is integers between 0 and P-1 and every operation between them is implictly done modulo P. For addition, subtraction and multiplication this means doing the operation in the usual way and then replacing the result with the remainder of its division by P. For division, "X / Z" means mutliplying (mod P) X by the modular inverse of Z mod P.

A convenient way to define the modular inverse of Z mod P is as Z^(P-2) mod P, that is Z to the power of 2^255-21 mod P. It is also a practical way of computing it, using a square-and-multiply method.

The four operations +, -, *, / modulo P are known as the field operations. Techniques for efficient implementation of the field operations are outside the scope of this document.

A.1.2. Conversion to and from internal format

For the purpose of this section, we will define a Curve25519 point as a pair (X, Z) were X and Z are integers mod P (as defined above). Though public keys were defined to be strings of 32 bytes, internally they are represented as curve points. This subsection describes the conversion process as two functions: PubkeyToPoint and PointToPubkey.

```
PubkeyToPoint:
Input: a public key b_0, ..., b_31
Output: a Curve25519 point (X, Z)
    1. Set X = b_0 + 256 * b_1 + ... + 256^31 * b_31 mod P
    2. Set Z = 1
    3. Output (X, Z)

PointToPubkey:
Input: a Curve25519 point (X, Z)
Output: a public key b_0, ..., b_31
    1. Set x1 = X / Z mod P
    2. Set b_0, ... b_31 such that
        x1 = b_0 + 256 * b_1 + ... + 256^31 * b_31 mod P
    3. Output b_0, ..., b_31
```

A.1.3. Scalar Multiplication

We first introduce the DoubleAndAdd function, defined as follows (formulas taken from [EFD]).

```
DoubleAndAdd:
Input: two points (X2, Z2), (X3, Z3), and an integer mod P: X1
Output: two points (X4, Z4), (X5, Z5)
Constant: the integer mod P: a24 = 121666 = 0x01DB42
Variables: A, AA, B, BB, E, C, D, DA, CB are integers mod P
    1. Do the following computations mod P:
        A  = X2 + Z2
        AA = A2
        B  = X2 - Z2
        BB = B2
        E  = AA - BB
        C  = X3 + Z3
        D  = X3 - Z3
        DA = D * A
        CB = C * B
        X5 = (DA + CB)^2
        Z5 = X1 * (DA - CB)^2
        X4 = AA * BB
        Z4 = E * (BB + a24 * E)
    2. Output (X4, Z4) and (X5, Z5)
```

This may be taken as the abstract definition of an arbitrary-looking function. However, let's mention "the true meaning" of this function, without justification, in order to help the reader make more sense of it. It is possible to define operations "+" and "-" between Curve25519 points. Then, assuming (X2, Z2) - (X3, Z3) = (X1, 1), the DoubleAndAdd function returns points such that (X4, Z4) = (X2, Z2) + (X2, Z2) and (X5, Z5) = (X2, Z2) + (X3, Z3).

Taking the "+" operation as granted, we can define multiplication of a Curve25519 point by a positive integer as N * (X, Z) = (X, Z) + ... + (X, Z), with N point additions. It is possible to compute this operation, known as scalar multiplication, using an algorithm called the Montgomery ladder, as follows.

```
ScalarMult:
Input: a Curve25519 point: (X, 1) and a 255-bits integer: N
Output: a point (X1, Z1)
Variable: a point (X2, Z2)
    0. View N as a sequence of bits b_254, ..., b_0,
        with b_254 the most significant bit
        and b_0 the least significant bit.
    1. Set X1 = 1 and Z1 = 0
    2. Set X2 = X and Z2 = 1
    3. For i from 254 downwards to 0, do:
        If b_i == 0, then:
            Set (X2, Z2) and (X1, Z1) to the output of
            DoubleAndAdd((X2, Z2), (X1, Z1), X)
        else:
            Set (X1, Z1) and (X2, Z2) to the output of
            DoubleAndAdd((X1, Z1), (X2, Z2), X)
    4. Output (X1, Z1)
```

A.1.4. Conclusion

We are now ready to define the Curve25519 function itself.

```
Curve25519:
Input: a public key P and a secret key S
Output: a public key Q
Variables: two Curve25519 points (X, Z) and (X1, Z1)
    1. Set (X, Z) = PubkeyToPoint(P)
    2. Set (X1, Z1) = ScalarMult((X, Z), S)
    3. Set Q = PointToPubkey((X1, Z1))
    4. Output Q
```

The common public key G mentioned in the first paragraph of Section 2 is defined as G = PointToPubkey((9, 1).

A.2. Test vectors

The following test vectors are taken from [NaCl]. Compared to this reference, the private key strings have been applied the ClampC function of the reference and converted to integers in order to fit the description given in [Curve25519] and the present memo.

The secret key of party A is denoted by S_a, it public key by P_a, and similarly for party B. The shared secret is SS.

```
S_a = 0x6A2CB91DA5FB77B12A99C0EB872F4CDF
        4566B25172C1163C7DA518730A6D0770

P_a = 85 20 F0 09 89 30 A7 54 74 8B 7D DC B4 3E F7 5A
      0D BF 3A 0D 26 38 1A F4 EB A4 A9 8E AA 9B 4E 6A

S_b = 0x6BE088FF278B2F1CFDB6182629B13B6F
        E60E80838B7FE1794B8A4A627E08AB58

P_b = DE 9E DB 7D 7B 7D C1 B4 D3 5B 61 C2 EC E4 35 37
      3F 83 43 C8 5B 78 67 4D AD FC 7E 14 6F 88 2B 4F

 SS = 4A 5D 9D 5B A4 CE 2D E1 72 8E 3B F4 80 35 0F 25
      E0 7E 21 C9 47 D1 9E 33 76 F0 9B 3C 1E 16 17 42
```

A.3. Side-channel considerations

Curve25519 was specifically designed so that correct, fast, constant-time implementations are easier to produce. In particular, using a Montgomery ladder as described in the previous section ensures that, for any valid value of the secret key, the same sequence of field operations are performed, which eliminates a major source of side-channel leakage.

However, merely using Curve25519 with a Montgomery ladder does not prevent all side-channels by itself, and some point are the responsibility of implementors:

1. In step 3 of SclarMult, avoid branches depending on b_i, as well as memory access patterns depending on b_i, for example by using safe conditional swaps on the inputs and outputs of DoubleAndAdd.
2. Avoid data-dependant branches and memory access patterns in the implementation of field operations.

Techniques for implementing the field operations in constant time and with high performance are out of scope of this document. Let's mention however that, provided constant-time multiplication is available, division can be computed in constant time using exponentiation as described in Appendix A.1.1.

If using constant-time implementations of the field operations is not convenient, an option to reduce the information leaked this way is to replace step 2 of the SclarMult function with:

```
2a. Pick Z uniformly randomly between 1 and P-1 included
2b. Set X2 = X * Z and Z2 = Z
```

This method is known as randomizing projective coordinates. However, it is no guaranteed to avoid all side-channel leaks related to field operations.

Side-channel attacks are an active reseach domain that still sees new significant results, so implementors of the Curve25519 function are advised to follow recent security research closely.

Authors' Addresses

Yoav Nir
Check Point Software Technologies Ltd.
5 Hasolelim st.
Tel Aviv 6789735
Israel

Email: ynir.ietf@gmail.com

Simon Josefsson
SJD AB

Email: simon@josefsson.org

Network Working Group
Internet Draft
Intended status: Standard Track
Expires: November 14, 2015

K. Tran
Ericsson

May 14, 2015

Yang Data Model for Internet Protocol Security (IPSec)
draft-tran-ipecme-yang-ipsec-00.txt

Abstract

This document defines a YANG data model that can be used to configure and manage Internet Protocol Security (IPSec).

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), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html

This Internet-Draft will expire on November 14, 2015.

Copyright Notice

Table of Contents

1. Introduction

Internet Protocol Security (IPSec) is a suite of protocols that provides security to internet communications at the IP layer. This document defines a YANG data model that can be used to configure and manage the IPSec protocol.

2. Conventions used in this document

The key words "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].

In this document, these words will appear with that interpretation only when in ALL CAPS. Lower case uses of these words are not to be interpreted as carrying RFC-2119 significance.

In this document, the characters ">>" preceding an indented line(s) indicates a compliance requirement statement using the key words listed above. This convention aids reviewers in quickly identifying or finding the explicit compliance requirements of this RFC.

3. IPSec Configuration and Operation Model Overview

Figure 1 illustrates the IPSec configuration and operation model which contains IPSec, IKE, and IKEv2 modules.

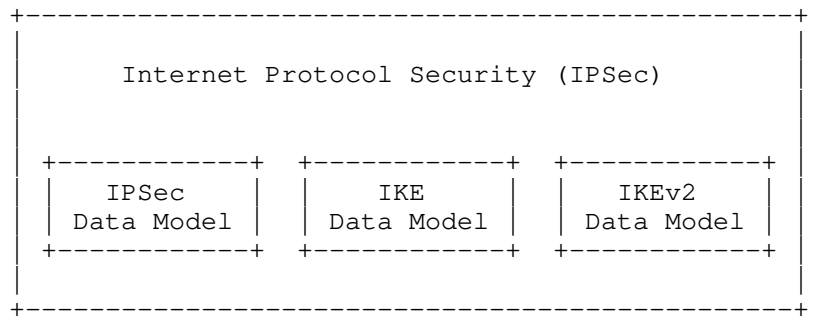


Figure 1. Overview of IPSec configuration and operation model structure

3.1. IPSec Configuration Data Model

The IPSec data model provides the appropriate leaves for configuring the IPSec protocol. The IPSec YANG data model has the following structure:

```
module: ietf-ipsec
   +--rw ipsec
      |  +--rw access-list* [name sequence-number]
      |     |  +--rw name                          string
      |     |  +--rw description?                  string
      |     |  +--rw sequence-number               uint32
      |     |  +--rw (protocol)?
      |     |     |  +--:(number)
      |     |     |     |  +--rw number?                      uint16
      |     |     |     |  +--rw (argument)?
      |     |     |     |     +--:(source-ipv4-address)
      |     |     |     |     |  +--rw source-ipv4-address?        inet:ipv4-address
      |     |     |     |     +--:(any)
      |     |     |     |        +--rw source-any?                 empty
      |     |     |  +--:(source-ipv4-address)
      |     |     |     |  +--rw source-ipv4-address?        inet:ipv4-address
      |     |     |  +--:(any)
      |     |     |     |  +--rw any?                        empty
      |     |     |  +--:(tcp)
      |     |     |     |  +--rw tcp?                        empty
```

```
|  |  |  +--:(udp)
|  |  |     +--rw udp?                          empty
|  |  +--rw (dest-address)?
|  |     +--:(dest-ipv4-address)
|  |     |  +--rw destination-ipv4-address?   inet:ipv4-address
|  |     +--:(dest-any)
|  |        +--rw dest-any?                    empty
|  +--rw alarms
|  |  +--rw hold-down?   uint8
|  +--rw qos
|  |  +--rw policy* [name]
|  |     +--rw name    string
|  |     +--rw pq
|  |        +--rw num-queues?   uint8
|  +--rw redundancy
|  |  +--rw inter-chassis?   empty
|  +--rw security-association
|  |  +--rw ipsec-sa* [name]
|  |     +--rw name                    string
|  |     +--rw anti-replay-window?   uint16
|  |     +--rw ip-comp?              empty
|  |     +--rw in
|  |     |  +--rw ah
|  |     |  |  +--rw spi?              uint32
|  |     |  |  +--rw description?     string
|  |     |  |  +--rw (authentication-algorithm)?
|  |     |  |     +--:(hmac-aes-xcbc)
|  |     |  |     |  +--rw hmac-aes-xcbc
|  |     |  |     |     +--rw key-str?   union
|  |     |  |     +--:(hmac-md5-96)
|  |     |  |     |  +--rw hmac-md5-96
|  |     |  |     |     +--rw key-str?   union
|  |     |  |     +--:(hmac-sha1-96)
|  |     |  |     |  +--rw hmac-sha1-96
|  |     |  |     |     +--rw key-str?   union
|  |     |  |     +--:(key-string)
|  |     |  |        +--rw key-string
|  |     |  |           +--rw key-str?   union
|  |     |  +--rw esp
|  |     |     +--rw description?      string
|  |     |     +--rw authentication
|  |     |     |  +--rw (authentication-algorithm)?
|  |     |     |     +--:(hmac-aes-xcbc)
|  |     |     |     |  +--rw hmac-aes-xcbc
|  |     |     |     |     +--rw key-str?   union
|  |     |     |     +--:(hmac-md5-96)
|  |     |     |     |  +--rw hmac-md5-96
|  |     |     |     |     +--rw key-str?   union
|  |     |     |     +--:(hmac-sha1-96)
```

```
|  |  |  |  |  |  +--rw hmac-sha1-96
|  |  |  |  |  |     +--rw key-str?   union
|  |  |  |  |  +--:(key-string)
|  |  |  |  |     +--rw key-string
|  |  |  |  |        +--rw key-str?   union
|  |  |  |  +--rw encryption
|  |  |  |     +--rw (encryption-algorithm)?
|  |  |  |        +--:(des3-cbc)
|  |  |  |        |  +--rw des3-cbd
|  |  |  |        |     +--rw key-str?   union
|  |  |  |        +--:(aes-128-cbc)
|  |  |  |        |  +--rw aes-128-cbc
|  |  |  |        |     +--rw key-str?   union
|  |  |  |        +--:(aes-192-cbc)
|  |  |  |        |  +--rw aes-192-cbc
|  |  |  |        |     +--rw key-str?   union
|  |  |  |        +--:(aes-256-cbc)
|  |  |  |        |  +--rw aes-256-cbc
|  |  |  |        |     +--rw key-str?   union
|  |  |  |        +--:(des-cbc)
|  |  |  |        |  +--rw des-cbc
|  |  |  |        |     +--rw key-str?   union
|  |  |  |        +--:(key-string)
|  |  |  |           +--rw key-string
|  |  |  |              +--rw key-str?   union
|  |  |  +--rw out
|  |  |     +--rw ah
|  |  |     |  +--rw spi?             uint32
|  |  |     |  +--rw description?     string
|  |  |     |  +--rw (authentication-algorithm)?
|  |  |     |     +--:(hmac-aes-xcbc)
|  |  |     |     |  +--rw hmac-aes-xcbc
|  |  |     |     |     +--rw key-str?   union
|  |  |     |     +--:(hmac-md5-96)
|  |  |     |     |  +--rw hmac-md5-96
|  |  |     |     |     +--rw key-str?   union
|  |  |     |     +--:(hmac-sha1-96)
|  |  |     |     |  +--rw hmac-sha1-96
|  |  |     |     |     +--rw key-str?   union
|  |  |     |     +--:(key-string)
|  |  |     |        +--rw key-string
|  |  |     |           +--rw key-str?   union
|  |  |     +--rw esp
|  |  |        +--rw description?      string
|  |  |        +--rw authentication
|  |  |        |  +--rw (authentication-algorithm)?
|  |  |        |     +--:(hmac-aes-xcbc)
|  |  |        |     |  +--rw hmac-aes-xcbc
|  |  |        |     |     +--rw key-str?   union
```

```
|  |  |           |     +--:(hmac-md5-96)
|  |  |           |     |  +--rw hmac-md5-96
|  |  |           |     |     +--rw key-str?   union
|  |  |           |     +--:(hmac-sha1-96)
|  |  |           |     |  +--rw hmac-sha1-96
|  |  |           |     |     +--rw key-str?   union
|  |  |           |     +--:(key-string)
|  |  |           |        +--rw key-string
|  |  |           |           +--rw key-str?   union
|  |  |           +--rw encryption
|  |  |              +--rw (encryption-algorithm)?
|  |  |                 +--:(des3-cbc)
|  |  |                 |  +--rw des3-cbd
|  |  |                 |     +--rw key-str?   union
|  |  |                 +--:(aes-128-cbc)
|  |  |                 |  +--rw aes-128-cbc
|  |  |                 |     +--rw key-str?   union
|  |  |                 +--:(aes-192-cbc)
|  |  |                 |  +--rw aes-192-cbc
|  |  |                 |     +--rw key-str?   union
|  |  |                 +--:(aes-256-cbc)
|  |  |                 |  +--rw aes-256-cbc
|  |  |                 |     +--rw key-str?   union
|  |  |                 +--:(des-cbc)
|  |  |                 |  +--rw des-cbc
|  |  |                 |     +--rw key-str?   union
|  |  |                 +--:(key-string)
|  |  |                    +--rw key-string
|  |  |                       +--rw key-str?   union
|  |  +--rw proposal
|  |  |  +--rw ipsec-proposal* [name]
|  |  |     +--rw name        string
|  |  |     +--rw ah?         ike-integrity-algorithm-t
|  |  |     +--rw esp
|  |  |     |  +--rw authentication?   ike-integrity-algorithm-t
|  |  |     |  +--rw encryption?       ike-encryption-algorithm-t
|  |  |     +--rw ip-comp?    empty
|  |  |     +--rw lifetime
|  |  |        +--rw kbytes?    uint32
|  |  |        +--rw seconds?   uint32
|  |  +--rw policy
|  |     +--rw ipsec-policy* [name]
|  |        +--rw name                        string
|  |        +--rw description?                string
|  |        +--rw anti-replay-window?         uint32
|  |        +--rw perfect-forward-secrecy
|  |        |  +--rw dh-group?   diffie-hellman-group-t
|  |        +--rw seq* [seq-id]
|  |           +--rw seq-id         uint32
```

```
|           +--rw description?   string
|           +--rw proposal?      leafref
```

3.2. IKE Configuration Data Model

The IKE data model provides the appropriate leaves for configuring the IKE protocol. The IKE YANG data model has the following structure:

```
+--rw ike
|  +--rw proposal* [name]
|  |  +--rw name              string
|  |  +--rw description?      string
|  |  +--rw dh-group          diffie-hellman-group-t
|  |  +--rw encryption
|  |  |  +--rw algorithm?   ike-encryption-algorithm-t
|  |  +--rw lifetime          uint32
|  |  +--rw authentication
|  |     +--rw algorithm?       ike-integrity-algorithm-t
|  |     +--rw preshared-key?   empty
|  |     +--rw rsa-signature?   empty
|  +--rw keepalive?   empty
|  +--rw policy* [name]
|     +--rw name                                string
|     +--rw mode
|     |  +--rw aggressive?   empty
|     |  +--rw main?         empty
|     +--rw connection-type                     connection-type-t
|     +--rw pre-shared-key?                     union
|     +--rw validate-certificate-identity?   empty
|     +--rw seq* [seq-id]
|     |  +--rw seq-id      uint32
|     |  +--rw proposal?   leafref
|     +--rw identity
|        +--rw local
|        |  +--rw (identity)?
|        |     +--:(ipv4-address)
|        |     |  +--rw ipv4-address?            inet:ipv4-address
|        |     +--:(ipv6-address)
|        |     |  +--rw ipv6-address?            inet:ipv6-address
|        |     +--:(fqdn-string)
|        |     |  +--rw fqdn-string?             inet:domain-name
|        |     +--:(rfc822-address-string)
|        |     |  +--rw rfc822-address-string?   string
|        |     +--:(dnX509)
|        |        +--rw dnX509?                  string
|        +--rw remote
```

```
|           +--rw (identity)?
|              +--:(ipv4-address)
|              |  +--rw ipv4-address?            inet:ipv4-address
|              +--:(ipv6-address)
|              |  +--rw ipv6-address?            inet:ipv6-address
|              +--:(fqdn-string)
|              |  +--rw fqdn-string?             inet:domain-name
|              +--:(rfc822-address-string)
|              |  +--rw rfc822-address-string?   string
|              +--:(dnX509)
|                 +--rw dnX509?                  String
```

3.3. IKEv2 Configuration Data Model

The IKEv2 data model provides the appropriate leaves for configuring the IKEv2 protocol. The IKEv2 YANG data model has the following structure:

```
+--rw ikev2
|  +--rw proposal* [name]
|  |  +--rw name                        string
|  |  +--rw description?                string
|  |  +--rw dh-group                    diffie-hellman-group-t
|  |  +--rw encryption
|  |  |  +--rw algorithm?   ike-encryption-algorithm-t
|  |  +--rw pseudo-random-function    pseudo-random-function-t
|  |  +--rw authentication
|  |     +--rw algorithm?   ike-integrity-algorithm-t
|  +--rw policy* [name]
|     +--rw name                               string
|     +--rw authentication
|     |  +--rw preshared-key?   empty
|     |  +--rw rsa-signature?   empty
|     +--rw lifetime                           uint32
|     +--rw address-allocation
|     |  +--rw aaa?   empty
|     +--rw connection-type                    connection-type-t
|     +--rw pre-shared-key?                    union
|     +--rw validate-certificate-identity?   empty
|     +--rw seq* [seq-id]
|     |  +--rw seq-id      uint32
|     |  +--rw proposal?   leafref
|     +--rw identity
|     |  +--rw local
|     |  |  +--rw (identity)?
```

```
|      |  |     +--:(ipv4-address)
|      |  |     |  +--rw ipv4-address?            inet:ipv4-address
|      |  |     +--:(ipv6-address)
|      |  |     |  +--rw ipv6-address?            inet:ipv6-address
|      |  |     +--:(fqdn-string)
|      |  |     |  +--rw fqdn-string?             inet:domain-name
|      |  |     +--:(rfc822-address-string)
|      |  |     |  +--rw rfc822-address-string?   string
|      |  |     +--:(dnX509)
|      |  |        +--rw dnX509?                  string
|      |  +--rw remote
|      |     +--rw (identity)?
|      |        +--:(ipv4-address)
|      |        |  +--rw ipv4-address?            inet:ipv4-address
|      |        +--:(ipv6-address)
|      |        |  +--rw ipv6-address?            inet:ipv6-address
|      |        +--:(fqdn-string)
|      |        |  +--rw fqdn-string?             inet:domain-name
|      |        +--:(rfc822-address-string)
|      |        |  +--rw rfc822-address-string?   string
|      |        +--:(dnX509)
|      |           +--rw dnX509?                  string
|      +--rw description?                    string
```

3.4. IPSec Operation Data Model

The IPSec data model provides the appropriate leaves for operational states of the IPSec protocol. The IPSec YANG data model has the following structure:

```
+--ro ipsec-state
   +--ro policy*
   |  +--ro name?                       string
   |  +--ro anti-replay-window?         uint32
   |  +--ro perfect-forward-secrecy?    diffie-hellman-group-t
   |  +--ro seq*
   |     +--ro seq-id?          uint32
   |     +--ro proposal-name?   string
   +--ro proposal*
   |  +--ro name?       string
   |  +--ro ah?         ike-integrity-algorithm-t
   |  +--ro esp
   |  |  +--ro authentication?   ike-integrity-algorithm-t
   |  |  +--ro encryption?       ike-encryption-algorithm-t
   |  +--ro ip-comp?    empty
   |  +--ro lifetime
   |     +--ro kbytes?    uint32
   |     +--ro seconds?   uint32
   +--ro hold-down?   uint32
   +--ro sa*
      +--ro name?                        string
      +--ro anti-replay-window?          uint16
      +--ro ip-comp?                     empty
      +--ro spi?                         uint32
      +--ro description?                 string
      +--ro authentication-algorithm?    ike-integrity-algorithm-t
      +--ro encryption-algorithm?        ike-encryption-algorithm-t
```

3.5. IKE Operation Data Model

The IKE data model provides the appropriate leaves for operational states of the IKE protocol. The IKE YANG data model has the following structure:

```
+--ro ike-state
|  +--ro proposal*
|  |  +--ro name?              string
|  |  +--ro lifetime?          uint32
|  |  +--ro encryption?        ike-encryption-algorithm-t
|  |  +--ro dh-group?          diffie-hellman-group-t
|  |  +--ro authentication?    ike-integrity-algorithm-t
|  +--ro policy*
|     +--ro name?               string
|     +--ro description?        string
|     +--ro mode?               enumeration
|     +--ro connection-type?    connection-type-t
|     +--ro local-identity?     inet:ipv4-address-no-zone
|     +--ro remote-identity?    inet:ipv4-address-no-zone
|     +--ro pre-shared-key?     string
|     +--ro seq?                uint32
|     +--ro proposal?           string
```

3.6. IKEv2 Operation Data Model

The IKEv2 data model provides the appropriate leaves for operational sattes of the IKEv2 protocol. The IKEv2 YANG data model has the following structure:

```
+--ro ikev2-state
|  +--ro proposal*
|  |  +--ro name?                      string
|  |  +--ro pseudo-random-function?   pseudo-random-function-t
|  |  +--ro authentication?           ike-integrity-algorithm-t
|  |  +--ro encryption?               ike-encryption-algorithm-t
|  |  +--ro dh-group                  diffie-hellman-group-t
|  +--ro policy*
|     +--ro name?              string
|     +--ro description?       string
|     +--ro mode?              enumeration
|     +--ro connection-type?   connection-type-t
|     +--ro local-identity?    inet:ipv4-address-no-zone
|     +--ro remote-identity?   inet:ipv4-address-no-zone
|     +--ro pre-shared-key?    string
|     +--ro seq?               uint32
|     +--ro proposal?          string
```

3.7. RPC Operation

This section defines a list of RPC support for IPSec protocol.

```
rpcs:
   +---x clear-ipsec-group
   |  +--ro input
   |     +--ro alarm-hold-down?     uint8
   |     +--ro ipsec-policy-name?   leafref
   +---x clear-ike-group
   |  +--ro input
   |     +--ro proposal?   leafref
   +---x clear-ikev2-group
      +--ro input
         +--ro proposal?   leafref
```

4. IPSec YANG Module

```
<CODE BEGINS> file "ietf-ipsec@2015-04-22.yang"

module ietf-ipsec {
  namespace "urn:ietf:params:xml:ns:yang:ietf-ipsec";
  prefix "eipsec";


  import ietf-inet-types {
    prefix inet;
  }

  import ietf-yang-types {
    prefix yang;
  }

  organization "Ericsson AB.";

  contact "Web:   <http://www.ericsson.com>";

  description
    "This YANG module defines the configuration and operational
     state data for Internet Protocol Security (IPSec) on
     IETF draft.
     Copyright (c) 2015 Ericsson AB.
     All rights reserved.";

  revision 2015-04-22 {
    description
      "Initial revision.";
    reference
      "YANG Data model for Internet Protocol Security - IPSec";
  }

  /*--------------------*/
  /* Typedefs           */
  /*--------------------*/

  typedef authentication-method-t {
    type enumeration {
      enum psk {
        value 0;
        description
          "Pre-Sharing Keys.";
      }
      enum certificate {
```

```
          value 1;
          description
            "Certificate.";
        }
      }
      description
        "Available authentication methods.";
    }

    /* IKEv2 Exchange Types (ET) */
    typedef ikev2-exchange-type-t {
      type enumeration {
        enum ikev2-et-ike-sa-init {
          value 34;
          description
            "ikev2-et-ike-sa-init - RFC 7296.";
        }
        enum ikev2-et-ike-auth {
          value 35;
          description
            "ikev2-et-ike-auth - RFC 7296.";
        }
        enum ikev2-et-create-child-sa {
          value 36;
          description
            "ikev2-et-create-child-sa - RFC 7296.";
        }
        enum ikev2-et-informational {
          value 37;
          description
            "ikev2-et-informational - RFC 7296.";
        }
        enum ikev2-et-ike-session-resume {
          value 38;
          description
            "ikev2-et-ike-session-resume - RFC 7296.";
        }
        enum ikev2-et-gsa-auth {
          value 39;
          description
            "ikev2-et-gsa-auth - RFC 7296.";
        }
        enum ikev2-et-gsa-registration {
          value 40;
          description
            "ikev2-et-gsa-registration - RFC 7296.";
        }
        enum ikev2-et-gsa-rekey {
          value 41;
```

```
          description
            "ikev2-et-gsa-rekey - RFC 7296.";
        }
       }
       description
         "IKEv2 Exchange Types (ET).";
     }

      /* Transform Type Values (TTV), RFC 7296 */
      typedef transform-type-value-t {
       type enumeration {
         enum ttv-reserved-0 {
           value 0;
           description
             "ttv-reserved-0 - Transform Type Value Reserved "+
             "(RFC 7296).";
         }
         enum ttv-encr {
           value 1;
           description
             "ttv-encr - Transform Type Value 1,
              Encryption Algorithm "+
             "(ENCR) used in IKE and ESP.";
         }
         enum ttv-prf {
           value 2;
           description
             "ttv-prf - Transform Type Value 2, "+
             "Pseudo-Random Function(PRF) used in IKE.";
         }
         enum ttv-integ {
           value 3;
           description
             "ttv-integ - Transform Type Value 3, Integrity Algorithm"+
             " (INTEG) used in IKE, AH, optional ESP.";
         }
         enum ttv-dh {
           value 4;
           description
             "ttv-dh - Transform Type Value 4, Diffie-Hellman (DH) "+
             "used in IKE, optional AH and ESP.";
         }
         enum ttv-esn {
           value 5;
           description
             "ttv-esn - Transform Type Value 5, Extended Sequence "+
             "Numbers (ESN) used in AH and ESP.";
         }
       }
```

```
       description
         "Transform Type Values (RFC 7296).";
     }

      /* IKEv2 Transform Attribute Types (TAT) */
      typedef ikev2-transform-attribute-type-t {
       type enumeration {
         enum ikev2-tat-reserved-0 {
           value 0;
           description
             "ikev2-tat-reserved-0 - IKEv2 Transform Attribute "+
             "Type Reserved-0 (RFC 7296).";
         }
         enum ikev2-tat-reserved-1 {
           value 1;
           description
             "ikev2-tat-reserved-1 - IKEv2 Transform Attribute "+
             "Type Reserved-1 (RFC 7296).";
         }
         enum ikev2-tat-reserved-13 {
           value 13;
           description
             "ikev2-tat-reserved-13 - IKEv2 Transform Attribute "+
             "Type Reserved-13 (RFC 7296).";
         }
         enum ikev2-tat-key-length {
           value 41;
           description
             "ikev2-tat-key-length - IKEv2 Transform Attribute "+
             "Type KEY LENGTH (in bits) (RFC 7296).";
         }
       }
       description
         "IKEv2 Transform Attribute Types (TAT) (RFC 7296).";
     }

      /* Transform Type 1 (Encryption Algorithm Transform IDs) */
     typedef ike-encryption-algorithm-t {
       type enumeration {
         enum encr-reserved-0 {
           value 0;
           description
             "encr-reserved-0 --> RFC_5996.";
         }
         enum encr-des-iv4 {
           value 1;
           description
             "encr-des-iv4 --> RFC_5996.";
         }
```

```
        enum encr-des {
          value 2;
          description
            "encr-des --> RFC_5996.";
        }
        enum encr-3des {
          value 3;
          description
            "encr-3des --> RFC_5996.";
        }
        enum encr-rc5 {
          value 4;
          description
            "encr-rc5 --> RFC_5996.";
        }
        enum encr-idea {
          value 5;
          description
            "encr-idea --> RFC_5996.";
        }
        enum encr-cast {
          value 6;
          description
            "encr-cast --> RFC_5996.";
        }
        enum encr-blowfish {
          value 7;
          description
            "encr-blowfish --> RFC_5996.";
        }
        enum encr-3idea {
          value 8;
          description
            "encr-3idea --> RFC_5996.";
        }
        enum encr-des-iv32 {
          value 9;
          description
            "encr-des-iv32 --> RFC_5996.";
        }
        enum encr-reserved-10 {
          value 10;
          description
            "encr-reserved-10 --> RFC_5996.";
        }
        enum encr-null {
          value 11;
          description
            "encr-null --> RFC_5996.";
```

```
        }
        enum encr-aes-cbc {
          value 12;
          description
            "encr-aes-cbc --> RFC_5996.";
        }
        enum encr-aes-ctr {
          value 13;
          description
            "encr-aes-ctr --> RFC_5996.";
        }
        enum encr-aes-ccm-8 {
          value 14;
          description
            "encr-aes-ccm-8 --> RFC_5996.";
        }
        enum encr-aes-ccm-12 {
          value 15;
          description
            "encr-aes-ccm-12 --> RFC_5996.";
        }
        enum encr-aes-ccm-16 {
          value 16;
          description
            "encr-aes-ccm-16 --> RFC_5996.";
        }
        enum encr-reserved-17 {
          value 17;
          description
            "encr-reserved-17 --> RFC_5996.";
        }
        enum encr-aes-gcm-8-icv {
          value 18;
          description
            "encr-aes-gcm-8-icv --> RFC_5996.";
        }
        enum encr-aes-gcm-12-icv {
          value 19;
          description
            "encr-aes-gcm-12-icv --> RFC_5996.";
        }
        enum encr-aes-gcm-16-icv {
          value 20;
          description
            "encr-aes-gcm-16-icv--> RFC_5996.";
        }
        enum encr-null-auth-aes-gmac {
          value 21;
          description
```

```
            "encr-null-auth-aes-gmac --> RFC_5996.";
          }
          enum encr-ieee-p1619-xts-aes {
            value 22;
            description
              "encr-ieee-p1619-xts-aes --> Reserved for "+
              "IEEE P1619 XTS-AES.";
          }
          enum encr-camellia-cbc {
            value 23;
            description
              "encr-camellia-cbc --> RFC_5996.";
          }
          enum encr-camellia-ctr {
            value 24;
            description
              "encr-camellia-ctr --> RFC_5996.";
          }
          enum encr-camellia-ccm-8-icv {
            value 25;
            description
              "encr-camellia-ccm-8-icv --> RFC_5996.";
          }
          enum encr-camellia-ccm-12-icv {
            value 26;
            description
              "encr-camellia-ccm-12-icv --> RFC_5996.";
          }
          enum encr-camellia-ccm-16-icv {
            value 27;
            description
              "encr-camellia-ccm-16-icv --> RFC_5996.";
          }
          enum encr-aes-cbc-128 {
            value 1024;
            description
              "encr-aes-cbc-128 --> RFC_5996.";
          }
          enum encr-aes-cbc-192 {
            value 1025;
            description
              "encr-aes-cbc-192 --> RFC_5996.";
          }
          enum encr-aes-cbc-256 {
            value 1026;
            description
              "encr-aes-cbc-256 --> RFC_5996.";
          }
          enum encr-blowfish-128 {
```

```
          value 1027;
          description
            "encr-blowfish-128 --> RFC_5996.";
        }
        enum encr-blowfish-192 {
          value 1028;
          description
            "encr-blowfish-192 --> RFC_5996.";
        }
        enum encr-blowfish-256 {
          value 1029;
          description
            "encr-blowfish-256 --> RFC_5996.";
        }
        enum encr-blowfish-448 {
          value 1030;
          description
            "encr-blowfish-448 --> RFC_5996.";
        }
        enum encr-camellia-128 {
          value 1031;
          description
            "encr-camellia-128 --> RFC_5996.";
        }
        enum encr-camellia-192 {
          value 1032;
          description
            "encr-camellia-192 --> RFC_5996.";
        }
        enum encr-camellia-256 {
          value 1033;
          description
            "encr-camellia-256 --> RFC_5996.";
        }
      }
      description
        "Transform Type 1 - Internet Key Exchange (IKE) "+
        "encryption algorithms.";
    }

    /* Transform Type 2 (Pseudo-Random Function PRF) */
    typedef pseudo-random-function-t {
      type enumeration {
        enum prf-reserved-0 {
          value 0;
          description
            "prf-reserved-0 --> RFC_2104.";
        }
        enum prf-hmac-md5 {
```

```
          value 1;
          description
            "prf-hmac-md5 --> RFC_2104.";
        }
        enum prf-hmac-sha1 {
          value 2;
          description
            "prf-hmac-sha1 --> RFC2104.";
        }
        enum prf-hmac-tiger {
          value 3;
          description
            "prf-hmac-tiger --> RFC2104.";
        }
        enum prf-aes128-xcbc {
          value 4;
          description
            "prf-aes128-xcbc --> RFC_4434.";
        }
        enum prf-hmac-sha2-256 {
          value 5;
          description
            "prf-hmac-sha2-256 --> RFC_4434.";
        }
        enum prf-hmac-sha2-384 {
          value 6;
          description
            "prf-hmac-sha2-384 --> RFC_4434.";
        }
        enum prf-hmac-sha2-512 {
          value 7;
          description
            "prf-hmac-sha2-512 --> RFC_4434.";
        }
        enum prf-aes128-cmac {
          value 8;
          description
            "prf-aes128-cmac --> RFC_4615.";
        }
      }
      description
        "Available Pseudo-Random Functions (PRF).";
    }

     /* Transform Type 3 (Integrity Algorithm) */
    typedef ike-integrity-algorithm-t {
      type enumeration {
        enum auth-none {
          value 0;
```

```
          description
            "auth-none --> RFC_5996.";
        }
        enum auth-hmac-md5-96 {
          value 1;
          description
            "auth-hmac-md5-96 --> RFC_5996.";
        }
        enum auth-hmac-sha1-96 {
          value 2;
          description
            "auth-hmac-sha1-96 --> RFC_5996.";
        }
        enum auth-des-mac {
          value 3;
          description
            "auth-des-mac --> RFC_5996.";
        }
        enum auth-kpdk-md5 {
          value 4;
          description
            "auth-kpdk-md5 --> RFC_5996.";
        }
        enum auth-aes-xcbc-96 {
          value 5;
          description
            "auth-aes-xcbc-96 --> RFC_5996.";
        }
        enum auth-hmac-md5-128 {
          value 6;
          description
            "auth-hmac-md5-128 --> RFC_5996.";
        }
        enum auth-hmac-sha1-160 {
          value 7;
          description
            "auth-hmac-sha1-160 --> RFC_5996.";
        }
        enum auth-aes-cmac-96 {
          value 8;
          description
            "auth-aes-cmac-96 --> RFC_5996.";
        }
        enum auth-aes-128-gmac {
          value 9;
          description
            "auth-aes-128-gmac --> RFC_5996.";
        }
        enum auth-aes-192-gmac {
```

```
          value 10;
          description
            "auth-aes-192-gmac --> RFC_5996.";
        }
        enum auth-aes-256-gmac {
          value 11;
          description
            "auth-aes-256-gmac --> RFC_5996.";
        }
        enum auth-hmac-sha2-256-128 {
          value 12;
          description
            "auth-hmac-sha2-256-128 --> RFC_5996.";
        }
        enum auth-hmac-sha2-384-192 {
          value 13;
          description
            "auth-hmac-sha2-384-192 --> RFC_5996.";
        }
        enum auth-hmac-sha2-512-256 {
          value 14;
          description
            "auth-hmac-sha2-512-256 --> RFC_5996.";
        }
        enum auth-hmac-sha2-256-96 {
          value 1024;
          description
            "auth-hmac-sha2-256-96.";
        }
      }
      description
        "Transform Type 3 - Internet Key Exchange (IKE) "+
        "Integrity Algorithms.";
    }

    /* Transform Type 4 (Diffie-Hellman Group) */
    typedef diffie-hellman-group-t {
      type enumeration {
        enum group-none {
          value 0;
          description
            "group-none --> RFC_5996.";
        }
        enum modp-768-group-1 {
          value 1;
          description
            "modp-768-group-1 --> RFC_5996.";
        }
        enum modp-1024-group-2 {
```

```
          value 2;
          description
            "modp-1024-group-2 --> RFC_5996.";
        }
        enum modp-1536-group-5 {
          value 5;
          description
            "modp-1536-group-5 --> RFC_3526.";
        }
        enum modp-2048-group-14 {
          value 14;
          description
            "modp-2048-group-14 --> RFC_3526.";
        }
        enum modp-3072-group-15 {
          value 15;
          description
            "modp-3072-group-15 --> RFC_3526.";
        }
        enum modp-4096-group-16 {
          value 16;
          description
            "modp-4096-group-16 --> RFC_3526.";
        }
        enum modp-6144-group-17 {
          value 17;
          description
            "modp-6144-group-17 --> RFC_3526.";
        }
        enum modp-8192-group-18 {
          value 18;
          description
            "modp-8192-group-18 --> RFC_3526.";
        }
        enum recp-256-group-19 {
          value 19;
          description
            "recp-256-group-19 --> RFC_6989. 256-bit"+
            " Random ECP Group.";
        }
        enum recp-384-group-20 {
          value 20;
          description
            "recp-384-group-20 --> RFC_6989. 384-bit"+
            " Random ECP Group.";
        }
        enum recp-521-group-21 {
          value 21;
          description
```

```
            "recp-521-group-21 --> RFC_6989. 521-bit"+
            " Random ECP Group.";
          }
          enum modp-1024-160-pos-group-22 {
            value 22;
            description
              "modp-1024-160-pos-group-22 --> RFC_6989."+
              " 1024-bit MODP Group with"+
              " 160-bit Prime Order Subgroup (POS).";
          }
          enum modp-2048-224-pos-group-23 {
            value 23;
            description
              "modp-2048-224-pos-group-23 --> RFC_6989."+
              " 2048-bit MODP Group with"+
              " 224-bit Prime Order Subgroup (POS).";
          }
          enum modp-2048-256-pos-group-24 {
            value 24;
            description
              "modp-2048-256-pos-group-24 --> RFC_6989."+
              " 2048-bit MODP Group with"+
              " 256-bit Prime Order Subgroup (POS).";
          }
          enum recp-192-group-25 {
            value 25;
            description
              "recp-192-group-25 --> RFC_6989."+
              " 192-bit Random ECP Group.";
          }
          enum recp-224-group-26 {
            value 26;
            description
              "recp-224-group-26 --> RFC_6989."+
              " 224-bit Random ECP Group.";
          }
        }
        description
          "Diffie-Hellman Groups (RFC 5996).";
      }


      /* Transform Type 5 (Extended Sequence Numbers
         Transform ESN IDs) */
      typedef extended-sequence-number-t {
        type enumeration {
          enum esn-none {
            value 0;
            description
```

```
            "esn-none - Extended Sequence Number None --> RFC_7296.";
        }
        enum esn-1 {
          value 1;
          description
            "esn-1 - Extended Sequence Number --> RFC_7296.";
        }
      }
      description
        "Extended Sequence Number (RFC 7296).";
    }


    typedef connection-type-t {
      type enumeration {
        enum initiator-only {
          value 0;
          description
            "initiator-only: ME will act as initiator for"+
            " bringing up IKEv2"+
            " session with its IKE peer.";
        }
        enum responder-only {
          value 1;
          description
            "responder-only: ME will act as responder for"+
            " bringing up IKEv2"+
            " session with its IKE peer.";
        }
        enum both {
          value 2;
          description
            "both: ME can act as initiator or responder.";
        }
      }
      description
        "Connection type for IKE session.";
    }

    typedef transport-protocol-name-t {
      type enumeration {
        enum tcp {
          value 1;
          description
            "Transmission Control Protocol (TCP) Transport Protocol.";
        }
        enum udp {
          value 2;
          description
```

```
            "User Datagram Protocol (UDP) Transport Protocol";
        }
        enum sctp {
          value 3;
          description
            "Stream Control Transmission Protocol (SCTP) Transport "+
            "Protocol";
        }
        enum icmp {
          value 4;
          description
            "Internet Control Message Protocol (ICMP) Transport "+
            "Protocol";
        }
      }
      description
        "Enumeration of well known transport protocols.";
    }

    typedef preshared-key-t {
      type string;
      description
        "Derived string used as Pre-Shared Key.";
    }



    /*---------------------*/
    /*   grouping          */
    /*---------------------*/

     /* The following groupings are used in both configuration data
        and operational state data */
    grouping name-grouping {
      description
        "This grouping provides a leaf identifying the name.";
      leaf name {
        type string;
        description
          "Name of a identifying.";
      }
      leaf description {
        type string;
        description
          "Specify the description.";
      }
    }

    grouping sequence-number-grouping {
```

```
      description
        "This grouping provides a leaf identifying
         a sequence number.";
      leaf sequence-number {
        type uint32 {
          range "1..4294967295";
        }
        description
          "Specify the sequence number.";
      }
    }

    grouping description-grouping {
      description
        "description for free use.";
      leaf description {
        type string;
        description
          "description for free use.";
      }
    }

    grouping traffic-selector-grouping {
      description
        "Traffic selector to be used for SA negotiation.";
      leaf traffic-selector-id {
        type string;
        mandatory true;
        description
          "Traffic selector identifier.";
      }
      leaf protocol-name {
        type transport-protocol-name-t;
        description
          "Specifies the protocol selector.";
      }
      leaf address-range {
        type string;
        mandatory true;
        description
          "Specifies the IPv4 or IPv6 address range.";
      }
    }


    grouping ike-general-proposal-grouping {
      description
        "IKE proposal.";
      leaf name {
```

```
          type string;
          mandatory true;
          description
            "IKE Proposal identify.";
        }
        leaf description {
          type string;
          description
            "Specify the description.";
        }

        leaf dh-group {
          type diffie-hellman-group-t;
          mandatory true;
          description
            "Specifies a Diffie-Hellman group.";
        }
        container encryption {
          description
            "Specify IKE Proposal encryption configuration";
          leaf algorithm {
             type ike-encryption-algorithm-t;
             description
               "Specifies an Encryption Algorithm.";
          }
        }
      }

      grouping ike-proposal-grouping {
        description
          "Configure the IKE Proposal";
        uses ike-general-proposal-grouping;

        leaf lifetime {
          type uint32;
           mandatory true;
          description
            "Configure lifetime for IKE SAs
             0: for no timeout.
             300 .. 99999999:  IKE SA lifetime in seconds.";
        }
        container authentication {
          description
            "Specify IKE Proposal authentication configuration";
          leaf algorithm {
            type ike-integrity-algorithm-t;
            description
              "Specify the authentication algorithm";
          }
```

```
        leaf preshared-key {
          type empty;
          description
            "Use pre-shared key based authentication";
        }
        leaf rsa-signature {
          type empty;
          description
            "Use signature based authentication by using
             PKI certificates";
        }
      }
    }

    grouping ikev2-proposal-grouping {
      description
        "Holds an IKEv2 transform proposal used during "+
        "IKEv2 SA negotiation.  Multiple IKEv2 Transforms "+
        " can be proposed during an IKEv2 session initiation "+
        "in an ordered list.";
      uses ike-general-proposal-grouping;

      leaf pseudo-random-function {
        type pseudo-random-function-t;
        mandatory true;
        description
          "Specifies Pseudo Random Function for IKEv2 key exchange";
      }
      container authentication {
        description
          "Specify IKEv2 Proposal authentication configuration";
        leaf algorithm {
          type ike-integrity-algorithm-t;
          description
            "Specify the authentication algorithm";
        }
      }
    }

    grouping ipsec-proposal-grouping {
      description
        "Configure IPSec Proposal";
      leaf name {
         type string;
         mandatory true;
         description
           "IPSec proposal identifier.";
      }
      leaf ah {
```

```
          type ike-integrity-algorithm-t;
          description
            "Configure Authentication Header (AH).";
        }
        container esp {
          description
            "Configure Encapsulating Security Payload (ESP).";
          leaf authentication {
            type ike-integrity-algorithm-t;
            description
              "Configure ESP authentication";
          }
          leaf encryption {
            type ike-encryption-algorithm-t;
            description
              "Configure ESP encryption";
          }
        }
        leaf ip-comp{
          type empty;
          description
            "Enable IPSec proposal IP-COMP which uses the IP Payload "+
            "compression protocol to compress IP Security (IPSec) "+
            "packets before encryption";
        }
        container lifetime {
          description
            "Configure lifetime for IPSEC SAs";
          leaf kbytes {
            type uint32 {
              range "128..2147483647";
            }
            description
              "Enter lifetime kbytes for IPSEC SAs";
          }
          leaf seconds {
            type uint32 {
              range "300..99999999";
            }
            description
              "Enter lifetime seconds for IPSEC SAs
              0: lifetime of 0 for no timeout
              300..99999999: IPSec SA lifetime in seconds";
          }
        }
      }

      grouping identity-grouping {
        description
```

```
        "Identification type. It is an union identity, "+
        "possible type as follows: "+
        "a) ID_FQDN: A fully-qualified domain name string. "+
        "  An example of a ID_FQDN is, example.com. "+
        "  The string MUST not contain any terminators "+
        "(e.g., NULL, CR, etc.). "+
        "b) ID_RFC822_ADDR: A fully-qualified RFC822 email "+
        "   address string, An example of a ID_RFC822_ADDR is, "+
        "   jsmith@example.com. The string MUST not contain "+
        "   any terminators. "+
        "c) ID_IPV4_ADDR: A single four (4) octet IPv4 address. "+
        "d) ID_IPV6_ADDR: A single sixteen (16) octet IPv6 address. "+
        "e) DN_X509: Distinguished name in the X.509 tradition.";
      choice identity {
        description
          "Choice of identity.";
        leaf ipv4-address {
          type inet:ipv4-address;
          description
            "Specifies the identity as a single four (4)
             octet IPv4 address.
             An example is, 10.10.10.10. ";
        }
        leaf ipv6-address {
          type inet:ipv6-address;
          description
            "Specifies the identity as a single sixteen (16) "+
            "octet IPv6 address. "+
            "An example is, "+
            "FF01::101, 2001:DB8:0:0:8:800:200C:417A .";
        }
        leaf fqdn-string {
          type inet:domain-name;
          description
            "Specifies the identity as a Fully-Qualified
             Domain Name (FQDN) string.
             An example is: example.com.
             The string MUST not contain any terminators
             (e.g., NULL, CR, etc.).";
        }
        leaf rfc822-address-string {
          type string;
          description
            "Specifies the identity as a fully-qualified RFC822
             email address string.
             An example is, jsmith@example.com.
             The string MUST not contain any terminators
             (e.g., NULL, CR, etc.).";
        }
```

```
        leaf dnX509 {
          type string;
          description
            "Specifies the identity as a distinguished name
             in the X.509 tradition.";
        }
      }
    } /* grouping identity-grouping */

    grouping ike-general-policy-profile-grouping {
      description
        "IKE policy.";
      leaf connection-type {
        type connection-type-t;
        mandatory true;
        description
          "Specify the IKE connection type";
      }
      leaf pre-shared-key {
        type union {
          type string {
            length "16";
          }
          type yang:hex-string {
            length "40";
          }
        }
        description
          "Specify IKE pre-shared-key value";
      }
      leaf validate-certificate-identity {
        type empty;
        description
          "Validate Remote-ID payload against the
          ID's available in the certificate";
      }
      list seq {
        key seq-id;
        description
          "list of sequence of policy.";
        leaf seq-id {
          type uint32 {
            range "1..429496729";
          }
          description
            "Sequence Number";
        }
        leaf proposal {
          type leafref {
```

```
            path "/eipsec:ike/eipsec:proposal"+
                 "/eipsec:name";
          }
          description
            "IKE Proposal reference.";
        }
      }
      container identity {
        description
          "Specify IKE identity value";
        container local {
          description
            "Specify the identity of the local IP Security (IPSec)
            tunnel endpoint in an Internet Key Exchange (IKE)
            policy to use when negotiating IKE request with a
            remote peer.";
          uses identity-grouping;
        }
        container remote {
          description
            "Specify the identity of the remote IP Security (IPSec)
            tunnel endpoint in an
             Internet Key Exchange (IKE) policy to use when
             negotiating IKE request with a remote peer.";
          uses identity-grouping;
        }
      }
    }

    grouping ike-policy-mode-grouping {
      description
        "IKE Policy Mode";
      container mode {
        description
          "Specify IKE mode configuration";
        leaf aggressive {
          type empty;
          description
            "Set IKE Aggressive mode";
        }
        leaf main {
          type empty;
          description
            "Set IKE Main mode";
        }
      }
    }

    grouping ike-policy-profile-grouping {
```

```
      description
        "Configure IKE policy";
      leaf name {
        type string;
        mandatory true;
        description
          "Specify an IKE policy name";
      }
      uses ike-policy-mode-grouping;
      uses ike-general-policy-profile-grouping;
    }

    grouping ikev2-policy-profile-grouping {
      description
        "Common information for multiple IKE sessions
        to be instantiated on a managed element.;
         One or more Ikev2Session instances might refer
         to this instance.";
      leaf name {
        type string;
        mandatory true;
        description
          "Value component of the RDN.";
      }
      container authentication {
        description
          "Specify IKE Proposal authentication configuration";
        leaf preshared-key {
          type empty;
          description
            "Use pre-shared key based authentication";
        }
        leaf rsa-signature {
          type empty;
          description
            "Use signature based authentication by using
            PKI certificates";
        }
      }
      leaf lifetime {
        type uint32;
        mandatory true;
        description
          "Configure lifetime for IKE SAs
           0: for no timeout.
           300 .. 99999999:  IKE SA lifetime in seconds.";
      }

      container address-allocation {
```

```
        must "../connection-type == 'responder-only'" {
          description
            "address-allocation can be configured only with
            responder-only in ike2 policy";
        }
        leaf aaa {
          type empty;
          description
            "IRAC address allocation by AAA";
        }
        description
          "Specify IKE IRAS address allocation option";
      }
      uses ike-general-policy-profile-grouping;

      leaf description {
        type string;
        description
          "Specify the description.";
      }
    }

    grouping ipsec-policy-grouping {
      description
        "Holds configuration information for IPSec policies.";
      leaf name {
        type string;
        mandatory true;
        description
          "IPSec Policy Identification";
      }
      leaf description {
        type string;
        description
          "Specify the description.";
      }

      leaf anti-replay-window {
        type uint32 {
          range "0 | 32..1024";
        }
        description
          "Configure replay window size
          0: to disable anti-replay-window
          32..1024: IPSec anti-replay-window size in multiple of 32";
      }
      container perfect-forward-secrecy {
        description
          "Configure Perfect Forward Secrecy (PFS) for IPSec Policy";
```

```
        leaf dh-group {
          type diffie-hellman-group-t;
          description
            "Configure Diffie-Hellman group for
             perfect-forward-secrecy";
        }
      }
      list seq {
        key seq-id;
        description
          "Specify IPSEC proposal sequence number";
        leaf seq-id {
          type uint32;
          description
            "Sequence ID";
        }
        leaf description {
          type string;
          description
            "Specify the description.";
        }

        leaf proposal {
          type leafref {
            path "/eipsec:ipsec/"+
                 "eipsec:proposal/eipsec:ipsec-proposal/eipsec:name";
          }
          description
            "IKE proposal reference.";
        }
      }
    }

    grouping key-string-grouping {
      description
        "Configure key for authentication algorithm";
      leaf key-str {
        type union {
          type string {
            length "16";
          }
          type yang:hex-string {
            length "40";
          }
        }
        description
          "Key string input is either string value (length of 16)
          or hexadecimal (length of 40)";
      }
```

```
    }

    grouping ipsec-sa-ah-grouping {
      description
        "Configure Authentication Header (AH) for
         Security Association (SA)";
      container ah {
        description
          "Configure Authentication Header (AH) for SA";
        leaf spi {
          type uint32 {
            range "256..131071";
          }
          description
            "Configure Security Parameter Index (SPI) value";
        }
        leaf description {
          type string;
          description
            "Specify the description.";
        }

        choice authentication-algorithm {
          description
            "choice for authentication algorithm to set for AH";
          case hmac-aes-xcbc {
            container hmac-aes-xcbc {
              description
                "Set the authentication algorithm to hmac-aes-xcbc";
              uses key-string-grouping;
            }
          }
          case hmac-md5-96 {
            container hmac-md5-96 {
              description
                "Set the authentication algorithm to hmac-md5-96";
              uses key-string-grouping;
            }
          }
          case hmac-sha1-96 {
            container hmac-sha1-96 {
              description
                "Set the authentication algorithm to hmac-sha1-96";
              uses key-string-grouping;
            }
          }
          case key-string {
            container key-string {
              description
```

```
                "Configure key for authentication algorithm";
              uses key-string-grouping;
            }
          }
        }
      }
    }

    grouping ipsec-sa-esp-grouping {
      description
        "Configure IPSec Encapsulation Security Payload (ESP)";
      container esp {
        description
          "Set IPSec Encapsulation Security Payloer (ESP)";
        leaf description {
          type string;
          description
            "Specify the description.";
        }

        container authentication {
          description
            "Configure authentication for IPSec
             Encapsulation Secutiry Payload (ESP)";
          choice authentication-algorithm {
            description
              "choice for authentication algorithm to set";
            case hmac-aes-xcbc {
              container hmac-aes-xcbc {
                description
                  "Set the authentication algorithm to hmac-aes-xcbc";
                uses key-string-grouping;
              }
            }
            case hmac-md5-96 {
              container hmac-md5-96 {
                description
                  "Set the authentication algorithm to hmac-md5-96";
                uses key-string-grouping;
              }
            }
            case hmac-sha1-96 {
              container hmac-sha1-96 {
                description
                  "Set the authentication algorithm to hmac-sha1-96";
                uses key-string-grouping;
              }
            }
            case key-string {
```

```
              container key-string {
                description
                  "Configure key for authentication algorithm";
                uses key-string-grouping;
              }
            }
          }
        }
        container encryption {
          description
            "Configure encryption for IPSec
             Encapsulation Secutiry Payload (ESP)";
          choice encryption-algorithm {
            description
              "type of encryption";
            case des3-cbc {
              container des3-cbd {
                description
                  "Set the encryption algorithm to des3-cbc";
                uses key-string-grouping;
              }
            }
            case aes-128-cbc {
              container aes-128-cbc {
                description
                  "Set the encryption algorithm to aes-128-cbc";
                uses key-string-grouping;
              }
            }
            case aes-192-cbc {
              container aes-192-cbc {
                description
                  "Set the encryption algorithm to aes-192-cbc";
                uses key-string-grouping;
              }
            }
            case aes-256-cbc {
              container aes-256-cbc {
                description
                  "Set the encryption algorithm to aes-256-cbc";
                uses key-string-grouping;
              }
            }
            case des-cbc {
              container des-cbc {
                description
                  "Set the encryption algorithm to des-cbc";
                uses key-string-grouping;
              }
```

```
            }
            case key-string {
              container key-string {
                description
                  "Configure key for encryption algorithm";
                uses key-string-grouping;
              }
            }
          }
        }
      }
    }

    grouping ipsec-acl-dest-grouping {
      description
        "IPSEC ACL destination.";
      /* For destination */
      choice dest-address {
        description
          "destination address.";
        case dest-ipv4-address {
          leaf destination-ipv4-address {
            type inet:ipv4-address;
            description
              "Destination IPv4 Address A.B.C.D/0..32.";
          }
        }
        case dest-any {
          leaf dest-any {
            type empty;
            description
              "Match Any Destination IPv4 Address.";
          }
        }
      }
    }

    grouping ipsec-acl-seq-protocol-number-grouping {
      description
        "IPSec ACL Sequence protocol number.";
      leaf number {
        type uint16 {
          range "0..255";
        }
        description
          "Specify protocol number.";
      }
      choice argument {
        description
```

```
          "Source IPv4 address.";
        case source-ipv4-address {
          leaf source-ipv4-address {
            type inet:ipv4-address;
            description
              "Source IPv4 Address A.B.C.D/0..32.";
          }
        }
        case any {
          /* For source */
          leaf source-any {
            type empty;
            description
              "Match Any Source IPv4 Address.";
          }
        }
      }
    }

    grouping ipsec-acl-seq-ip-address-grouping {
      description
        "IPSec ACL Sequence IP Address.";
      leaf source-ipv4-address {
        type inet:ipv4-address;
        description
          "Source is IPv4 Address A.B.C.D/0..32.";
      }
    }

    grouping ipsec-acl-seq-any-grouping {
      description
        "IPSec ACL Sequence Any.";
      leaf any {
        type empty;
        description
          "Source is Any.";
      }
    }

    grouping ipsec-acl-seq-tcp-grouping {
      description
        "IPSec ACL Sequence TCP.";
      leaf tcp {
        type empty;
        description
          "Source is TCP protocol.";
      }
    }
```

```
    grouping ipsec-acl-seq-udp-grouping {
      description
        "IPSec ACL Sequence for UDP.";
      leaf udp {
        type empty;
        description
          "Source is UDP protocol.";
      }
    }

    grouping ipsec-acl-grouping {
      description
        "IPSec ACL";
      list access-list {
        key "name sequence-number";
        uses name-grouping;
        uses sequence-number-grouping;
        description
          "Configure the IPSec access-list.";
        choice protocol {
          description
            "IPSec ACL protocol.";
          case number {
            uses ipsec-acl-seq-protocol-number-grouping;
          }
          case source-ipv4-address {
            uses ipsec-acl-seq-ip-address-grouping;
          }
          case any {
            uses ipsec-acl-seq-any-grouping;
          }
          case tcp {
            uses ipsec-acl-seq-tcp-grouping;
          }
          case udp {
            uses ipsec-acl-seq-udp-grouping;
          }
        }
        uses ipsec-acl-dest-grouping;
      }
    }

    grouping ipsec-df-bit-grouping {
      description
        "IPSec Dont Fragment (DF) bit for IP header.";
      container df-bit {
        description
          "Configure Don't Fragment (DF) bit for IP Header.";
        leaf clear {
```

```
          type empty;
          description
            "Clear DF bit for outer IP header.";
        }
        leaf propagate {
          type empty;
          description
            "Propagate DF bit for outer IP header.";
        }
        leaf set {
          type empty;
          description
            "Set DF bit for outer IP header.";
        }
      }
    }

    grouping ipsec-profile-grouping {
      description
        "IPSec profile.";
      list profile {
        key "name";
        uses name-grouping;
        uses ipsec-df-bit-grouping;
        description
          "Configure the IPSec Profile.";
        leaf mtu {
          type uint32 {
            range "256..1600";
          }
          description
            "Set the MTU.";
        }
        list seq {
          key "sequence-number";
          uses sequence-number-grouping;
          description
            "IPSec Access List sequence number.";
          leaf policy {
            type leafref {
              path "/eipsec:ipsec/eipsec:policy"+
                   "/eipsec:ipsec-policy/eipsec:name";
            }
            description
              "Specify IPSec policy name.";
          }
          leaf access-list {
            type leafref {
              path "/econtext:contexts/econtext:context/"+
```

```
                    "econtext:name/econtext:ipsec"+
                    "/econtext:access-list/econtext:name";
              }
              description
                "Specify IPSec access-list name.";
            }
          }
        }
      }

      /*--------------------*/
      /* Configuration Data */
      /*--------------------*/
      container ike {
        description
          "Configuration IPSec IKE";
        /* The following is for <configure> */
        list proposal {
           key "name";
           uses ike-proposal-grouping;
           description
             "Configure IKE proposal";
        }
        leaf keepalive {
          type empty;
          description
            "Enables sending Dead Peer Detection (DPD) messages "+
            "to Internet Key Exchange (IKE) peers.";
        }
        list policy {
          key "name";
          uses ike-policy-profile-grouping;
          description
            "Configure IKE Policy Profile.";
        }
      }

      container ikev2 {
        description
          "Configuration IPSec IKEv2";
        /* The following is for <configure> */
        list proposal {
           key "name";
           uses ikev2-proposal-grouping;
           description
             "Configure IKEv2 proposal";
        }
        list policy {
          key "name";
```

```
        uses ikev2-policy-profile-grouping;
        description
          "IKEv2 Policy Profile";
      }
    }

    container ipsec {
      description
        "Configuration IPSec";
      uses ipsec-acl-grouping;
      container alarms {
        description
          "Configure the IPSec alarm for tunnels";
        leaf hold-down {
          type uint8 {
            range "1..120";
          }
          description
            "Hold-down time (in seconds) before tunnel
             alarms are generated";
        }
      }
      container qos {
        description
          "Configure the IPSec QoS priority queuing policy";
        list policy {
          key "name";
          leaf name {
            type string;
            description
              "Specify IPSec QoS priority queuing name";
          }
          description
            "Configure IPSec QoS priority queuing name";
          container pq {
            description
              "Configure IPSec QoS priority queuing policy";
            leaf num-queues {
              type uint8 {
                range "1 | 4";
              }
              description
                "IPSec QoS Number of queues is either 1 or 4";
            }
          }
        }
      }
      container redundancy {
        description
```

```
          "Configure redundancy for IPSec";
        leaf inter-chassis {
          type empty;
          description
            "Set redundancy at chassis level";
        }
      }
      container security-association {
        description
          "Configure the IPSec Security Association (SA)";
        list ipsec-sa {
          key "name";
          leaf name {
            type string;
            description
              "Specify IPSec Security Association (SA) name";
          }
          description
            "Configure IPSec Security Association (SA)";
          leaf anti-replay-window {
            type uint16 {
              range "0 | 32..1024";
            }
            description
              "Specify replay window size";
          }
          leaf ip-comp {
            type empty;
            description
              "Enables IPCOMP, which uses the IP payload compression
               protocol to compress IP security (IPsec) packets
               before encryption";
          }
          container in {
            description
              "Configure inbound SA";
            uses ipsec-sa-ah-grouping;
            uses ipsec-sa-esp-grouping;
          }
          container out {
            uses ipsec-sa-ah-grouping;
            uses ipsec-sa-esp-grouping;
            description
              "Configure outbound SA";
          }
        }
      }
      container proposal {
        description
```

```
          "IPSec Proposal Profile";
        list ipsec-proposal {
          key "name";
          uses ipsec-proposal-grouping;
          description
            "Configure the IP Security (IPSec) proposal";
        }
      }
      container policy {
        description
          "Configure the IPSec policy";
        list ipsec-policy {
          key "name";
          uses ipsec-policy-grouping;
          description
            "Specify an IPSec policy name";
        }
      }
    }



    /*---------------------------*/
    /* Operational State Data   */
    /*---------------------------*/
    grouping ike-proposal-state-components {
      description
        "IKE Proposal operational state";
      list proposal {
        description
          "Operational data for IKE Proposal";
        leaf name {
          type string {
            length "1..50";
          }
          description
            "Name of the IKE proposal.";
        }
        leaf lifetime {
          type uint32;
          units "seconds";
          description
            "lifetime";
        }
        leaf encryption {
          type ike-encryption-algorithm-t;
          description
            "Encryption algorithm";
        }
```

```
        leaf dh-group {
          type diffie-hellman-group-t;
          description
            "Diffie-Hellman group.";
        }
        leaf authentication {
          type ike-integrity-algorithm-t;
          description
            "authentication";
        }
      }
    }

    grouping ike-policy-state-grouping {
      description
        "IKE Policy State.";
      list policy {
        description
          "Operational data for IKE policy";
        leaf name {
          type string {
            length "1..50";
          }
          description
            "Name of the IKE Policy.";
        }
        leaf description {
          type string;
          description
            "Description for IKE Policy.";
        }
        leaf mode {
          type enumeration {
            enum aggressive {
              description
                "Aggressive mode.";
            }
            enum main {
              description
                "Main mode.";
            }
          }
          description
            "IKE policy mode.";
        }
        leaf connection-type {
          type connection-type-t;
          description
            "IKE policy connection type.";
```

```
        }
        leaf local-identity {
          type inet:ipv4-address-no-zone;
          description
            "IP address of the local identity.";
        }
        leaf remote-identity {
          type inet:ipv4-address-no-zone;
          description
            "IP address of the remote identity.";
        }
        leaf pre-shared-key {
          type string;
          description
            "Pre-shared key";
        }
        leaf seq {
          type uint32;
          description
            "sequence number";
        }
        leaf proposal {
          type string;
          description
            "proposal name";
        }
      }
    }

    grouping ikev2-proposal-state-components {
      description
        "IKEv2 Operational state";
      list proposal {
        description
          "IKEv2 proposal operational data";
        leaf name {
          type string;
          description
            "Name of IKEv2 Proposal.";
        }
        leaf pseudo-random-function {
          type pseudo-random-function-t;
          description
            "Pseudo Random Function for IKEv2.";
        }
        leaf authentication {
          type ike-integrity-algorithm-t;
          description
            "authentication";
```

```
        }
        leaf encryption {
          type ike-encryption-algorithm-t;
          description
            "Encryption algorithm";
        }
        leaf dh-group {
          type diffie-hellman-group-t;
          mandatory true;
          description
            "Diffie-Hellman group.";
        }
      }
    }

    grouping ipsec-policy-state-grouping {
      description
        "IPSec operational state";
      list policy {
        description
          "IPSec policy operational data";
        leaf name {
          type string;
          description
            "IPSec Policy name.";
        }
        leaf anti-replay-window {
          type uint32;
          description
            "replay window size";
        }
        leaf perfect-forward-secrecy {
          type diffie-hellman-group-t;
          description
            "Diffie-Hellman group for perfect-forward-secrecy";
        }
        list seq {
          description
            "Sequence number";
          leaf seq-id {
            type uint32;
            description
              "Sequence number";
          }
          leaf proposal-name {
            type string;
            description
              "IPSec proposal name";
          }
```

```
        }
      }
    }
    grouping ipsec-proposal-state-grouping {
      description
        "IPSec proposal operational data";
      list proposal {
        description
          "IPSec proposal operational data";
        leaf name {
          type string;
          description
            "IPSec Proposal name";
        }
        leaf ah {
          type ike-integrity-algorithm-t;
          description
            "Authentication Header (AH).";
        }
        container esp {
          description
            "Encapsulating Security Payload (ESP).";
          leaf authentication {
            type ike-integrity-algorithm-t;
            description
              "ESP authentication";
          }
          leaf encryption {
            type ike-encryption-algorithm-t;
            description
              "ESP encryption";
          }
        }
        leaf ip-comp{
          type empty;
          description
            "IPSec proposal IP-COMP which uses the IP Payload "+
            "compression protocol to compress IP Security (IPSec) "+
            "packets before encryption";
        }
        container lifetime {
          description
            "lifetime for IPSEC SAs";
          leaf kbytes {
            type uint32;
            description
              "lifetime kbytes for IPSEC SAs";

          }
```

```
          leaf seconds {
            type uint32;
            description
              "lifetime seconds for IPSEC SAs";
          }
        }
      }
    }

    grouping ipsec-alarms-state-grouping {
      description
        "IPSec alarms operational data";
      leaf hold-down {
        type uint32;
        description
          "Hold-down value";
      }
    }

    grouping ipsec-sa-ah-state-grouping {
      description
        "IPSec SA's AH operational data";

      leaf spi {
        type uint32;
        description
          "Security Parameter Index (SPI) value";
      }
      leaf description {
        type string;
        description
          "the description.";
      }
      leaf authentication-algorithm {
        type ike-integrity-algorithm-t;
        description
          "Authentication algorithm";
      }
      leaf encryption-algorithm {
        type ike-encryption-algorithm-t;
        description
          "Encryption algorithm";
      }
    }


    grouping ipsec-sa-state-grouping {
      description
        "IPSec Security Association Operational data";
```

```
      list sa {
        description
          "IPSec SA operational data";
        leaf name {
          type string;
          description
            "Specify IPSec Security Association (SA) name";
        }
        leaf anti-replay-window {
          type uint16;
          description
            "replay window size";
        }
        leaf ip-comp {
          type empty;
          description
            "Enables IPCOMP, which uses the IP payload compression
             protocol to compress IP security (IPsec) packets before
             encryption";
        }
        uses ipsec-sa-ah-state-grouping;
      }
    }


    container ike-state {
      config "false";
      uses ike-proposal-state-components;
      uses ike-policy-state-grouping;
      description
        "Contain the operational data for IKE.";
    }

    container ikev2-state {
      config "false";
      uses ikev2-proposal-state-components;
      uses ike-policy-state-grouping;
      description
        "Contain the operational data for IKEv2.";
    }

    container ipsec-state {
      config "false";
      uses ipsec-policy-state-grouping;
      uses ipsec-proposal-state-grouping;
      uses ipsec-alarms-state-grouping;
      uses ipsec-sa-state-grouping;
      description
        "Contain the operational data for IPSec.";
```

```
}


/*--------------------*/
/* RPC                */
/*--------------------*/
rpc clear-ipsec-group {
  description
    "RPC for clear ipsec states";
  input {
    leaf alarm-hold-down {
      type uint8;
      description
        "IPSec alarm hold-down";
    }
    leaf ipsec-policy-name {
      type leafref {
        path "/eipsec:ipsec/eipsec:policy/"+
             "eipsec:ipsec-policy/eipsec:name";
      }
      description
        "IPSec Policy name.";
    }
  }
}

rpc clear-ike-group {
  description
    "RPC for clear IKE states";
  input {
    leaf proposal {
      type leafref {
        path "/eipsec:ike/eipsec:proposal/"+
             "eipsec:name";
      }
      description
        "IPSec IKE Proposal name.";
    }
  }
}

rpc clear-ikev2-group {
  description
    "RPC for clear IKEv2 states";
  input {
    leaf proposal {
      type leafref {
        path "/eipsec:ikev2/eipsec:proposal/"+
```

```
                  "eipsec:name";
            }
            description
              "IPSec IKEv2 Proposal name.";
          }
        }
      }

   } /* module ericsson-ipsec */

   <CODE ENDS>
```

5. Security Considerations

The configuration, state, and action data defined in this document are designed to be accessed via the NETCONF protocol [RFC6241]. The data model by itself does not create any security implications. The security considerations for the NETCONF protocol are applicable. The NETCONF protocol used for sending the data supports authentication and encryption.

6. References

6.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2234] Crocker, D. and Overell, P.(Editors), "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, Internet Mail Consortium and Demon Internet Ltd., November 1997.

[RFC6020] Bjorklund, M., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, October 2010.

[RFC6021] Schoenwaelder, J., "Common YANG Data Types", RFC 6021, October 2010.

[RFC6241] Enns, R., Bjorklund, M., Schoenwaelder, J., and A. Bierman, "Network Configuration Protocol (NETCONF)", RFC 6241, June 2011.

[RFC7296] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., Kivinen, T., "Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 5996, October 2014.

[RFC6071] Frankel, S., Krishnan, S., "IP Security (IPSec) and Internet Key Exchange (IKE) Document Roadmap", February 2011.

6.2. Informative References

[RFC6087] Bierman, A., "Guidelines for Authors and Reviewers of YANG Data Model Documents", RFC 6087, January 2011.

Authors' Addresses

Khanh Tran
Ericsson
300 Holger Way
San Jose, CA 95134
USA

Email: khanh.x.tran@ericsson.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: November 23, 2015

H. Wang
V. Nagaraj
X. Chen
Huawei Technologies
May 22, 2015

Yang Data Model for IKE
draft-wang-ipsecme-ike-yang-00

Abstract

This document describes a YANG data model for the IKE (Internet Key Exchange) protocol. The model covers the IKE protocol configuration, operational state, remote procedural calls, and event notifications data.

Requirements Language

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

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 November 23, 2015.

Copyright Notice

Table of Contents

1. Introduction

The Network Configuration Protocol (NETCONF) [RFC6241] is a network management protocol that defines mechanisms to manage network devices. YANG [RFC6020] is a modular language that represents data structures in an XML tree format, and is used as a data modeling language for the NETCONF.

This document introduces a YANG data model for the IKE (Internet Key Exchange) protocol. There are two IKE protocols defined in IETF namely IKEv1(IKE version 1) and IKEv2(IKE version 2). IKEv1 protocol is obsolete now. The model discussed in this document covers IKEv2 [RFC7296] and other generic enhancements that pertain to the base protocol operation.

The data model is defined for following constructs that are used for managing the IKE protocol: configuration, operational state, remote procedural calls, and event notifications data.

2. IKE YANG Model Organization

2.1. Overview

The model discussd in this document covers IKEv2 [RFC7296] and other generic enhancements that pertain to the base protocol operation. The cryptographic algorithms are deliberately separated from ietf-ike model so that these algorithms can be updated or replaced without affecting the standardization progress of the rest of the IKE yang model. IPsec yang model, basic cryptographic algorithms for IPsec and basic IPsec type defines will be left out of this model to support IPsec basic information defined in [RFC4301]. IPsec yang model will be defined in separate document. IKE data model has the following relationship with IPsec module and other modules.

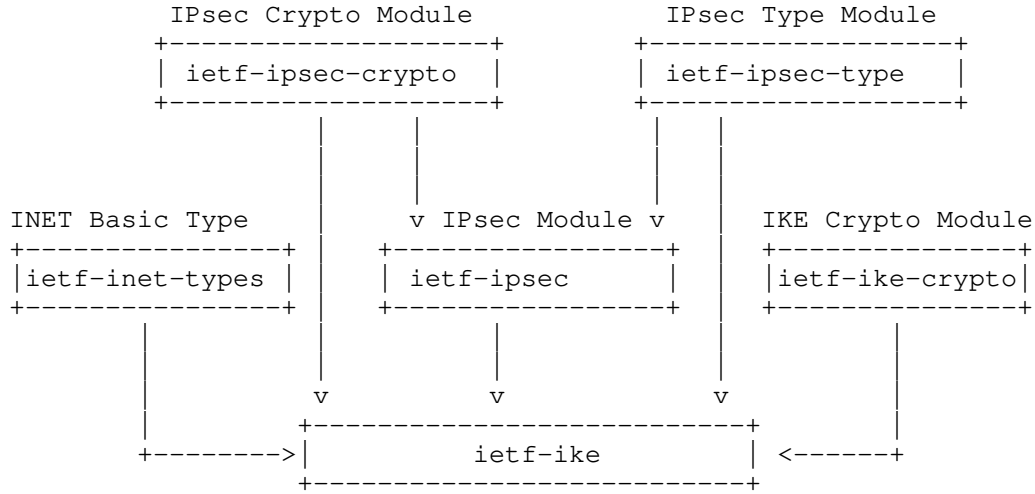


Figure 1: Relationship of IKE with IPsec module and other modules

This model aims to address only the core IKE parameters as per RFC 7296 [RFC7296].

This model does not cover any applications running on top of IKE nor does it cover any OAM procedures for IKE. Current revision only describes one address family of type "ipv4". The "ipv6" specific IKE configuration will be covered in later revision.

The figure below describes the overall structure of the IKE Yang model :

```
module: ietf-ike
   +--rw ike-global-configuration
   |  ...
   +--rw ipsec-proposal
   |  ...
   +--rw ike-proposal
   |  ...
   +--rw ike-peer
   |  ...
   +--rw ipsec-policy
   |  +--rw policy-entries* [policy-name sequence-number]
   |  |  ...
   |  +--rw policy-template-entries* [policy-name sequence-number]
   |     ...
   +--rw ipsec-interface-map
   |  ...
   +--ro ike-sa
   |  ...
   +--ro ipsec-sa
      ...
rpcs:
   +---x reset-ike-sa
   |  ...
   +---x reset-ipsec-sa
      ...
notifications:
   +---n dpd-failure
   |  ...
   +---n peer-authentication-failure
   |  ...
   +---n ike-reauth-failure
   |  ...
   +---n ike-rekey-failure
   |  ...
   +---n ipsec-rekey-failure
      ...
```

2.2. Configuration

This specification defines the configuration parameters for IKE protocols version2 (IKEv2). This specification only supports ipv4 address type for IKE.

2.2.1. IPsec Global Configuration

The IKE global configuration includes some configuration that is common and applicable for all the IKE peers. This includes IKE local name, NAT-Keep-Alive interval, DPD Idle timeout, DPD interval, DPD retry count etc.

```
 +--rw ike-global-configuration
      +--rw (df-flag)?
      |  +--:(set)
      |  |  +--rw set?                        empty
      |  +--:(clear)
      |  |  +--rw clear?                      empty
      |  +--:(copy)
      |     +--rw copy?                       empty
      +--rw stateful-frag-check?      boolean
      +--rw life-time-kb?             uint32
      +--rw life-time-second?         uint32
      +--rw (anti-replay)?
      |  +--:(enable)
      |  |  +--rw enable?                     empty
      |  |  +--rw (anti-replay-windows-size)?
      |  |     +--:(size-32)
      |  |     +--:(size-64)
      |  |     +--:(size-128)
      |  |     +--:(size-256)
      |  |     +--:(size-512)
      |  |     +--:(size-1024)
      |  +--:(disable)
      |     +--rw disable?                    empty
      +--rw inbound-dscp?             uint16
      +--rw outbound-dscp?            uint16
      +--rw local-name?               string
      +--rw nat-keepalive-interval?   uint16
      +--rw dpd-interval?             uint16
```

2.2.2. IPsec Proposal Configuration

The IPsec proposal container will be used to include the configuration items related to the IPsec tunnel like tunnel protocol (sp, ah), tunnel encapsulation mode (tunnel/transport),

authentication algorithm for ah/esp and encryption algorithm for esp etc

```
   +--rw ipsec-proposal
      +--rw ipsec-proposal-entries* [proposal-name]
         +--rw proposal-name                    string
         +--rw (protocol)?
            +--:(ah)
            |  +--rw ah                                 empty
            |  +--rw ah-authentication-algorithm?    ipsec-crypto:ipsec-authenti
cation-algorithm
            +--:(esp)
               +--rw esp                              empty
               +--rw esp-authentication-algorithm?   ipsec-crypto:ipsec-authenti
cation-algorithm
               +--rw esp-encryption-algorithm?       ipsec-crypto:ipsec-encrypti
on-algorithm
```

2.2.3. IKE Proposal Configuration

The IKE proposal container is mainly use to hold information related to the IKE SA establishment parameters. These parameters are mainly negotiated between the IKE peers at the time of SA establishment. The various parameters in this container are proposal number, authentication method, integrity algorithm, encryption algorithm, Psuedo-Random function (prf), dh group, reauth , rekey lifetime etc

```
    +--rw ike-proposal
      +--rw ike-proposal-entries* [proposal-number]
         +--rw proposal-number        uint32
         +--rw auth-method?           ike-auth-method
         +--rw integrity-algorithm?   ike-crypto:ike-integrity-algorithm
         +--rw encrypt-algorithm?     ike-crypto:ike-encryption-algorithm
         +--rw prf-algorithm?         ike-crypto:ike-prf-algorithm
         +--rw dh-group?              ike-crypto:ike-dh-group
         +--rw reauth-interval?       uint32
         +--rw life-time?             uint32
```

2.2.4. IKE Peer Configuration

The IKE peer container will hold information about peer. The IKE peer is an entity that is going to establish security association with the remote peer. The main configuration parameters related to the IKE peer are: Key information, Name, proposal number, ID type, remote address, local address, certificate information etc

```
      +--rw ike-peer
         +--rw ike-peer-entries* [peer-name]
            +--rw peer-name               string
            +--rw ike-proposal-number?    ike-proposal-number-ref
            +--rw PresharedKey?           string
            +--rw nat-traversal?          boolean
            +--rw (local-id-type)?
            |  +--:(ip)
            |  |  +--rw ip?                      empty
            |  +--:(fqdn)
            |  |  +--rw fqdn?                    empty
            |  +--:(dn)
            |  |  +--rw dn?                      empty
            |  +--:(user_fqdn)
            |     +--rw user_fqdn?               empty
            +--rw local-id?               string
            +--rw remote-id?              string
            +--rw low-remote-address?     inet:ip-address
            +--rw high-remote-address?    inet:ip-address
            +--rw certificate?            string
            +--rw auth-address-begin?     inet:ip-address
            +--rw auth-address-end?       inet:ip-address
```

2.2.5. IPsec Policy Configuration

The IPsec policy container will hold values related to the IPsec policy that is bound to an interface (tunnel or physical interface). The information contained in the IPsec policy will determine the characteristics of the tunnel that is going to be establishment. The main attributes related to IPsec policy are: ACL, PFS (to do an additional DH exchange), peer name, IPsec proposal number, policy name, sequence number, policy-mode (ISAKMP, Template etc)

```
       +--rw ipsec-policy
         +--rw policy-entries* [policy-name sequence-number]
         |  +--rw policy-name               string
         |  +--rw sequence-number           uint32
         |  +--rw (policy-mode)?
         |     +--:(isakmp)
         |        |  +--rw isakmp?                    empty
         |        |  +--rw local-address?             inet:ip-address
         |        |  +--rw binding-interface-name?    string
         |        |  +--rw (acl)?
         |        |  |  +--:(acl-number)
         |        |  |  |  +--rw acl-number?               uint32
         |        |  |  +--:(advance-acl)
         |        |  |     +--rw advance-acl?              string
         |        |  +--rw pfs?                       ike-crypto:ike-dh-group
```

```
|      |  +--rw peer-name?                ike-peer-name-ref
|      |  +--rw (df-flag)?
|      |  |  +--:(set)
|      |  |  |  +--rw set?                      empty
|      |  |  +--:(clear)
|      |  |  |  +--rw clear?                    empty
|      |  |  +--:(copy)
|      |  |     +--rw copy?                     empty
|      |  +--rw stateful-frag-check?      boolean
|      |  +--rw life-time-kb?             uint32
|      |  +--rw life-time-second?         uint32
|      |  +--rw (anti-replay)?
|      |  |  +--:(enable)
|      |  |  |  +--rw enable?                   empty
|      |  |  |  +--rw (anti-replay-windows-size)?
|      |  |  |     +--:(size-32)
|      |  |  |     +--:(size-64)
|      |  |  |     +--:(size-128)
|      |  |  |     +--:(size-256)
|      |  |  |     +--:(size-512)
|      |  |  |     +--:(size-1024)
|      |  |  +--:(disable)
|      |  |     +--rw disable?                  empty
|      |  +--rw inbound-dscp?             uint16
|      |  +--rw outbound-dscp?            uint16
|      |  +--rw ipsec-proposal* [proposal-name]
|      |     +--rw proposal-name    ipsec-proposal-name-ref
|      +--:(template)
|         +--rw template?                  empty
|         +--rw template-name              ipsec-policy-template-name-ref
+--rw policy-template-entries* [policy-name sequence-number]
   +--rw policy-name                string
   +--rw sequence-number            uint32
   +--rw local-address?             inet:ip-address
   +--rw binding-interface-name?    string
   +--rw (acl)?
   |  +--:(acl-number)
   |  |  +--rw acl-number?                    uint32
   |  +--:(advance-acl)
   |     +--rw advance-acl?                   string
   +--rw pfs?                       ike-crypto:ike-dh-group
   +--rw peer-name?                 ike-peer-name-ref
   +--rw (df-flag)?
   |  +--:(set)
   |  |  +--rw set?                           empty
   |  +--:(clear)
   |  |  +--rw clear?                         empty
   |  +--:(copy)
```

```
|     +--rw copy?                      empty
+--rw stateful-frag-check?      boolean
+--rw life-time-kb?             uint32
+--rw life-time-second?         uint32
+--rw (anti-replay)?
|  +--:(enable)
|  |  +--rw enable?                    empty
|  |  +--rw (anti-replay-windows-size)?
|  |     +--:(size-32)
|  |     +--:(size-64)
|  |     +--:(size-128)
|  |     +--:(size-256)
|  |     +--:(size-512)
|  |     +--:(size-1024)
|  +--:(disable)
|     +--rw disable?                   empty
+--rw inbound-dscp?             uint16
+--rw outbound-dscp?            uint16
+--rw ipsec-proposal* [proposal-name]
   +--rw proposal-name    ipsec-proposal-name-ref
```

2.2.6. IPsec Interface Map Configuration

The IPsec interface map container will have information related to the interface on which IPsec policy will be applied. It will have information like IPsec policy name, tunnel protocol, tunnel name , enable-disable UNR route generation etc

```
+--rw ipsec-interface-map
  +--rw policy-interface* [interface-name]
     +--rw interface-name        string
     +--rw policy-name           ipsec-policy-name-ref
     +--rw generate-unr-route?   boolean
```

2.3. Operational State

The Operational state of the IKE SA or IPsec SA can be queried and obtained from the respective container. All the attributes/items in this container are read-only attributes and they reflect the run-time information of any established IKE SA.

2.3.1. IKE SA Container State

The IKE SA container is used to maintain information related to the IKE SA established. This SA is a run-time data structure that is created and has information about established SA like SPI, local and remote address, established time, remaining life time, dh group, auth method, prf, encryption algorithm , integrity algorithm etc

```
   +--ro ike-sa
      +--ro ike-sa-entries* [initiator-spi responder-spi]
         +--ro remote-address?         inet:ip-address
         +--ro local-address?          inet:ip-address
         +--ro initiator-spi           uint64
         +--ro responder-spi           uint64
         +--ro remote-id?              string
         +--ro local-id?               string
         +--ro auth-method?            ike-auth-method
         +--ro integrity-algorithm?    ike-crypto:ike-integrity-algorithm
         +--ro encryption-algorithm?   ike-crypto:ike-encryption-algorithm
         +--ro prf-algorithm?          ike-crypto:ike-prf-algorithm
         +--ro dh-group?               ike-crypto:ike-dh-group
         +--ro sa-established-time?    string
         +--ro remaining-time?         uint32
```

2.3.2. IPsec SA State

The IPsec SA container is used to maintain information related to the IPsec SA established. This is a run-time data structure that is created and has information about established SA like SPI, local and remote address, remaining life time, protocol etc

```
   +--ro ipsec-sa
         +--ro ipsec-sa-entries*
            +--ro remote-address?   inet:ip-address
            +--ro local-address?    inet:ip-address
            +--ro responder-spi?    uint32
            +--ro remaining-time?   uint32
```

2.4. Actions

This model defines a list of RPCs that allow performing an action or executing a command on the protocol. For example, it allows clearing (reset) IKE SAs, IPsec SAs, statistics etc. The model makes an effort to provide different level of control so that a user is able to either clear all, or clear all of a given type, or clear a specific entity.

2.4.1. IKE SA reset action

This operation type is executed when the user wants to delete IKE SAs. The command gives a flexibility to delete all SAs or a particular SA only based on remote address, connection-id etc.

```
      +---x reset-ike-sa
         +---w input
         |  +---w (peer-info)?
         |     +--:(peer-id)
         |     |  +---w peer-id?        string
         |     +--:(peer-address)
         |        +---w peer-address?   inet:ip-address
         +--ro output
            +--ro status?   string
```

2.4.2. IPsec SA reset action

This operation type is executed when the user wants to delete IPsec SAs. The command gives a flexibility to delete all SAs or a particular SA only based on remote address, policy sequence number, remote peer address etc.

```
   +---x reset-ipsec-sa
         +---w input
         |  +---w (sa-info)?
         |     +--:(parameters)
         |     |  +---w (peer-info)
         |     |  |  +--:(peer-id)
         |     |  |  |  +---w peer-id?        string
         |     |  |  +--:(peer-address)
         |     |  |     +---w peer-address?   inet:ip-address
         |     |  +---w protocol        ipsec-type:ipsec-protocol
         |     |  +---w spi             ipsec-type:ipsec-spi
         |     +--:(remote-peer)
         |     |  +---w remote-peer
         |     |     +---w (peer-info)?
         |     |        +--:(peer-id)
         |     |        |  +---w peer-id?        string
         |     |        +--:(peer-address)
         |     |           +---w peer-address?   inet:ip-address
         |     +--:(policy)
         |        +---w policy?         ipsec-policy-name-ref
         +--ro output
            +--ro status?   string
```

2.5. Notifications

This model defines a list of notifications to inform client of important events detected during the protocol operation. These events include events related to changes in the operational state of an IKE SA, IPsec SA, Statistics etc.

2.5.1. DPD failure

This notification type is reported to the NETCONF client when there is a peer that is not responding to the DPD Keep Alive messages.

```
    +---n dpd-failure
         +--ro peer-id?   string
```

2.5.2. Peer Authentication failure

This notification type is reported to the NETCONF client when the peer authentication has failed due to either invalid key, certificate, invalid id etc

```
+---n peer-authentication-failure
    +--ro peer-id?   string
```

2.5.3. IKE Reauth failure

This notification type is reported to the NETCONF client when the re-authentication of the peer has failed. Reauth can fail due to many reasons like proposal mismatch during re-auth procedure, packet drop, dead peer etc

```
+---n ike-reauth-failure
  +--ro peer-id?   string
```

2.5.4. IKE Rekey failure

This notification type is reported to the NETCONF client when the IKE SA rekey has failed. The rekey is an operation used to refresh the IKE SA keys. It can fail due to proposal mismatch during rekey procedure, packet drop, dead peer etc

```
+---n ike-rekey-failure
    +--ro peer-id?     string
    +--ro old-i-spi?   uint64
  +--ro old-r-spi?   uint64
```

2.5.5. IPsec Rekey failure

This notification type is reported to the NETCONF client when the IKE SA rekey has failed. The rekey is an operation used to refresh the IPsec SA keys. It can fail due to proposal mismatch during rekey procedure, packet drop, dead peer etc

```
    +---n ipsec-rekey-failure
         +--ro peer-id?            string
         +--ro old-inbound-spi?    ipsec-type:ipsec-spi
         +--ro old-outbound-spi?   ipsec-type:ipsec-spi
```

3. IKE Yang Module

To support separately upgrade the algorithm part, the base data model and the algorithm part are defined as two separately parts.

3.1. IKE Basic Yang Module

```
module ietf-ike {
    namespace "urn:ietf:params:xml:ns:yang:ietf-ike";
    // replace with IANA namespace when assigned
    prefix "ike";

    import "ietf-ipsec-crypto" {
        prefix "ipsec-crypto";
    }

    import "ietf-ike-crypto" {
        prefix "ike-crypto";
    }

    import "ietf-inet-types" {
        prefix "inet";
    }

    import "ietf-ipsec-type" {
        prefix "ipsec-type";
    }

    import "ietf-ipsec" {
        prefix "ipsec";
    }

    organization "Huawei Technologies India Pvt Ltd";
    contact "stonewater.wang@huawei.com";
    description "IKE Yang module define";

    revision 2015-04-18 {
        description "Initial revision.";
        reference "RFC XXXX: IKE Yang Modules";
    }
```

```
    grouping ipsec-common-configuration {
        choice df-flag {
            default copy;
            case set {
                leaf set {
                    type empty;
                    description
                        "Set the df bit when encapsulate IPsec tunnel.";
                }
            }
            case clear {
                leaf clear {
                    type empty;
                    description
                        "Clear the df bit when encapsulate IPsec tunnel.";
                }
            }
            case copy {
                leaf copy {
                    type empty;
                    description
                        "Copy the inner IP header df bit.";
                }
            }

            description
                "It indicates how to process the df bit when encapsulate IPsec t
unnel.";
        }
        leaf stateful-frag-check {
            type boolean;
            default false;
            description "Whether stateful fragment checking applies.";
        }
        leaf life-time-kb {
            type uint32;
            units "KB";
            default 2000000;

            description "IPsec SA Life time in KB.";
        }
        leaf life-time-second {
            type uint32;
            units "Second";
            default 18400;
            description "IPsec SA Life time in Seconds";
        }
        choice anti-replay {
            default enable;
```

```
            case enable {
               leaf enable {
                    type empty;
                    description "Enable Anti-replay";
               }
               choice anti-replay-windows-size {

                    case size-32;
                    case size-64;
                    case size-128;
                    case size-256;
                    case size-512;
                    case size-1024;
                    default size-1024;
                    description "It indicate the size of anti-replay window";
               }
            }
            case disable {
                leaf disable {
                    type empty;
                    description "Disable Anti-replay";
                }
            }
            description "Whether enable or disable anti-replay";
        }

        leaf inbound-dscp {
            type uint16 {
                range "0..63";
            }
            default 0;
            description "Inbound DSCP value";
        }
        leaf outbound-dscp {
            type uint16 {
                range "0..63";
            }
            default 0;
            description "Outbound DSCP value";
        }
        description "Common IPsec configurations";
    }

    grouping choose-ipsec-peer {
        choice peer-info {
            case peer-id {
                leaf peer-id {
                    type string;
```

```
                    description "Peer ID";
                }
            }
            case peer-address {
                leaf peer-address {
                    type inet:ip-address;
                    description "Peer IP Address";
                }
            }
            description "Reset according to peer information";
        }
        description "IKE peer information when do reset operation";
    }

    typedef ike-peer-name-ref {
        type leafref {
            path "/ike-peer/ike-peer-entries/peer-name";
        }
        description "reference to ike peer name";
    }

    typedef ike-proposal-number-ref {
        type leafref {
            path "/ike-proposal/ike-proposal-entries/proposal-number";
        }
        description "reference to ike proposal number";
    }

    typedef ipsec-proposal-name-ref{
        type leafref {
            path "/ipsec-proposal/ipsec-proposal-entries/proposal-name";
        }
        description "reference to ike proposal name";
    }

    typedef ipsec-policy-template-name-ref {
        type leafref {
            path "/ipsec-policy/policy-template-entries/policy-name";
        }
        description "reference to ipsec policy template name";
    }

    typedef ike-auth-method {
        type enumeration {
            enum pre-share {
                description "Select pre-shared key message as the authentication
 method";
            }
            enum rsa-digital-signature {
```

```
                description "Select rsa digital signature as the authentication
method";
            }
            enum dss-digital-signature {
                description "Select dss digital signature as the authentication
method";
            }
        }
        description "IKE authentication methods";
    }

    typedef ipsec-policy-name-ref {
        type leafref {
            path "/ipsec-policy/policy-entries/policy-name";
        }
        description "reference to ipsec policy name";
    }

    container ike-global-configuration {
        description "Global IKE configurations";

        uses ipsec-common-configuration;

        leaf local-name {
            type string;
            description "Global local name configuration, if it is not configed,
                         ip address will be used as default. If configing special
                         local name for special peer, it will overwrite the globa
l
                         name configuration when negotion with that peer.";
        }
        leaf nat-keepalive-interval {

            type uint16 {
                range "5..300";
            }
            units "Seconds";
            default 20;
            description "Global nat keepalive interval";
        }
        leaf dpd-interval {
            type uint16 {
                range "10..3600";
            }

            units "Seconds";
            default 30;
            description "Global DPD interval";
        }
    }
```

```
    container ipsec-proposal {
        description "IPsec proposal information";

        list ipsec-proposal-entries {
            key "proposal-name";
            description "IPsec proposal information";

            leaf proposal-name {
                type string;
                mandatory true;
                description "Name of IPsec proposal.";
            }
            choice protocol {
                default esp;
                case ah {
                    leaf ah {
                        type empty;
                        mandatory true;
                        description "Choose AH as IPsec protocol";
                    }
                    leaf ah-authentication-algorithm {
                        type ipsec-crypto:ipsec-authentication-algorithm;
                        must "ah-authentication-algorithm != 'null'" {
                            error-message "AH authentication algorithm MUST not
be null";
                            description "AH authentication algorithm MUST not be
 null";

                        }
                        default sha2-256;
                        description "IPsec authentication algorithm for AH";
                    }
                    description "Choose AH as IPsec protocol";
                }
                case esp {
                    leaf esp {
                        type empty;
                        description "Choose ESP as IPsec protocol";
                    }
                    leaf esp-authentication-algorithm {
                        type ipsec-crypto:ipsec-authentication-algorithm;
                        default sha2-256;
                        description "IPsec authentication algorithm for ESP";
                    }

                    leaf esp-encryption-algorithm {
                        type ipsec-crypto:ipsec-encryption-algorithm;
                        default aes-256;
                        description "IPsec encryption algorithm for ESP";
```

```
                    }
                    must "esp-authentication-algorithm != 'null' or esp-encrypti
on-algorithm != 'null'" {

                        error-message "ESP authentication algorithm and encrypti
on algorithm can not be both null";
                        description "ESP authentication algorithm and encryption
 algorithm can not be both null";
                    }
                    description "Choose ESP as IPsec protocol";
                }
                description "Choose IPsec protocol";
            }

        } //End of IPsecProposalEntries
    }//End of IPsec Proposal

    container ike-proposal {
        description "IKE proposal information";

        list ike-proposal-entries {

            key "proposal-number";
            description "IKE proposal information";

            leaf proposal-number {
                type uint32;
                mandatory true;
                description "Proposal seq-number of ike proposal";
            }
            leaf auth-method {

                type ike-auth-method;
                default pre-share;
                description "authentication method of ike peer";
            }
            leaf integrity-algorithm {

                type ike-crypto:ike-integrity-algorithm;
                default hmac-sha2-256;
                description "integrity algorithm of ike protocol";
            }
            leaf encrypt-algorithm {

                type ike-crypto:ike-encryption-algorithm;
                default aes-cbc-256;
                description "Encryption algorithm of ike protocol";
            }
            leaf prf-algorithm {
                type ike-crypto:ike-prf-algorithm;
                default hmac-sha2-256;
```

```
                description "Prf algorithm of ike protocol";
            }
            leaf dh-group {

                type ike-crypto:ike-dh-group;
                must "dh-group != 'dh-group-none'" {
                    error-message "DH Group MUST be configurated";
                    description "DH Group MUST be configurated";
                }
                default dh-group-2;

                description "DH group of ike protocol";
            }
            leaf reauth-interval {

                type uint32 {
                    range "60..604800";
                }
                units "Seconds";
                default 86400;
                description "Reauth interval time of IKE protocol";
            }
            leaf life-time {
                type uint32 {
                    range "60..604800";
                }
                units "Seconds";
                default 86400;
                description "IKE SA life time";
            }

        } //End of IKEProposal
    }
    container ike-peer {
        description "IKE peer information";

        list ike-peer-entries {

            key "peer-name";
            description "IKE peer information";

            leaf peer-name {
                type string;
                mandatory true;
                description "Name of IKE peer";
            }

            leaf ike-proposal-number {
```

```
                type ike-proposal-number-ref;
                description "IKE proposal number referenced by IKE peer";
            }

            leaf PresharedKey {
                type string;
                description "Preshare key";
            }

            leaf nat-traversal {
                type boolean;
                default false;
                description "Enable/Disable nat traversal";
            }

            choice local-id-type {
                default ip;
                case ip {
                    leaf ip {
                        type empty;
                        description "IP address";
                    }
                }
                case fqdn {
                    leaf fqdn {
                        type empty;
                        description "Fully Qualifed Domain name ";
                    }
                }
                case dn {
                    leaf dn {
                        type empty;
                        description "Domain name";
                    }
                }
                case user_fqdn {
                    leaf user_fqdn {
                        type empty;
                        description "User FQDN";
                    }
                }
                description "Local ID type";
            }
            leaf local-id {
                type string;
                description "Local ID Name. When IP is used as local ID type,
                             it is ignored. If it is not configurated,
                             global local name will be used.";
```

```
            }
            leaf remote-id {
                type "string";
                description "ID of IKE peer";
            }
            leaf low-remote-address {
                type inet:ip-address;
                description "Low range of remote address";
            }
            leaf high-remote-address {
                type inet:ip-address;
                description "High range of remote address";
            }
            leaf certificate {
                type string;
                description "Certificate file name";
            }
            leaf auth-address-begin {
                type inet:ip-address;
                description "The begin range of authenticated peer address";
            }
            leaf auth-address-end {
                type inet:ip-address;
                description "The end range of authenticated peer address";
            }
        }

    }//End of IKEPeerEntries
    container ipsec-policy {
        description "IPsec policy information";

        grouping policy-content {
            leaf local-address {
                type inet:ip-address;
                description
                    "Local address used by IKE when negotiate with peer,
                     if it is not configed, the interface address with bind
                     this ipsec policy will be used.";
            }
            leaf binding-interface-name {
                type string;
                description "The interface that the policy is already bind with"
;
            }
            choice acl {
                case acl-number {
                    leaf acl-number {
                        type uint32 {
                            range "3000..3999";
```

```
                        }
                        description "Config common acl as IPsec traffic selector
";
                    }

                }
                case advance-acl {
                    leaf advance-acl {
                        type string {
                            length "1..32";
                        }
                        description "Config advance acl as IPsec traffic selecto
r";
                    }
                }
                description "Config acl as IPsec traffic selector";
            }


            leaf pfs {
                type ike-crypto:ike-dh-group;
                default dh-group-none;
                description
                    "Whether choose different DH group with IKE SA when create
                     ipsec SA to increase perfect forwarding security";
            }

            leaf peer-name {
                type ike-peer-name-ref;
                description "The ike peer binding with this policy";
            }

            uses ipsec-common-configuration {
                description "The common configuration of IPsec SA";
            }

            list ipsec-proposal {
                key "proposal-name";
                max-elements "6";
                description "The ipsec-proposals binding with the policy";

                leaf proposal-name {
                    type ipsec-proposal-name-ref;
                    description "The ipsec-proposals binding with the policy";
                }
            }
            description "IPsec policy content";
        }

        list policy-entries {
```

```
            key "policy-name sequence-number";
            description "IPsec policy information";

            leaf policy-name {
                type string;
                mandatory true;
                description "IPsec policy group name";

            }
            leaf sequence-number {
                type uint32;
                mandatory true;
                description "IPsec policy sequence number";
            }
            choice policy-mode {

                case isakmp {
                    leaf isakmp {
                        type empty;
                        description "Common ISAKMP IPsec policy";
                    }
                    uses policy-content {
                        description "common ipsec policy content";
                    }
                }
                case template {
                    leaf template {
                        type empty;
                        description "ISAKMP IPsec policy created using template"
;
                    }
                    leaf template-name {
                        type ipsec-policy-template-name-ref;
                        mandatory true;
                        description
                            "The IPsec policy template name which is used to cre
ate this policy";
                    }
                }
                default isakmp;
                description "IPsec policy mode";

            }

        }

        list policy-template-entries {
            key "policy-name sequence-number";
            description "IPsec policy template define";
```

```
            leaf policy-name {
                type string {
                    length "1..15";
                }
                mandatory true;
                description "IPsec policy template name";
            }
            leaf sequence-number {
                type uint32;
                mandatory true;
                description "Sequence number of policy template";

            }
            uses policy-content {
                description "common ipsec policy content";
            }
        }

    }
    container ipsec-interface-map {
        description "The map information between IPsec policy and interface";

        list policy-interface {
            key "interface-name";
            description "The map information between IPsec policy and interface"
;

            leaf interface-name {
                type string;
                mandatory true;
                description "Interface name which will bind IPsec policy";
            }
            leaf policy-name {
                type ipsec-policy-name-ref;
                mandatory true;
                description "IPsec policy name";
            }
            leaf generate-unr-route {
                type boolean;
                default false;
                description "Whether generate UNR route";
            }
        }
    }

    container ike-sa {
        config false;
        description "IKE SA informations";
```

```
        list ike-sa-entries {

            key "initiator-spi responder-spi";

            description "IKE SA informations";

            leaf remote-address {
                type inet:ip-address;
                description "The IP address of the remote peer";
            }
            leaf local-address {
                type inet:ip-address;
                description "The IP address of local";
            }
            leaf initiator-spi {
                type uint64;
                description "The SPI of initiator";
            }
            leaf responder-spi {
                type uint64;
                description "The SPI of responder";
            }
            leaf remote-id {
                type string;
                description "The ID of the remote peer";
            }
            leaf local-id {
                type string;
                description "The ID of local";
            }
            leaf auth-method {
                type ike-auth-method;
                description "The authentication method of IKE peer";
            }
            leaf integrity-algorithm {
                type ike-crypto:ike-integrity-algorithm;
                description "The integrity algorithm chosen by IKE negotiation";
            }
            leaf encryption-algorithm {
                type ike-crypto:ike-encryption-algorithm;
                description "The encryption algorithm chosen by IKE negotiation"
;
            }
            leaf prf-algorithm {
                type ike-crypto:ike-prf-algorithm;
                description "The PRF algorithm chosen by IKE negotiation";
            }
            leaf dh-group {
                type ike-crypto:ike-dh-group;
```

```
                description "The DH group chosen by IKE negotiation";
            }
            leaf sa-established-time {
                type string;
                description "The establish time of the IKE SA";
            }
            leaf remaining-time {
                type uint32;
                description "The remain life time of IKE SA";
            }

        }

    }
    container ipsec-sa {
        config false;
        description "IPsec SA information";

        list ipsec-sa-entries {

            description "IPsec SA information";

            leaf remote-address {
                type inet:ip-address;
                description "The IP address of the remote tunnel end-point";
            }
            leaf local-address {
                type inet:ip-address;
                description "The IP address of local tunnel end-point";
            }
            leaf responder-spi {
                type uint32;
                description "The SPI of responder";
            }

            leaf remaining-time {
                type uint32;
                description "The remain life time of IPsec SA";
            }

        }

    }

    rpc reset-ike-sa {
        description "Reset IKE SA";
        input {
```

```
            uses choose-ipsec-peer;
            description "Reset IKE SA";
        }
        output {
            leaf status {
                type string;
                description "Operation status";
            }
        }
    }
    rpc reset-ipsec-sa {
        description "Reset IPsec SA";
        input {
            choice sa-info {
                case parameters {
                    uses choose-ipsec-peer {
                        refine "peer-info" {
                            mandatory true;
                        }
                    }
                    leaf protocol {
                        type ipsec-type:ipsec-protocol;
                        mandatory true;
                        description "SA protocol";
                    }
                    leaf spi {

                        type ipsec-type:ipsec-spi;
                        mandatory true;
                        description "SA SPI";
                    }
                    description "Reset according to special parameters";
                }

                case remote-peer {
                    container remote-peer {
                        uses choose-ipsec-peer;
                        description "Reset according to remote peer";
                    }
                }
                case policy {
                    leaf policy {
                        type ipsec-policy-name-ref;
                        description "Reset according to IPsec policy name";
                    }
                }
                description "Reset according to special information";
```

```
            }

        }
        output {
            leaf status {
                type string;
                description "Operation status";
            }
        }
    }

    notification dpd-failure{
        description "IKE peer DPD detect failure";
        leaf peer-id {
            type string;
            description "Peer ID";
        }
    }

    notification peer-authentication-failure {
        description "Peer authentication fail when negotication";
        leaf peer-id {
            type string;
            description "The ID of remote peer";
        }
    }

    notification ike-reauth-failure {
        description "IKE peer reauthentication fail";
        leaf peer-id {
            type string;
            description "The ID of remote peer";
        }
    }

    notification ike-rekey-failure {
        description "IKE SA rekey failure";
        leaf peer-id {
            type string;
            description "The ID of remote peer";
        }
        leaf old-i-spi {
            type uint64;
            description "old SPI";
        }
        leaf old-r-spi {
            type uint64;
            description "old SPI";
```

```
        }
    }

    notification ipsec-rekey-failure {
        description "IPsec SA rekey failure";
        leaf peer-id {
            type string;
            description "The ID of remote peer";
        }
        leaf old-inbound-spi {
            type ipsec-type:ipsec-spi;
            description "old inbound SPI";
        }
        leaf old-outbound-spi {
            type ipsec-type:ipsec-spi;
            description "old outbound SPI";
        }
    }


}
```

3.2. IKE Algorithm Yang Module

```
   module ietf-ike-crypto {
     namespace "urn:ietf:params:xml:ns:yang:ietf-ike-crypto";
     prefix ike-crypto;

     organization "Huawei Technologies India Pvt Ltd";
     contact
       "stonewater.wang@huawei.com";
     description
       "IKE Crypto Yang";
     reference "RFC 7296: Internet Key Exchange Protocol Version 2";

     revision 2015-04-18 {
       description
         "Initial revision.";
       reference "RFC 7296: Internet Key Exchange Protocol Version 2";
     }

     typedef ike-integrity-algorithm {
       type enumeration {
         enum "hmac-md5-96" {
           description
             "HMAC-MD5-96 Integrity Algorithm";
         }
         enum "hmac-sha1-96" {
```

```
          description
            "HMAC-SHA1-96 Integrity Algorithm";
        }
        enum "hmac-sha2-256" {
          description
            "HMAC-SHA2-256 Integrity Algorithm";
        }
        enum "hmac-sha2-384" {
          description
            "HMAC-SHA2-384 Integrity Algorithm";
        }
        enum "hmac-sha2-512" {
          description
            "HMAC-SHA2-512 Integrity Algorithm";
        }
      }
      description
        "typedef for ike integrity algorithm.";
    }

    typedef ike-encryption-algorithm {
      type enumeration {
        enum "des-cbc" {
          description
            "DES-CBC Encryption algorithm";
        }
        enum "3des-cbc" {
          description
            "3DES-CBC Encryption algorithm";
        }
        enum "aes-cbc-128" {
          description
            "AES-CBC-128 Encryption algorithm";
        }
        enum "aes-cbc-192" {
          description
            "AES-CBC-192 Encryption algorithm";
        }
        enum "aes-cbc-256" {
          description
            "AES-CBC-256 Encryption algorithm";
        }
      }
      description
        "typedef for ike encryption algorithm.";
    }

    typedef ike-prf-algorithm {
```

```
      type enumeration {
        enum "hmac-md5-96" {
          description
            "HMAC-MD5-96 PRF Algorithm";
        }
        enum "hmac-sha1-96" {
          description
            "HMAC-SHA1-96 PRF Algorithm";
        }
        enum "hmac-sha2-256" {
          description
            "HMAC-SHA2-256 PRF Algorithm";
        }
        enum "hmac-sha2-384" {
          description
            "HMAC-SHA2-384 PRF Algorithm";
        }
        enum "hmac-sha2-512" {
          description
            "HMAC-SHA2-512 PRF Algorithm";
        }
      }
      description
        "typedef for ike prf algorithm.";
    }

    typedef ike-dh-group {
      type enumeration {
        enum "dh-group-none" {
          description
            "None Diffie-Hellman group";
        }
        enum "dh-group-1" {
          description
            "768 bits Diffie-Hellman group";
        }
        enum "dh-group-2" {
          description
            "1024 bits Diffie-Hellman group";
        }
        enum "dh-group-5" {
          description
            "1536 bits Diffie-Hellman group";
        }
        enum "dh-group-14" {
          description
            "2048 bits Diffie-Hellman group";
        }
```

```
      }
      description
        "typedef for ike dh group";
    }
  }
```

4. IANA Considerations

This document registers the following URIs in the IETF XML registry [RFC3688]. Following the format in [RFC3688], the following registration is requested to be made.

URI: urn:ietf:params:xml:ns:yang:ietf-ike XML: N/A, the requested URI is an XML namespace.

URI: urn:ietf:params:xml:ns:yang:ietf-ike-crypto XML: N/A, the requested URI is an XML namespace.

This document registers a YANG module in the YANG Module Names registry [RFC6020].

name: ietf-ike namespace: urn:ietf:params:xml:ns:yang:ietf-ike prefix: ike reference: [RFC7296]

name: ietf-ike-crypto namespace: urn:ietf:params:xml:ns:yang:ietf-ike-crypto prefix: ike-crypto reference: [RFC7296]

5. Security Considerations

The YANG module defined in this memo is designed to be accessed via the NETCONF protocol[RFC6241]. The lowest NETCONF layer is the secure transport layer and the mandatory-to-implement secure transport is SSH [RFC6242]. The NETCONF access control model [RFC6536] provides means to restrict access for particular NETCONF users to a pre-configured subset of all available NETCONF protocol operations and content. There are a number of data nodes defined in the YANG module which 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.

6. Acknowledgements

7. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, January 2004.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

[RFC6020] Bjorklund, M., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, October 2010.

[RFC6241] Enns, R., Bjorklund, M., Schoenwaelder, J., and A. Bierman, "Network Configuration Protocol (NETCONF)", RFC 6241, June 2011.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, June 2011.

[RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, March 2012.

[RFC7296] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", STD 79, RFC 7296, October 2014.

Authors' Addresses

Honglei Wang
Huawei Technologies
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: stonewater.wang@huawei.com

Vijay Kumar Nagaraj
Huawei Technologies
Huawei Technologies India Pvt Ltd
Bangalore 560008
India

Email: vijay.kn@huawei.com

Xia Chen
Huawei Technologies
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: xiachen@huawei.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 17, 2015

H. Wang
V. Nagaraj
X. Chen
Huawei Technologies
June 15, 2015

Yang Data Model for IPsec
draft-wang-ipsecme-ipsec-yang-00

Abstract

This document describes a YANG data model for the IPsec(Internet Protocol Security) protocol. The model covers the IPsec protocol operational state and remote procedural calls.

Requirements Language

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

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 December 17, 2015.

Copyright Notice

Table of Contents

1. Introduction

The Network Configuration Protocol (NETCONF) [RFC6241] is a network management protocol that defines mechanisms to manage network devices. YANG [RFC6020] is a modular language that represents data structures in an XML tree format, and is used as a data modeling language for the NETCONF.

This document introduces a YANG data model for the IPsec(Internet Protocol Security) protocol[RFC4301]. The data model is defined for following constructs that are used for managing the IPsec protocol: operational state and remote procedural calls.

2. IPsec YANG Model Organization

2.1. Overview

The model discussed in this document covers IPsec[RFC4301] and other generic enhancements that pertain to the base protocol operation. The cryptographic algorithms are deliberately separated from ietf-ipsec model so that these algorithms can be updated or replaced

without affecting the standardization progress of the rest of the IPsec yang model.

^: import

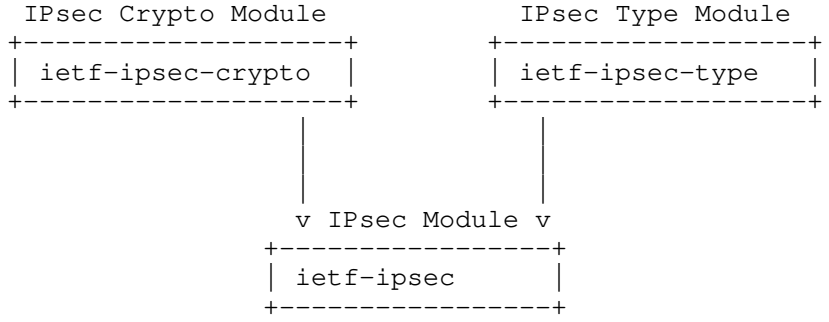


Figure 1: Relationship of IPsec module and other modules

This model aims to address only the core IPsec parameters as per [RFC4301]. This model does not cover any applications running on top of IPsec nor does it cover any OAM procedures for IPsec. Current revision only describes SAD and SPD, PAD will be covered in later revision.

Different IPsec implements may have different behaviors, e.g. a host may directly bind IPsec SA with socket, then SPD is not necessary; while a gateway may supply interfaces for IKE[RFC7296] to modify IPsec SPD entries. So we defined only the basic prototype of the data model, and all the databases are defined as read only. Any other extension and augment of the data model are left for implements.

The figure below describes the overall structure of the IPsec Yang model:

```
module: ietf-ipsec
   +--ro sad
   |  ...
   +--ro spd
   |  ...
   +--ro ipsec-global-statistics
      +--ro ipv4
      |  ...
      +--ro ipv6
      |  ...
      +--ro global
         ...
rpcs:
   +---x reset-ipv4
   |  ...
   +---x reset-ipv6
   |  ...
   +---x reset-global
      ...
```

2.2. Operational State

The Operational state of the IPsec can be queried and obtained from the respective container. All the attributes/items in this container are read-only attributes and they reflect the run-time information of IPsec database.

2.2.1. IPsec SAD State

The IPsec SAD(Security Association Database) container maintains information related to the IPSEC SAs established in a system. This is a run-time data structure that is created upon the first SA being established. The key for fetching SA in this database is the triplet: SPI, Protocol and Destination address of the SA to be fetched form the SA database.

The SAD entries also contain information about the IPSEC tunnel like direction, SA-type (manual or VPN SA), sequence number, anti-replay window size, protocol mode, ipsec algorithm info, life time in Seconds/Bytes etc, NAT traversal info, path-mtu, dscp etc.

```
+--ro sad
   +--ro sad-entries* [spi security-protocol direction]
      +--ro spi                                ipsec-type:ipsec-spi
      +--ro security-protocol                  ipsec-type:ipsec-protocol
      +--ro direction                          ipsec-type:ipsec-traffic-direction
      +--ro sa-type?                           enumeration
      +--ro sequence-number?                   uint64
      +--ro sequence-number-overflow-flag?   boolean
      +--ro anti-replay-enable-flag?         boolean
      +--ro anti-replay-window-size?         uint64
      +--ro ah-auth-algorithm?                 ipsec-crypto:ipsec-authentication-a
lgorithm
      +--ro esp-integrity-algorithm?         ipsec-crypto:ipsec-authentication-a
lgorithm
      +--ro esp-encrypt-algorithm?           ipsec-crypto:ipsec-encryption-algor
ithm
      +--ro life-time
      |  +--ro life-time-in-seconds?          uint32
      |  +--ro remain-life-time-in-seconds?   uint32
      |  +--ro life-time-in-byte?             uint32
      |  +--ro remain-life-time-in-byte?      uint32
      +--ro protocol-mode?                     ipsec-type:ipsec-mode
      +--ro tunnel-mode-process-info
      |  +--ro local-address?              string
      |  +--ro remote-address?             string
      |  +--ro bypass-df?                  enumeration
      |  +--ro dscp-flag?                  boolean
      |  +--ro stateful-frag-check-flag?   boolean
      +--ro dscp*                            uint8
      +--ro path-mtu?                        uint16
      +--ro nat-traversal-flag?              boolean
```

2.2.2. IPsec SPD State

The IPSEC SPD(Security Policy Database) container maintains policy information related to the IPSEC SAs established in a system. This is a run-time data structure that is created when the first IPSEC policy is created.

The SPD entries also contain information about the traffic selectors, protect action (permit, deny), protocol information etc as shown below. Based on these information the IPSEC module processes the outbound and inbound traffic.

```
+--ro spd
   +--ro spd-entries*
      +--ro name*
      |  +--ro name-type?     ipsec-type:ipsec-spd-name
      |  +--ro name-string?   string
      |  +--ro name-binary?   binary
      +--ro pfp-flag?            boolean
      +--ro traffic-selector*
      |  +--ro local-address-low?     inet:ip-address
      |  +--ro local-address-high?    inet:ip-address
      |  +--ro remote-address-low?    inet:ip-address
      |  +--ro remote-address-high?   inet:ip-address
      |  +--ro next-protocol-low?     uint16
      |  +--ro next-protocol-high?    uint16
      |  +--ro local-port-low?        inet:port-number
      |  +--ro local-port-high?       inet:port-number
      |  +--ro remote-port-high?      inet:port-number
      |  +--ro remote-port-low?       inet:port-number
      +--ro operation?           ipsec-type:ipsec-spd-operation
      +--ro protect-operation
         +--ro spd-ipsec-mode?           ipsec-type:ipsec-mode
         +--ro esn-flag?                 boolean
         +--ro spd-ipsec-protocol?       ipsec-type:ipsec-protocol
         +--ro tunnel-mode-additional
         |  +--ro local-address?              string
         |  +--ro remote-address?             string
         |  +--ro bypass-df?                  enumeration
         |  +--ro dscp-flag?                  boolean
         |  +--ro stateful-frag-check-flag?   boolean
         +--ro spd-algorithm*
            +--ro ah-auth-algorithm?         ipsec-crypto:ipsec-authentication-a
lgorithm
            +--ro esp-integrity-algorithm?   ipsec-crypto:ipsec-authentication-a
lgorithm
            +--ro esp-encrypt-algorithm?     ipsec-crypto:ipsec-encryption-algor
ithm
```

2.2.3. IPsec Global Statistics

The IPSEC Global Statistics container is used to maintain information related to all the IPSEC tunnels established in the system. These could be related to IPv4 IPSEC tunnels or IPv6 IPSEC tunnels.

The information maintained includes: traffic sent/received on an IPSEC tunnel like number of outbound/inbound packets, number of outbound/inbound bytes, number of packets dropped, number of replayed packets, number of packet authentication failures, number of packets dropped due to queue full, number of packets dropped due to deny policy, number of packet dropped due to being malformed, number of packets dropped due to being too large.

```
   +--ro ipsec-global-statistics
       +--ro ipv4
       |  +--ro inbound-packets?         uint64
       |  +--ro outbound-packets?        uint64
       |  +--ro inbound-bytes?           uint64
       |  +--ro outbound-bytes?          uint64
       |  +--ro inbound-drop-packets?    uint64
       |  +--ro outbound-drop-packets?   uint64
       |  +--ro dropped-packet-detail
       |     +--ro sa-non-exist?   uint64
       |     +--ro queue-full?     uint64
       |     +--ro auth-failure?   uint64
       |     +--ro malform?        uint64
       |     +--ro replay?         uint64
       |     +--ro large-packet?   uint64
       |     +--ro invalid-sa?     uint64
       |     +--ro policy-deny?    uint64
       |     +--ro other-reason?   uint64
       +--ro ipv6
       |  +--ro inbound-packets?         uint64
       |  +--ro outbound-packets?        uint64
       |  +--ro inbound-bytes?           uint64
       |  +--ro outbound-bytes?          uint64
       |  +--ro inbound-drop-packets?    uint64
       |  +--ro outbound-drop-packets?   uint64
       |  +--ro dropped-packet-detail
       |     +--ro sa-non-exist?   uint64
       |     +--ro queue-full?     uint64
       |     +--ro auth-failure?   uint64
       |     +--ro malform?        uint64
       |     +--ro replay?         uint64
       |     +--ro large-packet?   uint64
       |     +--ro invalid-sa?     uint64
       |     +--ro policy-deny?    uint64
       |     +--ro other-reason?   uint64
       +--ro global
          +--ro inbound-packets?         uint64
          +--ro outbound-packets?        uint64
          +--ro inbound-bytes?           uint64
          +--ro outbound-bytes?          uint64
          +--ro inbound-drop-packets?    uint64
          +--ro outbound-drop-packets?   uint64
          +--ro dropped-packet-detail
             +--ro sa-non-exist?   uint64
             +--ro queue-full?     uint64
             +--ro auth-failure?   uint64
             +--ro malform?        uint64
             +--ro replay?         uint64
```

```
            +--ro large-packet?   uint64
            +--ro invalid-sa?     uint64
            +--ro policy-deny?    uint64
            +--ro other-reason?   uint64
```

2.3. Actions

This model defines a list of RPCs that allow performing an action or executing a command on the protocol. In current version of this document, we only defined how to reset IPsec statistics, other actions are left for later version of this document.

2.3.1. IPsec statistics reset action

This operation type is executed when the user wants to reset IPSEC SA statistics. The operation will reset the global IPSEC4 statistics in the system.

```
rpcs:
   +---x reset-ipv4
   |  +---w input
   |  |  +---w ipv4?   empty
   |  +--ro output
   |     +--ro status?   string
   +---x reset-ipv6
   |  +---w input
   |  |  +---w ipv6?   empty
   |  +--ro output
   |     +--ro status?   string
   +---x reset-global
      +---w input
      |  +---w ipv6?   empty
      +--ro output
         +--ro status?   string
```

3. IPsec Yang Module

To support separately upgrade the algorithm part, the algorithm part is defined as separately part.

3.1. IPsec Yang Module

```
module ietf-ipsec {
  namespace "urn:ietf:params:xml:ns:yang:ietf-ipsec";
  prefix ipsec;

  import ietf-ipsec-crypto {
    prefix ipsec-crypto;
```

```
  }
  import ietf-inet-types {
    prefix inet;
  }
  import ietf-ipsec-type {
    prefix ipsec-type;
  }

  organization "Huawei Technologies India Pvt Ltd";
  contact
    "stonewater.wang@huawei.com";
  description
    "IPsec Yang";

  revision 2015-04-18 {
    description
      "Initial revision.";
    reference "RFC XXX: IPsec Yang Modules";
  }

  grouping ipsec-tunnel-mode-info {
    description
      "common infomations when using IPsec tunnel mode";
    leaf local-address {
      type string;
      description
        "Local address of IPsec tunnel mode";
    }
    leaf remote-address {
      type string;
      description
        "Remote address of IPsec tunnel mode";
    }
    leaf bypass-df {
      type enumeration {
        enum "set" {
          description
            "Set the df bit";
        }
        enum "clear" {
          description
            "Clear the df bit";
        }
        enum "copy" {
          description
            "Copy the df bit from inner header";
        }
      }
```

```
      description
        "This flag indicates how to process tunnel mode df flag";
    }
    leaf dscp-flag {
      type boolean;
      description
        "This flag indicate whether bypass DSCP or map to unprotected DSCP value
s (array) if needed to restrict bypass of DSCP values.";
    }
    leaf stateful-frag-check-flag {
      type boolean;
      description
        "This flag indicates whether stateful fragment checking will be used.";
    }
  }

  grouping traffic-selector {
    description
      "IPsec traffic selector information";
    leaf local-address-low {
      type inet:ip-address;
      description
        "Low range of local address";
    }
    leaf local-address-high {
      type inet:ip-address;
      description
        "High range of local address";
    }
    leaf remote-address-low {
      type inet:ip-address;
      description
        "Low range of remote address";
    }
    leaf remote-address-high {
      type inet:ip-address;
      description
        "High range of remote address";
    }
    leaf next-protocol-low {
      type uint16;
      description
        "Low range of next protocol";
    }
    leaf next-protocol-high {
      type uint16;
      description
        "High range of next protocol";
    }
```

```
    leaf local-port-low {
      type inet:port-number;
      description
        "Low range of local port";
    }
    leaf local-port-high {
      type inet:port-number;
      description
        "High range of local port";
    }
    leaf remote-port-high {
      type inet:port-number;
      description
        "Low range of remote port";
    }
    leaf remote-port-low {
      type inet:port-number;
      description
        "High range of remote port";
    }
  }

  grouping ipsec-algorithm-info {
    description
      "IPsec algorithm information used by SPD and SAD";
    leaf ah-auth-algorithm {
      type ipsec-crypto:ipsec-authentication-algorithm;
      description
        "Authentication algorithm used by AH";
    }
    leaf esp-integrity-algorithm {
      type ipsec-crypto:ipsec-authentication-algorithm;
      description
        "Integrity algorithm used by ESP";
    }
    leaf esp-encrypt-algorithm {
      type ipsec-crypto:ipsec-encryption-algorithm;
      description
        "Encryption algorithm used by ESP";
    }
  }

  grouping ipsec-stat {
    leaf inbound-packets {

        type uint64;
        config false;
        description "Inbound Packet count";
```

```
    }
    leaf outbound-packets {
        type uint64;
        config false;
        description "Outbound Packet count";
    }
    leaf inbound-bytes {
        type uint64;
        config false;
        description "Inbound Packet bytes";
    }
    leaf outbound-bytes {
        type uint64;
        config false;
        description "Outbound Packet bytes";
    }

    leaf inbound-drop-packets {
        type uint64;
        config false;
        description "Inbound dropped packets count";
    }
    leaf outbound-drop-packets {
        type uint64;
        config false;
        description "Outbound dropped packets count";
    }
    container dropped-packet-detail {
        description "The detail information of dropped packets";
        leaf sa-non-exist {
            type uint64;
            config false;
            description "The dropped packets counts caused by SA non-exist.";
        }
        leaf queue-full {
            type uint64;
            config false;
            description "The dropped packets counts caused by full processing qu
eue";
        }

        leaf auth-failure {
            type uint64;
            config false;
            description "The dropped packets counts caused by authentication fai
lure";
        }

        leaf malform {
            type uint64;
```

```
            config false;
            description "The dropped packets counts of malform";
        }
        leaf replay {
            type uint64;
            config false;
            description "The dropped packets counts of replay";
        }
        leaf large-packet {
            type uint64;
            config false;
            description "The dropped packets counts of too large";
        }
        leaf invalid-sa {
            type uint64;
            config false;
            description "The dropped packets counts of invalid SA";
        }
        leaf policy-deny {
            type uint64;
            config false;
            description "The dropped packets counts of denyed by policy";
        }
        leaf other-reason {
            type uint64;
            config false;
            description "The dropped packets counts of other reason";
        }
    }
    description "IPsec statistics information";

  }

  container sad {

    config false;

    description
      "The IPsec SA database";

    list sad-entries {
      key "spi security-protocol direction";
      description
        "The SA entries information";
      leaf spi {
        type ipsec-type:ipsec-spi;
        description
          "Security parameter index of SA entry.";
```

```
      }
      leaf security-protocol {
        type ipsec-type:ipsec-protocol;
        description
          "Security protocol of IPsec SA.";
      }
      leaf direction {
        type ipsec-type:ipsec-traffic-direction;
        description
          "It indicates whether the SA is inbound SA or out bound SA.";
      }
      leaf sa-type {
        type enumeration {
          enum "manual" {
            description
              "Manual IPsec SA";
          }
          enum "isakmp" {
            description
              "ISAKMP IPsec SA";
          }
        }
        description
          "It indicates whether the SA is created by manual or by dynamic protoc
ol.";
      }
      leaf sequence-number {
        type uint64;
        description
          "Current sequence number of IPsec packet.";
      }
      leaf sequence-number-overflow-flag {
        type boolean;
        description
          "The flag indicating whether overflow of the sequence number counter s
hould prevent transmission of additional packets on the SA, or whether rollover
is permitted.";
      }
      leaf anti-replay-enable-flag {
        type boolean;
        description
          "It indicates whether anti-replay is enable or disable.";
      }
      leaf anti-replay-window-size {
        type uint64;
        description
          "The size of anti-replay window.";
      }
      uses ipsec-algorithm-info;
      container life-time {
        leaf life-time-in-seconds {
```

```
          type uint32;
          description
            "SA life time in seconds";
        }
        leaf remain-life-time-in-seconds {
          type uint32;
          description
            "Remain SA life time in seconds";
        }
        leaf life-time-in-byte {
          type uint32;
          description
            "SA life time in bytes";
        }
        leaf remain-life-time-in-byte {
          type uint32;
          description
            "Remain SA life time in bytes";
        }
        description
          "SA life time information";
      }
      leaf protocol-mode {
        type ipsec-type:ipsec-mode;
        description
          "It indicates whether tunnel mode or transport mode will be used.";
      }
      container tunnel-mode-process-info {
        when "protocol-mode = 'tunnel'" {
          description
            "External information of SA when SA works in tunnel mode.";
        }
        uses ipsec-tunnel-mode-info;
        description
          "External information of SA when SA works in tunnel mode.";
      }
      leaf-list dscp {
        type uint8 {
          range "0..63";
        }
        description
          "When traffic matchs SPD, the DSCP values used to filter traffic";
      }
      leaf path-mtu {
        type uint16;
        description
          "Path MTU valie";
      }
```

```
      leaf nat-traversal-flag {
        type boolean;
        description
          "Whethe the SA is used to protect traffic that nedds nat traversal";
      }
    }
  }
  container spd {
    config false;
    description
      "IPsec security policy database information";

    list spd-entries {
      description
        "IPsec SPD entry information";
      list name {
        description
          "SPD name information.";
        leaf name-type {
          type ipsec-type:ipsec-spd-name;
          description
            "SPD name type.";
        }
        leaf name-string {
          when "name-type = 'id_rfc_822_addr' or name-type = 'id_fqdn'" {
            description
              "when name type is id_rfc_822_addr or id_fqdn, the name are saved
in string";
          }
          type string;
          description
            "SPD name content";
        }
        leaf name-binary {
          when "name-type = 'id_der_asn1_dn' or name-type = 'id_key'" {
            description
              "when name type is id_der_asn1_dn or id_key, the name are saved in
 binary";
          }
          type binary;
          description
            "SPD name content";
        }
      }
      leaf pfp-flag {
        type boolean;
        description
          "populate from packet flag";
      }
      list traffic-selector {
```

```
        min-elements 1;
        uses traffic-selector;
        description
          "Traffic selectors of SAD entry";
      }
      leaf operation {
        type ipsec-type:ipsec-spd-operation;
        description
          "It indicates how to process the traffic when it matches the security
policy.";
      }
      container protect-operation {
        when "operation = 'protect'" {
          description
            "How to protect the traffic when the SPD operation is protect";
        }
        leaf spd-ipsec-mode {
          type ipsec-type:ipsec-mode;
          description
            "It indicates which mode is chosen when the traffic need be protecte
d by IPsec.";
        }
        leaf esn-flag {
          type boolean;
          description
            "It indicates whether ESN is used.";
        }
        leaf spd-ipsec-protocol {
          type ipsec-type:ipsec-protocol;
          description
            "It indicates which protocol (AH or ESP) is chosen.";
        }
        container tunnel-mode-additional {
          when "spd-ipsec-mode = 'tunnel'" {
            description
              "Additional informations when choose tunnel mode";
          }
          uses ipsec-tunnel-mode-info;
          description
            "When use tunnel mode, the additional information of SPD.";
        }
        list spd-algorithm {
          min-elements 1;
          uses ipsec-algorithm-info;
          description
            "Algorithms defined in SPD, ordered by decreasing priority.";
        }
        description
          "How to protect the traffic when the SPD operation is protect";
      }
```

```
    }
  }

  container ipsec-global-statistics {
    config false;
    description "IPsec global statistics";

    container ipv4 {
        description "IPsec statistics of IPv4";
        uses ipsec-stat;
    }

    container ipv6 {
        description "IPsec statistics of IPv6";
        uses ipsec-stat;
    }

    container global {
        description "IPsec statistics of global";
        uses ipsec-stat;
    }
  }


    rpc reset-ipv4 {
        description "Reset IPsec IPv4 statistics";
        input {
            leaf ipv4 {
                type empty;
                description "Reset IPsec IPv4 statistics";
            }
        }
        output {
            leaf status {
                type string;
                description "Operation status";
            }
        }

    }
    rpc reset-ipv6 {
        description "Reset IPsec IPv6 statistics";
        input {
            leaf ipv6 {
                type empty;
                description "Reset IPsec IPv6 statistics";
            }
        }
```

```
        output {
            leaf status {
                type string;
                description "Operation status";
            }
        }

    }
    rpc reset-global {
        description "Reset IPsec global statistics";
        input {
            leaf ipv6 {
                type empty;
                description "Reset IPsec global statistics";
            }
        }
        output {
            leaf status {
                type string;
                description "Operation status";
            }
        }

    }

}
```

3.2. IPsec Algorithm Yang Module

```
   module ietf-ipsec-crypto {
     namespace "urn:ietf:params:xml:ns:yang:ietf-ipsec-crypto";
     prefix ipsec-crypto;

     organization "Huawei Technologies India Pvt Ltd";
     contact
       "stonewater.wang@huawei.com";
     description
       "IPsec Crypto Yang";
     reference
       "RFC 4301: Security Architecture for the Internet Protocol";

     revision 2015-04-18 {
       description
         "Initial revision.";
       reference
         "RFC 4301: Security Architecture for the Internet Protocol";
     }
```

```
    typedef ipsec-authentication-algorithm {
      type enumeration {
        enum "null" {
          value 0;
          description
            "null";
        }
        enum "md5" {
          value 1;
          description
            "MD5 authentication algorithm";
        }
        enum "sha1" {
          value 2;
          description
            "SHA1 authentication algorithm";
        }
        enum "sha2-256" {
          value 3;
          description
            "SHA2-256 authentication algorithm";
        }
        enum "sha2-384" {
          value 4;
          description
            "SHA2-384 authentication algorithm";
        }
        enum "sha2-512" {
          value 5;
          description
            "SHA2-512 authentication algorithm";
        }
      }
      description
        "typedef for ipsec authentication algorithm";
    }

    typedef ipsec-encryption-algorithm {
      type enumeration {
        enum "null" {
          description
            "null";
        }
        enum "des" {
          description
            "DES encryption algorithm";
        }
        enum "3des" {
```

```
            description
              "3DES encryption algorithm";
          }
          enum "aes-128" {
            description
              "AES-128 encryption algorithm";
          }
          enum "aes-192" {
            description
              "AES-192 encryption algorithm";
          }
          enum "aes-256" {
            description
              "AES-256 encryption algorithm";
          }
        }
        description
          "typedef for ipsec encryption algorithm";
      }
    }
```

3.3. IPsec Type Yang Module

```
module ietf-ipsec-type {
  namespace "urn:ietf:params:xml:ns:yang:ietf-ipsec-type";
  prefix ipsec-type;

  organization "Huawei Technologies India Pvt Ltd";
  contact
    "stonewater.wang@huawei.com";
  description
    "common type define for ipsec protocol Yang";
  reference "RFC 4301: Security Architecture for the Internet Protocol";

  revision 2015-04-18 {
    description
      "Initial revision.";
    reference "RFC 4301: Security Architecture for the Internet Protocol";
  }

  typedef ipsec-mode {
    type enumeration {
      enum "transport" {
        description
          "Transport mode";
      }
      enum "tunnel" {
        description
```

```
          "Tunnel mode";
      }
    }
    description
      "type define of ipsec mode";
  }

  typedef ipsec-protocol {
    type enumeration {
      enum "ah" {
        description
          "AH Protocol";
      }
      enum "esp" {
        description
          "ESP Protocol";
      }
    }
    description
      "type define of ipsec security protocol";
  }

  typedef ipsec-spi {
    type uint32 {
      range "1..max";
    }
    description
      "SPI";
  }

  typedef ipsec-spd-name {
   type enumeration {
    enum id_rfc_822_addr {
        description
          "Fully qualified user name string.";
      }
      enum id_fqdn {
        description
          "Fully qualified DNS name.";
      }
      enum id_der_asn1_dn {
        description
          "X.500 distinguished name.";
      }
      enum id_key {
        description
          "IKEv2 Key ID.";
      }
```

```
    }
    description
      "IPsec SPD name type";
  }

  typedef ipsec-traffic-direction {
    type enumeration {
      enum inbound {
        description
          "Inbound traffic";
      }
      enum outbound {
        description
          "Outbound traffic";
      }
    }
    description
      "IPsec traffic direction";
  }

  typedef ipsec-spd-operation {
    type enumeration {
      enum protect {
        description
          "PROTECT the traffic with IPsec";
      }
      enum bypass {
        description
          "BYPASS the traffic";
      }
      enum discard {
        description
          "DISCARD the traffic";
      }
    }
    description
      "The operation when traffic matches IPsec security policy";
  }

}
```

4. IANA Considerations

This document registers the following URIs in the IETF XML registry [RFC3688]. Following the format in [RFC3688], the following registration is requested to be made.

URI: urn:ietf:params:xml:ns:yang:ietf-ipsec XML: N/A, the requested URI is an XML namespace.

URI: urn:ietf:params:xml:ns:yang:ietf-ipsec-crypto XML: N/A, the requested URI is an XML namespace.

URI: urn:ietf:params:xml:ns:yang:ietf-ipsec-type XML: N/A, the requested URI is an XML namespace.

This document registers a YANG module in the YANG Module Names registry [RFC6020].

name: ietf-ipsec namespace: urn:ietf:params:xml:ns:yang:ietf-ipsec prefix: ipsec reference: [RFC4301]

5. Security Considerations

The YANG module defined in this memo is designed to be accessed via the NETCONF protocol[RFC6241]. The lowest NETCONF layer is the secure transport layer and the mandatory-to-implement secure transport is SSH [RFC6242]. The NETCONF access control model [RFC6536] provides means to restrict access for particular NETCONF users to a pre-configured subset of all available NETCONF protocol operations and content. There are a number of data nodes defined in the YANG module which 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.

6. Acknowledgements

7. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, January 2004.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

[RFC6020] Bjorklund, M., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, October 2010.

[RFC6241] Enns, R., Bjorklund, M., Schoenwaelder, J., and A. Bierman, "Network Configuration Protocol (NETCONF)", RFC 6241, June 2011.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, June 2011.

[RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, March 2012.

[RFC7296] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", STD 79, RFC 7296, October 2014.

Authors' Addresses

Honglei Wang
Huawei Technologies
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: stonewater.wang@huawei.com

Vijay Kumar Nagaraj
Huawei Technologies
Huawei Technologies India Pvt Ltd
Bangalore 560008
India

Email: vijay.kn@huawei.com

Xia Chen
Huawei Technologies
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: xiachen@huawei.com