IKEv2 の Scyhter による評価結果

国立研究開発法人 情報通信研究機構

1. 基本情報

- ✧ 名前

  Internet Key Exchange Protocol Version 2 (IKEv2)

- ✧ 機能

  IPsec の前に実行する相互認証・鍵交換プロトコル。 IKEv1 との互換性は確保されていない。 暗号化方式として 3DES、ハッシュ関数として SHA-1 が規定されている。

- ✧ 関連する標準

  RFC5996 (https://tools.ietf.org/html/rfc5996)

2. Scyther の文法による記述

2.1. プロトコル仕様

```
// data type
usertype String;
usertype SecurityParameterIndex;
usertype SecurityAssociation; //security association (supported cryptographic algorithms)
usertype TrafficSelector;
// constants (or easily expectable variables)
const SPIi, SPIr: SecurityParameterIndex;
const SAi1, SAi2, SAr1, SAr2: SecurityAssociation;
const TSi, TSr: TrafficSelector;
// (cryptographic) functions
hashfunction mac, prf, prfplus;
const DHg1, DHg2: Function; //finite field exponentital
const EncKey, MacKey, PrfKey: String;
//for pre-shared key based version
```

```
macro SharedSecret = k(I,R);
const KeyPad4IKEv2: String;
macro Kaut = prf(SharedSecret, KeyPad4IKEv2);
// packet specification
// message 1:------------------------------
macro KEi = DHg1(Ki);
macro ikev2-1-header = (SPIi);//corresponding to HDR. version numbers and
flags are omitted.
//send: I->R
macro ikev2-1-payload-I = (SAi1, KEi, Ni);
macro ikev2-1-I = (ikev2-1-header, ikev2-1-payload-I);
//recv: I->R
macro ikev2-1-payload-R = (SAi1, Gi, Ni);
macro ikev2-1-R = (ikev2-1-header, ikev2-1-payload-R);
//For executable
//send: I->R
macro ikev2-1-Ix = ikev2-1-I;
//recv: I->R
macro ikev2-1-payload-Rx = (SAi1, DHg1(Ki), Ni);
macro ikev2-1-Rx = (ikev2-1-header, ikev2-1-payload-Rx);
// message 2:------------------------------
macro KEr = DHg1(Kr);
macro ikev2-2-header = (SPIi, SPIr); //no information on HDR found.
//recv: R->I
macro ikev2-2-payload-I = (SAr1, Gr, Nr);
macro ikev2-2-I = (ikev2-2-header, ikev2-2-payload-I);
//send: R->I
macro ikev2-2-payload-R = (SAr1, KEr, Nr);
macro ikev2-2-R = (ikev2-2-header, ikev2-2-payload-R);
//For executable
//recv: R->I
macro ikev2-2-payload-Ix = (SAr1, DHg1(Kr), Nr);
```

```
macro ikev2-2-Ix = (ikev2-2-header, ikev2-2-payload-Ix);
//send: R->I
macro ikev2-2-Rx = ikev2-2-R;
// key derivation from here -------------
macro SKEYSEEDi = prf(Ni, Nr, DHg2(Gr, Ki));
macro SKai-I = prfplus(SKEYSEEDi, MacKey, I, R, SPIi, SPIr);
macro SKar-I = prfplus(SKEYSEEDi, MacKey, R, I, SPIi, SPIr);
macro SKei-I = prfplus(SKEYSEEDi, EncKey, I, R, SPIi, SPIr);
macro SKer-I = prfplus(SKEYSEEDi, EncKey, R, I, SPIi, SPIr);
macro SKpi-I = prfplus(SKEYSEEDi, PrfKey, I, R, SPIi, SPIr);
macro SKpr-I = prfplus(SKEYSEEDi, PrfKey, R, I, SPIi, SPIr);
macro SKEYSEEDr = prf(Ni, Nr, DHg2(Gi, Kr));
macro SKai-R = prfplus(SKEYSEEDr, MacKey, I, R, SPIi, SPIr);
macro SKar-R = prfplus(SKEYSEEDr, MacKey, R, I, SPIi, SPIr);
macro SKei-R = prfplus(SKEYSEEDr, EncKey, I, R, SPIi, SPIr);
macro SKer-R = prfplus(SKEYSEEDr, EncKey, R, I, SPIi, SPIr);
macro SKpi-R = prfplus(SKEYSEEDr, PrfKey, I, R, SPIi, SPIr);
macro SKpr-R = prfplus(SKEYSEEDr, PrfKey, R, I, SPIi, SPIr);
//For executable
macro SKEYSEEDix = prf(Ni, Nr, DHg2(DHg1(Kr), Ki));
macro SKai-Ix = prfplus(SKEYSEEDix, MacKey, I, R, SPIi, SPIr);
macro SKar-Ix = prfplus(SKEYSEEDix, MacKey, R, I, SPIi, SPIr);
macro SKei-Ix = prfplus(SKEYSEEDix, EncKey, I, R, SPIi, SPIr);
macro SKer-Ix = prfplus(SKEYSEEDix, EncKey, R, I, SPIi, SPIr);
macro SKpi-Ix = prfplus(SKEYSEEDix, PrfKey, I, R, SPIi, SPIr);
macro SKpr-Ix = prfplus(SKEYSEEDix, PrfKey, R, I, SPIi, SPIr);
macro SKEYSEEDrx = prf(Ni, Nr, DHg2(DHg1(Ki), Kr));
macro SKai-Rx = prfplus(SKEYSEEDrx, MacKey, I, R, SPIi, SPIr);
macro SKar-Rx = prfplus(SKEYSEEDrx, MacKey, R, I, SPIi, SPIr);
macro SKei-Rx = prfplus(SKEYSEEDrx, EncKey, I, R, SPIi, SPIr);
macro SKer-Rx = prfplus(SKEYSEEDrx, EncKey, R, I, SPIi, SPIr);
macro SKpi-Rx = prfplus(SKEYSEEDrx, PrfKey, I, R, SPIi, SPIr);
```

```
macro SKpr-Rx = prfplus(SKEYSEEDrx, PrfKey, R, I, SPIi, SPIr);
// message 3:-----------------------------
macro IDi = I;
macro ikev2-3-header = (SPIi, SPIr); //no information on HDR found.
//send I->R
macro ikev2-3-AUTH-I = prf(Kaut, ikev2-1-I, Nr, prf(SKpi-I, IDi));
macro ikev2-3-payload-I = (IDi, ikev2-3-AUTH-I, SAi2, TSi, TSr);
macro ikev2-3-ae-I = ({ikev2-3-payload-I}SKei-I, mac(ikev2-3-payload-I,
SKai-I));
macro ikev2-3-I = (ikev2-3-header, ikev2-3-ae-I);//send
//recv I->R
macro ikev2-3-AUTH-R = prf(Kaut, ikev2-1-R, Nr, prf(SKpi-R, IDi));
macro ikev2-3-payload-R = (IDi, ikev2-3-AUTH-R, SAi2, TSi, TSr);
macro ikev2-3-ae-R = ({ikev2-3-payload-R}SKei-R, mac(ikev2-3-payload-R,
SKai-R));//recv
macro ikev2-3-R = (ikev2-3-header, ikev2-3-ae-R);//recv
//For executable
//send I->R
macro ikev2-3-AUTH-Ix = prf(Kaut, ikev2-1-Ix, Nr, prf(SKpi-Ix, IDi));
macro ikev2-3-payload-Ix = (IDi, ikev2-3-AUTH-Ix, SAi2, TSi, TSr);
macro       ikev2-3-ae-Ix       =       ({ikev2-3-payload-Ix}SKei-Ix,
mac(ikev2-3-payload-Ix, SKai-Ix));
macro ikev2-3-Ix = (ikev2-3-header, ikev2-3-ae-Ix);//send
//recv I->R
macro ikev2-3-AUTH-Rx = prf(Kaut, ikev2-1-Rx, Nr, prf(SKpi-Rx, IDi));
macro ikev2-3-payload-Rx = (IDi, ikev2-3-AUTH-Rx, SAi2, TSi, TSr);
macro       ikev2-3-ae-Rx       =       ({ikev2-3-payload-Rx}SKei-Rx,
mac(ikev2-3-payload-Rx, SKai-Rx));//recv
macro ikev2-3-Rx = (ikev2-3-header, ikev2-3-ae-Rx);//recv
// message 4:-----------------------------
macro IDr = R;
macro ikev2-4-header = (SPIi, SPIr); //no information on HDR found.
```

```
//recv R->I
macro ikev2-4-AUTH-I = prf(Kaut, ikev2-2-I, Ni, prf(SKpr-I, IDr));
macro ikev2-4-payload-I = (IDr, ikev2-4-AUTH-I, SAr2, TSi, TSr);
macro ikev2-4-ae-I = ({ikev2-4-payload-I}SKer-I, mac(ikev2-4-payload-I,
SKar-I));
macro ikev2-4-I = (ikev2-4-header, ikev2-4-ae-I);
//send R->I
macro ikev2-4-AUTH-R = prf(Kaut, ikev2-2-R, Ni, prf(SKpr-R, IDr));
macro ikev2-4-payload-R = (IDr, ikev2-4-AUTH-R, SAr2, TSi, TSr);
macro ikev2-4-ae-R = ({ikev2-4-payload-R}SKer-R, mac(ikev2-4-payload-R,
SKar-R));
macro ikev2-4-R = (ikev2-4-header, ikev2-4-ae-R);
//For executable
//recv R->I
macro ikev2-4-AUTH-Ix = prf(Kaut, ikev2-2-Ix, Ni, prf(SKpr-Ix, IDr));
macro ikev2-4-payload-Ix = (IDr, ikev2-4-AUTH-Ix, SAr2, TSi, TSr);
macro       ikev2-4-ae-Ix       =       ({ikev2-4-payload-Ix}SKer-Ix,
mac(ikev2-4-payload-Ix, SKar-Ix));
macro ikev2-4-Ix = (ikev2-4-header, ikev2-4-ae-Ix);
//send R->I
macro ikev2-4-AUTH-Rx = prf(Kaut, ikev2-2-Rx, Ni, prf(SKpr-Rx, IDr));
macro ikev2-4-payload-Rx = (IDr, ikev2-4-AUTH-Rx, SAr2, TSi, TSr);
macro       ikev2-4-ae-Rx       =       ({ikev2-4-payload-Rx}SKer-Rx,
mac(ikev2-4-payload-Rx, SKar-Rx));
macro ikev2-4-Rx = (ikev2-4-header, ikev2-4-ae-Rx);
// helper protocols
protocol @oracles (DH-naked)
{
  // (g^x)^y -> (g^y)^x
  role DH-naked
    {
      var x, y: Ticket;
```

```
      recv_!dh1(DH-naked,DH-naked, DHg2(DHg1(x),y));
      send_!dh2(DH-naked,DH-naked, DHg2(DHg1(y),x));
    }
}
protocol @executable (MSG3, MSG4)
{
  // message 3
  role MSG3
    {
      var Ki, Kr, Ni, Nr: Nonce;
      var I, R: Agent;

      recv_!msg31(MSG3,MSG3, ikev2-3-Ix);
      send_!msg32(MSG3,MSG3, ikev2-3-Rx);
    }
  // message 4
  role MSG4
    {
      var Ki, Kr, Ni, Nr: Nonce;
      var I, R: Agent;

      recv_!msg41(MSG4,MSG4, ikev2-4-Rx);
      send_!msg42(MSG4,MSG4, ikev2-4-Ix);
    }
}
protocol IKEv2-mac (I, R)
{
  // Peer (Initiator)
  role I
    {
      // variables
```

```
      fresh Ki, Ni: Nonce;
      var Nr: Nonce;
      var Gr: Ticket; //g^Kr
      // sequence
      send_1(I,R, ikev2-1-I);
      recv_2(R,I, ikev2-2-I);
      claim(I, Running, R, Ni, Nr);
      claim(I, Running, R, KEi, Gr);
      send_!3(I,R, ikev2-3-I);
      recv_!4(R,I, ikev2-4-I);

    }

  // Server (Responder)
  role R
    {
      // variables
      fresh Kr, Nr: Nonce;
      var Ni: Nonce;
      var Gi: Ticket; //g^Ki
      // sequence
      recv_1(I,R, ikev2-1-R);
      send_2(R,I, ikev2-2-R);

      recv_!3(I,R, ikev2-3-R);
      claim(R, Running, I, Ni, Nr);
      claim(R, Running, I, Gi, KEr);
      send_!4(R,I, ikev2-4-R);
    }
}
```

2.2. 攻撃者モデル

Scyther はデフォルトで Dolev-Yao モデルを想定しており、特に記載すべき項目はない。

2.3. セキュリティ要件

```
// ロール I のセキュリティ要件
claim(I, SKR, SKai-I);
claim(I, SKR, SKei-I);
claim(I, SKR, SKpi-I);
claim(I, SKR, Kaut);
claim(I, Alive);
claim(I, Weakagree);
claim(I, Commit, R, Ni, Nr);
claim(I, Commit, R, KEi, Gr);
// ロール R のセキュリティ要件
claim(R, SKR, SKai-R);
claim(R, SKR, SKei-R);
claim(R, SKR, SKpi-R);
claim(R, SKR, Kaut);
claim(R, Alive);
claim(R, Weakagree);
claim(R, Commit, I, Ni, Nr);
claim(R, Commit, I, Gi, KEr);
```

3. Scyther による評価結果

3.1. 出力

Scyther で評価可能なセキュリティプロパティについての、bounded な評価結果では、攻撃は発見されなかった。

```
claim                    IKEv2-mac,I                                SKR_I3
prfplus(prf(Ni,Nr,DHg2(Gr,Ki)),MacKey,I,R,SPIi,SPIr)      Ok          [no
attack within bounds]
claim                    IKEv2-mac,I                                SKR_I4
prfplus(prf(Ni,Nr,DHg2(Gr,Ki)),EncKey,I,R,SPIi,SPIr)      Ok          [no
```

```
attack within bounds]
claim                    IKEv2-mac,I                                    SKR_I5
prfplus(prf(Ni,Nr,DHg2(Gr,Ki)),PrfKey,I,R,SPIi,SPIr)      Ok          [no
attack within bounds]
claim    IKEv2-mac,I       SKR_I6   prf(k(I,R),KeyPad4IKEv2)           Ok
[no attack within bounds]
claim    IKEv2-mac,I      Alive_I7          -         Ok        [no attack
within bounds]
claim    IKEv2-mac,I      Weakagree_I8      -         Ok        [no attack
within bounds]
claim    IKEv2-mac,I      Commit_I9         (R,Ni,Nr)         Ok        [no
attack within bounds]
claim    IKEv2-mac,I      Commit_I10        (R,DHg1(Ki),Gr)  Ok        [no
attack within bounds]
claim                    IKEv2-mac,R                                    SKR_R3
prfplus(prf(Ni,Nr,DHg2(Gi,Kr)),MacKey,I,R,SPIi,SPIr)      Ok          [no
attack within bounds]
claim                    IKEv2-mac,R                                    SKR_R4
prfplus(prf(Ni,Nr,DHg2(Gi,Kr)),EncKey,I,R,SPIi,SPIr)      Ok          [no
attack within bounds]
claim                    IKEv2-mac,R                                    SKR_R5
prfplus(prf(Ni,Nr,DHg2(Gi,Kr)),PrfKey,I,R,SPIi,SPIr)      Ok          [no
attack within bounds]
claim    IKEv2-mac,R       SKR_R6   prf(k(I,R),KeyPad4IKEv2)           Ok
[no attack within bounds]
claim    IKEv2-mac,R      Alive_R7          -         Ok        [no attack
within bounds]
claim    IKEv2-mac,R      Weakagree_R8      -         Ok        [no attack
within bounds]
claim    IKEv2-mac,R      Commit_R9         (I,Ni,Nr)         Ok        [no
attack within bounds]
claim    IKEv2-mac,R      Commit_R10        (I,Gi,DHg1(Kr))  Ok        [no
```

attack within bounds]

3.2. 攻撃の解説

前述のとおり、攻撃は発見されなかった。

4. 形式化

4.1. 方針

IETF の通信プロトコルでは、ペイロードの階層ごとに、ペイロードタイプ、ペイロード長などが記載されている。しかし、これらの情報は冗長であり、また、Scyther では長さなどの情報をチェックすることはできない。そこで、メッセージのペイロードを特定するための、最低限のメッセージタイプ情報は残し、それ以外の認証に関係ないと思われる情報はモデルから削除した。

Security parameter index, security association など、予測可能である値は定数とみなした。

IKE では Diffie-Hellman (DH)鍵交換を用いて鍵共有を行い、さらに共有したセッション鍵と事前共有情報（事前共有鍵、通信相手の公開鍵など）を用いて相互認証を行う。DH 鍵交換は、指数演算処理が可換であることを利用しているが、Scyther では処理の可換性を記述することができない。このため、通信メッセージを途中で意図的に書き換え、送信パケットと受信パケットが異なるように記述することで、送信者、受信者にとって意味のあるデータとなるモデル化を行っている。

4.2. 妥当性

抽象化は行っているが、情報が大きく損なわれたり、特殊なデータ変換を用いたりはしていないので、妥当な形式化であると考える。

4.3. 検証ツールとの相性

プロトコル仕様、攻撃者モデルを記述するにあたって、特に制限はなかった。(DH 鍵交換の形式化については、「方針」「妥当性」の項を参照。)

セキュリティ要件を記述するにあたって、Scyther では、鍵長など数値化された情報を記述することができない。同様に辞書攻撃について評価することはできない。これは暗号プロトコルではなく、暗号プリミティブの安全性として評価されるべき項目である。データの完全性、秘匿性は本評価の対象である認証プロトコルの範囲外であるため、評価を行っていない。

相互認証については、暗号プロトコルが満たすべき性質が記載されていない。しかし、リプレイ攻撃に対する耐性を謳っていること、鍵共有を目的としていることから、

injective agreement が達成されるべきセキュリティプロパティであるとみなした。

ただし、Scyther では injective な性質について評価ができない。セッション鍵は RAND と事前共有鍵 k(A,P)から生成されるため、RAND の共有が成功していれば、セッション鍵は秘密裡に共有できたと考える。そこで、特定データについて non-injective agreement が成立していることを確認する記述を行っている。

Scyther で記述できないプロセスはないが、上記のように DH 鍵交換のモデル化は直感的とはいえない。また、送受信データの対応がないようなモデル化を行っているため、そのままでは Scyther で評価することができない。このため、上記のモデルでは、データの対応付けを行う補助的な暗号プロトコル(helper protocol)を定義し、攻撃者による暗号プロトコルの実行をサポートすることで、評価を可能としている。これらの helper protocol のメッセージに関する記述が大量に発生するため、モデル記述が肥大化し、バグが混入しやすくなっている。

4.4. 検証ツール適用時の性能

検証時間は 10 分だった。実行環境は以下のとおり。

- CPU：Intel Xeon E5502 1.87GHz x 4CPU(SMP)
- メモリ：48GB

5. 備考

本文書は、**総務省「暗号・認証技術等を用いた安全な通信環境推進事業に関する実証実験の請負　成果報告書」**からの引用である。