Needham-Schroeder symmetric -key 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

Needham-Schroeder symmetric-key

✧ 機能

共通鍵暗号を用いた相互認証プロトコル。

✧ 関連する文書

R.Needham and M.Schroeder, "Using encryption for authentication in large networks of computers," Communications of the ACM, 21(12), December 1978.

2. Scyther の文法による記述

2.1. プロトコル仕様

```
usertype SessionKey;
const dec: Function;
const inc: Function;
inversekeys(inc, dec);
protocol Needham-Schroeder-SymmetricKey(A, B, S)
{
        role A
        {
                fresh Na: Nonce;
                var Nb: Nonce;
                var Kab: SessionKey;
                send_1(A, S, (A, B, Na));
                recv_2(S, A, {Na, B, Kab, {Kab, A}k(B, S)}k(A, S));
                send_3(A, B, {Kab, A}k(B, S));
```

```
                recv_4(B, A, {Nb}Kab);
                send_5(A, B, {dec(Nb)}Kab);
        }
        role B
        {
                fresh Nb: Nonce;
                var Kab: SessionKey;
                recv_3(A, B, {Kab, A}k(B, S));
                send_4(B, A, {Nb}Kab);
                recv_5(A, B, {dec(Nb)}Kab);
        }
        role S
        {
                fresh Kab: SessionKey;
                var Na: Nonce;
                recv_1(A, S, (A, B, Na));
                send_2(S, A, {Na, B, Kab, {Kab, A}k(B, S)}k(A, S));
        }
}
```

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 A のセキュリティ要件
claim_a1(A, Secret, Kab);
claim_a2(A, Alive);
claim_a3(A, Weakagree);
claim_a4(A, Niagree);
claim_a5(A, Nisynch);
// ロール B のセキュリティ要件
claim_b1(B, Secret, Kab);
claim_b2(B, Alive);
claim_b3(B, Weakagree);
```

```
claim_b4(B, Niagree);
claim_b5(B, Nisynch);
// ロールSのセキュリティ要件
claim_s1(S, Secret, Kab);
```

3. Scyther による評価結果

3.1. 出力

暗号プロトコルの実行セッション数に制限をおいた評価（すなわち bounded）では、攻撃は発見されなかった。

```
claim id [Needham-Schroeder-SymmetricKey,a1], Secret(Kab)       :
No attacks.
claim id [Needham-Schroeder-SymmetricKey,a2], Alive    : No attacks.
claim id [Needham-Schroeder-SymmetricKey,a3], Weakagree         :
No attacks.
claim id [Needham-Schroeder-SymmetricKey,a4], Niagree  : No attacks.
claim id [Needham-Schroeder-SymmetricKey,a5], Nisynch  : No attacks.
claim id [Needham-Schroeder-SymmetricKey,b1], Secret(Nb)        :
No attacks within bounds.
claim id [Needham-Schroeder-SymmetricKey,b2], Alive    : No attacks.
claim id [Needham-Schroeder-SymmetricKey,b3], Weakagree         :
No attacks.
claim id [Needham-Schroeder-SymmetricKey,b4], Niagree  : No attacks
within bounds.
claim id [Needham-Schroeder-SymmetricKey,b5], Nisynch  : No attacks
within bounds.
claim id [Needham-Schroeder-SymmetricKey,s1], Secret(Kab)       :
No attacks.
```

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

認証にはLoweによる定義を含めて、いくつかのセキュリティ要件が定義されている。一方、元の文献では目標とするセキュリティ要件が定義されていない。そこで、本評価では、Scytherで選択可能な認証に関する性質すべてについて評価を行った。

4.2. 妥当性

Needham-Schroederプロトコルは既に抽象化されている暗号プロトコルであり、単純化は不要であった。

4.3. 検証ツールとの相性

プロトコル仕様、攻撃者モデル、セキュリティ要件を記述するにあたって、特に制限はなかった。

4.4. 検証ツール適用時の性能

検証時間は0.8秒だった。実行環境は以下のとおり。

- CPU：AMD Phenom X4 9750B (2.4GHz)
- メモリ：1.7GB

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。

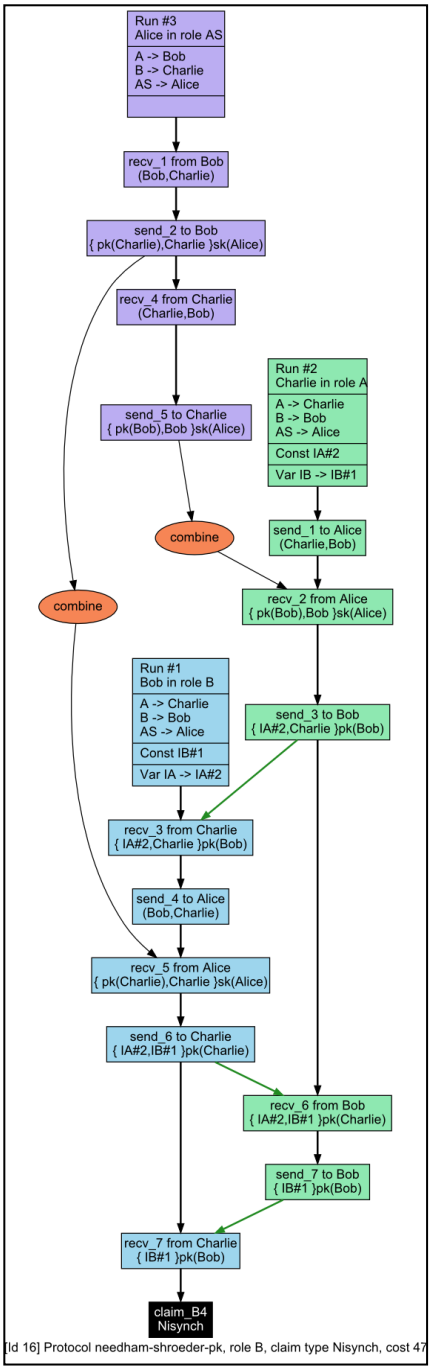
Run #3
Alice in role AS
A -> Bob
B -> Charlie
AS -> Alice
recv_1 from Bob
(Bob,Charlie)
send_2 to Bob
{ pk(Charlie),Charlie }sk(Alice)
recv_4 from Charlie
(Charlie,Bob)
send_5 to Charlie
{ pk(Bob),Bob }sk(Alice)
Run #2
Charlie in role A
A -> Charlie
B -> Bob
AS -> Alice
Const IA#2
Var IB -> IB#1
combine
send_1 to Alice
(Charlie,Bob)
combine
recv_2 from Alice
{ pk(Bob),Bob }sk(Alice)
Run #1
Bob in role B
A -> Charlie
B -> Bob
AS -> Alice
Const IB#1
Var IA -> IA#2
send_3 to Bob
{ IA#2,Charlie }pk(Bob)
recv_3 from Charlie
{ IA#2,Charlie }pk(Bob)
send_4 to Alice
(Bob,Charlie)
recv_5 from Alice
{ pk(Charlie),Charlie }sk(Alice)
send_6 to Charlie
{ IA#2,IB#1 }pk(Charlie)
recv_6 from Bob
{ IA#2,IB#1 }pk(Charlie)
send_7 to Bob
{ IB#1 }pk(Bob)
recv_7 from Charlie
{ IB#1 }pk(Bob)
claim_B4
Nisynch
[Id 16] Protocol needham-shroeder-pk, role B, claim type Nisynch, cost 47

図. 1.　攻撃に関する解説