TMN

Author(s): M. Tatebayashi, N. Matsuzaki, and D.B. Newman 1989
Last modified January 16, 2003

Summary: Distribution of a fresh symmetric key and authentication. Symmetric keys, trusted server and public keys (only the public key of the server is used).

Protocol specification (in common syntax)

```
A, B, S :  principal
Ka, Kb :   key
PK, SK :   principal -> key (keypair)

1.   A  ->  S  :   B, {Ka}PK(S)
2.   S  ->  B  :   A
3.   B  ->  S  :   A, {Kb}PK(S)
4.   S  ->  A  :   B, {Kb}Ka
```

Description of the protocol rules

We assume that both `A` and `B` initially know the public key `PK(S)` of `S`.

`Ka`, `Kb` are session symmetric keys freshly created by `A`, resp. `B`.

In message `4`, `Kb` is encrypted using a symmetric key algorithm with the key `Ka`. Hence, the encryption operators used in `4` on one hand and in `1` and `3` on the other hand differ (though the notation is the same).

Remark

The binary operator {`Kb`}`Ka` in the last message can be intepreted either by a xor operator or by another symmetric key encryption algorithm, according to the implementation of the protocol.

This choice may be important, as the attack 4. below shows.

Requirements

The protocol must guaranty the secrecy of the new shared key `Kb`: in every session, the value of `Kb` must be known only by the participants playing the roles of `A` and `B` in that session.

The protocol must guaranty the secrecy of the auxiliary fresh key `Ka`: in every session, the value of `Ka` must be known only by the participants playing the roles of `A` and `S` in that session.

References

[TMN89], see also [LR97].

Claimed attacks

1. [LR97]. Authentication and secrecy failure: the intruder `I` impersonates `A`, and uses a session auxiliary key `Ki` of his choice to learn the established session key `Kb` in the last message.

```
1.   I(A)  ->   S    :   B, {Ki}PK(S)
2.    S    ->   B    :   A
3.    B    ->   S    :   A, {Kb}PK(S)
4.    S    ->  I(A)  :   B, {Kb}Ki
```

Note that this is a very simple attack without parallel session or replay.

2. [LR97]. Authentication failure: the intruder `I` impersonates `B` and establishes a new session key `Ki` of his choice.

```
1.    A    ->   S    :   B, {Ka}PK(S)
2.    S    ->  I(B)  :   A
3.   I(B)  ->   S    :   A, {Ki}PK(S)
4.    S    ->  I(A)  :   B, {Ki}Ka
```

This attack demonstrates actually more than an authentication flaw, because the established session key is known to the intruder. With the following additional fifth message representing further communications between `A` and `B` using the new established shared key `Kb`:

```
5.   A  ->  B   :    {X}Kb
```

the protocol would not guaranty the secrecy of `X` as expected.

3. [LR97]. Parallel session and replay attack combining the above attacks 1 and 2. Secrecy and authentication failure: at the end of the second session, the intruder knows the established session key `Kb`.

i.1.	I(A)	->	S	:	B, {Ki}PK(S)
i.2.	S	->	B	:	A
i.3.	B	->	S	:	A, {Kb}PK(S)
i.4.	S	->	I(A)	:	B, {Kb}Ki
ii.1.	A	->	S	:	B, {Ka}PK(S)
ii.2.	S	->	I(B)	:	A
ii.3.	I(B)	->	S	:	A, {Kb}PK(S)
ii.4.	S	->	I(A)	:	B, {Kb}Ka

Note that after this attack, `A` and `B` shall communicate with the compromised session key `Kb`. This was not the case with attacks 1 and 2, because during these attacks, the authentication had been performed only with one honest principal.

4. The following secrecy attack, described in [Sim88, Sim94], see also [TMN89], doesn't rely on an authentication failure but on algebraic properties of the encryption method.

It assumes that the symmetric key encryption is performed by a operator `+` such that:

```
(x+y)+y  =  x     (1)
x+(x+y)  =  y     (1')
```

Hence, the protocol reads:

1.	A	->	S	:	B, {Ka}PK(S)
2.	S	->	B	:	A
3.	B	->	S	:	A, {Kb}PK(S)
4.	S	->	A	:	B, Kb + Ka

We `A`, knowing `Ka`, receives the message `4`, he can obtain `Kb` by (1).

Let `*` be a multiplication operator such that the public key encryption algorithm verifies, for all public key `PK(U)`:

```
{x * {y}PK(U)}PK(U) = {x*y}PK(U)    (2)
```

Moreover, we assume a partial division operator (associated to `*`).

These hypotheses are satisfied e.g. if the following choices are made for the operators:

- `+` is `xor`,
- `{x}n` is `x^3 mod n` (with `x < n`),
- `*` is integer multiplication.

The attack is then the following. The intruder `I` has learned the message `3` from a first session `i`, and will use the server `S` as an oracle in a second session `ii` to learn the key `Kb`. `D` is the identity of an honest principal (which can be `A` or `B` or anyone else).

```
 i.3.      B    ->  I(S)   :    A, {Kb}PK(S)
 ii.1.     I    ->   S     :    D, {Ki * {Kb}PK(S)}PK(S) ( = {Ki*Kb}PK(S) by (2) )
 ii.2.     S    ->  I(D)   :    I
 ii.3.    I(D)  ->   S     :    I, {Kd}PK(S)
 ii.4.     S    ->   I     :    D, Kd + (Ki * Kb)
```

`Ki` and `Kd` are arbitrary values chosen by `I`.

After receiving `ii.4`, `I` can compute `Ki * Kb` = `Kd + (Kd + (Ki * Kb))`, using (1'), and hence `Kb`.

Note that in this attack, the server `S` cannot detect the replay of `{Kb}PK(S)` in message `ii.1` because it is multiplied with the arbitrary value `Ki`.

Comment sent by Ralf Treinen (*13/01/2003*)

Ralf Treinen has submitted the above claimed attack number 4.

Citations

[LR97] G. Lowe and A. W. Roscoe. Using CSP to detect errors in the TMN protocol. *Software Engineering*, 23(10):659–669, 1997.

[Sim88] Gustavus J. Simmons. An impersonation-proof identity verification scheme. In *Advances in Cryptology: Proceedings of Crypto 87*, volume 293 of *LNCS*, pages 211–215. Springer-Verlag, 1988.

[Sim94] Gustavus J. Simmons. Cryptoanalysis and protocol failure. *Communications of the ACM*, 37(11):56–65, November 1994.

[TMN89] M. Tatebayashi, N. Matsuzaki, and D.B. Newman. Key distribution protocol for digital mobile communication systems. In *Advance in Cryptology — CRYPTO '89*, volume 435 of *LNCS*, pages 324–333. Springer-Verlag, 1989.