

Cryptographic protocols: design and analysis

David Wagner
University of California, Berkeley

Warmup

Establishing a secure channel with a challenge-response protocol:

1. $A \rightarrow B : A$
2. $B \rightarrow A : N_B$
3. $A \rightarrow B : [N_B]_{K_A^{-1}}$
4. $A \rightarrow B : \{\text{message}\}_{K_B}$
5. $A \rightarrow B : \{\text{message}'\}_{K_B}$
- ...

Can you spot the flaw?

X.509 standard #1

Sending a signed, encrypted message to B :

$$1. A \rightarrow B : A, [T_A, B, \{\text{message}\}_{K_B}]_{K_A^{-1}}$$

This has a subtle issue, depending upon how it is used.

Breaking X.509 standard #1

Look again:

$$1. A \rightarrow B : A, [T_A, B, \{\text{message}\}_{K_B}]_{K_A^{-1}}$$

There's no reason to believe the sender was ever aware of the contents of the message. Signatures imply approval but not authorship.

An Attack on X.509 #1

Example: Proving yourself by sending a password.

Attacker M intercepts Alice's encrypted password:

$$1. A \rightarrow B : A, [T_A, B, \{\text{password}\}_{K_B}]_{K_A^{-1}}$$

Then M extracts $\{\text{password}\}_{K_B}$, and sends

$$1'. M \rightarrow B : M, [T_M, B, \{\text{password}\}_{K_B}]_{K_M^{-1}}$$

Now M is in, without needing to know the password.

Another Attack on X.509 #1

Example: Secure auctions.

The same attack provides an easy way for M to send in a copy of A 's bid under his own name, without needing to know what A 's bid was.

Lessons

An important difference between

- Authentication as *endorsement* (i.e., taking responsibility).
- Authentication as a way of *claiming credit*.

Encrypting before signing provides a secure way of assigning responsibility, but an insecure way to establishing credit.

Moral: sign before encrypting.

Credits: Abadi and Needham.

TMN

A, B establish a shared key k_B using the help of a fast server S :

1. $A \rightarrow S : \{k_A\}_{K_S}$
2. $B \rightarrow S : \{k_B\}_{K_S}$
3. $S \rightarrow A : k_A \oplus k_B$

A recovers k_B as $k_A \oplus (k_A \oplus k_B)$.

What's the flaw?

Breaking TMN

Let's play spot the oracle!

The attack: Given $\{k_B\}_{K_S}$, M, M' can conspire to recover k_B :

- 1'. $M \rightarrow S : \{k_B\}_{K_S}$
- 2'. $M' \rightarrow S : \{k_{M'}\}_{K_S}$
- 3'. $S \rightarrow M : k_B \oplus k_{M'}$

Now M, M' can recover k_B from $\{k_B\}_{K_S}$.

This lets eavesdroppers recover session keys established by other parties.

Credits: Simmons.

Needham-Schroeder

A, B establish a secure channel, given knowledge of each other's public key:

1. $A \rightarrow B : \{A, N_A\}_{K_B}$
2. $B \rightarrow A : \{N_A, N_B\}_{K_A}$
3. $A \rightarrow B : \{N_B, \text{message}\}_{K_B}$

This protocol was published in 1978. In 1996, Gavin Lowe found a flaw. Can you find it, too?

The Lowe attack

Suppose Alice initiates a session with dishonest Dave:

$$1. A \rightarrow D : \{A, N_A\}_{K_D}$$

Dave can then convince Bob he is Alice:

$$1'. D \rightarrow B : \{A, N_A\}_{K_B}$$

$$2'. B \rightarrow A : \{N_A, N_B\}_{K_A}$$

$$3. A \rightarrow D : \{N_B, \text{message}\}_{K_D}$$

$$3'. D \rightarrow B : \{N_B, \text{message}'\}_{K_B}$$

The Smash protocol

A, B establish a secure channel, given knowledge of each other's public key:

1. $A \rightarrow B : \{A, N_A\}_{K_B}$
2. $B \rightarrow A : \{N_A\}_{K_A}$
3. $A \rightarrow B : \{N_A, \text{message}\}_{K_B}$

Yes, this too has a flaw. What is it?

Smashing the Smash

Nothing prevents Zorro from lying about his identity in the first message:

1. $Z \rightarrow B : \{A, N_Z\}_{K_B}$
2. $B \rightarrow A : \{N_Z\}_{K_A}$
3. $Z \rightarrow B : \{N_Z, \text{message}\}_{K_B}$