Announcements

• Project 1 has been graded:
  – Average: 85

• Project 2 is due Today

• Project 3 will be available on the web site this afternoon
  – Due date: Nov. 19th
Mutual Public Key Decryption

• Exchange nonces

\[ K_B \{ n_A, A \} \]
\[ K_A \{ n_A, n_B \} \]
\[ n_B \]
Unilateral Digital Signatures

• $S_A\{M\}$ is A’s signature on message M.
• Unilateral authentication with nonces:

\[ \text{A} \quad n_B \quad n_A, B, S_A\{n_A, n_B, B\} \]

The $n_A$ prevents chosen plaintext attacks.
Primary Attacks

- Replay.
- Interleaving.
- Reflection.
- Forced delay.
- Chosen plaintext.
Primary Controls

• Replay:
  – use of challenge-response techniques
  – embed target identity in response.

• Interleaving
  – link messages in a session with chained nonces.

• Reflection:
  – embed identifier of target party in challenge response
  – use asymmetric message formats
  – use asymmetric keys.
Primary Controls, continued

• Chosen text:
  – embed self-chosen random numbers ("confounders") in responses
  – use “zero knowledge” techniques.

• Forced delays:
  – use nonces with short timeouts
  – use timestamps in addition to other techniques.
General Principles

• Don’t do anything more than necessary until confidence is built.
  – Initiator should prove identity before the responder does any “expensive” action (like encryption)

• Embed the intended recipient of the message in the message itself

• Principal that generates a nonce is the one that verifies it

• Before encrypting an untrusted message, add “salt” (i.e. a nonce) to prevent chosen plaintext attacks

• Use asymmetric message formats (either in “shape” or by using asymmetric keys) to make it harder for roles to be switched
Multiple Use of Keys

• Risky to use keys for multiple purposes.
• Using an RSA key for both authentication and signatures may allow a chosen-text attack.
• B attacker/verifier, \( n_B = H(M) \) for some message \( M \).

\[ \text{B, pretending to be A} \]

\[ n_B \quad \rightarrow \quad k_A\{n_B\} \quad \rightarrow \quad M, k_A\{H(M)\} \]
Effective Control

- Notice how the protocol described earlier foils this. Here’s the protocol:

  - B -> A: n_B
  - A -> B: n_A, B, k_A{n_A, n_B, B}
  - B(A) -> C: M, k_A{n_A, H(M), B}
  - C finds that k_A{n_A, H(M), B} ≠ k_A{H(M)} and rejects the signature.

- Here’s what happens:
  - B -> A: n_B
  - A -> B: n_A, B, k_A{n_A, n_B, B}
  - B(A) -> C: M, k_A{n_A, H(M), B}
  - C finds that k_A{n_A, H(M), B} ≠ k_A{H(M)} and rejects the signature.
Key Establishment

• Symmetric keys.
  – Point-to-Point.
  – Needham-Schroeder.
  – Kerberos.
Symmetric Keys

• Key establishment using only symmetric keys requires use of pre-distribution keys to get things going.

• Then protocol can be based on:
  – Point to point distribution, or
  – Key Distribution Center (KDC).
Point-to-Point

- Should also use timestamps & nonces.
- Session key should include a validity duration.
Key Distribution Centers

Give me a key to talk with Bart

Here is the key

Tom gave us this session key
Distribution Center Setup

• A wishes to communicate with B.
• T (trusted 3rd party) provides session keys.
• T has a key $K_{AT}$ in common with A and a key $K_{BT}$ in common with B.
• A authenticates T using a nonce $n_A$ and obtains a session key from T.
• A authenticates to B and transports the session key securely.
Needham-Schroeder Protocol

1. \( A \rightarrow T : \quad A, B, n_A \)

2. \( T \rightarrow A : \quad K_{AT}\{K_S, n_A, B, K_{BT}\{K_S, A}\} \)
   \( A \) decrypts with \( K_{AT} \) and checks \( n_A \) and \( B \). Holds \( K_S \) for future correspondence with \( B \).

3. \( A \rightarrow B : \quad K_{BT}\{K_S, A\} \)
   \( B \) decrypts with \( K_{BT} \).

4. \( B \rightarrow A : \quad K_S\{n_B\} \)
   \( A \) decrypts with \( K_S \).

5. \( A \rightarrow B : \quad K_S\{n_B - 1\} \)
   \( B \) checks \( n_B - 1 \).
Attack Scenario 1

1. A $\rightarrow$ T : A, B, $n_A$

2. T $\rightarrow$ C (A) : $K_{AT}\{k, n_A, B, K_{BT}\{K_S, A}\}$

C is unable to decrypt the message to A; passing it along unchanged does no harm. Any change will be detected by A.
Attack Scenario 2

1. $A \rightarrow C (T): A, B, n_A$
2. $C (A) \rightarrow T: A, C, n_A$
3. $T \rightarrow A: K_{AT}\{K_S, n_A, C, K_{CT}\{K_S, A}\}$

Rejected by $A$ because the message contains $C$ rather than $B$. 
Attack Scenario 3

1. A $\rightarrow$ C (T) : A, B, $n_A$

2. C $\rightarrow$ T : C, B, $n_A$

3. T $\rightarrow$ C : $K_{CT}\{K_S, n_A, B, K_{BT}\{K_S, C}\}$

4. C (T) $\rightarrow$ A : $K_{CT}\{K_S, n_A, B, K_{BT}\{K_S, C}\}$

A is unable to decrypt the message.
Attack Scenario 4

1. $C \rightarrow T : C, B, n_A$
2. $T \rightarrow C : K_{CT\{K_S, n_A, B, K_{BT\{K_S, C\}}\}}$
3. $C (A) \rightarrow B : K_{BT\{K_S, C\}}$

B will see that the purported origin (A) does not match the identity indicated by the distribution center.
Valid Attack

- The attacker records the messages on the network (in particular, the messages sent in step 3).
- Consider an attacker that manages to get an old session key $K_S$.
- That attacker can then masquerade as Alice:
  - Replay starting from step 3 of the protocol, but using the message corresponding to $K_S$.

- Could be prevented with time stamps.