CS 408
Cryptography & Internet Security

Lectures 20, 21:
Key Establishment (KE) Protocols
Needham-Schroeder Protocol

Announcement

- Course evaluations are available online until May 5, 2015, at:

  http://moodle.njit.edu

  or

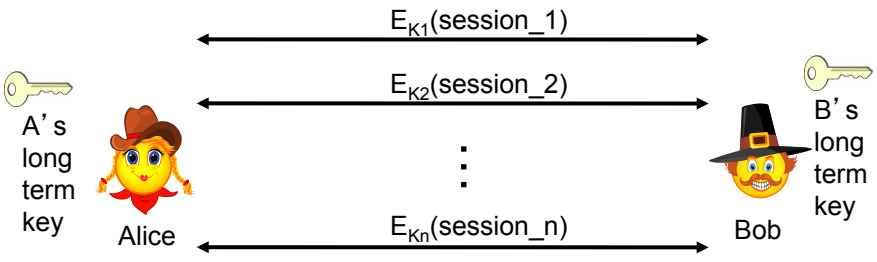
  http://survey.njit.edu/courseeval
  - It should take less than 3 minutes to complete.
  - All responses are strictly confidential.

- Final exam will be on May 11, 2015 between 11:30am – 2:00pm in FMH 319
  - http://www.njit.edu/registrar/exams/finalexams.php

Secure Communication

- If two parties, Alice and Bob, want to talk securely with each other, they should employ some cryptographic primitives (encryption, MAC, digital signatures, etc.)
- For this, they need to have a shared secret key.
- How to establish this shared key?

Long-term keys vs. Session keys



- $K_1, K_2, \ldots, K_n$ are session keys
  - Used for a short amount of time (*e.g.*, one session)
- Why use session keys?
  - To limit available ciphertext for cryptanalytic attacks
  - To limit exposure in case of key compromise
  - Efficiency

Perfect Forward Secrecy

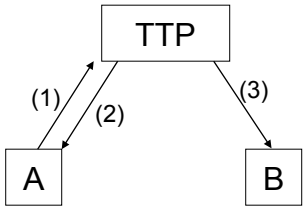
- A key establishment (KE) protocol has perfect forward secrecy if compromise of long-term keys does not compromise past session keys
  - Past session keys must be erased from memory after being used (i.e., after the session ends)
  - Future session keys cannot be protected

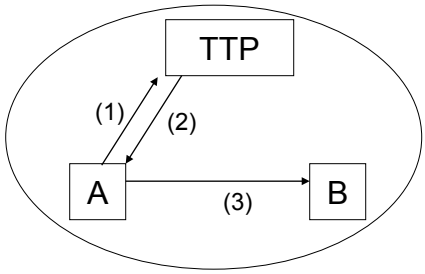
Known-key Attack

- A KE protocol is vulnerable to a known-key attack if compromise of a session key allows compromise of other session keys
- The need for *key independence* between different sessions

Key Establishment Using a KDC

- A network with n users
- Insecure communication channels
- Solutions to the n^2 key distribution problem
  - Use centralized key management
  - Use public key cryptography
- Key distribution center (KDC)
  - It is a trusted third party (TTP)
  - Must be online





Notation

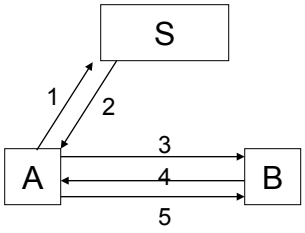
- A, B — principals participating in a protocol
- S — server (the TTP)
- T — timestamp
- N — nonce (a value used only once, to ensure *freshness*; prevents replay attacks)
- Δt — an interval of time (validity period or expiration time)
- $K_{a,b}$ — key shared between A, B
- $\{X\}_K$ — X encrypted under K
- B $\Rightarrow$ A: $\{T_1+1\}_{Ka,b}$ means B sends to A a timestamp incremented by 1, encrypted under key $K_{a,b}$

Needham-Schroeder protocol

Goal: establish a session key $K_{a,b}$ between A and B, and ensure mutual authentication between A and B

We assume that A and B have each already established a secret key with S (these are called long term keys: $K_{a,s}$, $K_{b,s}$)

1. A ⇒ S: A, B, N_1
2. S ⇒ A: $\{N_1, B, K_{a,b}, \{K_{a,b}, A\}_{Kb,s}\}_{Ka,s}$
3. A ⇒ B: $\{K_{a,b}, A\}_{Kb,s}$
4. B ⇒ A: $\{N_2\}_{Ka,b}$
5. A ⇒ B: $\{N_2\text{-}1\}_{Ka,b}$



$\{K_{a,b}, A\}_{Kb,s}$ is a *ticket* from S for B

• after the protocol, A and B share a secret key $K_{a,b}$ with each other (session key)

Needham-Schroeder protocol - analysis

- We need nonce N_1 to prevent replay attacks and ensure that A is really talking to S
- Otherwise, Eve can reuse an old message from S, which uses a ticket $\{K_{a,b}, A\}_{Kb,s}$, in which $K_{b,s}$ is an old key that was stolen by Eve

Eve records an old message $\{B, K_{a,b}, \{K_{a,b}, A\}_{Kb,s}\}_{Ka,s}$ from a previous run of the protocol
Eve steals key $K_{b,s}$; B changes $K_{b,s}$

1. A ⇒ S: A, B
2. E ⇒ A: $\{B, K_{a,b}, \{K_{a,b}, A\}_{Kb,s}\}_{Ka,s}$
3. A ⇒ B: $\{K_{a,b}, A\}_{Kb,s}$
   E gets $K_{a,b}$ and can impersonate B to A
4. E ⇒ A: $\{N_2\}_{Ka,b}$
5. A ⇒ B: $\{N_2\text{-}1\}_{Ka,b}$

steps 4,5 don't prevent the attack

Needham-Schroeder protocol - analysis

- Message 2: identities are included to ensure each party knows who is at the other end of the communication
- Otherwise, Eve can impersonate Bob (if Bob's identity is not included in message 2):

1. $A \Rightarrow S$: A, B, N_1
   Eve intercepts and modifies message 1 to: A, E, N_1
2. $S \Rightarrow A$: $\{N_1, K_{a,e}, \{K_{a,e}, A\}_{Ke,s}\}_{Ka,s}$
3. A establishes a secret key with E

- Protocol was criticized for doubly encrypting the ticket

Needham-Schroeder protocol - analysis

- Messages 4 and 5: nonce N_2 is required so that A can prove knowledge of the key (A authenticates herself to B)

1. $A \Rightarrow S$: A, B, N_1
2. $S \Rightarrow A$: $\{N_1, B, K_{a,b}, \{K_{a,b}, A\}_{Kb,s}\}_{Ka,s}$
3. $A \Rightarrow B$: $\{K_{a,b}, A\}_{Kb,s}$
4. $B \Rightarrow A$: $\{N_2\}_{Ka,b}$
5. $A \Rightarrow B$: $\{N_2\text{-}1\}_{Ka,b}$

(otherwise, Eve could replay an old message 3 and pretend to be A)

Needham-Schroeder protocol - flaws

- B never proves knowledge of the session key
  What would you do?
- Corrected protocol:

1. $A \Rightarrow S: A, B, N_1$
2. $S \Rightarrow A: \{N_1, B, K_{a,b}, \{K_{a,b}, A\}_{Kb,s}\}_{Ka,s}$
3. $A \Rightarrow B: \{K_{a,b}, A\}_{Kb,s}, \{N_3\}_{Ka,b}$
4. $B \Rightarrow A: \{N_3\text{-}1, N_2\}_{Ka,b}$
5. $A \Rightarrow B: \{N_2\text{-}1\}_{Ka,b}$

- The updated protocol provides *mutual authentication* (A knows she's talking to B, and B knows he's talking to A *at the time when protocol is performed*)

Needham-Schroeder protocol - flaws

Are we done?
NO!

1. $A \Rightarrow S: A, B, N_1$
2. $S \Rightarrow A: \{N_1, B, K_{a,b}, \{K_{a,b}, A\}_{Kb,s}\}_{Ka,s}$
3. $A \Rightarrow B: \{K_{a,b}, A\}_{Kb,s}, \{N_3\}_{Ka,b}$
4. $B \Rightarrow A: \{N_3\text{-}1, N_2\}_{Ka,b}$
5. $A \Rightarrow B: \{N_2\text{-}1\}_{Ka,b}$

What if Eve gets hold of an old session key $K_{a,b}$?
She can impersonate A by replaying an old message 3!

What would you do?

Needham-Schroeder protocol - flaws

- Fix 1: S includes a timestamp T

1. A ⇒ S: A, B, N_1
2. S ⇒ A: $\{T, N_1, B, K_{a,b}, \{K_{ab}, A, T\}_{Kb,s}\}_{Ka,s}$
3. A ⇒ B: $\{K_{a,b}, A, T\}_{Kb,s}$, $\{N_3\}_{Ka,b}$
4. B ⇒ A: $\{N_3\text{-}1, N_2\}_{Ka,b}$
5. A ⇒ B: $\{N_2\text{-}1\}_{Ka,b}$

Replay of an old message 3 will be recognized as old and ignored by B!

Requires synchronized clocks

- Only loose clock synchronization is necessary

Needham-Schroeder protocol - flaws

- Fix 2: add two more messages

1. A ⇒ B: I'm A, I want to talk to you
2. B ⇒ A: $\{N_4\}_{Kb,s}$
3. A ⇒ S: A, B, N_1, $\{N_4\}_{Kb,s}$
4. S ⇒ A: $\{N_1, B, K_{a,b}, \{K_{a,b}, A, N_4\}_{Kb,s}\}_{Ka,s}$
5. A ⇒ B: $\{K_{a,b}, A, N_4\}_{Kb,s}$, $\{N_3\}_{Ka,b}$
6. B ⇒ A: $\{N_3\text{-}1, N_2\}_{Ka,b}$
7. A ⇒ B: $\{N_2\text{-}1\}_{Ka,b}$

B accepts message 5 only if it contains the same nonce B used in message 2!