Protocols for Authentication and Key Establishment

2005 CIMPA-UNESCO School
IISc Bangalore

Anish Mathuria
DA-IICT

Main References

- **Handbook of Applied Cryptography** - Menezes, Oorschot and Vanstone (CRC)
- **Protocol for Authentication and Key Establishment** - Boyd and Mathuria (Springer)

Module Outline

- Lecture 1 (this lecture)
  - » Key transport
- Lecture 2 (tomorrow, 11-1)
  - » Entity authentication
  - » Key agreement
- Lecture 3 (tomorrow, 2-4)
  - » Group key agreement
  - » Password-based protocols

Ideal Security Protocol

- Does the protocol meet the requirements?
  - » N.B. requirements must be precise
- Not fragile
  - » Must work when adversary tries to break it
  - » Works even if environment changes
- Minimizes computational and/or communication cost
- Very difficult to satisfy all of these!

Key establishment

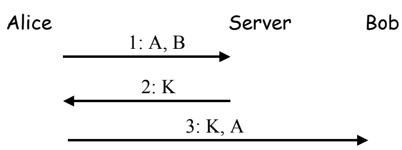
- Secure communications using cryptography requires use of (session) keys that must be shared by participants
- If participants do not physically meet, keys have to be established using a suitable protocol

Classification

- Key transport
  - » one party creates a shared secret, and securely transfers it to other(s)
- Key agreement
  - » parties jointly create a shared secret

Key Transport Protocols

First Protocol Attempt



- K = session key for A and B generated by S
- Is this secure?
- No, the key is not secret !

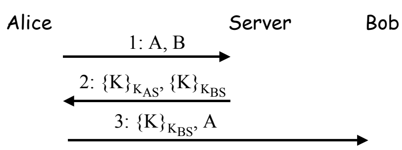
Assumption 1

- The adversary can eavesdrop on all messages sent in a protocol
- Countermeasure
  - » Make K confidential by encrypting it with another key
- Long-term keys necessary
  - » Symmetric key
  - » Private, public key pair

Notations

- $\{M\}_K$: encryption of M with symmetric key K
  - » Assume encryption provides both confidentiality and integrity
- $E_X(M)$: encryption of M with public key of entity X
- $sig_X(M)$: digital signature of M using the private key of entity X
  - » Assume not a message-recovering signature (but it can be)

Second Protocol Attempt



- Server shares key K_{AS} with Alice, key K_{BS} with Bob, key K_{CS} with Carol, etc.
- Is this secure?

Assumption 2

- The adversary can alter all messages sent in a protocol using any information available
- The adversary can re-route any message to any principal
- The adversary can generate and insert completely new messages

Authentication Attack

Alice — Charlie — Server

1: A, B (Alice → Charlie)

1': A, C (Charlie → Server)

2: $\{K\}_{K_{AS}}, \{K\}_{K_{CS}}$ (Server → Alice)

3: $\{K\}_{K_{CS}}$, A (Alice → Charlie)

- What went wrong?
- Alice obtains session key K with Bob
- But K was intended as session key for Alice and Charlie!

Authentication Property

- Alice and Bob should have assurance of the identity of the other party who can obtain K
- How to achieve this?

Third Protocol Attempt

Alice — Server — Bob

1: A, B (Alice → Server)

2: $\{K, B\}_{K_{AS}}, \{K, A\}_{K_{BS}}$ (Server → Alice)

3: $\{K, A\}_{K_{BS}}$ (Alice → Bob)

- Bob's (Alice's) ID is bound to K
  - » Proves that server will reveal K to Bob (Alice) only
  - » Works only if encryption algorithm provides integrity
- This protocol prevents the authentication attack
- Is it secure? See the next slide …

Security Assumption 4

- An adversary can obtain the value of the session key used in any sufficiently old previous run of the protocol

Replay Attack

Alice — Charlie as server — Bob

1: A, B (Alice → Charlie)

2: $\{K^*, B\}_{K_{AS}}, \{K^*, A\}_{K_{BS}}$ (Charlie → Alice)

3: $\{K^*, A\}_{K_{BS}}$ (Alice → Bob)

- K* = old session key between A and B
- What's went wrong?
- Charlie knows K*!

Freshness

- Alice and Bob should have assurance that K is newly generated
- One secure method for achieving freshness
  - » Challenge sent from Alice to Server
  - » Only server can provide the correct response
  - » Challenge chosen so that replay is not possible
- For challenge, a random value or "number used once" (nonce)

Final Protocol Attempt

Bob Alice Server

1: B, N_B (Bob → Alice)

2: A, B, N_A, N_B (Alice → Server)

3: $\{K, B, N_A\}_{K_{AS}}, \{K, A, N_B\}_{K_{BS}}$ (Server → Alice)

4: $\{K, A, N_B\}_{K_{BS}}$ (Alice → Bob)

- N_A, N_B = nonces generated by A and B resp.
- This protocol protects against replay attack

Protocol Using Timestamps

Alice Bob Server

1: A, B (Bob → Server)

2: $\{K, B, T_S\}_{K_{AS}}, \{K, A, T'_S\}_{K_{BS}}$ (Server → Bob)

3: $\{K, B, T_S\}_{K_{AS}}$ (Bob → Alice)

- T_S, T'_S = timestamps generated by S

Security Assumption 5

- The adversary can start any number of parallel protocol runs between any principals including different runs involving the same principals and with principals taking the same or different protocol roles
- This is a common source of protocol failures

Attack Strategies

- Replay
  - » Adversary records information in the protocol and sends it to the same, or a different, principal, possibly during a later protocol run
- Reflection
  - » Adversary sends protocol messages back to the principal who sent them
- Typing
  - » Adversary replaces a message field of one type with a message field of another type

Attack Strategies (2)

- Denial of service
  - » Adversary prevents or hinders legitimate principals from completing the protocol
- Certificate manipulation
  - » Adversary chooses or modifies certification information
- Protocol interaction
  - » Adversary uses one protocol to attack another protocol

Some Attacks

Otway-Rees protocol

Alice Bob Server

1: M, A, B, $\{N_A, M, A, B\}_{K_{AS}}$

2: M, A, B, $\{N_A, M, A, B\}_{K_{AS}}$, $\{N_B, M, A, B\}_{K_{BS}}$

3: M, $\{N_A, K\}_{K_{AS}}$, $\{N_B, K\}_{K_{BS}}$

4: M, $\{N_A, K\}_{K_{AS}}$

- Is it secure? See the next slide

Typing Attack

- Adv. sends A's message back to A

Alice Adv. as Bob Server

M, A, B, $\{N_A, M, A, B\}_{K_{AS}}$

M, $\{N_A, M, A, B\}_{K_{AS}}$

(instead of M, $\{N_A, K\}_{K_{AS}}$)

- Thus A may be fooled into accepting (M, A, B) as the new session key

BAN Otway-Rees protocol

Alice Bob Server

1: M, A, B, $\{N_A, M, A, B\}_{K_{AS}}$

2: M, A, B, $\{N_A, M, A, B\}_{K_{AS}}$, N_B, $\{M, A, B\}_{K_{BS}}$

3: M, $\{N_A, K\}_{K_{AS}}$, $\{N_B, K\}_{K_{BS}}$

4: M, $\{N_A, K\}_{K_{AS}}$

- N_B is sent unencrypted in message 2
- Is it secure? See the next slide

Boyd and Mao's Attack

- Assume C obtains $\{M, C, B\}_{K_{BS}}$ by running the protocol with B

Adv. as Alice — Bob — Server

1: $M, A, B, \{N_C, M, C, B\}_{K_{CS}}$

2: $M, A, B, \{N_A, M, A, B\}_{K_{AS}}, N_B, \{M, C, B\}_{K_{BS}}$

(instead of …, N_B, $\{M, A, B\}_{K_{BS}}$)

3: $M, \{N_C, K\}_{K_{CS}}, \{N_B, K\}_{K_{BS}}$

4: $M, \{N_C, K\}_{K_{CS}}$

- Thus B accepts K as a session key with A, although it is shared with C !

Woo-Lam Protocol

Alice — Bob — Server

1: N_A

2: N_B

3: $\{A, B, N_A, N_B\}_{K_{AS}}$

4: $\{A, B, N_A, N_B\}_{K_{AS}}, \{A, B, N_A, N_B\}_{K_{BS}}$

5: $\{B, N_A, N_B, K\}_{K_{AS}}, \{A, N_A, N_B, K\}_{K_{BS}}$

6: $\{B, N_A, N_B, K\}_{K_{AS}}$

- This has to work …
- Is it secure? See the next slide

Lowe's Attack

- Session I

Adv As Alice — Bob — Adv as Server

1: B

2: N_B

3: string1

4: string1, $\{A, B, B, N_B\}_{K_{BS}}$

9: string3, $\{A, B, N_B, N_B'\}_{K_{BS}}$

10: string3

- Session II

Adv As Alice — Bob — Adv as Server

5: N_B

6: N_B'

7: string2

8: string2, $\{A, B, N_B, N_B'\}_{K_{BS}}$

Wide-mouthed-frog protocol

Alice Server Bob

1: A, $\{T_A, B, K\}_{K_{AS}}$ → 2: A, $\{T_S, A, K\}_{K_{BS}}$ →

- T_A, T_B = timestamps generated by A and S
- This has to work…
- Is it secure? See next slide

Replay attack

Alice Server Bob

1: A, $\{T_A, B, K\}_{K_{AS}}$ → 2: $\{T_S, A, K\}_{K_{BS}}$ →

Adv. as Bob Server Adv. as Alice

1': B, $\{T_S, A, K\}_{K_{BS}}$ → 2': $\{T'_S, B, K\}_{K_{AS}}$ →

Adv. as Alice Server Adv. as Bob

1'': A, $\{T'_S, B, K\}_{K_{AS}}$ → 2'': $\{T''_S, A, K\}_{K_{BS}}$ →

- Adv. can continue in this fashion until session key is discarded and then fool A or B into accepting the key again