On Efficient Key Agreement Protocols

Anish Mathuria
DA-IICT, Gandhinagar

IIT Kanpur Hacker’s Workshop
March 5, 2005

Key Establishment Protocols

- Cryptographic protocols that use long-term keys in order to setup short-term (session) keys
- Needed
  » to limit available cipher-text
  » to limit exposure in the event of (session) key compromise
  » to create independence across communications sessions or applications
Classification

• Key transport protocol
  » one party creates, and securely transfers it to the other(s).
• Key agreement protocol: key establishment technique in which
  » a shared secret is derived by two (or more) parties

Notation

• \( \{M\}_K \): encryption of \( M \) with symmetric key \( K \)
• Properties
  » Only some who knows \( K \) can read \( M \) (confidentiality)
  » Only some who knows \( K \) can construct \( \{M\}_K \) (integrity)
First Protocol Attempt

Alice       Server       Bob

1: A, B

2: \(\{K\}_{K_{AS}}, \{K\}_{K_{BS}}\)

3: \(\{K\}_{K_{BS}}, A\)

- \(K\) = session key for A and B generated by Server
- Server shares key \(K_{AS}\) with Alice, key \(K_{BS}\) with Bob, key \(K_{CS}\) with Carol, etc.
- Is this secure?

Assumption 1

- 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
**Attack**

Alice

Server

Charlie

1: A, B

2: $\{K\}_{KAS}, \{K\}_{KCS}$

3: $\{K\}_{KCS}, A$

1’: A, C

- What went wrong?
- Alice accepts K as a session key with Bob
- But K is known to Charlie!

---

**Key exclusivity**

- A (respectively, B) should have assurance that K is known only to A and B and any mutually trusted parties
- This property is often called implicit key authentication (Menezes, Oorschot and Vanstone)
  - N.B. this property is actually a confidentiality property (Gollmann)
Second Protocol Attempt

Bob’s (Alice’s) ID is bound to K
  » Proves that server will reveal K to Bob (Alice) only
  » Encryption must ensure message integrity

This protocol protects against the previous attack

Is it secure? See the next slide …

Security Assumption 2

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

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

Key Freshness

- Alice and Bob should have assurance that K is newly generated
- One 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)
Nonce-based protocol

1: \( B, N_B \) 
2: \( A, B, N_A, N_B \) 
3: \( \{K, B, N_A\}_{K_{AS}}, \{K, A, N_B\}_{K_{BS}} \) 
4: \( \{K, A, N_B\}_{K_{BS}} \)

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

Timestamp-based protocol

1: \( A, B \) 
2: \( \{K, B, T_S\}_{K_{AS}}, \{K, A, T'_S\}_{K_{BS}} \) 
3: \( \{K, A, T'_S\}_{K_{BS}} \) 

- Server adds timestamps \( T_S, T'_S \) to its messages
- Recipient checks that timestamp lies within an appropriate time window
- Synchronization of clocks required
Two Protocol Metrics [Gong]

- **Metric 1:** Total number of messages required by a protocol
- **Metric 2:** Total number of rounds required by a protocol
  » This metric gives a measure of the time needed to execute the protocol

Gong’s Results

- Protocols using nonce-based challenge-response require exactly one more message and round than timestamp-based protocols
- For certain protocol classes no protocols can be both message and round optimal
3-msg/2-round timestamp-based protocol

1. \(A \rightarrow S: A, B\)
2. \(S \rightarrow A: \{B, K, T_S\}_{KAS}\)
3. \(S \rightarrow B: \{A, K, T_S\}_{KBS}\)

- Message 2 and 3 sent in parallel
- This protocol is both message and round optimal

Boyd's 3-msg protocol

\[
\begin{align*}
\text{Bob} & \quad \text{Server} & \quad \text{Alice} \\
1: & A, B, N_A & 2: \{A, B, K_S\}_{KAS}, \{A, B, K_S\}_{KBS}, N_A \\
3: & N_B, \{A, B, K_S\}_{KAS} & 1: A, B, N_A
\end{align*}
\]

- \(K = MAC_{K_S}(N_A, N_B)\)
- Adv. cannot compute \(K\) without knowledge of \(K_S\)
- \(MAC\) is "secure" against collision so key freshness is provided
Boyd’s 2-round protocol

1. $A \rightarrow S$: A, B
2. $A \rightarrow B$: A, $N_A$
3. $S \rightarrow B$: $\{A, B, K_S\}_{K_{BS}}$
4. $S \rightarrow A$: $\{A, B, K_S\}_{K_{AS}}$
5. $B \rightarrow A$: B, $N_B$

- Sent in parallel: messages 1 and 2, and messages 3, 4, and 5
- No protocol can be both message and round optimal

Forward Secrecy

- Property that compromise of long-term keys does not compromise past session keys
- A protocol which encrypts the session key with a long-term key (public or symmetric) cannot provide forward secrecy
Diffie-Hellman based variant

1. $A \rightarrow S: A, B, g^N_A$
2. $S \rightarrow B: \{A, B, k_S\}_{K_{AS}}, \{A, B, k_S\}_{K_{BS}}, g^N_A$
3. $B \rightarrow A: \{A, B, k_S\}_{K_{AS}}, g^N_B$

- The session key $K = g^{N_A N_B K_S}$
- What if adversary finds long-term keys $K_{AS}$ and $K_{BS}$?
  - If $A$ and $B$ destroy $N_A$ and $N_B$ after computing $K$, Adv. cannot discover past keys (although subsequent sessions are compromised)
- Protocol provides forward secrecy

Bauer-Berson-Feiertag Attack

- If $A$'s long term key is compromised and subsequently, Adv. can still masquerade as $A$ to $B$
- As noted by Boyd his protocols are vulnerable to the BBF attack
- Alternative protocols using nonce-based challenge-response are not vulnerable to this attack
New 3-msg protocol

1. \( \text{A} \to \text{S}: \text{A}, \{\text{B}, g^{\text{NA}}\}_{K_{\text{AS}}} \)
2. \( \text{S} \to \text{B}: \{\text{A}, g^{\text{NA}}\}_{K_{\text{BS}}} \)
3. \( \text{B} \to \text{A}: g^{\text{NB}} \)

- Session key \( K = g^{\text{NANB}} \)
- B has assurance that \( g^{\text{NA}} \) did indeed come from A, and so B has assurance that only A knows K
- A has assurance that \( g^{\text{NA}} \) will be revealed only to B, and so A has assurance that only B knows K
- This protocol uses 3 rounds but it is easy to design a 2-round protocol with the same method (at the cost of additional messages)

Resistance to BBF Attack

1. \( \text{A} \to \text{S}: \text{A}, \{\text{B}, g^{\text{NA}}\}_{K_{\text{AS}}} \)
2. \( \text{S} \to \text{B}: \{\text{A}, g^{\text{NA}}\}_{K_{\text{BS}}} \)
3. \( \text{B} \to \text{A}: g^{\text{NB}} \)

- Suppose \( K_{\text{AS}} \) is compromised and replaced
- Possible attacking strategies
  1. Adv. replays the value \( \{\text{A}, g^{\text{NA}}\}_{K_{\text{BS}}} \) to B
  2. Adv. generates a value \( N_X \) and then sends \( \{\text{B}, g^{N_X}\}_{K_{\text{AS}}} \) to S
Conclusions

• New key agreement protocols that lower Gong’s bounds for nonce-based protocols
• More efficient in messages and rounds than existing provably secure protocols
• N.B. Proposed protocols lack security proof!

Thank You

• For more information: