Security Protocols and Verification

Design and Analysis of Cryptographic Protocols

Garance Frolla
Ely Marthouret
Ewan Decima

September / November 2025

1 Assumptions

We assume that at the beginning of the protocol agents A and B know the public key $K_{pub(C)}$ of any agent C. Moreover, we assume that all agents C shared a symmetric key with a trusted server S, named K_{CS}. N_a is a nonce generated by A, and N_b is a nonce generated by B. τ and λ respectively denote a timestamp and a lifetime. A generates K_{AB} with perfect randomness at each session. .

2 Protocol: First Attempt

1. $A \to S : \{|A, B, N_a, K_{AB}, pub(A)|\}_{K_{AS}}$
2. $S \to A : \left\{ \left| S, A, N_a + 1, \{|A, B, K_{AB}, N_a, \tau, \lambda, pub(A)|\}_{K_{BS}} \right| \right\}_{K_{AS}}$
3. $A \to B : \{|A, B, K_{AB}, N_a, \tau, \lambda, pub(A)|\}_{K_{BS}}$
4. $B \to S : \{|B, A, N_b|\}_{K_{BS}}$
5. $S \to B : \{|S, B, N_b + 1, h(K_{AB}, N_b)|\}_{K_{BS}}$
6. $B \to A : \{B, A, N_a + 1, h(K_{AB}, N_b), pub(B)\}_{pub(A)}$
7. $A \to B : \{A, B, N_b + 1, h(K_{AB}, N_a)\}_{pub(B)}$

2.1 Analysis

2.1.1 Pros

- A can decide when to access the session because he is responsible for sending the ticket $\{|A, B, K_{AB}, N_a, \tau, \lambda, pub(A)|\}_{K_{BS}}$ to B.
- A can verify the authenticity of the ticket's origin.
- Mutual authentication between A and B.
- A and B can prove the server's participation.
- A and B authenticate each other through nonces and hashes.
- Replay attack prevention with λ and τ.
- A and B verify the authenticity of the server.

2.1.2 Cons

- Complex and lengthy protocol.
- If clocks are not synchronized, timestamps can be exploited.
- If S is compromised, all communications are vulnerable.

3 Protocol: Second Attempt

1. $A \rightarrow S : \{|A, B, N_a, \lambda, K_{AB}, pub(A)|\}_{K_{AS}}$
2. $S \rightarrow A : \left\{ \left| S, A, N_a + 1, \{|A, B, K_{AB}, N_a, \tau, \lambda, pub(A)|\}_{K_{BS}} \right| \right\}_{K_{AS}}$
3. $A \rightarrow B : \{|A, B, K_{AB}, N_a, \tau, \lambda, pub(A)|\}_{K_{BS}}$
4. $B \rightarrow A : \{B, A, N_a + 1, h(K_{AB}, N_b), pub(B)\}_{pub(A)}$
5. $A \rightarrow B : \{A, B, N_b + 1, h(K_{AB}, N_a)\}_{pub(B)}$

3.1 Analysis

3.1.1 Pros

- Simpler and shorter protocol.
- A can decide when to access the session because he is responsible for sending the ticket $\{|A, B, K_{AB}, N_a, \tau, \lambda, pub(A)|\}_{K_{BS}}$.
- A can verify the authenticity of the ticket's origin.
- Maintains mutual authentication between A and B.
- A can prove the server's participation.

3.1.2 Cons

- B cannot verify the authenticity of the ticket $\{|A, B, K_{AB}, N_a, \tau, \lambda, pub(A)|\}_{K_{BS}}$.
- S does not know whether B has actually received the key K_{AB}.
- B must trust without verification.
- B cannot prove the server's participation.

4 Protocol: Third Attempt

1. $A \to S : \{|A, B, N_a, \lambda, K_{AB}, pub(A)|\}_{K_{AS}}$
2. $S \to B : \{|A, B, K_{AB}, N_a, \tau, \lambda, pub(A)|\}_{K_{BS}}$
3. $B \to A : \{B, A, N_a + 1, h(K_{AB}, N_b), pub(B)\}_{pub(A)}$
4. $A \to B : \{A, B, N_b + 1, h(K_{AB}, N_a)\}_{pub(B)}$

4.1 Analysis

4.1.1 Pros

- Very short and very simple protocol.
- A can decide when to access the session because he is responsible for sending the ticket $\{|A, B, K_{AB}, N_a, \tau, \lambda, pub(A)|\}_{K_{BS}}$.
- A can verify the authenticity of the ticket's origin.
- Maintains mutual authentication between A and B.

4.1.2 Cons

- A does not know whether S has processed his request.
- A does not know when B has received the key K_{AB}.
- No indication of success from the server.
- A could send messages before B has received the key K_{AB}.
- Impossible to prove the server's participation (lack of non-repudiation).