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

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

3 Protocol Description

This protocol begins with A generating a fresh nonce N_a and a perfectly random session key K_{AB}, then sending these along with the identities A and B to the trusted server S, encrypted under their shared key K_{AS}. Upon receiving this request, server S decrypts the message, verifies its validity, and creates a ticket containing A's identity, B's identity, the session key K_AB, the nonce N_a, and timing information (timestamp τ and lifetime λ), encrypting this ticket under K_{BS}. S then responds to A with a message containing S's identity, A's identity, the incremented nonce $N_a + 1$ for authentication, and the encrypted ticket for B, all encrypted under K_{AS}. A decrypts this response, verifies the nonce increment to confirm S's authenticity, and forwards the ticket to B by sending the encrypted message $\{|A, B, K_{AB}, N_a, \tau, \lambda|\}_{K_{BS}}$. B decrypts this ticket, verifies the timestamp is still valid, extracts the session key K_{AB}, then generates his own nonce N_b and requests confirmation from S by sending $\{|B, A, N_b|\}_{K_{BS}}$. Server S responds to B with the session key K_{AB} and incremented nonce $N_b + 1$, encrypted under K_{BS}, allowing B to verify S's authenticity and obtain the session key. The protocol concludes with mutual key confirmation where B sends A a message containing B's identity, A's identity, the incremented nonce $N_a + 1$, and a hash of the session key with B's nonce $h(K_{AB}, N_b)$, encrypted under A's public key, and A responds similarly with $\{A, B, N_b + 1, h(K_{AB}, N_a)\}_{K_{pub(B)}}$. Throughout the protocol, each party must verify message authenticity through proper decryption, correct nonce increments, and valid hash computations, aborting if any verification fails, ultimately establishing mutual authentication, session key agreement, and key confirmation while providing forward secrecy through A's random key generation.