Security Handshake Pitfalls

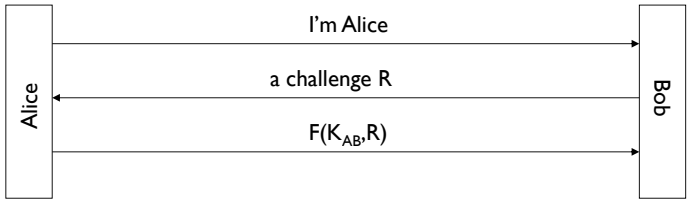
Ahmet Burak Can
Hacettepe University
abc@hacettepe.edu.tr

Cryptographic Authentication

- Password authentication is subject to eavesdropping
- Alternative: Cryptographic challenge-response
  - Symmetric key
  - Public key

Symmetric Key Challenge-Response

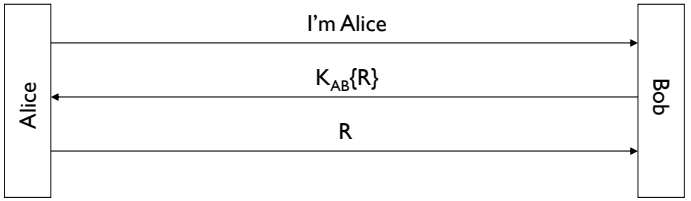
An example protocol:



- Authentication not mutual (login only)
- Subject to connection hijacking (login only)
- Subject to off-line password guessing (if K is derived from password)
- Bob's database has keys in the clear

Symmetric Key Challenge-Response

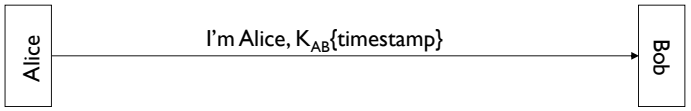
An alternative protocol:



- Requires reversible cryptography
- Subject to dictionary attack, without eavesdropping, if R is recognizable
- Can be used for mutual authentication if R is recognizable and has limited lifetime

Symmetric Key Challenge-Response

A one-message protocol:

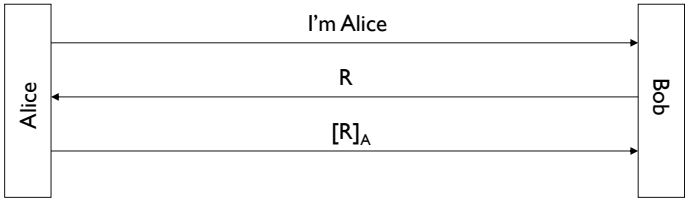


- Easy integration into password-sending systems
  - More efficient: Single message, stateless
  - Care needed against replays: timeout needed
  - Care needed if key is common across servers
  - Clock has to be protected as well
- Alternatively, with a hash function, send,

  I'm Alice, timestamp, $H(K_{AB}$,timestamp)

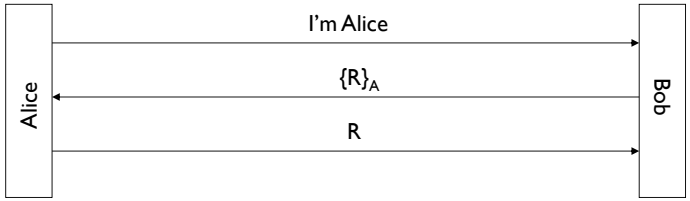
Public Key Challenge-Response

By signature:



Public Key Challenge-Response

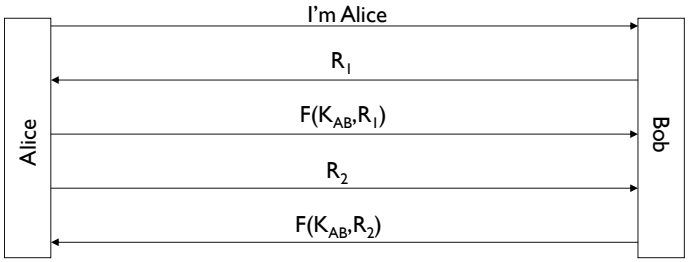
By decryption:



- Problem: Bob (or Trudy) can get Alice to sign/decrypt any text he chooses.
- Solutions:
  - Never use the same key for different purposes (e.g., for login and signature)
  - Use formatted challenges

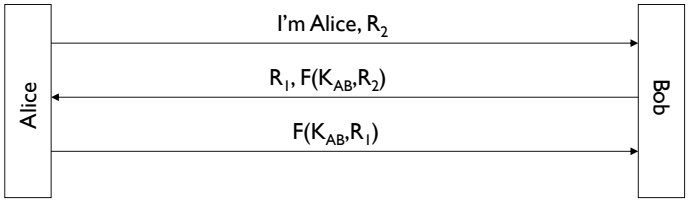
Mutual Authentication

An example protocol:



Mutual Authentication with Few Messages

Number of messages for mutual authentication can be reduced:

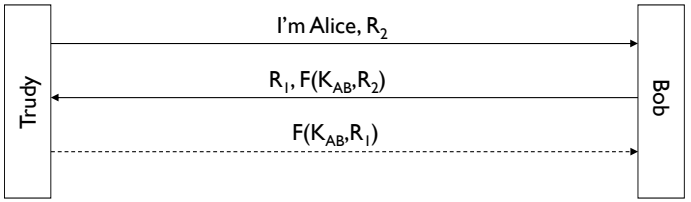


However, this protocol is vulnerable to

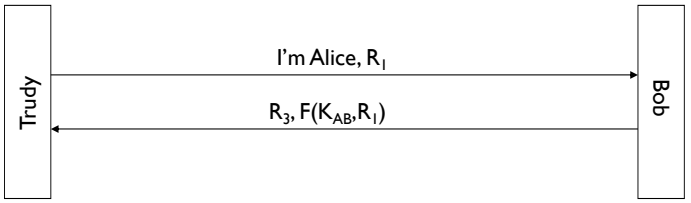
- Reflection attack
- Dictionary attack :Trudy can do dictionary attack against K_{AB} acting as Alice, without eavesdropping.

Reflection Attack:

Original session:



Decoy session:

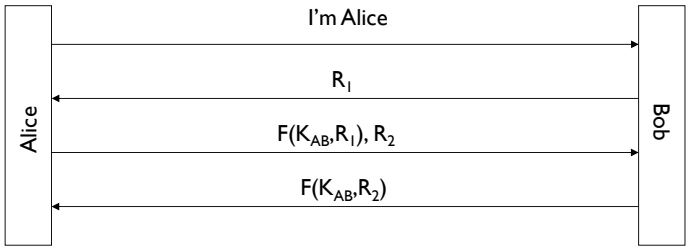


Results from Reflection Attack

- Solutions:
  - Different keys for Alice and Bob
  - Formatted challenges, different for Alice and Bob
- Principle:
  - Initiator should be the first to prove its identity

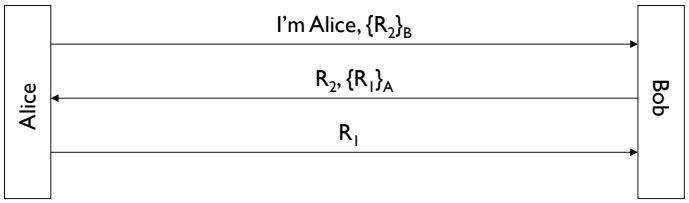
A Modified Mutual Authentication Scheme

- Solution against both problems:



- Dictionary attack is still possible if Trudy can impersonate Bob.

Mutual Authentication with Public Keys

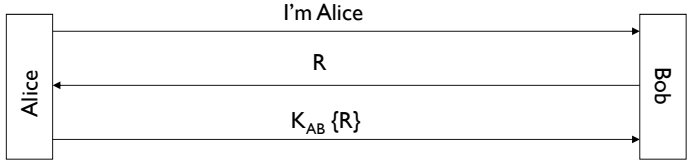


- Problem: How can the public/private keys be remembered by ordinary users?
  - Possibly, they can be retrieved from a server with password based authentication & encryption.

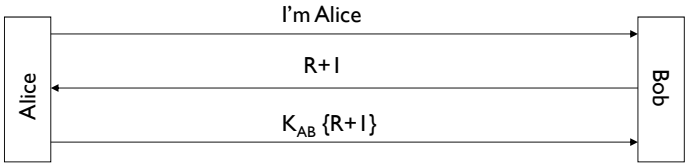
Session Key Establishment

- A session key is needed for integrity protection and encryption in a communication session. It must be
  - different for each session
  - unguessable by an eavesdropper
  - not $K_{AB}\{x\}$ for some x predictable/extractable by an attacker
- Session keys can be established by using
  - Symmetric encryption
  - Public key encryption

Session Key Establishment with Symmetric Encryption

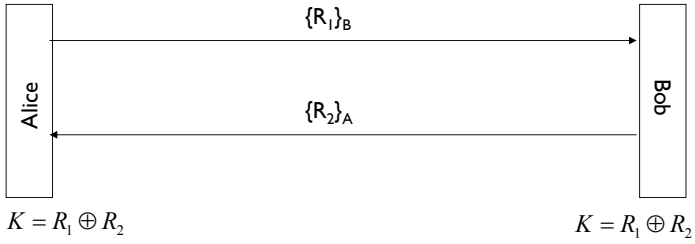


- Do not use $K_{AB}\{R\}$ or $K_{AB}\{R+1\}$
  - Take $(K_{AB}+1)\{R\}$ as the session key.



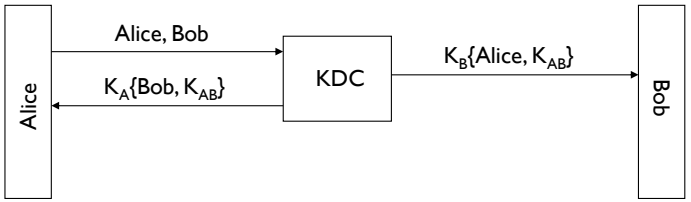
Session Key Establishment with Public Key Cryptosystem

- An alternative is to use Diffie-Helman key exchange algorithm.
- Another alternative with PKC, send additional random nonces $\{R\}_A$, $\{R\}_B$ and use them to derive a session key.



Key Establishment and Authentication with Key Distribution Center (KDC)

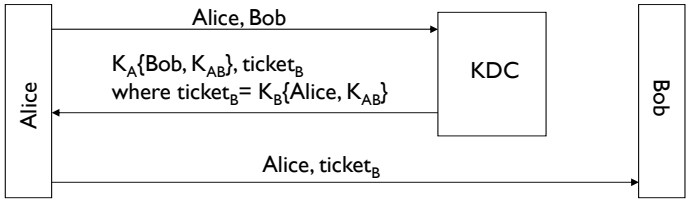
A simple protocol:



- Problem:
  - Potential delayed key delivery to Bob. (besides others)

Key Establishment and Authentication with KDC

- Another simple protocol:

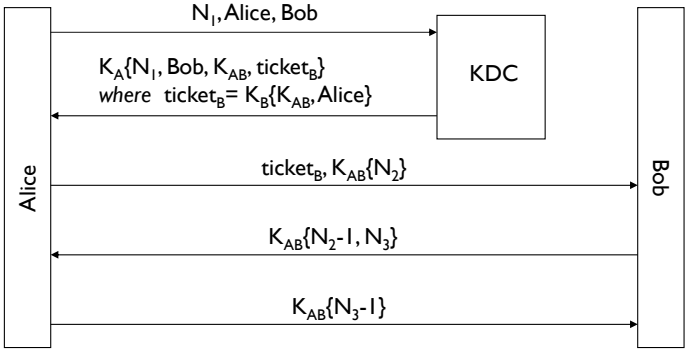


- Problems:
  - No freshness guarantee for K_{AB}
  - Alice & Bob need to authenticate

Nonces

- Nonce: Something created for one particular occasion
- Nonce types:
  - Random numbers
  - Timestamps
  - Sequence numbers
- Random nonces needed for unpredictability
- Obtaining random nonces from timestamps: encryption with a secret key.

Needham-Schroeder Protocol

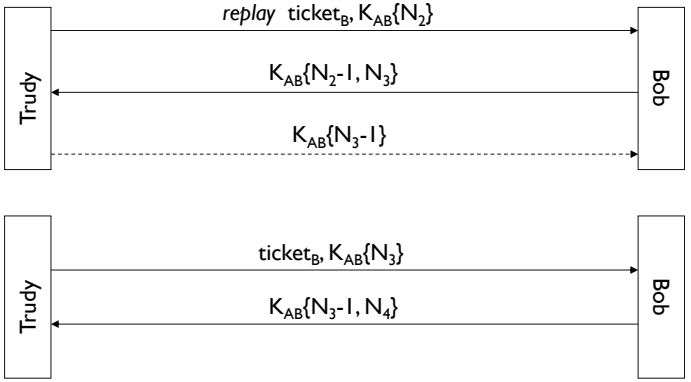


Needham-Schroeder Protocol

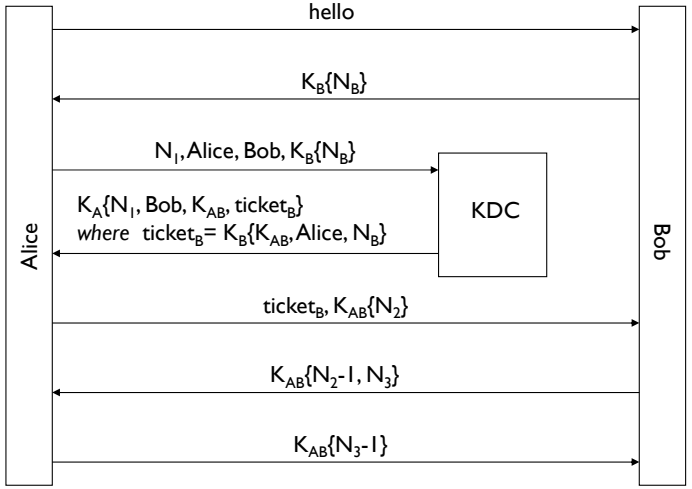
- Ticket is double-encrypted. (unnecessary)
- N_1: for authenticating KDC & freshness of K_{AB}.
- N_2, N_3: for key confirmation, mutual authentication
- Why are the challenges N_2, N_3 encrypted?
- Problem: Bob doesn't have freshness guarantee for K_{AB} (i.e., can't detect replays).

Replaying Tickets

- Messages should be integrity protected. Otherwise, cut-and-paste reflection attacks possible:



Expanded Needham-Schroeder Protocol



Protocol Performance Comparison

- Computational Complexity:
  (to minimize CPU time, power consumption)
  - Number of private-key operations
  - " " public-key "
  - " " bytes encrypted with secret key
  - " " bytes hashed
- Communication Complexity:
  - Number of message rounds
  - Bandwidth consumption