Authentication
Using Asymmetric Keys

Haipeng Dai

haipengdai@nju.edu.cn
313 CS Building
Department of Computer Science and Technology
Nanjing University
Authentication

- Problem: How do you prove to someone that you are whom you claim to be?
- Any system with access control must solve this problem.
- Goals:
  - 1. Mutual Authentication: each party authenticates itself to the other party.
  - 2. Key Establishment: establish a session key. This session key will be used to encrypt and decrypt messages between the two parties using symmetric key cryptography.
- Methods
  - Authentication with asymmetric keys
  - Authentication with symmetric keys
  - Human authentication
Authentication Using Asymmetric Keys

- Assumption
  - Everyone knows your public key
  - No one (except you) knows your private key

- Threat Model (i.e., what we assume attackers can do):
  - Message injection
    - Inject a new message into a channel, e.g., TCP poisoning attacks injecting TCP RESET.
  - Message modification
    - Modify a message in a channel
  - Message loss
    - Delete a message in a channel
  - Message replay
    - Replay an old message. The message is authentic, but old.
Here \( n \) denotes a nounce.

- An ideal nounce has two properties
  - Freshness (No repetition)
    - Each nounce is used at most once during any infinite execution of a protocol
  - Unpredictability
    - Knowing all nounces used in the past does not help to determine the next nounce to be used

- In practice, it is simulated using a large random number.

- Sometimes we only need the freshness property. In this case, we can use:
  - Increasing sequence number. The sender needs to remember the last sequence number. The numbers may increase randomly each time.
  - Real time, i.e., time stamp.
Question 1: Can we replace \( \{n\}_{PRA} \) by \( \{n\}_{PUA} \)?
- Answer: No. Everyone knows \( PU_A \) and can compute \( \{n\}_{PUA} \).

Question 2: What is wrong with this authentication protocol?
- Answer: No. An attacker can replay this message later to authenticate himself to Bob.
- How to fix this problem?
Now attackers cannot replay $\{n\}_{\text{PRA}}$.

Question: What is wrong with this authentication protocol?
- No session key is established.
- Authentication = mutual identity verification + session key establishment
- How to fix this problem?
Question: What is wrong with this authentication protocol?

Answer: Attackers can see $K_{ab}$ because they know Alice’s public key.

How to fix this problem?
Only Bob can decrypt \( {{\{n, K_{ab}\}}_{PRA}}_{PUB} \).


Question: What is wrong with this authentication protocol?

Answer: vulnerable to man-in-the-middle attacks:

- the attacker makes independent connections with the victims and relays/modifies/injects/deletes messages between them.
Man-in-the-middle (MITM) Attack on Version 4

- When Alice begins to talk to Robert, Robert starts to talk to Bob as Alice.

- Question: How to fix this problem?
  - Solution 1: use \(\{\{n, K_{ar}\}_{PRA}\}_{PUR}\) to replace \(\{\{n, K_{ar}\}_{PRA}\}_{PUR}\).
  - Solution 2: use \(\{\{n, R, K_{ar}\}_{PRA}\}_{PUR}\) to replace \(\{\{n, K_{ar}\}_{PRA}\}_{PUR}\).

- Principle: Encryption should be inside a signature, otherwise we need to include principal’s names.
Now only Alice and Bob can know the session key $K_{AB}$.

Question: What is wrong with this authentication protocol?

- Authentication = mutual identity verification + session key establishment
- Bob authenticates Alice, but Alice did not authenticate Bob.
- How to fix this problem?
Now, Bob authenticates Alice, Alice authenticates Bob, and a session key is established.

Question: which part of this protocol can be made more efficient?

Answer: replace \( \{m\}_{PRB} \) by \( \{m\}_{K_{ab}} \).

Note: an attacker can try to launch man-in-the-middle attack; however, it will not be successful because the attacker cannot learn \( K_{ab} \).
Final Version

Alice → A

n → {n, K_{ab}}_{\text{PUB}}_{\text{PRA}}, m → Bob

{m}_{K_{ab}} ← Bob

Alice → A

n → {n, K_{ab}}_{\text{PUB}}_{\text{PRA}}, m → Bob

{m}_{K_{ab}} ← Bob