Network Security

Authentication and Security Protocol Verification

Today

- Security Protocols
  - Authentication
  - Session Keys
  - Key Management
- Security Protocol Verification

The Principals in a Communication Session

- A: Alice, normally initiates the session
- B: Bob, usual recipient
- S: Sally, a trusted third party
- M: Mallory, a malicious intruder

Other Terminology

- *kab*: Symmetric key shared by Alice and Bob
- *{Msg}kab*: Message encrypted using key *kab*
- *PKa*: Alice's public key
- *PKa⁻¹*: Alice's private key

Authentication

- Corroborating the identity of an entity (person, process, etc.)

- You are what you _____________!

  - Have
  - Know
  - Look like

  - Feel like
  - See like
  - Are composed of

Basic Authentication

![Basic Authentication Dialog](Image)
Problems with Basic Authentication

- Passwords easy to intercept
- Passwords easy to guess
- Passwords easy to share
- No server authentication
  - Easy to fool client into sending password to malicious server
- One intercepted password gives eavesdropper access to many documents

A Password Protocol

Another Password Protocol

A Secure Password Protocol

Other Password Vulnerabilities

- The Main Password Vulnerability
  - Poor maintenance
    - Written down
    - Poorly chosen
      - Middle name, mother's maiden name, etc.
      - Too short
      - Repeated letters
    - Not changed

Other Password Vulnerabilities

- Guessing Attacks
  - # of 8 character possibilities
  - Weak (guessable) passwords
  - # of 4 digit possibilities?
  - Why are PINs not susceptible to guessing?
    - 3 try limit
    - Cameras at ATMs
  - Time intensive to try
    - If the interface is the ATM?
Other Password Vulnerabilities

- Dictionary attacks
  - Reduces the search space by factor of 10
  - Offline attack
  - Only available if there is no repeated try limit, or if the limit can be circumvented
  - Must be automated
- Counter Dictionary Attacks, Use:
  - Large password space
  - Numbers, special characters and capitalization
  - Acronyms (MhalL)
  - Keyboard pattern

Cryptographic Protocols

- Defined: Rules that detail the interaction between parties in a communication that are using cryptography for security.

- Simple Protocol: Using PGP, securely send a file to a colleague in the CS Department

  - To send:
    - Select the key of the intended recipient
    - Sign and encrypt the secret message
    - Select the email address of the intended recipient
    - Send the encrypted message via email

  - To receive:
    - Determine who the message came from
    - Select the key to match the originator
    - Decrypt the message
    - Check the signature

Potential Problems

- What if you get the same message twice?
- How will you know that you got the same message twice?
- How will you know that sent messages are received?

The Big Questions

- Does the protocol produce the expected results when run in a stable, friendly environment?
- Does the protocol produce the expected results when run in an unstable, hostile environment?
Simple Protocol

Goal: Two participants are in a session and both know that they are engaged.

A -> B: Na
B -> A: \{Na\}^k\{Nb\}^k
A -> B: Nb

Meaning of Messages

• Messages **reflect**:
  – Actions of principles
  – Data passed

• Messages **suggest**:
  – Local and global states of participants
  – Intentions of "sending" principles
  – Beliefs of "receiving" principles

Protocol Actions and Intents

Alice's Intent:
- Let Bob know that she wants to talk
- Give Bob a value they can use to authenticate the session.

Bob's Intent:
- Find out if it's really Alice by giving her a random value
- Let Alice know that he's ready to talk

Alice's Actions:
- Generate a random value
- Transmit that value to Bob

Bob's Actions:
- Decrypt her nonce and compare to original
- Decrypt Bob's value

Data Alice passes:

Data Bob passes:

Alice's Beliefs:
- Na hasn't been used before and cannot be guessed.
- She shares a key (k) with Bob

Bob's Beliefs:
- It's Alice that wants to talk
- Nb hasn't been used before and cannot be guessed.
- She has Bob's attention
Protocols Faults

- Protocols are prone to errors
- Protocols are vulnerable to attack
- Sophisticated intruders are ingenious
- Successful attacks put system integrity, security, etc. at risk.

Attack on the Simple Protocol

- Attack session
  1. A -> B: Na
  2. Mallory intercepts
  3. M -> A: Na
  4. A -> M: {Na}k{Na'}k
  5. M -> A: {Na}k{Na'}k
  6. A -> M: Na'

- Reference session

The Bird XOR Protocol

A => B: na
B => A: ({xor(na, B)}kab, {nb}kab)
A => B: nb

This prevents Mallory from using Alice as an oracle to encrypt na, as was done in the previous protocol, but...

Result of the Attack

- Alice believes that she is in a secure session with Bob, but Bob is not involved and is not aware that there is a session.

Big Deal! Maybe …

The Bird XOR Protocol

A => B: na
B => A: ({xor(na, B)}kab, {nb}kab)
A => B: nb

This value looks like a nonce to Alice

A => B: na

Mallory intercepts and returns Alice's message to Alice in the attack session. When you xor twice, the original value is returned.

So Mallory still used Alice as an Oracle.

Result of the Attack

- Alice believes that she is in a secure session with Bob, but Bob is not involved and is not aware that there is a session.
**Needham and Schroeder Paper**
1978

- Seminal paper on Cryptographic Protocols
- Identified several "canonical" protocols
- First to suggest verification of protocols
- Accidentally illustrated the first canonical protocol flaw found
  - The flaw was found two years later
  - The flaw reflects a very unlikely attack
  - Nonetheless, it illustrates how hard it is to produce reliable cryptographic protocols.

**Initial session**

A => S: (A, B, na)
S => A: {na, B, kab, {kab, A}kbs}kAs
A => B: {kab, A}kbs -- This message establishes the session key
B => A: {nb}kab
A => B: {nb-1}kab

The malicious intruder captures the key interchange messages and the ensuing session. Over time, he compromises kab, then initiates a new session with Bob.

**Result of the Attack**

- Alice believes that she is in a secure session with Bob, but she is actually in a secure session with Mallory, i.e. Mallory has masqueraded as Bob.

**Tatebayashi, Matsuzaki, and Newman**

**Public Key Protocol**

A => S: (A, B, {kas}PKs);
S => B: (A);
B => S: {kab}PKs;
S => A: {kab}kAs;
A => B: {important_data}kab;
Attack on the Tatebayashi, Matsuzaki, and Newman Public Key Protocol

\[\begin{align*}
A &\Rightarrow S: (A, B, \{\text{kas}\}PKs); \\
S &\Rightarrow B: (A); \text{ [Intercepted by Mallory]} \\
M &\Rightarrow S: \{\text{kam}\}PKs; \\
S &\Rightarrow A: \{\text{kam}\}kas; \\
A &\Rightarrow B: \{\text{important_data}\}\text{kam}; \text{ [Mallory intercepts]} \\
\end{align*}\]

Result of the Attack

- Alice believes that she is in a secure session with Bob, but she is actually in a secure session with Mallory, i.e. Mallory has masqueraded as Bob.
- Bob is left in the lurch (session started, but never finished)

CCITT X.509 Public Key Protocol

\[\begin{align*}
\text{Ta}, \text{Tb} &\text{ are timestamps} \\
\text{Xa}, \text{Xb} &\text{ are data that can be public} \\
\text{na}, \text{nb} &\text{ are nonces} \\
\text{Ya}, \text{Yb} &\text{ are private data} \\
\text{A} &\Rightarrow B: (A, \{\text{Ta}, \text{na}, B, \text{Xa}, \{\text{Ya}\}PKb\}PKa^{-1}) \\
\quad \quad \text{Privacy protected} \\
\quad \quad \text{Integrity protected} \\
\text{B} &\Rightarrow A: B, \{\text{Tb}, \text{nb}, A, \text{na}, \text{Xb}, \{\text{Yb}\}PKa\}PKb^{-1} \\
\text{A} &\Rightarrow B: \{\text{nb}\}PKa^{-1} \\
\end{align*}\]

Attack on the CCITT X.509 Public Key Protocol

\[\begin{align*}
A &\Rightarrow B: (A, \{\text{Ta}, \text{na}, B, \text{Xa}, \{\text{Ya}\}PKb\}PKa^{-1}) \text{ [Captured by Mallory]} \\
\text{New session (this is the attack session)} \\
M &\Rightarrow B: (A, \{\text{Ta}, \text{na}, B, \text{Xa}, \{\text{Ya}\}PKb\}PKa^{-1}) \text{ [Mallory acting as Alice]} \\
B &\Rightarrow A: B, \{\text{Tb}, \text{nb}, A, \text{na}, \text{Xb}, \{\text{Yb}\}PKa\}PKb^{-1}\text{[Intercepted]} \\
\text{Third session (this is also parallel session)} \\
A &\Rightarrow M: (A, \{\text{Ta}', \text{na}', M, \text{Xa}', \{\text{Ya}'\}PKm\}PKa^{-1}) \\
M &\Rightarrow A: M, \{\text{Tm}, \text{nb}, M, \text{na}', \text{Xm}, \{\text{Ym}\}PKa\}PKm^{-1} \\
A &\Rightarrow M: \{\text{nb}\}PKa^{-1} \\
M &\Rightarrow B: \{\text{nb}\}PKa^{-1} \\
\end{align*}\]

Result of the Attack

- Bob believes Alice recently sent Xa & Ya\(^1\)
- Bob believes that Alice has Xb and Yb and that only Alice knows Yb.
- Alice does not have Xb or Yb. Mallory has Xb (but not Yb).

\(^1\)Xa and Ya were components of the first message that was captured by Mallory. If that message was allowed to go to Bob, or if that session was suspended but Alice restarted it and resent Xa and Ya, then this is the second time that Bob received these exact messages.

The KP Protocol

\[\begin{align*}
A &\Rightarrow B: (A, \text{na}, B) \\
B &\Rightarrow S: (A, \text{na}, B, \text{nb}) \\
S &\Rightarrow B: \{\text{kab}, A, \text{nb}, \{A, \text{B, na, kab}\}kbs\}kbs \\
B &\Rightarrow A: \{A, \text{B, na, kab}\}kas, \{\text{na, nb}, B\}kab \\
A &\Rightarrow B: \{\text{nb}', A\}kab \\
\end{align*}\]
Attack on The KP Protocol

A1. A => B: (A, na, B) [ Intercepted by Mallory]

P1. M => A: (B, nm, A) [Mallory acting as Bob]

P2. A => S: (B, nm, A, na')

A1b. M => B: (A, na', B) [Mallory forwards Alice's msg, substituting the later nonce]

A2. B => S: (A, na', B, nb)

A3. S => B: (kab, A, nb, (A, B, na', kab)kas)kbs

A4. B => A: (A, B, na', kab)kas, (na', nb', B)kab

P3. M => A: (A, B, na', kab)kas [Mallory acting as KDC]

P4. A => B: kab, (na', nb', B)kab [ Intercepted by Mallory]

A5. M => B: (nb', A)kab

---

Result of the Attack

- Alice believes that she is in a secure session with Bob
- Bob believes he is in a secure session with Alice.
- Mallory has the session key for Bob and Alice's conversation.

---

A Taxonomy of Attacks

- Replay
- Oracle
- Parallel session
- Interleaving
- Man in the Middle
## Categories of Design Principles for Security Protocols

1. Explicit communication (1)  
2. Appropriate action (2)  
3. Naming (3)  
4. Encryption (4)  
5. Signing encrypted data (5)  
6. Timeliness (6-9)  
7. Recognizing messages (10)  
8. Trust (11)  


## Design Principals for Security Protocols

1. Messages should be self descriptive  
2. Conditions for execution should be clear  
   a. Be clear on the meaning of encryption  
   b. Be clear on how timeliness of messages is proved  
3. Cryptographically Bind the originator's ID to the message if the ID is important to the message's meaning  
4. Be clear on why encryption is done  

## Design Principals for Security Protocols

5. Encrypted vs signed  
   a. Don't infer that an originator knows the contents of an encrypted message  
   b. Always infer that an originator knows the contents of a signed, then encrypted message  
6. Be clear what properties you are assuming for nonces  
7. Predictable values can be used for freshness through effective challenge/response.

## Design Principals for Security Protocols

8. Effective timestamps require trusted clock synchronization  
9. A recently used key may still be old  
10. Cryptographically bind:  
   a. Related elements within messages  
   b. Messages to a session  
   c. Messages to their place in a session  
11. Trust relationships should be explicit, and based on judgment & policy rather than logic.

## Review

- Cryptographic Protocols  
  - Symmetric Key  
  - Public Key  
- Protocol Goals  
- Attacks and Flaws in Protocols  
- Analyzing Protocols  
- Principles for Protocol Design