Authentication Handshakes

• Secure communication almost always includes an initial authentication handshake.
  – Authenticate each other
  – Establish session keys
  – *This process is not trivial; flaws in this process undermines secure communication*

• This topic is about typical flaws
Authentication with Shared Secret

- **Weaknesses**
  - Authentication is not mutual; Trudy can convince Alice that she is Bob
  - Trudy can hijack the conversation after the initial exchange
  - If the shared key is derived from a password, Trudy can mount an off-line password guessing attack
  - Trudy may compromise Bob’s database and later impersonate Alice

Authentication with Shared Secret (Cont’d)

- **A variation**
  - Requires reversible cryptography
  - Other variations are possible
- **Weaknesses**
  - All the previous weaknesses remain
  - Trudy doesn’t have to see R to mount off-line password guessing if R has certain patterns (e.g., concatenated with a timestamp)
    - Trudy sends a message to Bob, pretending to be Alice
Authentication with Public Key

- Bob’s database is less risky
- Weaknesses
  - Authentication is not mutual; Trudy can convince Alice that she is Bob
  - Trudy can hijack the conversation after the initial exchange
  - Trudy can trick Alice into signing something
    - Use different private key for authentication

Authentication with Public Key (Cont’d)

A variation
Mutual Authentication

![Diagram of Mutual Authentication]

Optimize

Mutual Authentication (Cont’d)

• Reflection attack

![Diagram of Reflection Attack]
Reflection Attacks (Con’td)

• Lesson: Don’t have Alice and Bob do exactly the same thing
  – Different keys
    • Totally different keys
    • $K_{Alice-Bob} = K_{Bob-Alice} + 1$
  – Different Challenges
  – The initiator should be the first to prove its identity
    • Assumption: initiator is more likely to be the bad guy

Mutual Authentication (Cont’d)

• Password guessing

\begin{align*}
\text{Alice} & \quad \text{I’m Alice, } R_2 \\
& \quad R_1, f(K_{Alice-Bob}, R_2) \\
& \quad f(K_{Alice-Bob}, R_1) \\
\text{Bob} & \\
\end{align*}

\begin{align*}
\text{Alice} & \quad \text{I’m Alice} \\
& \quad R_1 \\
& \quad f(K_{Alice-Bob}, R_1), R_2 \\
& \quad f(K_{Alice-Bob}, R_2) \\
\text{Bob} & \\
\end{align*}

Countermeasure
Mutual Authentication (Cont’d)

• Public keys
  – Authentication of public keys is a critical issue

![Diagram showing mutual authentication with public keys]

Mutual Authentication (Cont’d)

• Mutual authentication with timestamps
  – Require synchronized clocks
  – Alice and Bob have to encrypt different timestamps

![Diagram showing mutual authentication with timestamps]
Integrity/Encryption for Data

- Communication after mutual authentication should be cryptographically protected as well
  - Require a session key established during mutual authentication

Establishment of Session Keys

- Secret key based authentication
  - Assume the following authentication happened.
    - Can we use $K_{Alice-Bob\{R\}}$ as the session key?
    - Can we use $K_{Alice-Bob\{R+1\}}$ as the session key?
    - In general, modify $K_{Alice-Bob}$ and encrypt $R$. Use the result as the session key.

\[ \begin{array}{c}
\text{Alice} \quad I'm \ Alice \\
\quad \downarrow R \\
\quad K_{Alice-Bob\{R\}} \\
\end{array} \quad \begin{array}{c}
\quad \uparrow R \\
\text{Bob} \quad K_{Alice-Bob\{R\}} \\
\end{array} \]
Establishment of Session Keys (Cont’d)

- Two-way public key based authentication
  - Alice chooses a random number R, encrypts it with Bob’s public key
    - Trudy may hijack the conversation
  - Alice encrypts and signs R
    - Trudy may save all the traffic, and decrypt all the encrypted traffic when she is able to compromise Bob
    - Less severe threat

Two-Way Public Key Based Authentication (Cont’d)

- A better approach
  - Alice chooses and encrypts $R_1$ with Bob’s public key
  - Bob chooses and encrypts $R_2$ with Alice’s public key
  - Session key is $R_1 \oplus R_2$
  - Trudy will have to compromise both Alice and Bob

- An even better approach
  - Alice and Bob establish the session key with Diffie-Hellman key exchange
  - Alice and Bob signs the quantity they send
  - Trudy can’t learn anything about the session key even if she compromises both Alice and Bob
Establishment of Session Keys (Cont’d)

- One-way public key based authentication
  - It’s only necessary to authenticate the service
    - Example: SSL
  - Encrypt R with Bob’s public key
  - Diffie-Hellman key exchange
    - Bob signs the D-H public key

Mediated Authentication (With KDC)

KDC operation (in principle)

- Some concerns
  - Trudy may claim to be Alice and talk to KDC
    - Trudy cannot get anything useful
  - Messages encrypted by Alice may get to Bob before KDC’s message
  - It may be difficult for KDC to connect to Bob
Mediated Authentication (With KDC)

KDC operation (in practice)

- Must be followed by a mutual authentication exchange
  - To confirm that Alice and Bob have the same key

Needham-Schroeder Protocol

- Classic protocol for authentication with KDC
  - Many others have been modeled after it (e.g., Kerberos)
- Nonce: A number that is used only once
  - Deal with replay attacks
Needham-Schroeder Protocol (Cont’d)

• A vulnerability
  – When Trudy gets a previous key used by Alice, Trudy may reuse a previous ticket issued to Bob for Alice
  – Essential reason
    • The ticket to Bob stays valid even if Alice changes her key

Expanded Needham-Schroeder Protocol

I want to talk to you

K_{Bob}\{N_B\}

N_1, Alice wants Bob, K_{Bob}\{N_B\}

Generate K_{AB}; extract N_B

K_{Alice}\{N_1\}, “Bob”, K_{AB}, ticket to Bob, where ticket to Bob = K_{Bob}\{K_{AB}\cdot Alice, N_B\}

K_{AB}\{N_2\}

ticket to Bob, K_{AB}\{N_2\}

K_{AB}\{N_2-1, N_3\}

K_{AB}\{N_3-1\}

• The additional two messages assure Bob that the initiator has talked to KDC since Bob generates N_B

I want to talk to you

K_{Bob}\{N_B\}

N_1, Alice wants Bob, K_{Bob}\{N_B\}

Generate K_{AB}; extract N_B

K_{Alice}\{N_1\}, “Bob”, K_{AB}, ticket to Bob, where ticket to Bob = K_{Bob}\{K_{AB}\cdot Alice, N_B\}

K_{AB}\{N_2\}

ticket to Bob, K_{AB}\{N_2\}

K_{AB}\{N_2-1, N_3\}

K_{AB}\{N_3-1\}
Otway-Rees Protocol

- Only has five messages
- KDC checks if $N_C$ matches in both cipher-texts
  - Make sure that Bob is really Bob