Man-in-the-Middle Attack

So far: \[ \text{Alice} \leftrightarrow \text{Bob} \]

What if Eve can do more than just listening?

Attacking Diffie-Hellman Key Exchange.

1. Eve chooses a random exponent \( z \).

\[
\text{public} \quad g^x \quad \text{Eve} \quad g^z \quad \text{Bob} \quad (C, g) \text{ is public}
\]

\[
\text{private} \quad x \quad \text{Eve} \quad z \quad \text{Eve} \quad y \quad \text{(Everything is under \( \mod p \).)}
\]

After the key exchange, Alice and Bob start to communicate using the key they thought they agreed on.

2. Alice and Eve agree on \( K_{AE} = g^{xz} \mod p \).

3. Bob and Eve agree on \( K_{BE} = g^{yz} \mod p \).

4. When Alice encrypts and sends a message \( m \), Eve intercepts the ciphertext \( \text{enc}(m, K_{AE}) \), decrypts it using \( K_{AE} \), encrypts it using \( K_{BE} \), and sends \( \text{enc}(m, K_{AE}) \) to Bob.

How to fix this?

Have Alice and Bob sign every message they send.

\[
\text{Alice} \quad \rightarrow \quad \text{Bob} \quad (g^x, \text{sig}_A(g^x))
\]

Eve intercepts this message, and wants to send \( (g^z, \text{sig}_A(g^z)) \) to Bob instead.

\[
\text{Eve cannot forge Alice's signature on } g^z.
\]

Q: How does Bob verify \( (g^x, \text{sig}_A(g^x)) \)?

A: With Alice's public key.

How can Bob make sure Alice's public key is indeed Alice's public key? (See next page)
Public key Infrastructure (PKI).

Intuition: Have a trusted third party that maintains all public keys.

When Bob wants to know/verify Alice's public key, Bob can ask Trent for Alice’s public key.

Moreover, Trent can sign a message

\[ \text{Sign}_{sk_{Trent}}(”\text{Alice’s public key is } pk_{Alice}”) \]

This is essentially a digital certificate.

This way, Bob only needs to know Trent's public key and Bob maintains certificates (published by Trent) for other people's public keys.

Q: What if Trent is hacked or corrupted?

A: Multiple CAs.

Certificate Authority (CA): an entity that issues digital certificates.

(e.g., a company whose business is to certify public keys, a government agency, etc.)

Hierarchy: a root CA certifies multiple intermediate CAs.

Example: Alice’s CA cannot verify Bob’s key directly.

Alice’s CA trusts Charlie’s CA,

Charlie’s CA trusts Bob’s CA.

Bob’s CA can verify Bob’s key.

Standard for Certificates

<table>
<thead>
<tr>
<th>eg, X.509</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Issuer:</strong> DigiCert ← name of CA</td>
<td></td>
</tr>
<tr>
<td><strong>Serial number:</strong></td>
<td>...</td>
</tr>
<tr>
<td><strong>Subject:</strong> Alice</td>
<td></td>
</tr>
</tbody>
</table>
| Alice’s organization and other info.
| **Subject’s public key:** Alice’s pk ← a large number |
| **Valid from:** YY-MM-DD hh:mm:ss |
| **Valid until:** ... |
| CA’s signature for everything above |