Introduction to Modern Cryptography

11th lecture:
Digital Signatures
Public-Key Infrastructures
**11th lecture (today):**
- Digital Signatures
- Public-Key Infrastructures

**last time:**
- RSA encryption
- CCA security

<table>
<thead>
<tr>
<th>Confidentiality</th>
<th>Secret Key</th>
<th>Public Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>Private-key encryption</td>
<td></td>
<td>Public-key encryption</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Authentication</th>
<th>Secret Key</th>
<th>Public Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>Message authentication codes (MAC)</td>
<td></td>
<td>Digital signatures</td>
</tr>
</tbody>
</table>
Certificates & Public-Key Infrastructures (PKI)

- use digital signatures to securely distribute public keys!

- a **digital certificate** is a signature, binding some entity to some public key

- For instance:
  \[
  \text{cert}_{C\rightarrow B} = \text{Sign}_{sk_C} \left( \text{“Bob’s key is pk}_B\text{”} \right)
  \]

- Standard used on the internet: **X.509**
Example of PKI: Certificate Authority (CA)

- completely trusted by everybody
- every user needs to know the CA’s public key $\text{pk}_{\text{CA}}$
- ship it bundled with software (e.g. in browsers)

- Single point of failure / trust
- Use multiple CAs instead
Certificate Chains and Delegation

- $\text{cert}_{B \rightarrow A} = \text{Sign}_{\text{sk}_B}$ ("Alice’s key is $\text{pk}_A$")

- If Alice wants to communicate to Dave who knows and trusts Charlie, she sends $\text{pk}_A, \text{cert}_{B \rightarrow A}, \text{pk}_B, \text{cert}_{C \rightarrow B}$

- "stronger trust": Dave learns $\text{pk}_B$ and needs to trust Bob to issue other certificates

image credit: http://gnutls.org/openpgp.html
Web Certificates

1. A Certification Authority distributes its CA root certificate via browser vendors to browsers. These root certificates reside in a "trust list" on the user's PC.

2. A company that wants its website to be secured, purchases a website certificate at the CA. This certificate is signed by the CA and guarantees the identity of the website to the users.

3. When a user wants to visit the secure website, the web browser will first ask the web server for the certificate. If its signature can be verified with the certificate of a CA in the trust list, the website certificate will be accepted. Then the website will be loaded into the browser, and all traffic between the browser and the website will be secured by using encryption.

credit: http://www.win.tue.nl/hashclash/rogue-ca/
Web of Trust (as in OpenPGP)

- every user decides individually whom to trust
- public keys can be signed by different people, e.g. at key-signing parties

- Dave implicitly trusts Bob’s \( \text{pk} \)
- Charlie signed Bob’s key but does not trust him

An example of the web of trust model

Image credit: http://gnutls.org/openpgp.html
Invalidating Certificates

- **insert expiry dates:**
  \[ \text{cert}_{C \to B} = \text{Sign} \ sk_C \left( \text{“Bob’s key is pk}_B \text{” , date } \right) \]
- when date has passed, get a new certificate

- **revocation:** (include a serial number with all certs)
  \[ \text{cert}_{C \to B} = \text{Sign} \ sk_C \left( \text{“Bob’s key is pk}_B \text{” , serial-# } \right) \]
- CA stores a list of (Bob, pk_B, serial-#)
- If skB is stolen, Bob alerts the CA
- CA creates **certificate revocation list (CRL)** with all serial-#s of revoked certificates, signs the list with date and publishes it
- verifying the certificate now requires checking if the serial-# has not been revoked