Encryption Methods

‣ Cæsar (substitution) cipher
  - ... frequency analysis

‣ “Unbreakable” cipher

‣ DES - Data Encryption Standard
  - 1977, symmetric key, 56-bit key, 64-bit data blocks

‣ AES - Advanced Encryption Standard
  - 1998, symmetric key, 128, 192, and 256-bit keys, 128-bit data blocks
Encryption Methods

‣ **Diffie-Hellman** key exchange
  - a method allows two parties that have no prior knowledge of each other to jointly establish a shared secret key over an insecure communications channel.

‣ **RSA** - Rivest, Shamir, and Adleman
  - 1978, public/private key algorithm, 1,024 to 4,096-bit keys (typically)
Authentication

Basic idea:
- use public/private key cryptography
- only possessor of private key could have encrypted something that decrypts using its public key

Problem: Replay Attack
- solution: use of a nonce

Still unaddressed: We need a trusted way to obtain someone’s public key
Message Integrity

- Basic idea:
  - use public/private key cryptography
  - send encrypted (with senders private key) hash of a message
  - if hash of the received message agrees with decrypted received hash, it is assumed that the message was not altered in transit

- Problems:
  - need a cryptographic hash function
  - need a public key distribution method
Message Integrity

\[ m = h(M) \]

\[ m' = h(M) \]

\[ \text{Ekpriv}_A(m) \]

\[ \text{Dkpub}_A(c) \]

If \( m == m' \), msg integrity is verified.

message digest encrypted with senders private key
Cryptographic Hash Functions

- **MD5** - Message Digest Algorithm
  - 1992, R. Rivest, digest size 128 bits

- **SHA-1** - Secure Hash Algorithm
  - 1995, NSA, digest size 160 bits
  - SHA-2, SHA-3 competition at NIST
Certificates

- Solving the public key distribution problem
- Trust (having somebody’s public key) is transitive
  - A trusts C and B trusts C
    → A can establish trust with B

- Where to start?
  - who to trust
  - how is the initial trust established

- Solution: Certificate Authority (CA)
Certificates

- **Goal**: A wants to prove its identity to B
- **Given**: A and B trust CA and both have CA’s public key
- **Broad approach**: Public key certificate
  - A’s public key encrypted with CA’s private key (ensures integrity of the key)
  - … plus additional information
- **Use**: A presents its certificate when initiating communication with B
Certificates - Questions

- **Man in the Middle Attack**: How does B know that it is A’s certificate and not an impostor’s one?
  - Include A’s human-readable identification

- **Replay Attack**: Attacker overhears/requests A certificate and presents it when pretending to be A
  - Use nonce encrypted with A’s public key during communication

- **Compromised certificate**: Either A’s or CA’s private keys are compromised
  - Limited validity and certificate revocation
Certificates - Issuance

- A gets a certificate from a CA
  - A generates public/private key pair
  - A generates Certificata Signing Request (CSR)
  - CA (hopefully) makes sure that it interacts with A and not an impostor
  - CA encrypts the certificate (public key + A’s identification + ...) with its own private key
  - certificate is delivered to A (can be done securely since A has CA’s public key)
Certificates - Use

- B authenticates A
  - B requests a certificate from A together with a nonce
  - A sends back the certificate and the nonce encrypted with A’s private key
  - B decrypts the certificate with CA’s public key and verifies that it was issued to A
  - B decrypts the nonce using A’s public key and verifies that the value matches the value sent