Digital Signatures

CSCI 454/554

- have looked at message authentication
  - but does not address issues of lack of trust
- digital signatures provide the ability to:
  - verify author, date & time of signature
  - authenticate message contents
  - be verified by third parties to resolve disputes
- hence include authentication function with additional capabilities
Digital Signature Properties

- must use information unique to sender
  - to prevent both forgery and denial
- must be relatively easy to produce
- must be relatively easy to recognize & verify
- be practical save digital signature in storage

Direct Digital Signatures

- involve only sender & receiver
- assumed receiver has sender’s public-key
- digital signature made by sender hashing entire message then sign with its private-key
- can encrypt using receivers public-key
- important that sign first then encrypt message & signature
- security depends on sender’s private-key
**Arbitrated Digital Signatures**

- involves use of arbiter A
  - validates any signed message
  - then dated and sent to recipient
- requires suitable level of trust in arbiter
- can be implemented with either private or public-key algorithms
- arbiter may or may not see message

**Digital Signature Standard (DSS)**

- uses the SHA hash algorithm
- designed by NIST & NSA in early 90's
- DSS is the standard, DSA is the algorithm
  - used for digital signature only
- a variant on ElGamal and Schnorr schemes
- creates a 320 bit signature, but with 512-1024 bit security
- security depends on difficulty of computing discrete logarithms
DSA Key Generation

- have shared global public key values \((p,q,g)\):
  - a large prime \(p\)
    - \(P\) is 512 to 1024 bits long
  - choose \(q\), a 160 bit prime factor of \(p-1\)
  - choose \(g = h^{(p-1)/q}\)
    - where \(h < p-1\), \(h^{(p-1)/q} \mod p > 1\)
- users choose private & compute public key:
  - choose \(x < q\) as private key
  - compute \(y = g^x \mod p\) as public key

(a) RSA Approach

(b) DSS Approach

\[ H(M) \]
**DSA Signature Creation**

- **to sign** a message $M$ the sender:
  - generates a random signature key $k, \ k<q$
  - $k$ must be random, be destroyed after use, and never be reused
- **then computes** signature pair:
  - $r = (g^k \pmod{p}) \pmod{q}$
  - $s = ((k^{-1} \cdot \text{SHA}(M) + x \cdot r) \pmod{q})$
- **sends** signature $(r,s)$ with message $M$

**DSA Signature Verification**

- **having received** $M$ & signature $(r,s)$
- **to verify** a signature, recipient computes:
  - $w = s^{-1} \pmod{q}$
  - $u1 = (\text{SHA}(M) \cdot w) \pmod{q}$
  - $u2 = (r \cdot w) \pmod{q}$
  - $v = (g^{u1} \cdot y^{u2} \pmod{p}) \pmod{q}$
- **if** $v=r$ then signature is verified
- **see** book web site for details of proof why
Summary

- have considered:
  - digital signatures
  - digital signature standard