

# Introduction to Modern Cryptography

## Lecture 10

### Digital Signatures

# Handwritten Signatures

- Relate an individual, through a handwritten signature, to a document.
- Signature can be **verified** against a prior authenticated one, signed in person.
- Should be **hard to forge**.
- Are **legally binding** (convince a third party, e.g. a judge).

# Digital Signatures: Desired Properties

- Relate an individual, through a **digital string**, to a document.
- Signature should be easy to **verify**.
- Should be **hard to forge**.
- Are **legally binding** (convince a third party, e.g. a judge).

# Diffie and Hellman (76)

## “New Directions in Cryptography”

Let  $E_A$  be Alice's public encryption key,  
and let  $D_A$  be Alice's private decryption key.

- To sign the message  $M$ , Alice computes the string  $y = D_A(M)$  and sends  $M, y$  to Bob.
- To verify this is indeed Alice's signature, Bob computes the string  $x = E_A(y)$  and checks  $x = M$ .

Intuition: Only Alice can compute  $y = D_A(M)$ , thus forgery should be computationally infeasible.

# Problems with “Pure” DH Paradigm

- Easy to **forge** signatures of random messages even without holding  $D_A$ :

**Bob** picks  $R$  arbitrarily, computes  $S = E_A(R)$ .

Then the pair  $(S, R)$  is a valid signature of **Alice** on the “message”  $S$ .

- Therefore the scheme is subject to **existential forgery**.
- “So what” ?

# Problems with “Pure” DH Paradigm

- Consider specifically RSA. Being multiplicative, we have (products mod  $N$ )

$$D_A(M_1M_2) = D_A(M_1) D_A(M_2).$$

- If  $M_2$  = “I OWE BOB \$20” and  $M_1$  = “100” then under certain encoding of letters we could get  $M_1M_2$  = “I OWE BOB \$2000” ...

# Standard Solution: Hash First

Let  $E_A$  be Alice's public encryption key,  
and let  $D_A$  be Alice's private decryption key.

- To sign the message  $M$ , Alice first computes the strings  $y=H(M)$  and  $z=D_A(y)$ . Sends  $M, z$  to Bob.
- To verify this is indeed Alice's signature, Bob computes the string  $y=E_A(z)$  and checks  $y=H(M)$ .
- The function  $H$  should be collision resistant, so that cannot find another  $M'$  with  $H(M)=H(M')$ .

# General Structure: Signature Schemes

- **Generation** of private and public keys (randomized).
- **Signing** (either deterministic or randomized)
- **Verification** (accept/reject) - usually deterministic.

# Schemes Used in Practice

- RSA
- El-Gamal *Signature* Scheme (85)
- The *DSS* (digital signature standard, adopted by NIST in 94 is based on a modification of El-Gamal signature.

# El-Gamal Signature Scheme

## Generation

- Pick a prime  $p$  of length 1024 bits such that DL in  $Z_p^*$  is hard.
- Let  $g$  be a generator of  $Z_p^*$ .
- Pick  $x$  in  $[2, p-2]$  at random.
- Compute  $y = g^x \bmod p$ .
- Public key:  $p, g, y$ .
- Private key:  $x$ .

# El-Gamal Signature Scheme

## Signing $M$

- Hash: Let  $m=H(M)$ .
- Pick  $k$  in  $[1,p-2]$  relatively prime to  $p-1$  at random.
- Compute  $r=g^k \bmod p$ .
- Compute  $s=(m-rx)k^{-1} \bmod (p-1)$  (\*\*\*)
- Output  $r$  and  $s$ .

# El-Gamal Signature Scheme

Verify  $M, r, s, PK$

- Compute  $m = H(M)$ .
- Accept if  $0 < r < p$  and  $y^r r^s = g^m \pmod{p}$ .  
else reject.
- What's going on?

By (\*\*\*)  $s = (m - rx)k^{-1} \pmod{p-1}$ , so  
 $sk + rx = m$ . Now  $r = g^k$  so  $r^s = g^{ks}$ , and  $y = g^x$   
so  $y^r = g^{rx}$ , implying  $y^r r^s = g^m$ .

## Signatures vs. MACs

Suppose parties  $A$  and  $B$  share the secret key  $K$ . Then  $M, MAC_K(M)$  convinces  $A$  that indeed  $M$  originated with  $B$ . But in case of dispute  $A$  cannot convince a judge that  $M, MAC_K(M)$  was sent by  $B$ , since  $A$  could generate it herself.