Introduction to Cryptography
CS 355

Lecture 30

Digital Signatures

Announcements

- Wednesday’s lecture cancelled

- Friday will be guest lecture by Prof. Cristina Nita-Rotaru on Identification Schemes
  - This topic will be covered in the final exam

Lecture Outline

- Digital signatures
- Security requirements
- The RSA signatures
- The El Gamal signatures
- The DSA signatures

Where Does This Fit?

	Secret Key Setting	Public Key Setting
Secrecy / Confidentiality	Stream cipher Block cipher + encryption modes	Public key encryption: RSA, El Gamal, etc.
Authenticity / Integrity	MAC	Digital Signatures

Digital Signatures: The Problem

- Consider the real-life example where a person pays by credit card and signs a bill; the seller verifies that the signature on the bill is the same with the signature on the card
- Contracts, they are valid if they are signed.
- Can we have a similar service in the electronic world?

Digital Signatures

- Digital Signature: a data string which associates a message with some originating entity.
- Digital Signature Scheme:
  - a signing algorithm: takes a message and a (private) signing key, outputs a signature
  - a verification algorithm: takes a (public) key verification key, a message, and a signature
- Provides:
  - Authentication
  - Data integrity
  - Non-Repudiation (MAC does not provide this.)

Adversarial Goals

- **Total break**: adversary is able to find the secret for signing, so he can forge then any signature on any message.
- **Selective forgery**: adversary is able to create valid signatures on a message chosen by someone else, with a significant probability.
- **Existential forgery**: adversary can create a pair (message, signature), s.t. the signature of the message is valid.

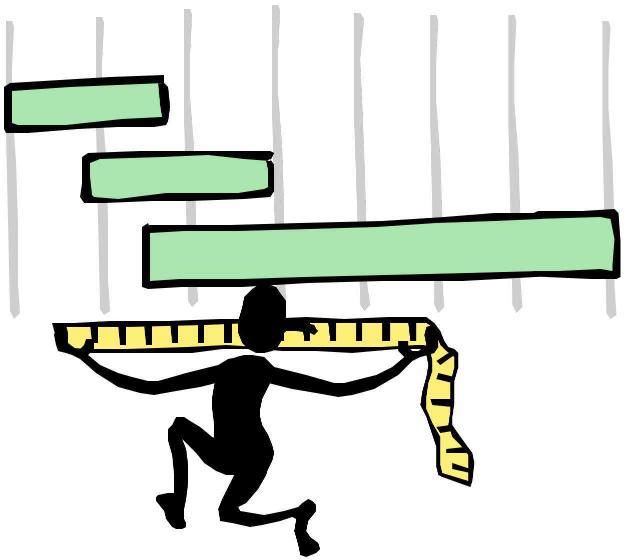
- A signature scheme can not be perfectly secure; it can only be computationally secure.
- Given enough time and adversary can always forge Alice's signature on any message.

Attack Models for Digital Signatures

- **Key-only attack**: Adversary knows only the verification function (which is supposed to be public).
- **Known message attack**: Adversary knows a list of messages previously signed by Alice.
- **Chosen message attack**: Adversary can choose what messages wants Alice to sign, and he knows both the messages and the corresponding signatures.

Digital Signatures and Hash

- Very often digital signatures are used with hash functions, hash of a message is signed, instead of the message.
- Hash function must be:
  - Pre-image resistant
  - Weak collision resistant
  - Strong collision resistant



RSA Signatures

Key generation (as in RSA encryption):

- Select 2 large prime numbers of about the same size, p and q
- Compute $n = pq$, and $\Phi = (q - 1)(p - 1)$
- Select a random integer e, $1 < e < \Phi$, s.t. $\gcd(e, \Phi) = 1$
- Compute d, $1 < d < \Phi$ s.t. $ed \equiv 1 \bmod \Phi$

Public key: (e, n)

Secret key: d,

RSA Signatures (cont.)

Signing message M

- Verify $0 < M < n$
- Compute $S = M^d \bmod n$

Verifying signature S

- Use public key (e, n)
- Compute $S^e \bmod n = (M^d \bmod n)^e \bmod n = M$

Note: in practice, a hash of the message is signed and not the message itself.

RSA Signatures (cont.)

Example of forging

- Attack based on the multiplicative property of property of RSA.

  $y_1 = sig_K(x_1)$

  $y_2 = sig_K(x_2)$, then

  $ver_K(x_1 x_2 \text{ mod } n, y_1 y_2 \text{ mod } n) = \text{true}$

- So adversary can create the valid signature

  $y_1 y_2$ mod n on the message $x_1 x_2$ mod n
- This is an existential forgery using a known message attack.

El Gamal Signature

Key Generation (as in ElGamal encryption)

- Generate a large random prime *p* such that DLP is infeasible in Z_p and a generator g of the multiplicative group Z_p of the integers modulo *p*
- Select a random integer *a*, $1 \leq a \leq p-2$, and compute

$$y = g^a \bmod p$$

- Public key is (p; g ; $\beta = g^a$)
- Private key is *a*.

- Recommended sizes: 1024 bits for p and 160 bits for a.

ElGamal Signature (cont.)

Signing message M

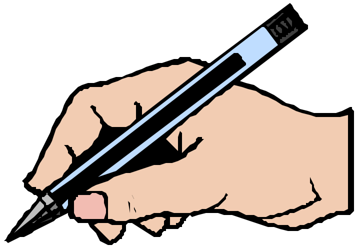
- Select random k, $1 \leq k \leq p\text{-}1$, $k \in Z_{p-1}^{*}$
- Compute

  $$r = g^k \bmod p$$

  $$s = k^{-1}(h(M) - ar) \bmod (p-1)$$

- Signature is: (r,s)
- Size of signature is double size of p

NOTE: h is a hash function



ElGamal Signature (cont.)

Signature is: (r, s)

$r = g^k \bmod p$

$s = k^{-1}(h(M) - ar) \bmod (p-1)$

Verification

- Verify that r is in Z_{p-1}^* : $1 \leq r \leq p-1$
- Compute

  $v_1 = \beta^r r^s \bmod p$

  $v_2 = g^{h(M)} \bmod p$
- Accept iff $v_1=v_2$

QuickTime™ and a TIFF (Uncompressed) decompressor are needed to see this picture.

ElGamal Signature (cont.)

Security of ElGamal signature

- Weaker than DLP
- k must be unique for each message signed
- Hash function h must be used, otherwise easy for an existential forgery attack
  - without h, a signature on $M \in Z_p$, is (r,s) s.t. $\beta^r r^s = g^M \bmod p$
  - choose u,v s.t. gcd(v,p-1)=1, then let $r=g^u\beta^v \bmod p=g^{u+av} \bmod p$, and let $s=-rv^{-1} \bmod (p-1)$
  - then $\beta^r r^s = g^{ar} (g^{u+av})^s = g^{ar} g^{avs} g^{us} = g^{ar} g^{av(-rv\text{^}\{-1\})} g^{us} = g^{ar} g^{-ar} g^{us} = g^{us}$
  - i.e., (r,s) is a signature of the message us

ElGamal Signature (Continued)

- $0 < r < p$ must be checked, otherwise easy to forge a signature on any message if an valid signature is available.
  - given M, and $r=g^k$, $s=k^{-1}(h(M) - ar) \bmod (p-1)$
  - for any message M', let $u=h(M') / h(M) \bmod (p-1)$
  - computes $s'=su \bmod (p-1)$ and r' s.t.
    $r' \equiv ru \pmod{p-1}$ AND $r' \equiv r \pmod{p}$, then
    $\beta^{r'} r^{s'} = \beta^{ru} r^{su} = (\beta^{r} r^{s})^{u} = (g^{h(M)})^{u} = g^{h(M')}$

Digital Signature Algorithm (DSA)

Specified as FIPS 186

Key generation

- Select a prime q of 160-bits
- Choose $0 \leq t \leq 8$
- Select $2^{511+64t} < p < 2^{512+64t}$ with $q \mid p-1$
- Let α be a generator of Z_p^*, and set
  $g = \alpha^{(p-1)/q} \bmod p$
- Select $1 \leq a \leq q-1$
- Compute $\beta = g^a \bmod p$

Public key: (p, q, g, β)

Private key: a

DSA

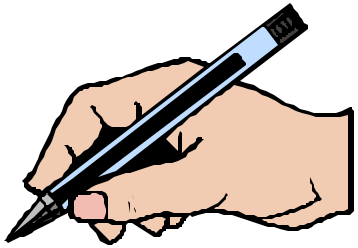
Signing message M:

- Select a random integer k, $0 < k < q$
- Compute

  $k^{-1} \bmod q$

  $\mathbf{r = (g^k \bmod p) \bmod q}$

  $\mathbf{s = k^{-1} (h(M) + ar) \bmod q}$
- Signature: (r, s)



Note: FIPS recommends the use of SHA-1 as hash function.

DSA

Signature: (r, s)

$r = (g^k \bmod p) \bmod q$

$s = k^{-1} (h(M) + ar) \bmod q$

Verification

- Verify $0 < r < q$ and $0 < s < q$, if not, invalid
- Compute

  $u_1 = h(M)s^{-1} \bmod q$,

  $u_2 = rs^{-1} \bmod q$
- Valid iff $r = (g^{u_1} \beta^{u_2} \bmod p) \bmod q$

  $g^{u_1} \beta^{u_2} = g^{h(M)s\text{^}\{-1\}} g^{ars\text{^}\{-1\}} = g^{(h(M)+ar)s\text{^}\{-1\}} = g^k \pmod p$

Coming Attractions …

- Security protocols