Digital Signatures

Outline

- [1] Introduction
- [2] Security Requirements for Signature Schemes
- [3] The ElGamal Signature Scheme
- [4] Variants of the ElGamal Signature Scheme
  - The Digital Signature Algorithm
  - The Elliptic Curve DSA

[1] Introduction

- A signature scheme consists of two components: a signing algorithm and a verification algorithm
- Alice can sign a message x using a private signing algorithm sig
- The resulting signature sig(x) can subsequently be verified using a public verification algorithm ver
- Given a pair (x,y), the verification algorithm returns an answer "true" or "false" depending on whether the signature is valid.

(Def) A signature scheme is a 5-tuple (P,A,K,S,V):

- *P* is a finite set of possible messages
- *A* is a finite set of possible signatures
- *K* is a finite set of possible keys
- For each key K, there is a signing algorithm sig_k in S and a verification algorithm ver_k in V such that:
  - ver(x,y) = true if and only if y=sig(x)

- A pair (x,y) is a signed message

- The functions sig_k and ver_k should be polynomial-time computable functions

- Given a message x, it should be computationally infeasible for anyone other than Alice to compute a signature y such that $ver_k(x,y)$=true

- If Oscar can compute a pair (x,y) such that $ver_k(x,y)$=true and x was not previously signed by Alice, y is called a forgery

RSA signature scheme

$n = pq$

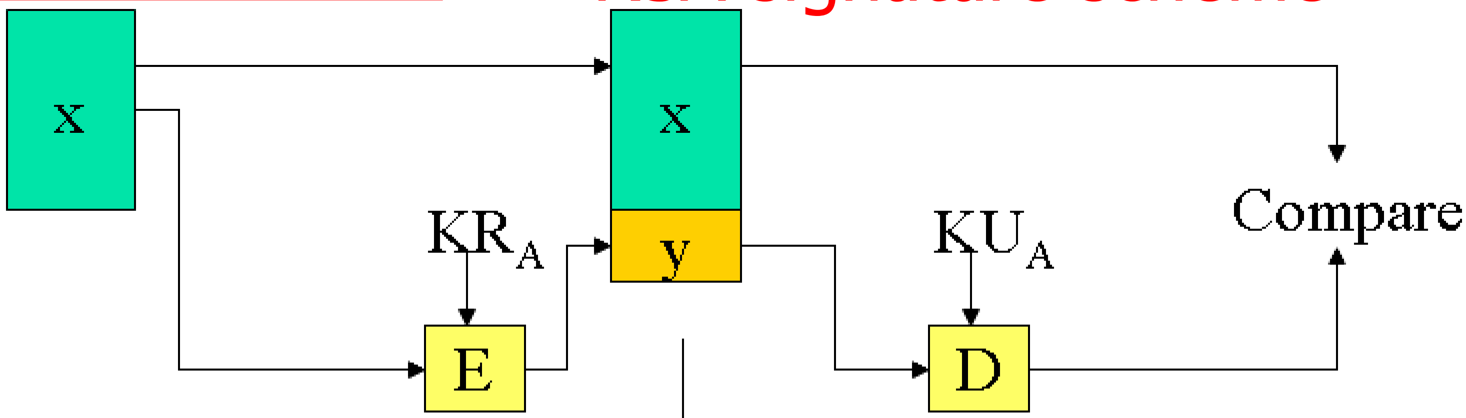
$d*e = 1 \pmod{\phi(n)}$

Signing key

$KR_A = (d, n)$

Verification key

$KU_A = (e, n)$



$sign_{KR_A}(x) = x^d \pmod{n}$

$ver_{KU_A}(y) =$ true iff $y^e \pmod{n} = x$

Signing

Verification

(RSA signature scheme)

- Let $n=pq$, p and q are primes. Define

  $\mathcal{K}=\{ (n,p,q,d,e) : n=pq, de=1 \bmod \Phi(n) \}$

- For each $K=(n,p,q,d,e)$ in $\mathcal{K}$, define

$$y = sig_K(x) = x^d \bmod n$$

and

$ver_K(x,y) = \text{true}$ <u>if and only if</u> $x = y^e \bmod n$

- Combine signing and encryption

Signing before encrypting is recommended. Since:

if Alice first encrypted x, then signed the result:

$$z=e_{Bob}(x) \text{ and } y=sig_{Alice}(z)$$

Oscar can replace y by his own signature

$$y'=sig_{Oscar}(z)$$

Bob may infer that the plaintext x originated with Oscar.

[2] Security Requirements for Signature Schemes

(1) Three attack models

- Key-only attack
  Oscar possesses Alice's public key
- Known message attack
  Oscar possesses a list of messages previously signed by Alice
- Chosen message attack
  Oscar requests Alice's signatures on a list of messages

(2) Three possible adversarial goals

- Total break

  Determine the signing key

- Selective forgery

  Forge a valid signature on a message chosen by someone else with non-negligible probability

- Existential forgery

  Forge a valid signature on a message which hasn't previously been signed by Alice

(3) Forgeries based on RSA signature scheme

1. Existential forgery using a key-only attack

2. Existential forgery using a known message attack

3. Selective forgery using a chosen message attack

1. Existential forgery using a key-only attack

For any y,

$$(x=y^e,\ y) \text{ satisfies } ver_k(x,y) = \text{true}$$

The use of hash functions in conjunction with signature schemes will eliminate this type of forging

2. Existential forgery using a known message attack

The attack is based on the multiplicative property of RSA.

Suppose $y_1=sig_k(x_1)$, $y_2=sig_k(x_2)$ are two messages previously signed by Alice.

Then $ver_k(x_1x_2 \bmod n, y_1y_2 \bmod n)=true$

3. Selective forgery using a chosen message attack

- Suppose Oscar wants to forge a signature on the message x, where x was possibly chosen by someone else. It is simple matter for him to find x_1, x_2 in Z_n such that $x = x_1 x_2 \bmod n$

- He asks Alice for the signatures on messages x_1 and x_2, which we denote by y_1 and y_2 respectively

- As in previous attack, $y_1 y_2 \bmod n$ is the signature for the message $x = x_1 x_2 \bmod n$

(4) Three attacks related to hash in signature scheme

1. Oscar may start with a valid signed message (x,y), where $y=sig_{Alice}(h(x))$. Then he computes $z=h(x)$ and attempts to find $x' \neq x$ such that $h(x')=h(x)$.

   If Oscar can do this, (x',y) would be a valid signed message
   (existential forgery using a known message attack)

   In order to prevent this type of attack, we require that h is second preimage resistant

2. Oscar first finds two messages $x' \neq x$ such that $h(x)=h(x')$. Oscar them gives x to Alice and persuades her to sign the message digest $h(x)$, obtaining y.

If Oscar can do this, (x',y) is a valid signed message (existential forgery using a chosen message attack)

In order to prevent this type of attack, we require that h is collision resistant

3. It is often possible with certain signature schemes to forge signatures on random message digests z (eg. RSA Signature Scheme).

If Oscar can compute a signature on some message digest z ($y=sig_{Alice}(z)$), and then he finds a message x such that $z=h(x)$. This (x,y) is a valid signed message (existential forgery using a key-only attack)

In order to prevent this type of attack, we require that h be a preimage resistant hash function

[3] ElGamal Signature Scheme

- ElGamal Signature Scheme was proposed in 1985
- The scheme is non-deterministic
- Its security is based on Discrete Logarithm Problem
  - The Discrete Logarithm Problem :
    given an element β belonging to $<\alpha>$, find an integer a such that $\alpha^a = \beta$

ElGamal signature scheme

Global public key

p

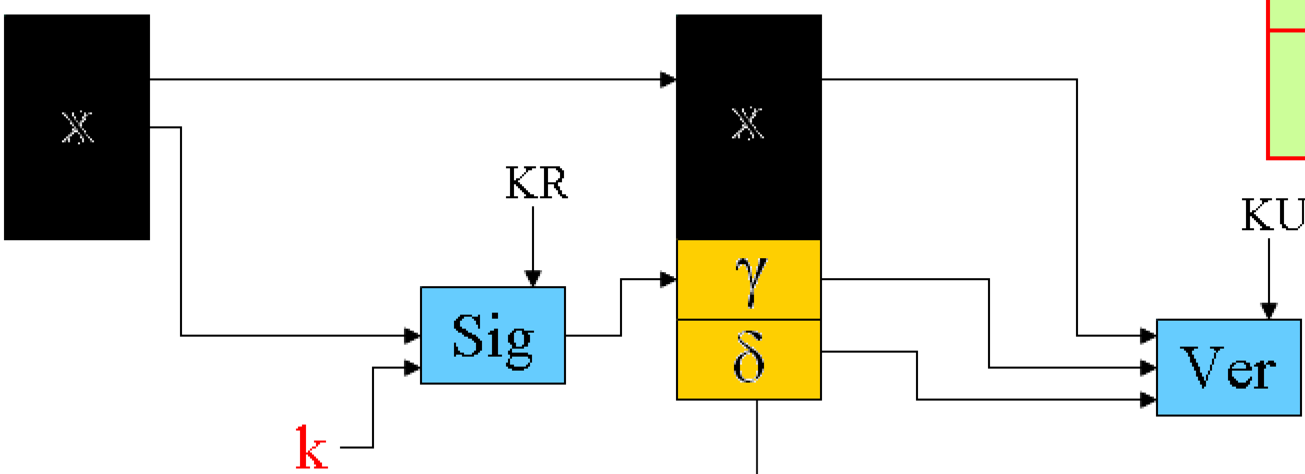
α

Signing key

$KR = a$

Verification key

$KU = \beta = \alpha^{a} \bmod p$



$$\gamma = \alpha^{k} \bmod p$$

$$\delta = (x - a\gamma)k^{-1} \bmod p-1$$

Test $\alpha^{x} \stackrel{?}{=} \beta^{\gamma}\gamma^{\delta} \bmod p$

Signing

Verification

(ElGamal signature scheme)

- Let p be a prime such that DL problem in Z_p is intractable, and let α be a primitive element in Z_p^*
  Define K={ (p,α,a,β) : $\beta = \alpha^a \bmod p$ }
  p,α,β are the public key, a is the private key

- For a (secret) random number k, define
  $sig_k(x,k)=(\gamma, \delta)$, where
  $\gamma = \alpha^k \bmod p$ and $\delta = (x - a\gamma)k^{-1} \bmod (p-1)$

- For a message (γ, δ), define

ver $(x, (\gamma, \delta))$=true iff. $\beta^{\gamma}\gamma^{\delta} = \alpha^{x} \bmod p$

- If the signature was constructed correctly, the verification will succeed since

$$\beta^{\gamma}\boldsymbol{\gamma}^{\boldsymbol{\delta}} = \underline{\alpha^{a\gamma}\alpha^{k\delta} = \alpha^{x} \bmod p}$$

By definition of b

- Example

We take p=467, α=2, a=127; then

$$\beta = 2^{127} \bmod 467 = 132$$

To sign the message x=100, Alice select k=213;

Then

$$\gamma = 2^{213} \bmod 467 = 29,$$

$$\delta = (100-127*29) * 213^{-1} \bmod 466 = 51$$

(100, (29,51)) is the signed message

Since (100, (29,51)) is valid, Bob will find that

$\beta^{\gamma}\gamma^{\delta}$ mod p = $132^{29} * 29^{51}$ mod 467 = 189

is identical with

α^{x} mod p = 2^{100} mod 467 = 189

Security of the ElGamal Signature Scheme

1. Selective forgery using a key only attack

Suppose Oscar tries to forge a signature (x,y) for a given message x, without knowing a

If he chooses a value γ and the tries to find δ, he must compute

$$\delta = \log_{\gamma} \alpha^{x} \beta^{-\gamma} \bmod p$$

It is an instance of DL problem

Unsuccessful forgery

2. Selective forgery using a key-only attack

- If he chooses a value δ and the tries to find γ, he must solve the equation

$$\beta^{\gamma}\gamma^{\delta} \bmod p = \alpha^{x} \bmod p$$

for the unknown value γ

- It is a problem for which no feasible solution is known

Unsuccessful forgery

3. Existential forgery using a key only attack

- If he chooses a value δ and γ, then tries to find x, he must compute

$$x = \log_{\alpha} \beta^{\gamma}\gamma^{\delta}$$

It is an instance of DL problem

Unsuccessful forgery

4. Existential forgery using a key only attack

- Unfortunately, an adversary is able to forge a signed message which can pass the verification

  Suppose i and j are integers in Z_{p-1} and gcd(j, p-1), the adversary can assign γ by

  $$\gamma = \alpha^{i} \beta^{j} \bmod p$$

  According to the above assignment, the verification condition is

  $$\alpha^{x} = \beta^{\gamma} (\alpha^{i} \beta^{j})^{\delta} \bmod p$$

It is equivalent to

$$\alpha^{x-i\delta} = \beta^{\gamma+j\delta} \bmod p$$

The congruence will be satisfied if

$$\begin{cases} x - i\delta = 0 \bmod p-1, \text{ and} \\ \gamma + j\delta = 0 \bmod p-1 \end{cases} \qquad (1)$$

Given i and j where gcd(j,p-1)=1, we can solve (1) for x and δ

$$\begin{cases} \gamma = \alpha^i \beta^j \bmod p \\ \delta = -\gamma j^{-1} \bmod p-1 \ (j^{-1} \text{ exist}) \\ x = -\gamma i j^{-1} \bmod p-1 \end{cases}$$

Since gcd(j,p-1)

The adversary constructed a valid signature

$(x, (\gamma, \delta))$

- Example

Let p=467, α=2, β =132.

Suppose the adversary chooses i=99 and j=179

$$\begin{cases} \gamma = 2^{99}132^{179} \bmod 467 & = 177 \\ \delta = -\gamma * 179^{-1} \bmod 466 & = 41 \\ x = -\gamma * 99 * 179^{-1} \bmod 466 & = 331 \end{cases}$$

It will pass the verification:

$$\beta^{\gamma}\gamma^{\delta} = 132^{117} * 117^{41} = 303 \bmod 467$$

$$\alpha^{x} = 2^{231} = 303 \bmod 467$$

5. Careless use of k will cause attacks:

1. When k is known, an adversary can obtain Alice's signing key since:

$$a = (x - k\delta) * \gamma^{-1} \bmod p-1$$

2. When identical k is used in signing two different messages, an adversary can obtain Alice's signing key

Suppose $(x_1,(\gamma_1,\delta_1))$ and $(x_2,(\gamma_2,\delta_2))$ are two signed messages, we have

$$\beta^{\gamma}\gamma^{\delta_1}=\alpha^{x_1} \bmod p$$

$$\beta^{\gamma}\gamma^{\delta_2}=\alpha^{x_2} \bmod p$$

Thus

$$\alpha^{x_1-x_2}=\gamma^{\delta_1-\delta_2} \bmod p$$

- Suppose $\gamma = \alpha^{k}$, then

$$\alpha^{x_1 - x_2} = \alpha^{k(\delta_1 - \delta_2)} \bmod p$$

which is equivalent to

$$x_1 - x_2 = k(\delta_1 - \delta_2) \bmod p-1$$

- Let $d=\gcd(\delta_1-\delta_2, p-1)$, define

$$x' = (x_1 - x_2)/d, \quad \delta' = (\delta_1 - \delta_2)/d, \quad p' = (p-1)/d$$

- Then the congruence becomes

$$x' = k\delta' \bmod p'$$

thus

$$k = (x' * \delta'^{-1}) + (i * p') \bmod p-1, \text{ for } 0 \le i \le d-1$$

Of these d candidate values, the correct k which is really used by Alice can be determined by testing the condition

$$\gamma = \alpha^k \bmod p$$

- Example

We take $p=17$, $\alpha=3$, $a=8$; then

$$\beta = 3^8 \bmod 17 = 16$$

Alice first signs $x_1=15$ using $k=5$

$$(15, (5,11))$$

Then she signs $x_2=10$ using $k=5$ again

$$(10, (5,10))$$

Oscar obtains:

$$(x_1=15, (\gamma_1=5, \delta_1=11))$$
$$(x_2=10, (\gamma_2=5, \delta_2=10))$$

Then he can compute

$$d=\gcd(\delta_1-\delta_2, p-1)=\gcd(1,16)=1$$

Thus these is only one candidate value of r

$$k = (x' * \delta'^{-1}) \bmod p-1$$
$$= (5 * 1) \bmod 16 = 5$$

Then he can obtain Alice's signing key by

$$
\begin{aligned}
a &= (x - k\delta) * \gamma^{-1} \bmod p-1 \\
&= (15 - 5*11) * 5^{-1} \bmod 16 \\
&= 8 * 13 \bmod 16 \\
&= 8
\end{aligned}
$$

[4] Variants of the ElGamal Signature Scheme

- **Digital Signature Algorithm (DSA)**
  - Proposed in 1991
  - Was adopted as a standard on December 1, 1994
- **Elliptic Curve DSA (ECDSA)**
  - FIPS 186-2 in 2000

$L = 0 \bmod 64$,
$512 \leq L \leq 1024$

Digital Signature Algorithm

- Let p be a L-bit prime such that the DL problem in Z_p^* is intractable, and let q be a 160-bit prime that divides p-1. Let α be a q_{th} root of 1 modulo p.

  Define $K = \{ (p,q,\alpha,a,\beta): \beta = \alpha^a \bmod p \}$

  p,q,α,β are the public key, a is private

- For a (secret) random number k, define

  sig (x,k)=(γ,δ), where

  $\gamma=(\alpha^k \bmod p) \bmod q$ and

  $\delta=(\text{SHA-1}(x)+a\gamma)k^{-1} \bmod q$

- For a message (x,(γ,δ)), verification is done by performing the following computations:

  $e_1=\text{SHA-1}(x)*\delta^{-1} \bmod q$

  $e_2=\gamma*\delta^{-1} \bmod q$

  ver(x,(γ,δ))=true iff. $(\alpha^{e_1}\beta^{e_2} \bmod p) \bmod q=\gamma$

- Notice that the verification requires to compute:

$$e_1 = SHA\text{-}1(x) * \delta^{-1} \bmod q$$
$$e_2 = \gamma * \delta^{-1} \bmod q$$

when $\delta = 0$ (it is possible!), Alice should re-construct a new signature with a new k

DSA Example

- Take $q=101$, $p=78q+1=7879$, $\alpha=170$, $a=75$; then $\beta=4567$

- To sign the message SHA-1(x)=22, Alice selects $k=50$;

  Then $\gamma=(170^{50} \bmod 7879) \bmod 101=94$,

  $\delta=(22+75*94)50^{-1} \bmod 101=97$

  (x, (94,97)) is the signed message

- The signature (94,97) on the message digest 22 can be verify by the following computations:

$\delta^{-1}=97^{-1} \bmod 101=25$

$e^{1}=22*25 \bmod 101=45$

$e^{2}=94*25 \bmod 101=27$

$(170^{45}*4567^{27} \bmod 7879) \bmod 101 = 94 = \gamma$

Elliptic Curve DSA

- Let p be a prime or a power of two, and let E be an elliptic curve defined over F_p. Let A be a point on E having prime order q, such that DL problem in <A> is infeasible.

  Define $\mathcal{K}$={ (p,q,E,A,m,B): B=mA }

  p,q,E,A,B are the public key, m is private

- For a (secret) random number k, define $sig_k(x,k)=(r,s)$,
  where $rA=(u,v)$, $r=u \bmod q$ and
  $s=k^{-1}(SHA\text{-}1(x)+mr) \bmod q$

- For a message $(x,(r,s))$, verification is done by performing the following computations:

  $i=SHA\text{-}1(x)*s^{-1} \bmod q$
  $j=r*s^{-1} \bmod q$
  $(u,v)=iA+jB$

  $ver(x,(r,s))=true$ if and only if $u \bmod q=r$