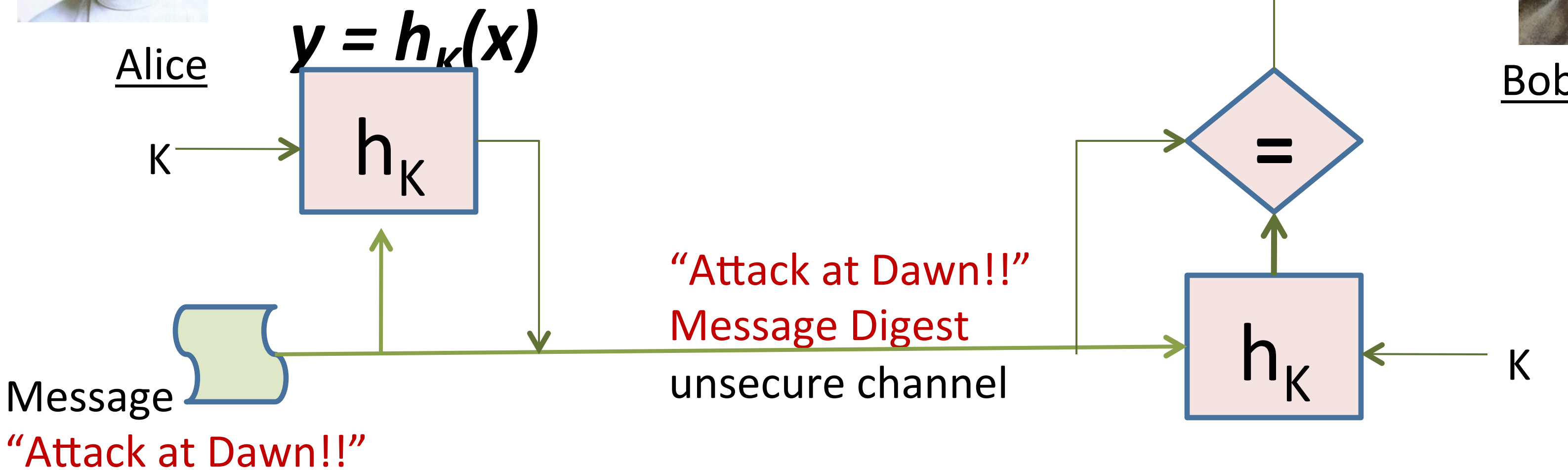
Signature Schemes

Chester Rebeiro
IIT Madras

STINSON : chapter 7

Recall : MACs



MACs allow Bob to be certain that

- the message has originated from Alice
- the message was not tampered during communication

MAC cannot

- prevent Bob from creating forgeries (i.e., messages in the name of Alice)
- cannot prove Authenticity to someone without sharing the secret key K

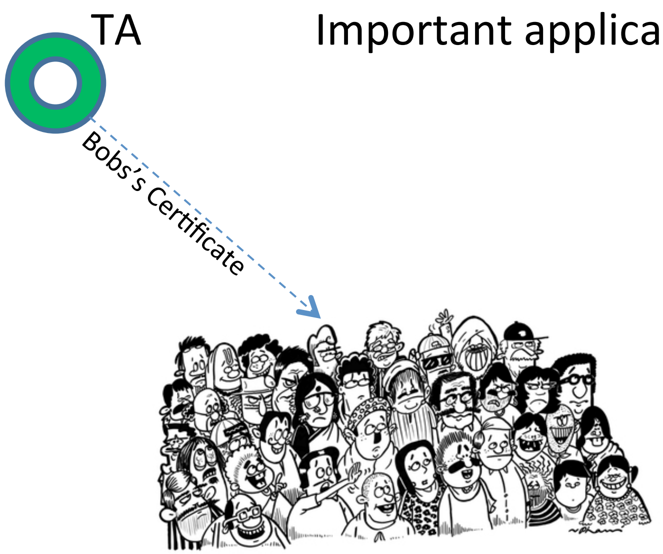
Digital Signatures solve both these problems

Digital Signatures

- A token sent along with the message that achieves
  - Authentication
  - Non-repudiation
  - Integrity
- Based on public key cryptography

Public key Certificates

Important application of digital signatures

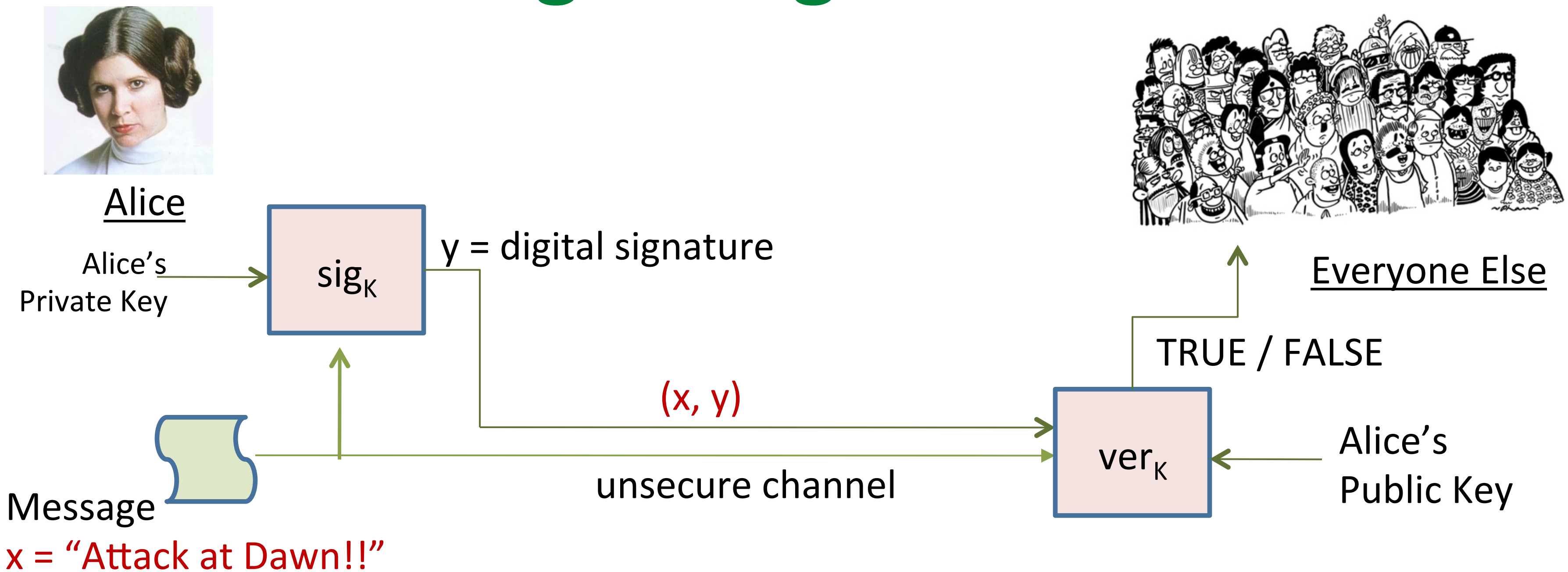


Bob's Certificate{
  Bob's public key in plaintext
  Signature of the certifying authority
  other information
}

To communicate with Bob, Alice gets his public key from a trusted authority (TA)
A trusted authority could be a Government agency, Verisign, etc.

A signature from the TA, ensures that the public key is authentic.

Digital Signature



Signing Function

y = sig$_a$(x)

Input : Message (x) and Alice's private key
Output: Digital Signature of Message

Verifying Function

ver$_b$(x, y)

Input : digital signature, message
Output : true or false

true if signature valid
false otherwise

Digital Signatures (Formally)

Definition : A *signature scheme* is a five-tuple $(\mathcal{P}, \mathcal{A}, \mathcal{K}, \mathcal{S}, \mathcal{V})$, where the following conditions are satisfied:

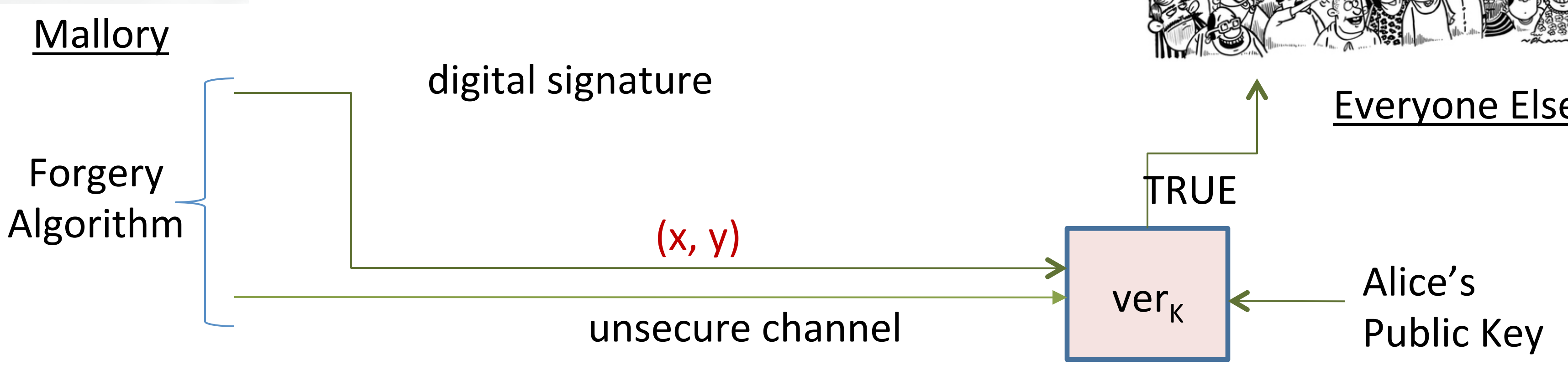
1. $\mathcal{P}$ is a finite set of possible *messages*
2. $\mathcal{A}$ is a finite set of possible *signatures*
3. $\mathcal{K}$, the *keyspace*, is a finite set of possible *keys*
4. For each $K \in \mathcal{K}$, there is a *signing algorithm* $\mathbf{sig}_K \in \mathcal{S}$ and a corresponding *verification algorithm* $\mathbf{ver}_K \in \mathcal{V}$. Each $\mathbf{sig}_K : \mathcal{P} \rightarrow \mathcal{A}$ and $\mathbf{ver}_K : \mathcal{P} \times \mathcal{A} \rightarrow \{true, false\}$ are functions such that the following equation is satisfied for every message $x \in \mathcal{P}$ and for every signature $y \in \mathcal{A}$:

$$\mathbf{ver}_K(x, y) = \begin{cases} true & \text{if } y = \mathbf{sig}_K(x) \\ false & \text{if } y \neq \mathbf{sig}_K(x). \end{cases}$$

A pair (x, y) with $x \in \mathcal{P}$ and $y \in \mathcal{A}$ is called a *signed message*.

CR

Forgery

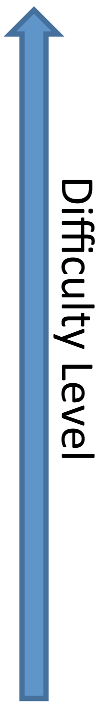


If Mallory can create a valid digital signature such that ver$_K$(x, y) = TRUE for a message not previously signed by Alice, then the pair (x, y) forms a forgery

Security Models for Digital Signatures

Assumptions   **Goals of Attacker**

- **Total break:**
  Mallory can determine Alice's private key
  (therefore can generate any number of signed messages)
- **Selective forgery:**
  Given a message x, Mallory can determine y, such
  that (x, y) is a valid signature from Alice
- **Existential forgery:**
  Mallory is able to create y for some x, such that
  (x, y) is a valid signature from Alice



Security Models for Digital Signatures

Assumptions Goals of Attacker

Weak
(needs a strong attacker)

- **Key-only attack :**
  Mallory only has Alice's public key
  (i.e. only has access to the verification function, *ver*)
- **Known-message attack :**
  Mallory only has a list of messages signed by Alice
  $(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4), \ldots..$
- **Chosen-message attack :**
  Mallory chooses messages $x_1, x_2, x_3, \ldots\ldots..$ and tricks
  Alice into providing the corresponding signatures y_1, y_2, y_3
  (resp.)

Strong

First Attempt making a digital signature (using RSA)

b, n public

a, p, q private

$n = pq;\ a \equiv b^{-1} \bmod \phi(n)$

$$
\begin{aligned}
&sig(x)\{\\
&\quad y \equiv x^{a} \bmod n\\
&\quad return\ (x, y)\\
&\}
\end{aligned}
$$

(x, y) →

$$
\begin{aligned}
&ver(x, y)\{\\
&\quad if\,(x \equiv y^{b} \bmod n)\ \ return\ \ TRUE\\
&\quad else\ return\ \ FALSE\\
&\}
\end{aligned}
$$

x is the message here
and (x, y) the signature

A Forgery for the RSA signature (First Forgery)

b, n public

a, p, q private

$n = pq;\ a \equiv b\text{-}1 \bmod \varphi(n)$

$sig(x)\{$

$\quad y \equiv x^a \bmod n$

$\quad return\ (x, y)$

$\}$

(x, y) →

$ver_K(x, y)\{$

$\quad if\ (x \equiv y^b \bmod n)\ \ return\ TRUE$

$\quad else\ return\ \ FALSE$

$\}$

$forgery()\{$

$\quad$ select a random y

$\quad compute\ x \equiv y^b \bmod n$

$\quad return\ (x, y)$

$\}$

Key only, existential forgery

Second Forgery

Suppose Alice creates signatures of two messages x_1 and x_2

$$y_1 = sig(x_1) \rightarrow y_1 \equiv x_1^a \bmod n \qquad (x_1, y_1)$$

$$y_2 = sig(x_2) \rightarrow y_2 \equiv x_2^a \bmod n \qquad (x_2, y_2)$$

Mallory can use the **multiplicative property of RSA** to create a forgery

$$(x_1 x_2 \bmod n, y_1 y_2 \bmod n) \quad is \ a \ forgery$$

$$y_1 y_2 \equiv x_1^a x_2^a \bmod n$$

Known message, existential forgery

RSA Digital Signatures

Incorporate a hash function in the scheme to prevent forgery

b, n public

a, p, q private

$$
\begin{aligned}
&sig(x)\{\\
&\quad z = h(x)\\
&\quad y \equiv z^a \bmod n\\
&\quad return\ (x, y)\\
&\}
\end{aligned}
$$

(x, y) →

$$
\begin{aligned}
&ver_K(x, y)\{\\
&\quad z = h(x)\\
&\quad if\,(z \equiv y^b \bmod n)\ \ return\ TRUE\\
&\quad else\ return\ \ FALSE\\
&\}
\end{aligned}
$$

x is the message here, (x, y) the signature
and h is a hash function

How does the hash function help?

Preventing the First Forgery

$$\begin{aligned}
&forgery()\{\\
&\quad \text{select a random } y\\
&\quad compute\ z' \equiv y^b \bmod n\\
&\quad compute\ I^{st}\ preimage:\ x\,st.\ z' = h(x)\\
&\quad return\ (x, y)\\
&\}
\end{aligned}$$

Forgery becomes equivalent to the first preimage attack on the hash function

How does the hash function help?

Preventing the Second Forgery

$$(x_1 x_2 \bmod n, y_1 y_2 \bmod n) \quad \textit{is difficult}$$

$$y_1 y_2 \equiv h(x_1)^a h(x_2)^a \bmod n$$

$$\not\equiv x_1{}^a x_2{}^a \bmod n$$

creating such a forgery is unlikely

How does the hash function help?

Another Forgery prevented

$$forgery(x, y)\{$$
$$\quad compute\ \ h(x)$$
$$\quad compute\ II^{nd}\ preimage:\ \ find\ \ x'\ s.t.\ \ h(x) = h(x')\ \ \ and\ \ x \neq x'$$
$$\quad return\ \ (x', y)$$
$$\}$$

Given a valid signature (x,y) find (x’,y)

creating such a forgery is equivalent to solving the 2nd preimage problem of the hash functionw

ElGamal Signature Scheme

- 1985
- Variant adopted by NIST as the DSA
  (DSA: standard for digital signature algorithm)
- Based on the difficult of the discrete log problem

ElGamal Signing

Initialization

Choose a large prime p

Let $\alpha \in Z_p^*$ be a primitive element

Choose $a \quad (0 < a \le p-1)$

Compute $\beta \equiv \alpha^a \bmod p$

Public Parameters : p, α, β

Private key : a

Signing Message x

$sig(x)\{$

$\quad$ select a secret random k $s.t.$ $\gcd(k, p-1) = 1$

$\quad \gamma \equiv \alpha^k \mod p$

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

$\quad y = (\gamma, \delta)$

$\quad return\ (x, y)$

$\}$

The use of a random secret k for every signature makes ElGamal non-deterministic

ElGamal Verifying

Initialization

Choose a large prime p

Let $\alpha \in Z_p^*$ be a primitive element

Choose $a \quad (0 < a \le p-1)$

Compute $\beta \equiv \alpha^a \bmod p$

Public Parameters : p, α, β

Private key : a

Verifying Signature (x,y)

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

$\quad compute \;\; t_1 \equiv \alpha^x \bmod p$

$\quad compute \;\; t_2 \equiv \beta^\gamma \gamma^\delta \bmod p$

$\quad if \;\; (t_1 = t_2)$

$\qquad return \;\; TRUE$

$\quad else$

$\qquad return \;\; FALSE$

$\}$

ElGamal Correctness

Initialization

Choose a large prime p

Let $\alpha \in Z_p^*$ be a primitive element

Choose $a \quad (0 < a \le p-1)$

Compute $\beta \equiv \alpha^a \bmod p$

Public Parameters : p, α, β

Private key : a

Signing Message x

$sig(x)\{$

select a secret random k

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

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

$y = (\gamma, \delta)$

$return\ (x, y)$

$\}$

Verifying Signature (x,y)

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

$compute\ t_1 \equiv \alpha^x \bmod p$

$compute\ t_2 \equiv \beta^\gamma \gamma^\delta \bmod p$

$if\ (t_1 = t_2)\ return\ TRUE$

$else\ return\ FALSE$

$\}$

correctness

First note that

$$a\gamma + k\delta \equiv x \bmod (p-1)$$

$$t_2 \equiv \beta^\gamma \gamma^\delta \bmod p \qquad\qquad t_1 \equiv \alpha^x \bmod p$$
$$\equiv (\alpha^a)^\gamma \cdot (\alpha^k)^\delta \bmod p$$
$$\equiv \alpha^{a\gamma + k\delta} \bmod p$$
$$\equiv \alpha^x \bmod p$$

if the signature is valid, $t_1 = t_2$

Example

$$\begin{aligned} p &= 467 \\ \alpha &= 2 \\ \text{a} &= 127 \\ \beta &\equiv \alpha^a \bmod p \\ &= 2^{127} \bmod 467 \\ &= 132 \end{aligned}$$

Signature of message x = 100

$$\begin{aligned} k &= 213 \quad (\textit{chosen randomly}) \\ k^{-1} &\bmod p-1 = 431 \\ \gamma &= \alpha^k \bmod p \\ &= 2^{213} \bmod 467 \\ &= 29 \\ \delta &= (x - a\gamma)k^{-1} \bmod p-1 \\ &= (100 - 2 \cdot 29)431 \bmod 466 \\ &= 51 \end{aligned}$$

Verifying

$$\begin{aligned} &\beta^\gamma \gamma^\delta \bmod p = 132^{29} 29^{51} \bmod 467 = 189 \\ &\alpha^x \bmod p = 2^{100} \bmod p = 189 \\ &\textit{TRUE} \end{aligned}$$

CR

Security of ElGamal Signature Scheme
(against Selective forgery)

Given an x, Mallory needs to find (γ,δ) such that $ver(x,(\gamma,\delta)) = TRUE$

Attempt 1

Choose a value for γ, then try to compute δ $s.t.$ $\beta^{\gamma}\gamma^{\delta} \equiv \alpha^{x} \bmod p$

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

This is the intractable discrete log problem

Attempt 2

Choose a value for δ, then try to compute γ $s.t.$ $\beta^{\gamma}\gamma^{\delta} \equiv \alpha^{x} \bmod p$

This is not related to the discrete log problem. There is no known solution for this.

Attempt 3

Choose value for γ and δ simultaneously, $s.t.$ $\beta^{\gamma}\gamma^{\delta} \equiv \alpha^{x} \bmod p$

No way known.

Security of ElGamal Signature Scheme *(against* Existential *forgery)*

Mallory needs to find an $(x, (\gamma,\delta))$ such that $ver(x,(\gamma,\delta)) = TRUE$

The one-parameter forgery

forgery

$$\text{choose some } i \qquad (0 \le i \le p-2).$$

$$\text{form } \gamma \equiv \alpha^{i}\beta \bmod p$$

$$\delta \equiv -\gamma \bmod (p-1)$$

$$x \equiv i\delta \bmod (p-1).$$

$$\text{then, } ver(x,(\gamma,\delta)) = TRUE$$

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

proof

$$RHS \equiv \beta^{\gamma}(\alpha^{i}\beta)^{\delta} \bmod p$$

$$\equiv \beta^{\gamma+\delta}\alpha^{i\delta} \bmod p$$

$$\equiv \alpha^{a\gamma+a\delta}\alpha^{i\delta} \bmod p$$

$$\equiv \alpha^{a\gamma-a\gamma+i\delta} \bmod p$$

$$\equiv \alpha^{i\delta} \bmod p$$

$$\equiv \alpha^{x} \bmod p = LHS$$

Security of ElGamal Signature Scheme *(against* Existential *forgery)*

Mallory needs to find an $(x, (\gamma, \delta))$ such that $ver(x, (\gamma, \delta)) = TRUE$

The two-parameter forgery

forgery

$$
\begin{aligned}
&choose\ some\ i, j && (0 \le i, j \le p-2;\ \gcd(j, p-1) = 1).\\
&form\ \ \gamma \equiv \alpha^{i} \beta^{j} \bmod p\\
&\qquad \delta \equiv -\gamma j^{-1} \bmod (p-1)\\
&\qquad x \equiv \gamma i j^{-1} \bmod (p-1).\\
&then,\ ver(x, (\gamma, \delta)) = TRUE
\end{aligned}
$$

Prevent Existential Forgeries by hashing the message

CR

Improper use of ElGamal's Signature Scheme

1. What if k is not a secret?

$$\textit{if } \gcd(\gamma, p-1) = 1 \textit{ then}$$
$$\text{secret } a \text{ can be computed as follows}$$
$$a = (x - k\delta)\gamma^{-1} \bmod (p-1).$$

$$
\begin{aligned}
&sig(x)\{\\
&\quad \text{select a secret random } k\\
&\quad \gamma \equiv \alpha^k \bmod p\\
&\quad \delta \equiv (x - a\gamma)k^{-1} \bmod p-1\\
&\quad y = (\gamma, \delta)\\
&\quad \textit{return } (x, y)\\
&\}
\end{aligned}
$$

The secret key 'a' is retrieved and Mallory can create many forgeries

Improper use of ElGamal's Signature Scheme

2. What if k is reused?

Lets say we have two different messages x_1 *and* x_2 *signed with the same* k

The signatures are (γ, δ_1) *and* (γ, δ_2) *then,*

$$
\begin{aligned}
&sig(x)\{\\
&\quad \text{select a secret random } k\\
&\quad \gamma \equiv \alpha^k \mod p\\
&\quad \delta \equiv (x - a\gamma)k^{-1} \mod p-1\\
&\quad y = (\gamma, \delta)\\
&\quad return\ (x, y)\\
&\}
\end{aligned}
$$

$$\beta^\gamma \gamma^{\delta_1} \equiv \alpha^{x_1} \pmod p$$

$$\beta^\gamma \gamma^{\delta_2} \equiv \alpha^{x_2} \pmod p.$$

dividing

$$\alpha^{x_1 - x_2} \equiv \gamma^{\delta_1 - \delta_2} \pmod p.$$

Representing in terms of α

$$\alpha^{x_1 - x_2} \equiv \alpha^{k(\delta_1 - \delta_2)} \pmod p,$$

=>

$$x_1 - x_2 \equiv k(\delta_1 - \delta_2) \pmod{p-1}.$$

CR

Improper use of ElGamal's Signature Scheme

$$x_1 - x_2 \equiv k(\delta_1 - \delta_2) \pmod{p-1}.$$

Now let $d = \gcd(\delta_1 - \delta_2, p-1)$. Since $d \mid (p-1)$ and $d \mid (\delta_1 - \delta_2)$, it follows that $d \mid (x_1 - x_2)$. Define

$$x' = \frac{x_1 - x_2}{d} \qquad \delta' = \frac{\delta_1 - \delta_2}{d} \qquad p' = \frac{p-1}{d}.$$

Then the congruence becomes:

$$x' \equiv k\delta' \pmod{p'}.$$

Since $\gcd(\delta', p') = 1$, we can compute

$$\epsilon = (\delta')^{-1} \bmod p'.$$

Then value of k is determined modulo p' to be

$$k = x'\epsilon \bmod p'.$$

This yields d candidate values for k:

$$k = x'\epsilon + ip' \bmod (p-1)$$

for some i, $0 \leq i \leq d-1$. Of these d candidate values, the (unique) correct one can be determined by testing the condition

$$\gamma \equiv \alpha^k \pmod{p}.$$

ElGamal Signature Length

- Generally p is a prime of length 1024 bits
- The signature comprises of (γ, δ) which is of length 2048 bits

Schnorr's Signature Scheme is a modification of the ElGamal signature scheme with signatures of length around 320 bits

Schnorr Group

Let p be a large prime and Z_p^* the corresponding multiplicative group

Select another prime q $(< p)$ such that $p \equiv 1 \bmod q$

$i.e.\ q \mid (p-1)$ or $p = qr + 1$

Choose a random h in the range $1 < h < p$ $s.t.$

$h^r \neq 1 \bmod p$

This h^r is the generator of a subgroup of order q

$note\ h^q \equiv 1 \bmod p$

CR

Schnorr Group and Discrete Log

- When p is used, best known technique to solve discrete log is index-calculus

  For a 1024 bit prime, the complexity of index calculus is approx 2^{80}

- In the subgroup q, the best attack is pollard-rho which has a birthday paradox complexity.

  Thus a subgroup of size 2^160 will provide the same level of security

DSA (Digital Signature Standard)

Initialization

Choose a large prime $p\,(1024\,bit)$

Choose another prime $q\,(160\ bit)\ s.t.\ q \mid p-1$

Find α of order q (α creates a subgroup of order q)

Choose $a \quad (0 < a \le q-1)$

Compute $\beta \equiv \alpha^a \bmod p$

Public Parameters : p, q, α, β

Private key : a

$\alpha^{(p-1)/q} \bmod p$

DSA (Signing Function)

Initialization

Choose a large prime $p\,(1024\,bit)$

Choose another prime $q\,(160\ bit)\ s.t.\ q \mid p-1$

Find α of order q (α creates a subgroup of order q)

Choose $a \quad (0 < a \le q-1)$

Compute $\beta \equiv \alpha^{a} \bmod p$

Public Parameters : $p,\ q,\ \alpha,\ \beta$

Private key : a

Signing Message x

$sig(x)\{$

select a secret random $k\ s.t.\ \gcd(k,q)=1$

$\gamma \equiv (\alpha^{k} \bmod p) \bmod q$

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

$y = (\gamma, \delta)$

$return\ (x, y)$

$\}$

The use of a random secret k for every signature makes ElGamal non-deterministic

DSA (Verifying Function)

Initialization

Choose a large prime $p\,(1024\,bit)$

Choose another prime $q\,(160\ bit)\ s.t.\ q \mid p-1$

Find α of order q (α creates a subgroup of order q)

Choose a $(0 < a \le q-1)$

Compute $\beta \equiv \alpha^a \bmod p$

Public Parameters : $p,\ q,\ \alpha,\ \beta$

Private key : a

Signing Message x

$sig(x)\{$

select a secret random $k\ s.t.\ \gcd(k,q)=1$

$\gamma \equiv (\alpha^k \bmod p) \bmod q$

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

$y = (\gamma, \delta)$

$return\ (x, y)$

$\}$

Verifying Signature

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

$compute\ \ w \equiv \delta^{-1} \bmod q$

$compute\ \ t_1 \equiv w \cdot SHA(x) \bmod q$

$compute\ t_2 \equiv w \cdot \gamma \bmod q$

$compute\ v \equiv (\alpha^{t_1} \cdot \beta^{t_2} \bmod p) \bmod q$

$if\ \ (v \equiv \gamma \bmod q)\ \ return\ TRUE$

$else\ \ return\ FALSE$

$\}$

DSA (Correctness)

Initialization

Public Parameters : $p,\ q,\ \alpha,\ \beta\ (\beta \equiv \alpha^{a} \bmod p)$

Private key : a

Signing Message x

$sig(x)\{$

select a secret random k s.t. $\gcd(k,q)=1$

$\gamma \equiv (\alpha^{k} \bmod p) \bmod q$

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

$y=(\gamma,\delta)$

$return\ (x,y)$

$\}$

Verifying Signature

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

$compute\ \ w \equiv \delta^{-1} \bmod q$

$compute\ \ t_1 \equiv w \cdot SHA(x) \bmod q$

$compute\ \ t_2 \equiv w \cdot \gamma \bmod q$

$compute\ \ v \equiv (\alpha^{t_1} \cdot \beta^{t_2} \bmod p) \bmod q$

$if\ \ (v \equiv \gamma \bmod q)\ \ return\ TRUE$

$else\ \ return\ FALSE$

$\}$

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

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

$$=(wSHA(x)+wa\gamma) \bmod q$$

$$k \equiv (t_1+at_2) \bmod q$$

$$\alpha^{k} \equiv \alpha^{(t_1+at_2) \bmod q} \bmod p$$

$$\alpha^{k} \equiv \alpha^{t_1}\beta^{t_2} \bmod p$$

Take mod q *on both sides*

$$\gamma \equiv (\alpha^{t_1}\beta^{t_2} \bmod p) \bmod q$$

Security of DSA

- There are two ways to attack the DSA (attempt to recover the secret a)
  - Target the large cyclic group Z_p
  - Target the smaller group Z_q

Could you techniques such as Index Calculus. For a 1024 bit p, this method offers security of 80 bits

Cannot apply Index Calculus relies on Pollard rho for solving the discrete log, For 160 bit q, this offers security of 80 bits

Security of DSA

- There are two ways to attack the DSA (attempt to recover the secret a)
  - Target the large cyclic group Z_p
  - Target the smaller group Z_q

Could you techniques such as Index Calculus. For a 1024 bit p, this method offers security of 80 bits

Cannot apply Index Calculus relies on Pollard rho for solving the discrete log, For 160 bit q, this offers security of 80 bits

Thus the size of p dictates the size of q.

p	q	Signature
1024	160	320
2048	224	448
3072	256	512

Schnorr Signature Scheme
(uses a hash function to get smaller signatures)

Initialization

Choose a large prime p $(of\ \ size\ 1024\ bits)$

Choose a smaller prime q $(of\ \ size\ 160\ bits)\ and\ \ q \mid (p-1)$

Let $\alpha_0 \in Z_p^*$ be a primitive element

then $\alpha = \alpha_0^{(p-1)/q} \bmod p$ is the q$^{\text{th}}$ root of $1 \bmod p$

Choose a randomly from $(0 \le a < q)$

Compute $\beta = \alpha^a \bmod q$

Private : a

Private : α, β, p, q

Signing Message x

$sig(x)\{$

  select a secret random k $s.t.\ 1 \le k \le q-1.$

  $\gamma = h(x \,\|\, \alpha^k \bmod p)$

  $\delta = k + a\gamma \bmod p$

  $y = (\gamma, \delta)$

  $return\ (x, y)$

$\}$

Verifying Signature (x,y)

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

  $compute\ \ t_1 \equiv h(x \,\|\, \alpha^{\delta}\beta^{-\gamma} \bmod p)$

  $if\ \ (t_1 = \gamma)\ \ return\ TRUE$

  $else\ \ return\ FALSE$

$\}$