暨南大學
JINAN UNIVERSITY

Lecture 8: Digital Signature Schemes

-Cryptographic Algorithms and Protocols

Huang, Xiujie (黄秀姐)

Office: Nanhai Building, #411

E-mail: t_xiujie@jnu.edu.cn

Dept. Computer Science

Review

- **Public-Key Cryptosystem (PKC)**

  - **RSA**
  - **ElGamal**

  → Encryption, **Signature**

Outline

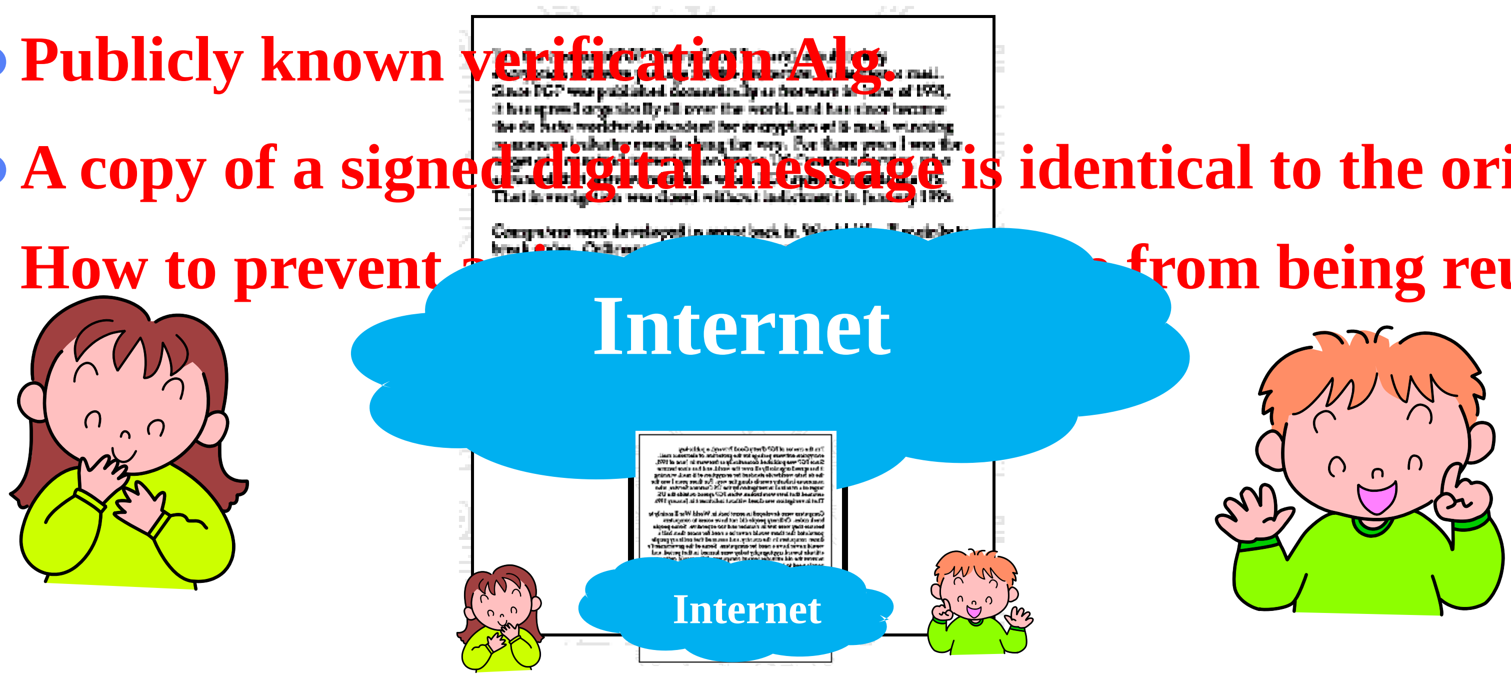
- <u>1. Introduction to Digital Signatures</u>
  - Basic Algorithms of DS
- 2. The RSA Signature
- 3. The ElGamal Signature
- 4. Security Requirements for Signature Schemes
- 5. Variants of the ElGamal Signature Schemes
  - Schnorr, DSA, ECDSA
- 6. Certificates
- 7. Signing and Encrypting, Hash

CS v.s. DS

To specify the person responsible for the signed message

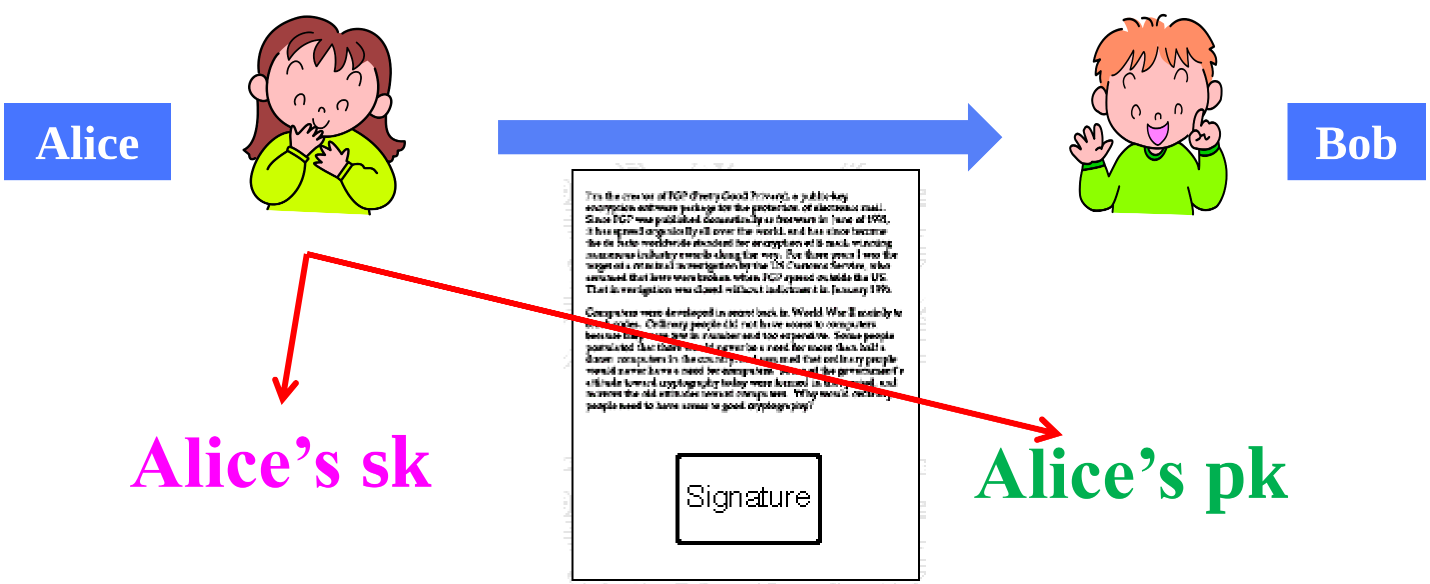
- **CS: a handwritten signature attached to a document**
- **DS: signing a electronic document**
  - **To bind the signature to the message**
  - **Publicly known verification Alg.**
  - **A copy of a signed digital message is identical to the original.**

    How to prevent a … e from being reused?



Basic Requirements of DS

- The signature can not be denied by the creator.
- The signature can not be forged by any other person.
- When a dispute happens, the reliable intermediary can verify the signature.



Three Basic Algorithms of DS

- **Set-up of Key Generation:** $\text{KeyGen}(\theta) = (\text{pk}, \text{sk})$

- **Signature:** $\text{sig}_{\text{sk}}(x) = y$

sig_{sk} is private

- **Verification:** $\text{ver}_{\text{pk}}(x,y) = \begin{cases} \textit{true}, \text{ if } y = \text{sig}_{\text{sk}}(x) \\ \textit{false}, \text{ if } y \neq \text{sig}_{\text{sk}}(x) \end{cases}$

ver_{pk} is public

The pair (x,y) is called a signed message.

Definition 8.1

Standards and Laws/Rules of DS

- **1994, US, Digital Signature Standard**
- **1995, China, GB 15851-1995 (Standard)**

- **1997, EU, 《EU Directive on a Community Framework for Electronic Signatures》**
- **2000, US, 《Electronic Signatures in Global and National Commerce Act》**
- **2004, China, 《中华人民共和国电子签名法》**

Useful Links

- **Some Useful Links:**
  - **An interesting introduction of digital signature**

    http://www.youdzone.com/signature.html
  - **中国数字认证网**

    http://www.gov.cn/flfg/2005-06/27/content_9785.htm#
  - **《中华人民共和国电子签名法》2004**

    http://www.gov.cn/flfg/2005-06/27/content_9785.htm#

Outline

- 1. Introduction to Digital Signatures
  - Basic Algorithms of DS
- 2. The RSA Signature
- 3. The ElGamal Signature
- 4. Security Requirements for Signature Schemes
- 5. Variants of the ElGamal Signature Schemes
  - Schnorr, DSA, ECDSA
- 6. Certificates
- 7. Signing and Encrypting, Hash

RSA Signature Scheme

- **Set-up of Key Generation:**
  1) Generate two large odd primes p and q such that $p \neq q$
  2) Compute $n = pq$ and $\phi(n) = (p-1)(q-1)$
  3) Choose a random number b $(1 < b < \phi(n))$ such that $\gcd(b, \phi(n))=1$
  4) Compute $a \equiv b^{-1} \pmod{\phi(n)}$
  5) Output: pk = (n, b), sk = (p, q, a)

Signer's public and secret keys

- **Signature:** $\mathrm{sig}_{\mathrm{sk}}(x) = x^a \bmod n$

- **Verification:** $\mathrm{ver}_{\mathrm{pk}}(x,y) = \mathrm{true} \leftrightarrow x \equiv y^b \pmod{n}$

Cryptosystem 8.1

Outline

- 1. Introduction to Digital Signatures
  - Basic Algorithms of DS
- 2. The RSA Signature
- 3. The ElGamal Signature
- 4. Security Requirements for Signature Schemes
- 5. Variants of the ElGamal Signature Schemes
  - Schnorr, DSA, ECDSA
- 6. Certificates
- 7. Signing and Encrypting, Hash

The ElGamal Signature Scheme

◆ **Set-up of Key Generation:**

1) **Generate a large prime p such that the DLP in Z_p^* is infeasible**
2) **Choose a primitive element $\alpha \in Z_p^*$**
3) **Choose a random number a $(0 < a < q)$ and Compute $\beta \equiv \alpha^a \pmod{p}$**
4) **Output: pk = (p, α, β), sk = (a)**

◆ **Signature:**

Non-deterministic:
one message,
many signatures

1) **Choose a secret random number k in Z_{p-1}^***
2) **Compute $sig_{sk}(x) = \boxed{(\gamma, \delta)}$ where $\gamma = \alpha^k \bmod p$ & $\delta = (x-a\gamma)k^{-1} \bmod p\text{-}1$**

◆ **Verification: $ver_{pk}(x, (\gamma, \delta)) = \text{true} \leftrightarrow \boxed{\beta^\gamma \gamma^\delta \equiv \alpha^x \pmod{p}}$**

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

$$= \beta^\gamma\alpha^{(x-a\gamma)} = \alpha^x\beta^\gamma\alpha^{(-a\gamma)} = \alpha^x$$

An Example of the ElGamal Signature

Example 8.1 on Pages 316

Suppose we take $p = 467$, $\alpha = 2$, $\boxed{a = 127}$; then

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

Suppose Alice wants to sign the message $x = 100$ and she chooses the random value $\boxed{k = 213}$ (note that $\gcd(213, 466) = 1$ and $213^{-1} \bmod 466 = 431$). Then

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

and

$$\delta = (100 - 127 \times 29)431 \bmod 466 = 51.$$

$\mathrm{sig}_{sk}(x) = (\gamma, \delta)$

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

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

Anyone can verify this signature by checking that

$$132^{29} 29^{51} \equiv 189 \pmod{467}$$

and

$$2^{100} \equiv 189 \pmod{467}.$$

$\mathrm{ver}_{pk}(x, (\gamma, \delta)) = \text{true}$

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

Hence, the signature is valid. $\square$

Outline

- **1. Introduction to Digital Signatures**
  - **Basic Algorithms of DS**
- **2. The RSA Signature**
- **3. The ElGamal Signature**
- **4. Security Requirements for Signature Schemes**
- **5. Variants of the ElGamal Signature Schemes**
  - **Schnorr, DSA, ECDSA**
- **6. Certificates**
- **7. Signing and Encrypting, Hash**

Attack Models

- **Key-only attack**

Oscar possesses Alice's public key, i.e., the verification function, $\mathbf{ver}_K$.

- **Known message attack**

Oscar possesses a list of messages previously signed by Alice, say

$$(x_1, y_1), (x_2, y_2), \ldots,$$

where the x_i's are messages and the y_i's are Alice's signatures on these messages (so $y_i = \mathbf{sig}_K(x_i)$, $i = 1, 2, \ldots$).

- **Chosen message attack**

Oscar requests Alice's signatures on a list of messages. Therefore he chooses messages $x_1, x_2, \ldots,$ and Alice supplies her signatures on these messages, namely, $y_i = \mathbf{sig}_K(x_i)$, $i = 1, 2, \ldots$.

Adversarial Goals

- **Total break**

The adversary is able to determine Alice's private key, i.e., the signing function $\mathbf{sig}_K$. Therefore he can create valid signatures on any message.

- **Selective forgery**

With some non-negligible probability, the adversary is able to create a valid signature on a message chosen by someone else. In other words, if the adversary is given a message x, then he can determine (with some probability) the signature y such that $\mathbf{ver}_K(x, y) = \mathtt{true}$. The message x should not be one that has previously been signed by Alice.

- **Existential forgery**

The adversary is able to create a valid signature for at least one message. In other words, the adversary can create a pair (x, y) where x is a message and $\mathbf{ver}_K(x, y) = \mathtt{true}$. The message x should not be one that has previously been signed by Alice.

Security Criteria of DS

- **Computational security: computational effort**

- **Provable security: reduction**

- **Unconditional security: even with infinite computational resources**

- **A DS cannot be unconditionally secure since the verification algorithm is public.**

Security of the RSA Signature

- Existential forgery using <u>a key-only attack</u>
  - Choose a signature y
  - Forge: (x, y) using the verification algorithm: $x = y^b \bmod n$
- Existential forgery using <u>a known message attack</u>
  - Know (x_1, y_1) & (x_2, y_2)
  - Forge: $(x_1 x_2 \bmod n, y_1 y_2 \bmod n)$
- Selective forgery using <u>a chosen message attack</u>
  - Given x
  - Forge: (x, y) by factoring $x = x_1 x_2 \bmod n$ and queries

Security of the ElGamal Signature

- **1.Existential forgery under a key-only attack**
- **2.Existential forgery under a known message attack**
- **3.Misuse**
  - **3.1The random value *k* used in signing is revealed.**
  - **3.2Use the same *k* to sign two different messages.**

Security of the ElGamal Signature

1.Existential forgery under a key-only attack

Suppose i and j are integers such that $0 \leq i \leq p-2$, $0 \leq j \leq p-2$, and suppose we express γ in the form $\gamma = \alpha^i \beta^j \bmod p$. Then the verification condition is

$$\alpha^x \equiv \beta^\gamma (\alpha^i \beta^j)^\delta \pmod p.$$

This is equivalent to

$$\alpha^{x-i\delta} \equiv \beta^{\gamma+j\delta} \pmod p.$$

This latter congruence will be satisfied if

$$x - i\delta \equiv 0 \pmod{p-1}$$

and

$$\gamma + j\delta \equiv 0 \pmod{p-1}.$$

Given i and j, we can easily solve these two congruences modulo $p-1$ for δ and x, provided that $\gcd(j, p-1) = 1$. We obtain the following:

$$\gamma = \alpha^i \beta^j \bmod p,$$

$$\delta = -\gamma j^{-1} \bmod (p-1), \quad \text{and}$$

$$x = -\gamma i j^{-1} \bmod (p-1).$$

By the way in which we constructed (γ, δ), it is clear that it is a valid signature for the message x.

- **Verification:** $ver_{pk}(x, (\gamma, \delta)) = true \leftrightarrow \beta^\gamma \gamma^\delta \equiv \alpha^x \pmod p$

Security of the ElGamal Signature

2.Existential forgery under a known message attack

Suppose (γ, δ) is a valid signature for a message x. Then it is possible for Oscar to sign various other messages. Suppose h, i and j are integers, $0 \leq h, i, j \leq p-2$, and $\gcd(h\gamma - j\delta, p-1) = 1$. Compute the following:

$$\lambda = \gamma^h \alpha^i \beta^j \bmod p$$

$$\mu = \delta\lambda(h\gamma - j\delta)^{-1} \bmod (p-1), \quad \text{and}$$

$$x' = \lambda(hx + i\delta)(h\gamma - j\delta)^{-1} \bmod (p-1).$$

Then, it is tedious but straightforward to check that the verification condition

$$\beta^\lambda \lambda^\mu \equiv \alpha^{x'} \pmod p$$

holds. Hence (λ, μ) is a valid signature for x'.

◆ **Verification:** $\mathrm{ver}_{pk}(x, (\gamma, \delta)) = \mathrm{true} \leftrightarrow \beta^\gamma \gamma^\delta \equiv \alpha^x \pmod p$

Security of the ElGamal Signature

3.Misuse

- **3.1The random value *k* used in signing is revealed.**

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

Once a is known, then the system is completely broken and Oscar can forge signatures at will.

Security of the ElGamal Signature

3.Misuse

3.2Use the same *k* to sign two different messages.

Suppose (γ, δ_1) is a signature on x_1 and (γ, δ_2) is a signature on x_2. Then we have

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

and

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

Thus

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

Writing $\gamma = \alpha^k$, we obtain the following equation in the unknown k:

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

which is equivalent to

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

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

$$x' = \frac{x_1 - x_2}{d}$$

$$\delta' = \frac{\delta_1 - \delta_2}{d}$$

$$p' = \frac{p-1}{d}.$$

◆ **Verification:** $ver_{pk}(x, (\gamma, \delta)) = true \leftrightarrow \beta^\gamma \gamma^\delta \equiv \alpha^x \pmod p$

Security of the ElGamal Signature

3.Misuse

- **3.2Use the same *k* to sign two different messages.**

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:

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

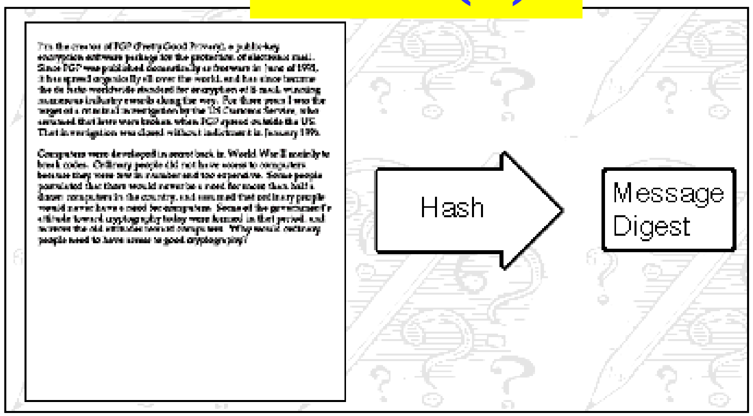
for some i, $0 \leq i \leq d-1$. Of these d candidate values, the (unique) correct one can be determined by testing the condition

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

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

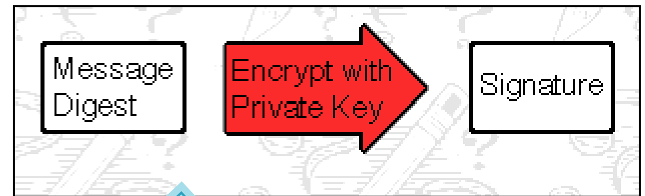
Signatures and Hash Functions

$z = h(x)$

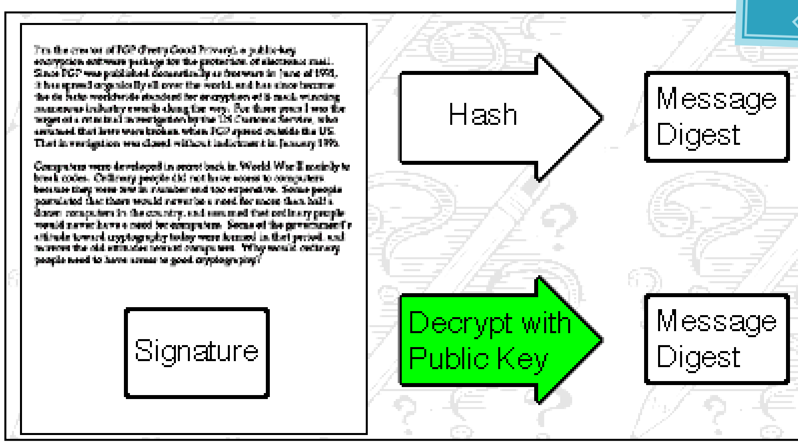


+

$y = \mathrm{sig}_{sk}(z)$
or $\mathrm{Enc}_{sk}(z)$



(x,y)



$z = h(x)$

$\mathrm{ver}_{pk}(z,y) = \mathrm{true}$
or $z = \mathrm{Dec}_{pk}(y)$

Outline

- 1. Introduction to Digital Signatures
  - Basic Algorithms of DS
- 2. The RSA Signature
- 3. The ElGamal Signature
- 4. Security Requirements for Signature Schemes
- 5. Variants of the ElGamal Signature Schemes
  - Schnorr, DSA, ECDSA
- 6. Certificates
- 7. Signing and Encrypting, Hash

Variants of the ElGamal Signature

- **The Schnorr Signature Scheme**
  - Proposed in August 1989
- **The Digital Signature Algorithm**
  - Proposed in August 1991
  - Published in May 1994, Adopted as a standard in Dec. 1994 by NIST
  - A Combination of Hash and Signature (ideas from Schnorr Sig)
- **The Elliptic Curve DSA**

Schnorr Signature Scheme

Cryptosystem 8.3

- **Set-up of Key Generation:**
  1) **Generate** a large prime p such that the DLP in Z_p^* is infeasible
  2) **Choose** a large prime q that divides p-1.
  3) **Choose** a qth root of 1 modulo p, i.e., $\alpha \in Z_p^*$
  4) **Choose** a random number a $(0 < a < q)$ **and Compute** $\beta \equiv \alpha^a \pmod p$
  5) **Choose** a secure hash function $h : \{0,1\}^* \to Z_q$
  6) **Output:** pk = (p, q, α, β, h), sk = (a)

- **Signature:**
  1) **Choose** a secret random number k in Z_{q-1}
  2) **Compute** $\mathrm{sig}_{sk}(x) = \boxed{(\gamma, \delta)}$

     where $\boxed{\gamma = h\,(x \| \alpha^k \bmod p)}$ & $\boxed{\delta = k + a\gamma \bmod q}$

Non-deterministic:
one message,
many signatures

- **Verification:** $\mathrm{ver}_{pk}(x, (\gamma, \delta)) = \text{true} \leftrightarrow \boxed{h(x \| \alpha^{\delta}\beta^{-\gamma} \bmod p) = \gamma}$

Digital Signature Algorithm

Cryptosystem 8.4

- **Set-up of Key Generation:**
  1) **Generate** a 2048-bit prime p such that the DLP in Z_p^* is infeasible
  2) **Choose** a 224-bit prime q that divides p-1.
  3) **Choose** a qth root of 1 modulo p, i.e., $\alpha \in Z_p^*$
  4) **Choose** a random number a ($0 < a < q$) **and Compute** $\beta \equiv \alpha^a \pmod p$
  5) **Output:** pk = (p, q, α, β), sk = (a)

- **Signature:**
  1) **Choose** a secret random number k in Z_{q-1}
  2) **Compute** $sig_{sk}(x) = \boxed{(\gamma, \delta)}$

     where $\boxed{\gamma = (\alpha^k \bmod p) \bmod q}$ & $\boxed{\delta = (\text{SHA3-224}(x)+a\gamma)\, k^{-1} \bmod q}$

Non-deterministic:
one message,
many signatures

- **Verification:** $ver_{pk}(x, (\gamma, \delta)) = \text{true} \leftrightarrow \boxed{(\alpha^{e1}\beta^{e2} \bmod p) \bmod q = \gamma}$

  where $\boxed{e1 = \text{SHA3-224}(x)\delta^{-1} \bmod q \ \ \&\ e2 = \gamma\, \delta^{-1} \bmod q}$

Summary

- 1. Introduction to Digital Signatures
  - Basic Algorithms of DS
- 2. The RSA Signature
- 3. The ElGamal Signature
- 4. Security Requirements for Signature Schemes
- 5. Variants of the ElGamal Signature Schemes
  - Schnorr, DSA, ECDSA
- 6. Certificates
- 7. Signing and Encrypting, Hash

Homework 7

Exercises: 8.3, 8.8 (for the cases of the ElGamal Signature Scheme.)

Thank you!

Questions?