Introducción a la Criptografía y a la Seguridad de la Información

Part 8
Digital Signatures

Jorge E. Camargo

Agenda

- Digital Signatures

➢How it works?

➢RSA Signature Scheme
  — In Practice
  — Example

➢ ElGamal Signature Scheme
  — In Practice
  — Example
  — Forging a Signature
  — Remarks

➢Digital Signature Standard (DSS)
  — Remarks

How does it work?

Assume you were going to send the draft of a contract to your lawyer in another town. You want to give your lawyer the assurance that it was unchanged from what you sent and that it is really from you.

- You copy-and-paste the contract into an e-mail note.
- Using special software, you obtain a message hash (mathematical summary) of the contract.
- You then use a private key that you have previously obtained from a public-private key authority to encrypt the hash. The encrypted hash becomes your digital signature of the message.

At the other end, your lawyer receives the message. To make sure it's intact and from you, your lawyer makes a hash of the received message. Your lawyer then uses your public key to decrypt the message hash or summary. If the hashes match, the received message is valid.

RSA Signature Scheme

Let $n = pq$, where p and q are primes. Let $\mathcal{M} = S = \mathbb{Z}_n$, and define

$\mathcal{K} = \{ (n, p, q, e, d) : n = pq, p, q$ prime, $ed \equiv 1 \bmod \phi(n) \}$.

The value (e, n) is the public key, and (d, n) is the private key.

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

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

and

$$ver_k^-(x, y) = \text{true} \Leftrightarrow x \equiv y^e \pmod{n}$$

$(x, y \in \mathbb{Z}_n)$.

Clearly, the security level of the RSA signature scheme is exactly the same as for the RSA cryptosystem.

In Practice

(a) Setup:

1) Generate two large random primes p and q ($p \neq q$ and $|p| \approx |q|$)

2) Compute $n = pq$ and $\phi = (p-1)(q-1)$

3) Select a random integer e, $1 < e < \phi$ such that GCD(e, ϕ)=1

4) Use EEA(ϕ, e) algorithm to find d such that $ed \equiv 1 \pmod{\phi}$

5) Publish (e, n) as verification (public) key

6) Keep (d, n) as signing (secret) key

(b) Signature Generation:

1) $y = sig_{(d,n)}$ (x) = PowerMod(x, d, n) where x is some message

(c) Signature Verification:

1) Compute $ver_{(e,n)}$ (x, y) = 1(true) if PowerMod(y, e, n) = x, otherwise $ver_{(e,n)}$ (x, y) = 0 (false)

Example

(a) Setup:

1) Let p=47, q=71

2) Let $n = pq$ =3337, ϕ = 46 × 70 = 3220

3) Choose e (at random) to be 79

4) Compute EEA(3220, 79) to find d

ϕ	e	q	$gcd(\phi, e)$	x	y	
3220	79	40	1	-25	1019	-> d=1019
79	60	1	1	19	-25	
60	19	3	1	-6	19	
19	3	6	1	1	-6	
3	1	3	1	0	1	
1	0	-	1	1	0	

5) Verification key = (79, 3337)

6) Signing key = (1019, 3337)

Message x =999

(b) Signature Generation:

1) $y = sig_{(1019,3337)}(999)$ = PowerMod(999, 1019, 3337)

i	9	8	7	6	5	4	3	2	1	0	
b_i	1	1	1	1	1	1	1	0	1	1	
x	999	835	2439	2415	786	54	3220	341	412	1264	$=y$

(c) Signature Verification:

1) Compute PowerMod(1264, 79, 3337)

i	6	5	4	3	2	1	0
b_i	1	0	0	1	1	1	1
y	1264	2610	1283	289	1212	1709	999

PowerMod(1264, 79, 3337) = 999 = $x \Rightarrow ver_{(79,3337)}(1264,999)$ = 1(*true*)

ElGamal Signature Scheme
1985

Let p be a prime such that the discrete logarithm problem in $\mathbb{Z}_p$ is intractable and let $\alpha \in \mathbb{Z}_p^*$ be a primitive element. Let $\mathcal{M} = \mathbb{Z}_p^*$, $S = \mathbb{Z}_p^* \times \mathbb{Z}_{p-1}$, and define

$$\mathcal{K} = \{(p, \alpha, a, \beta) : \beta \equiv \alpha^a \bmod p\}.$$

The values of p, α and β are public and a is secret. For $K = (p, \alpha, a, \beta) \in \mathcal{K}$ and for a secret random number $k \in \mathbb{Z}_{p-1}^*$ define

$$sig_K(x, k) = (\gamma, \delta)$$

where $\gamma = \alpha^k \pmod p$ and $\delta = (x - a\gamma)k^{-1} \pmod{p-1}$.
For $x, \gamma \in \mathbb{Z}_p^*$ and $\delta \in \mathbb{Z}_{p-1}$, define

$$ver_K(x, \gamma, \delta) = \text{true} \Leftrightarrow \beta^\gamma \gamma^\delta \equiv \alpha^x \pmod p$$

In Practice

(a) Setup:

1) Generate a large random prime p and a generator α of $\mathbb{Z}_p^*$
2) Select a random integer a, $1 \leq a \leq p - 2$
3) Compute β = PowerMod(α, a , p)
4) Publish (p, α, β) as signing (public) key
5) Keep a as verification (secret) key

(b) Signature Generation:

1) Select a random integer k, $1 \leq k \leq p - 2$
2) Compute γ = PowerMod(α, k, p)
3) Use EEA(k, $p - 1$) algorithm to find k^{-1}
4) Compute δ = (x − aγ) k^{-1} mod ($p - 1$), where x is the message
5) y = (γ, δ)

In Practice (cont.)

(c) Signature Verification:

1) Verify that $1 \leq \gamma \leq p - 1$; if not, then reject the signature
2) Compute v_1' = PowerMod(β, γ, p)
3) Compute v_1'' = PowerMod(γ, δ, p)
4) Compute v_1 = $v_1' \, v_1''$ mod p
5) Compute v_2 = PowerMod(α, x, p)
6) Accept the signature iff $v_1 = v_2$

Example

(a) Setup:

1) Let p =2579, α=2

2) Let a =765

3) β=PowerMod(2, 765, 2579) $b = 765_{10} = \langle 1011111101 \rangle_2$

i	9	8	7	6	5	4	3	2	1	0	
b_i	1	0	1	1	1	1	1	1	0	1	
α	2	4	32	2048	1700	461	2086	1246	2537	949	$=\beta$

4) Public (2579, 2, 949) as signing (public) key

5) Keep 765 as verification (secret) key

(b) Signature Generation:

1) k=853

2) γ =PowerMod(2, 853, 2579)

b = 853_{10} =⟨1101010101 ⟩$_2$

i	9	8	7	6	5	4	3	2	1	0	
b_i	1	1	0	1	0	1	0	1	0	1	
α	2	8	64	455	705	1135	1304	1710	2093	435	=γ

3) Compute EEA(2578, 853) to find k^{-1}

$p-1$	k	q	$\gcd(p-1,k)$	x	y	
2578	853	3	1	-404	1221	=k^{-1}
853	19	44	1	9	-404	
19	17	1	1	-8	9	
17	2	8	1	1	-8	
2	1	2	1	0	1	
1	0	-	1	1	0	

4) x=999, δ=(999-765×435)× 1221 mod 2578 =690

5) y =(435, 690)

(c) Signature Verification:

1) $1 \le 435 \le 2578$

2) v_1'= PowerMod(949, 435, 2579) $b = 435_{10} = \langle 110110011 \rangle_2$

i	9	8	7	6	5	4	3	2	1	0	
b_i	1	1	0	1	1	0	0	1	1		
β	949	65	1646	1676	1928	845	2221	1996	1710	$=v_1'$	

3) Compute v_1'' = PowerMod(435, 690, 2579)
$b = 690_{10} = \langle 1010110010 \rangle_2$

i	9	8	7	6	5	4	3	2	1	0	
b_i	1	0	1	0	1	1	0	0	1	0	
γ	435	958	719	1161	2248	1694	1788	1563	1670	1001	$=v_1''$

4) Compute $v_1 = 1710 \times 1001 \bmod 2579 = 1833$

5) Compute v_2 = PowerMod(2, 999, 2579)
$b = 999_{10} = \langle 1111100111 \rangle_2$

i	9	8	7	6	5	4	3	2	1	0	
b_i	1	1	1	1	1	0	0	1	1	1	
α	2	8	128	1820	1928	845	2221	1007	1004	1833	$=v_2$

6) We Accept the signature since $v_1 = v_2$

Forging a Signature

Let $i, j \in \mathbb{Z}$, $0 \leq i, j \leq p-2$ and $gcd(j, p-1) = 1$

Can Eve sign a random message $x = -\gamma i j^{-1} \bmod (p-1)$ by choosing $\gamma = \alpha^i \beta^j \bmod p$ and $\delta = -\gamma j^{-1} \bmod (p-1)$?

where j^{-1} is computed modulo $(p-1)$.

- Try answering this question by using p=467, α=2, β=132, i=99, j=179.

Remarks

- As for the ElGamal cryptosystem, the security of the ElGamal signature scheme is based on the discrete logarithm problem modulo p
- One feature that is different from RSA signature scheme is that, with the ElGamal method, there are many different signatures that are valid for a given message
- The ElGamal signature scheme is an example of signatures with appendix. The message is not easily recovered from the signature (γ, δ). The message x must be included in the verification process. This is in contrast with the RSA signature scheme, which is a message recovery scheme. In this case, the message is readily obtained from the signature y. Therefore, only y needs to be sent since anyone can deduce x as $y^e \pmod n$

Digital Signature Standard (DSS)

1993 by National Institute of Standards and Technology (NIST)

Let p be a 1024bit prime and let q be a 160bit prime that divides $p - 1$.

Let $\alpha \in \mathbb{Z}_p^*$ be an element of order q. Let $\mathcal{M} = \{0,1\}^*$, $S = \mathbb{Z}_q \times \mathbb{Z}_q$, and define

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

where $0 \leq a \leq q - 1$. The values p, q, α and β are the public key, and a is the private key.

For $K = (p, q, \alpha, a, \beta)$, and for a (secret) random number k, $1 \leq k \leq q - 1$, define

$$sig_K(x, k) = (\gamma, \delta),$$

Where

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

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

The DSS is a variant of ElGamal Signature Scheme

Digital Signature Standard (cont.)

For $x \in 0,1^*$ and $\gamma, \delta \in \mathbb{Z}_p^*$, verification is done by performing the following computations:

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

$ver_K^-(x, (\gamma, \delta)) = \text{true} \Leftrightarrow (\alpha^{e_1}\beta^{e_2} \bmod p) \bmod q = \gamma.$

Remarks

- Digital Signature Standard (DSS) is the digital signature algorithm (DSA) developed by the U.S. National Security Agency (NSA) to generate a digital signature for the authentication of electronic documents
- ElGamal scheme is a pair (γ, δ), where γ and δ are integers modulo p and $p - 1$, respectively. In 1994 it was already necessary to choose p as a 512-bit prime in order to make the ElGamal scheme secure. Thus, an ElGamal signature can be expected to have 1024 bits. This is too long for typical applications such as smart cards. Nowadays it would be preferable to choose a 1024-bit prime p, leading to 2048-bit ElGamal signatures.
- DSS signs 160-bit messages with a 320-bit signature, but the computations are done using a prime modulus p that has between 512 and 1024 bits.

References

- Pinzon, Yoan. “Introducción a la criptografía y a la seguridad de la información”, 2013.
- Menezes A., Handbook of applied Cryptografy, 5th Edition. CRC Press, 2001.
- A Graduate Course in Applied Cryptography by D. Boneh and V. Shoup
- H. Delfs and H. Knebl, Introduction to Cryptography, 3rd ed. 2015.
- J. A. Buchmann, Introduction to Cryptography. 2004.
- T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, Ch. 31 in Introduction to Algorithms, 3rd ed. 2009.
- B. Lomas de Zamora: Gradi. Hacking desde cero, Fox Andina, 2011.
- Secure Coding Working Group, Japan Smartphone Security Association (JSSEC), Android Application Secure Design/Secure Coding Guidebook, 2017.