Chapter 7: Digital signatures

Example: Assume that each user A uses a public-key cryptosystem (e_A, d_A).

Signing a message w by a user A, so that any user can verify the signature;

$$d_A(w)$$

Signing a message w by a user A so that only user B can verify the signature;

$$e_B(d_A(w))$$

Sending a message w, and a signed message digest of w, obtained through a hash function h:

$$(w, d_A(h(w)))$$

Example Assume Alice succeeds to factor the integer that Bob used, as modulus, to sign his will, using RSA, 20 years ago. Even the key has already expired, Alice could rewrite Bob's will, leaving fortune to her, and date it 20 years ago.

Moral: It may pay of to factor a single integers using many years of many computers power.

Digital signatures – basic goals

Digital signatures – basic goals

Digital signatures

Indeed, if an active enemy, called tamperer, intercepts the message, then he can compute

$$d_T(e_A(c)) = d_T(e_B(w))$$

and can send the outcome to Bob, pretending that it is from him/tamperer (without being able to decrypt/know the message).

Any public-key cryptosystem in which the plaintext and cryptotext spaces are the same can be used for digital signature.

Digital Signature Schemes I

Digital Signature Schemes II

Digital Signature Schemes II

Attacks on digital signatures

Attacks on digital signatures

A digital signature of one bit

Signature of a bit *b:*

$$(b, k_b).$$

Verification of such a signature

$$s_b = f\,(k_b)$$

SECURITY?

RSA signatures and their attacks

[illegible]

$s =$ [illegible]

[illegible]

Attacks

• It might happen that Bob accepts a signature not produced by Alice. Indeed, let Eve, using Alice's public key, compute w^e and say that (w^e, w) is a message signed by Alice.

Everybody verifying Alice's signature gets $w^e = w^e$.

• Some new signatures can be produced without knowing the secret key.

Indeed, is σ and σ are signatures for w_1 and w_2, then σ . and σ are signatures for w_1w_2 and w_1^{-1}.

ENCRYPTION versus SIGNATURE

ENCRYPTION versus SIGNATURE

PUBLIC-KEY SIGNATURES

Signing:	$d_U(w)$
Verification of the signature:	$e_U(d_U(w))$

DIGITAL SIGNATURE SYSTEMS – simplified version

$\{$

FROM PKC to DSS - again

Signing of a message w by a user A so that only user B can verify the signature;

$$e_B(d_A(w)).$$

Sending of a message w and a signed message digest of w obtained by using a (standard) hash function h:

$$(w, d_A(h(w))).$$

If only signature (but not the encryption of the message) are of importance, then it suffices that Alice sends to Bob

$$(w, d_A(w)).$$

ElGamal signatures

Signature of a message *w*: Let $r \in Z_{p-1}^*$ be randomly chosen and kept secret.

$$sig(w, r) = (a, b),$$

where $$a = q^{r} \bmod p$$

and $$b = (w - xa)r^{-1} \pmod{(p-1)}.$$

Verification: accept a signature *(a,b)* of w as valid if

$$y^a a^b \equiv q^w \pmod{p}$$

$$\text{(Indeed: } y^a a^b \equiv q^{ax} q^{rb} \equiv q^{ax + w - ax + k(p-1)} \equiv q^w \pmod{p})$$

ElGamal signatures - example

ElGamal signatures - example

Security of ElGamal signatures

Security of ElGamal signatures

1. If Eve chooses *a* and *b* and tries to determine such *w that (a,b) is signature of w*, then she has to compute discrete logarithm

$$\lg_q y^a a^b.$$

Hence, Eve can not sign a “random” message this way.

Forging and misusing of ElGamal signatures

Forging and misusing of ElGamal signatures

Digital Signature Standard

Design of DSA

1. The following global public key components are chosen:

- p - a random l-bit prime, $512 \leq l \leq 1024$, $l = 64k$.
- q - a random 160-bit prime dividing p -1.
- $r = h^{(p-1)/q} \bmod p$, where h is a random primitive element of Z_p, such that r>1 (observe that r is a q-th root of 1 mod p).

2. The following user's private key components are chosen:

- x - a random integer (once), $0 < x < q$, and $y = r^x \bmod p$ *is made public*.

3. Key is $K = (p, q, r, x, y)$

Digital Signature Standard

Verification of signature (a, b)

- compute $z = b^{-1} \bmod q$
- compute $u_1 = wz \bmod q$,

  $u_2 = az \bmod q$

verification:

$$ver_K(w, a, \mathrm{b}) = \text{ true} \iff (r^{u1} y^{u2} \bmod p) \bmod q = a$$

From ElGamal to DSA

From ElGamal to DSA

Any proposal for digital signature standard has to go through a very careful scrutiny. Why?

Encryption of a message is usually done only once and therefore it usually suffices to use a cryptosystem that is secure **at the time of the encryption**.

On the other hand, a signed message could be a contract or a will and it can happen that it will be needed to verify a signature **many years after the message is signed**.

Since ElGamal signature is no more secure than discrete logarithm, it is necessary to use large *p*, with at least 512 bits.

However, with ElGamal this would lead to signatures with at least 1024 bits what is too much for such applications as smart cards.

In DSA a 160 bit message is signed using 320-bit signature, but computation is done modulo with 512-1024 bits.

Observe that *y* and *a* are also *q*-roots of 1. Hence any exponents of *r*,*y* and *a* can be reduced module *q* without affecting the verification condition.

This allowed to change ElGamal verification condition: $y^{a} a^{b} = q^{w}$.

Fiat-Shamir signature scheme

Fiat-Shamir signature scheme

[illegible]

(2) Alice uses a publically known hash function h to compute

$$H=h(wx_1x_2\dots x_t)$$

and then uses first *kt* bits of H, denoted as b_{ij}, $1 \le i \le t$, $1 \le j \le k$ as follows.

(3) Alice computes $y_1,\dots,y_t$ [illegible]

(4) Alice sends to Bob *w*, all b_{ij} all y_i and also h

{ Bob already knows Alice's public key $v_1,\dots,v_k$ }

(5) Bob computes $z_1,\dots,z_k$ [illegible]

and verifies that the first $k \times t$ bits of $h(wx_1x_2\dots x_t)$ are the b_{ij} values that Alice has sent to him.

Security of this signature scheme is 2^{-kt}.

Advantage over the RSA-based signature scheme: only about 5% of modular multiplications are needed.

Sad story

Sad story

IV054 Ong-Schnorr-Shamir subliminal channel scheme

Ong-Schnorr-Shamir subliminal channel scheme

Yes. Alice and Bob create first the following communication scheme:

They choose a large n and an integer k such that gcd(n, k) = 1.

They calculate $h = k^{-2} \bmod n = (k^{-1})^2 \bmod n$.

Public key: h, n

Trapdoor information: k

Let secret message Alice wants to send be w (it has to be such that gcd(w, n) =1)

Denote a harmless message she uses by w' (it has to be such that gcd(w',n) = 1)

Signing by Alice: $S_1 =$ [illegible]

$S_2 =$ [illegible]

Signature: (S_1, S_2). Alice then sends to Bob (w', S_1, S_2)

Signature verification by Walter: $w' = S_1^2 - hS_2^2 \pmod n$

Decryption by Bob: [illegible]

One-time signatures

Signing of a message $x = x_1 \dots x_k \in \{0,1\}^k$

$$sig(x_1 \dots x_k) = (y_{1,x1}, \dots, y_{k,xk}) = (a_1, \dots, a_k)$$ - notation

and

$$ver_K(x_1 \dots x_k, a_1, \dots, a_k) = \text{true} \iff f(a_i) = z_{i,xi},\ 1 \le i \le k$$

Eve cannot forge a signature because she is unable to invert one-way functions.

Important note: Lampert signature scheme can be used to sign only one message.

Undeniable signatures I

Undeniable signatures I

Undeniable signatures II

Chaum-van Antwerpen undeniable signature schemes (CAUSS)

- p, r are primes $p = 2r + 1$
- $q \in Z_p^*$ is of order r;
- $1 \leq x \leq r - 1$, $y = q^x \bmod p$;
- G is a multiplicative subgroup of Z_p^* of order q (G consists of quadratic residues modulo p).

Key space: $K = \{p, q, x, y\}$; p, q, y are public, x € G is secret.

Signature: $s = sig_K(w) = w^x \bmod p$.

Fooling and Disallowed protocol

Disallowed protocol

Basic idea: After receiving a signature s Alice initiates two independent and unsuccessful runs of the verification protocol. Finally, she performs a “consistency check" to determine whether Bob has formed his responses according to the protocol.

- Alice chooses $e_1, e_2 \in Z_r^*$.
- Alice computes $c = s^{e1}y^{e2}$ mod p and sends it to Bob.
- Bob computes $d = c^{x\wedge(-1) \bmod r}$ mod p and sends it to Alice.
- Alice verifies that $d \neq w^{e1}q^{e2}$ (mod p).
- Alice chooses $f_1, f_2 \in Z_r^*$.
- Alice computes $C = s^{f1}y^{f2}$ mod p and sends it to Bob.
- Bob computes $D = C^{x\wedge(-1) \bmod r}$ mod p and sends it to Alice.

Fooling and Disallowed protocol

- Alice verifies that $D \neq w^{f1} q^{f2} \pmod{p}$.
- Alice concludes that s is a forgery iff

$$(dq^{-e2})^{f1} \equiv (Dq^{-f2})^{e1} \pmod{p}.$$

CONCLUSIONS

It can be shown:

Bob can convince Alice that an invalid signature is a forgery. In order to that it is sufficient to show that if $s \neq w^{x}$, then

$$(dq^{-e2})^{f1} \equiv (Dq^{-f2})^{e1} \pmod{p}$$

what can be done using congruency relation from the design of the signature system and from the disallowed protocol.

Bob cannot make Alice believe that a valid signature is a forgery, except with a very small probability.

Signing of fingerprints

Signing of fingerprints

Collision-free hash functions revisited

Definition A hash function h is strongly collision-free if it is computationally infeasible to find messages w and w' such that $h(w) = h(w')$.

Example 2: Eve computes a signature y on a random fingerprint z and then find an x such that $z = h(x)$. Would she succeed (x,y) would be a valid signature.

In order to prevent the above attack, it is required that in signatures we use one-way hash functions.

It is not difficult to show that for hash-functions (strong) collision-free property implies the one-way property.

Timestamping

A method for timestamping of signatures:

In the following *pub* denotes some publically known information that could not be predicted before the day of the signature (for example, stock-market data).

Timestamping by Bob of a signature on a message *w, using a hash function h.*

- Bob computes $z = h(w)$;
- Bob computes $z' = h(z \,||\, pub)$;
- Bob computes $y = sig(z')$;
- Bob publishes (z, pub, y) in the next days's newspaper.

It is now clear that signature was not be done after triple (z, pub, y) was published, but also not before the date *pub* was known.

Blind signatures

Blind signatures

Chum’s blind signatures

Chum’s blind signatures

Fail-then-stop signatures

Fail-then-stop signatures

Digital signatures with encryption and resending

2. Alice encrypts the signed message: $e_B(s_A(w))$ and sends it to Bob.

3. Bob decrypt the signed message: $d_B(e_B(s_A(w))) = s_A(w)$.

4. Bob verifies signature and recovers the message $v_A(s_A(w)) = w$.

Resending the message as a receipt

5. Bob signs and encrypts the message and sends to Alice $e_A(s_B(w))$.

6. Alice decrypts the message and verifies the signature.

Assume now: $v_x = e_x$, $s_x = d_x$ for all users x.

A surprising attack to the previous scheme

A surprising attack to the previous scheme

2. Later Mallot sends $e_B(s_A(w))$ to Bob pretending it is from him (from Mallot).

3. Bob decrypts and “verifies” the message by computing

$$e_M(d_B(e_B(d_A(w)))) = e_M(d_A(w)) \qquad \text{- a garbage.}$$

4. Bob goes on with the protocol and returns Mallot the receipt:

$$e_M(d_B(e_M(d_A(w))))$$

5. Mallot can then get w.

Indeed, Mallot can compute $e_A(d_M(e_B(d_M(e_M(d_B(e_M(d_A(w)))))))) = w$.

A MAN-IN-THE-MIDDLE attack

A MAN-IN-THE-MIDDLE attack

What can an active eavesdropper C do?

- C can learn $(e_A(e_A(w)\ B), A)$ and therefore $e_A(w')$, $w' = e_A(w)B$.
- C can now send to Alice the pair $(e_A(e_A(w')\ C), A)$.
- Alice, thinking that this is the step 1 of the protocol, acknowledges by sending the pair $(e_C(e_C(w')\ A), C)$ to C.
- C is now able to learn w' and therefore also $e_A(w)$.
- C now sends to Alice the pair $(e_A(e_A(w)\ C), A)$.
- Alice acknowledges by sending the pair $(e_C(e_C(w)\ A), C)$.
- C is now able to learn w.

Probabilistic signature schemes - PSS

Signing: of a message $w \in \{0,1\}^*$.

1. Choose random $r \in \{0,1\}^k$ and compute $m = h(w \,||\, r)$.
2. Compute $G(m) = (G_1(m), G_2(m))$ and $y = m \,||\, (G_1(m) \oplus r) \,||\, G_2(m)$.
3. Signature of w is $\sigma = f^{-1}(y)$.

Verification of a signed message (w, σ).

- Compute $f(\sigma)$ and decompose $f(\sigma) = m \,||\, t \,||\, u$, where $|m| = l$, $|t| = k$ and $|u| = n - (k+l)$.
- Compute $r = t \oplus G_1(m)$.
- Accept signature σ if $h(w \,||\, r) = m$ and $G_2(m) = u$; otherwise reject it.

Authenticated Diffie-Hellman key exchange

1. Alice chooses a random x and Bob chooses a random y.
2. Alice computes $q^{x} \bmod p$, and Bob computes $q^{y} \bmod p$.
3. Alice sends q^{x} to Bob.
4. Bob computes $K = q^{xy} \bmod p$.
5. Bob sends q^{y} and $e_K(s_B(q^{y}, q^{x}))$ to Alice.
6. Alice computes $K = q^{xy} \bmod p$.
7. Alice decrypts $e_K(s_B(q^{y}, q^{x}))$ to obtain $s_B(q^{y}, q^{x})$.
8. Alice verifies, using an authority, that v_B is Bob's verification algorithm.
9. Alice uses v_B to verify Bob's signature.
10. Alice sends $e_K(s_A(q^{x}, q^{y}))$ to Bob.
11. Bob decrypts, verifies v_A, and verifies Alice's signature.

An enhanced version of the above protocol is known as Station-to-Station protocol.

Security of digital signatures

Security of digital signatures

Treshold Signature Schemes

Treshold Signature Schemes

Digital Signatures - Observation

Digital Signatures - Observation

SPECIAL TYPES of DIGITAL SIGNATURES

SPECIAL TYPES of DIGITAL SIGNATURES

GROUP SIGNATURES

GROUP SIGNATURES

IV054 Unconditionally secure digital signatures

Unconditionally secure digital signatures