Cryptography

Digital Signatures

Professor: Marius Zimand

Digital signatures are meant to realize

- authentication of the sender
- nonrepudiation

(Note that authentication of sender is also achieved by MACs.)

How about this idea for a digital signature?

Scan your handwritten signature and append it to the document?

Does not work: anyone can take it and append it to their document.

We see that the signature must depend on the document.

A solution that works: use a PKC

We have seen that a PKC which satisfies $E_k(D_k(m)) = m$ can be used for digital signatures.

Example: RSA.

Alice has n_A, p_A, q_A, e_A, d_A.

She signs message m with $m^{d_A}(\mod n_A)$. So she sends to Bob

$$(m, m^{d_A}(\mod n_A))$$

Bob receives this and he needs to check the validity of the signature.

The verification algorithm consists in computing $(m^{d_A})^{e_A}(\mod n_A)$. If this equal to m, the signature is valid, otherwise it is not.

This is the idea for all digital signature algorithms:

- producing the signature requires the knowledge of the secret key owned by the signer.
- the signature must have a property linking it to the message and anyone (having the public key) can test that property.

In practice, people are using specialized digital signatures algorithms.

The generic schema for a digital signature is as follows:

digital signature schema: $(\mathcal{M}, \mathcal{A}, \mathcal{K}, S, V)$, where

- $\mathcal{M}$ is the set of messages
- $\mathcal{A}$ is the set of signatures
- $\mathcal{K}$ is the set of keys
- S is the signing algorithm. Formally,

  $$S = \{\text{sig}_k \mid \text{ one signature function for each key } k \in \mathcal{K}\},$$

  where each sig_k maps messages to signatures.

  The signing algorithm should be

  - efficient
  - secret (only the owner of k can do it).

- V is the verification algorithm, Formally,

  $$V = \{\text{ver}_k \mid \text{ one verification test for each key } k \in \mathcal{K}\},$$

  Requirement:

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

The verification test should be

- efficient
- public.

Security requirement (informallly): It should be computationally infeasible for someone who does not have k to produce a y that passes the test $\text{ver}(x, y)$.

El Gamal Signature Algorithm

parameters for the key

- p - large prime number
- α - primitive root of p
- a - randomly chosen in the range $\{2, \ldots, p-2\}$.
- $\beta = \alpha^a (\mod p)$ ("β is a in a box.")

(p, α, β) - public

a - secret key.

signature algorithm

We want to sign message m.

1. select random k such that $gcd(k, p-1) = 1$.
2. compute $r = \alpha^k (\mod p)$.
3. compute $s = k^{-1}(m - ar) (\mod (p-1))$.
4. The signature is the pair (r, s) and the signed message is $(m, (r, s))$.

verification algorithm

The user who makes the verification knows

- (p, α, β) - the public key
- $(m, (r, s))$ - the signed message

The verification is done by computing

$$v_1 = \beta^r \cdot r^s (\mod p) \text{ and } v_2 = \alpha^m (\mod p).$$

If $v_1 = v_2$, then the signature is valid, otherwise it is not.

Let's see that a good signature passes the verification test.

$$\begin{aligned} s &= k^{-1}(m - ar)(\mod (p-1)) \\ \Rightarrow sk &= m - ar(\mod (p-1)) \\ \Rightarrow m &= sk + ar(\mod (p-1)) \\ \Rightarrow v_2 &= \alpha^m = \alpha^{sk+ar}(\mod p) \\ &= (\alpha^a)^r \cdot (\alpha^k)^s = \beta^r \cdot r^s (\mod p). \end{aligned}$$

Informal analysis of security

Suppose that Eve wants to sign as Alice (who is the owner of the secret key a).

Eve needs an s that passes the verification test, i.e., she needs an s such that

$$\beta^r \cdot r^s = \alpha^m (\mod p).$$

So

$$r^s = \beta^{-r} \cdot \alpha^m (\mod p).$$

This is similar to the Discrete Log problem, so Eve cannot find such an s.

Digital Signature Algorithm (DSA)

proposed by NIST in 1991, and adopted as a federal standard in 1994.

parameters for the key

- q - prime number having 160 bits
- p - prime such that q divides $p-1$.
- g - primitive root of p.
- $\alpha = g^{(p-1)/q}(\mod p)$.

  Note that $\alpha^q = 1(\mod p)$.
- a - secret number randomly chosen in the range $\{0, \ldots, p-1\}$.
- $\beta = \alpha^a(\mod p)$

(p, q, α, β) - public key.

a - secret key

signature algorithm

1. select random secret k, $k \in \{1, \ldots, q-1\}$
2. compute $r = (\alpha^k(\mod p))(\mod q)$.
3. compute $s = k^{-1}(m + ar)(\mod q)$.
4. the signature is (r, s); the signed message is $(m, (r, s))$.

verification algorithm

1. compute $u_1 = s^{-1}m(\mod q)$.
2. compute $u_2 = s^{-1}r(\mod q)$.
3. compute $v = (\alpha^{u_1}\beta^{u_2}(\mod p))(\mod q)$.
4. if $v = r$, then accept the signature; otherwise reject the signature.

Let's check that a good signature passes the verification test.

$$\begin{aligned} m &= (sk - ar)(\mod (q)) \\ \Rightarrow s^{-1}m &= (k - a \cdot r \cdot s^{-1})(\mod q) \\ \Rightarrow \quad k &= s^{-1}m + a \cdot r \cdot s^{-1}(\mod q) \\ \Rightarrow &= u_1 + a \cdot u_2(\mod q). \end{aligned}$$

So

$$\begin{aligned} \alpha^k &= \alpha^{u_1 + a \cdot u_2}(\mod p) \\ &= \alpha^{u_1} \cdot \beta^{u_2}(\mod p). \end{aligned}$$

The first equality holds because $\alpha^q = 1(\mod p)$.

And finally:

$$(\alpha^k(\mod p))(\mod q) = (\alpha^{u_1} \cdot \beta^{u_2}(\mod p))(\mod q).$$

The LHS is r and the RHS is v, so $r = v$.

DSA is very similar to the El Gamal signature scheme; the major difference is that the r and s, the components of the signature, are guaranteed to be smaller than q, which has the fixed size of 160 bits. On the other hand, the security is based on

working in the larger field Z_p; thus we can pick an appropriate large size for p for which the DLP is hard and the signature will still be short.

(Toy) Example of DSA

- $q = 13, p = 4q + 1 = 53, g = 2$; g is a primitive root of p.
- $\alpha = g^{(p-1)/q}(\mod p) = 2^4(\mod 53) = 16$.
- $a = 3$ (the secret key)
- $\beta = \alpha^a(\mod p) = 16^3(\mod 53) = 15$.

Suppose we wish to sign the message $m = 5$.

For this

- we generate the ephemeral secret key $k = 2$
- compute $r = (\alpha^k(\mod p))(\mod q) = (16^2(\mod 53))(\mod 13) = 5$.
- compute $s = k^{-1}(m + ar)(\mod q) = 10$.
- the signature is the pair $(5, 10)$.

To verify the signature the receiver computes

- $u_1 = s^{-1}m(\mod q) = 7$
- $u_2 = s^{-1}r(\mod q) = 7$
- $v = (\alpha^{u_1} \cdot \beta^{u_2}(\mod p))(\mod q) = 5$.
- Note that $v = r$ and thus the signature is accepted as valid.

In DSA, it is recommended that p has at least 512 bits, and some experts say it is more prudent to have p with 1024 bits. Thus the order of magnitude of DSA parameters is about the same as for RSA. However, DSA is slower than RSA signature algorithm, because the DSA operations are more complicated. The verification algorithm in RSA requires only one exponentiation modulo a number that is, say 1024 bits long. In DSA, verification requires two exponentiations.

The main problem is that the DSA only needs working modulo a number with 160 bits, but because this size is too small for making the DLP hard, an extra "layer" has been introduced that works mod p, where p should have 512, or, better, 1024 bits.

DSA works in the group generated by α (this group has q elements, because, $\alpha^q = 1$, and thus, the first q powers form a pattern that repeats itself). We have defined the DLP in the multiplicative group of nonzero residues mod p (this group is denoted Z_p^*).

However, DLP can be formulated in any group, not only in Z_p^*. We can use in cryptography, any such group in which DLP is hard.

Such a group exists in the "world" of elliptic curves (EC). Thus there exists an EC-DSA which is very similar to the DSA that we have seen, only that the arithmetic is done in the "world" of EC. The advantage of EC is that it needs smaller parameters, and operations can be implemented faster.

The use of Hash Functions in Signature Schemes

It is very inefficient to apply a signature algorithm to a long message. Consequently, in general, we do not sign the message m but its digest $h(m)$ (where h is some hash function).

For example this is how we typically sign a message m with RSA.

- compute the digest $h(m)$.
- compute $s = h(m)^d (\mod n)$.
- the message + signature is (m, s).

The verification of (m, s) is done as follows:

- compute the digest $h(m)$.
- compute $s^e (\mod n)$.
- check that the above two numbers are equal.

Recall that we want a hash function to have the following properties:

- **preimage resistant:** given a digest, it should be hard to find a message having that digest.
- **weak collision resistant:** given a message m_1 it should be unfeasible to find another message m_2 that collides with m_1.
- **strong collision resistant:** it should be hard to find two messages that collide.

Let's see why these properties are necessary:

(1) suppose h is not preimage resistant.

Then Eve can do the following attack (we assume that RSA signature is used).

- Eve computes $h' = r^e (\mod n)$, for some random integer r.
- Eve also computes the pre-image of h' under h, i.e., Eve computes m such that $h(m) = h'$. Eve can do this under assumption (1).

- Eve now has the signed message (m, r).

In other words, Eve has first produced the digest, and then a message having that digest. Such a forgery is called an *existential forgery* in that the attacker may not have any control over the content of the message for which she has obtained a valid signature. A protocol allowing existential forgery is still considered to have a weakness.

(2) suppose h is not weak collision resistant.

Eve can do the following:

- Eve obtains your signature (m, s) on message m.
- Eve finds another message m' with $h(m) = h(m')$.
- Eve now has a valid (m', s) on the message m'.

(3) suppose h is not strong collision resistant.

The legitimate signer can do the following:

- the signer chooses two messages m and m' with $h(m) = h(m')$.
- the signer signs m (using the digest $h(m)$) and sends (m, s).
- later the signer repudiates this signature claiming it was a signature on m'.

The Birthday Paradox and attacks based on it

30 people are in a room. What is the probability that some pair of them have the same birthday?

Answer: 0.71

For what (smallest) value of n people in a room is the above probability at least 1/2?

Answer: 23

Why: let's estimate the probability that no pair of people have the same birthday.

$$1 \cdot \frac{364}{365} \cdot \frac{363}{365} \cdots \frac{365-22}{365} = 0.493.$$

So, Prob (there are 2 people with same birthday) = 1 - 0.493 = 0.507.

Let's look at the general case. Suppose that there are n types of objects labelled $\{1, 2, \ldots, n\}$ and we randomly pick k objects (there is an unlimited supply of each type of object).

Let $P(n, k)$ be the probability that there is at least one duplicate. We will use the inequality

$$(1 - x) \leq e^{-x}, \text{ for all } x \geq 0,$$

which can easily be proved using some Calculus I stuff.

Then

$$\begin{aligned} P(n,k) &= 1 - \frac{n-1}{n}\frac{n-2}{n}\ldots\frac{n-k+1}{n} \\ &= 1 - (1-\frac{1}{n})(1-\frac{2}{n})\ldots(1-\frac{k-1}{n}) \\ &> 1 - e^{-1/n}\cdot e^{-2/n}\cdot\ldots\cdot e^{-(k-1)/n} \\ &= 1 - e^{-[1/n+2/n+\ldots+(k-1)/n]} \\ &= 1 - e^{-(k(k-1)/(2n)} \\ &\approx 1 - e^{-k^2/(2n)}. \end{aligned}$$

Let's estimate for what k is $P(n,k) = 1/2$.

We need to have

$e^{-k^2/(2n)} = 1/2$

$\Rightarrow k^2/(2n) = \ln 2$

$\Rightarrow k = \sqrt{2(\ln 2)n} = 1.18\sqrt{n} \approx \sqrt{n}$.

The rule of a thumb is that for $k \approx \sqrt{n}$, chances of a collision are approx. $1/2$.

If a hash functions produces digest of length m bits, then there are $n = 2^m$ possible digests.

If we randomly pick $\sqrt{n} = 2^{m/2}$ random messages, there is a $1/2$ chance that two of them collide.

So if we want that the effort of the attacker is at least 2^{80}, we need to make the length of the digest $m = 160$.

Also note that if $k = \sqrt{2\lambda n}$ for some λ, then the probability that there is a duplicate is $\approx 1 - e^{-\lambda}$.

For the birthday attack (to be presented next), the relevant situation is a follows:

- there are n types of objects, and an unlimited supply of objects of each type.
- there is a group 1 of k random choices of objects
- there is a group 2 of k random choices of objects
- let $P(n, k)$ be the probability that a choice in group 1 = a choice in group 2.
- if $k = \sqrt{\lambda n}$, then $P(n, k) \approx 1 - e^{-\lambda}$.

Birthday paradox attack on signature schemes

Recall the basic scheme

$$m \to h(m) \to sign(h(m)).$$

Suppose that digest $h(m)$ is 50 bits long.

Suppose m is a sale contract and Bob (the real-estate agent) asks Alice (the buyer) to sign it.

m is a good contract and Alice wants to sign it.

She thinks that Bob may change m to a bad contract m' so that $h(m) = h(m')$, in which case $sign(h(m)) = sign(h(m'))$.

If h is a good hash function (looking like a random function) the chances that this happens are $1/2^{50}$, which is negligible, so Alice is not too worried.

However Bob does the following:

He finds 30 places in the contract where he can make a small change.

In this way he prepares:

- group 1: 2^{30} contracts (essentially the same) that are good for Alice.
- group 2: 2^{30} contracts (essentially the same) that are bad for Alice.

So $k = 2^{30}$ and $n = 2^{50}$ and $k = \sqrt{\lambda n}$ for $\lambda = 2^{10}$.

So with prob $1 - e^{-1024} \approx 1$, there is m in group1 and m' in group 2 that collide, i.e, $h(m) = h(m')$.

Bob asks Alice to sign m, which she does, but then changes m with m'.

Dear Anthony,

{This letter is / I am writing} to introduce {you to / to you} {Mr. / --} Alfred {P. / --}

Barton, the {new / newly appointed} {chief / senior} jewellery buyer for {our / the}

Northern {European / Europe} {area / division} . He {will take / has taken} over {the / --}

responsibility for {all / the whole of} our interests in {watches and jewellery / jewellery and watches}

in the {area / region} . Please {afford / give} him {every / all the} help he {may need / needs}

to {seek out / find} the most {modern / up to date} lines for the {top / high} end of the

market. He is {empowered / authorized} to receive on our behalf {samples / specimens} of the

{latest / newest} {watch and jewellery / jewellery and watch} products, {up / subject} to a {limit / maximum}

of ten thousand dollars. He will {carry / hold} a signed copy of this {letter / document}

as proof of identity. An order with his signature, which is {appended / attached}

{authorizes / allows} you to charge the cost to this company at the {above / head office}

address. We {fully / --} expect that our {level / volume} of orders will increase in

the {following / next} year and {trust / hope} that the new appointment will {be / prove}

{advantageous / an advantage} to both our companies.

Figure 11.8 A Letter in $2^{3\ 7}$ Variations [DAVI89]