Digital Signatures Schemes

Digital signatures are the electronic equivalent of the handwritten signatures. A *digital signature* is attached to the message in order to prove its origin authentication. As opposed to the ordinary signature, the digital signature depends on the corresponding message. The digital signature also depends on a secret parameter (the *signing key*) known only by the signer. The digital signatures must be verifiable - thus, some information (the *verification key*) must be broadcasted in order to make the verification process possible. The formal definition of a digital signature scheme is presented next.

Definition 1 A *digital signature scheme* is a system $\mathcal{S}ig = (\mathcal{P}, \mathcal{S}, \mathcal{K}, \mathcal{M}_s, \mathcal{M}_v)$ of non-empty finite sets, where

- $\mathcal{P}$ denotes the set of *signing atomic symbols*;
- $\mathcal{S}$ denotes the set of *signature atomic symbols*;
- $\mathcal{K}$ denotes the set of *keys*;
- $\mathcal{M}_s$ denotes the set of *signing methods*,
  $\mathcal{M}_s = \{sig_k | sig_k : \mathcal{P} \to \mathcal{S}, k \in \mathcal{K}\}$;
- $\mathcal{M}_v$ denotes the set of *verification methods*,
  $D = \{ver_k | ver_k : \mathcal{P} \times \mathcal{S} \to \{0,1\}, k \in \mathcal{K}\}$ such that

$$(\forall k \in \mathcal{K})(\forall x \in \mathcal{P})(\forall y \in \mathcal{S})(ver_k(x,y) = 1 \Leftrightarrow sig_k(x) = y).$$

As we presented above, any key k has two components: k_s, for signing, also referred to as *private (sub)key* and k_v, for verification, also referred to as *public (sub)key*.

RSA digital signature scheme

- `verification key`: $k_v = (N, e)$, where $N = p \cdot q$, p and q are distinct primes, and $e \in \mathbf{Z}^*_{\phi(N)}$;
- `signing key`: $k_s = (p, q, d)$, where d is a positive integer such that $e \cdot d \equiv 1 \; mod \; \phi(N)$;
- `signature generation`: the digital signature corresponding to a signing atomic symbol $x \in \mathbf{Z}_N$ is $y = x^d \; \texttt{mod} \; N$;
- `signature verification`: having a pair $(x, y) \in \mathbf{Z}_N \times \mathbf{Z}_N$, y is the correct signature with respect to x if and only if

$$x \stackrel{?}{=} y^e \; \texttt{mod} \; N.$$

Digital Signature Standard (DSS)

- `signing key`: $k_s = a$, $a \in \mathbf{Z}^*_q$;
- `verification key`: $k_v = (p, q, \alpha, \beta)$ where p is a large prime, such that $p - 1$ has a large prime divisor q, α is an element of order q, and $\beta = \alpha^a \; \texttt{mod} \; p$;
- `signature generation`: the digital signature corresponding to a signing atomic symbol $x \in \mathbf{Z}^*_q$ is the pair (γ, δ); where
  - $\gamma = (\alpha^{r^{-1}} \; \texttt{mod} \; p) \; \texttt{mod} \; q$,
  - $\delta = r \cdot (x + a \cdot \gamma) \; \texttt{mod} \; q$,

  where $r \in \mathbf{Z}^*_q$ is a parameter chosen by the signer;
- `signature verification`: having a pair $(x, (\gamma, \delta))$, the correctness of (γ, δ) with respect to x can be verified by testing

$$\gamma \stackrel{?}{=} (\alpha^{x \cdot \delta^{-1}} \cdot \beta^{\gamma \cdot \delta^{-1}} \texttt{mod} \; p) \; \texttt{mod} \; q.$$

  (all the operations from exponents are performed modulo q)

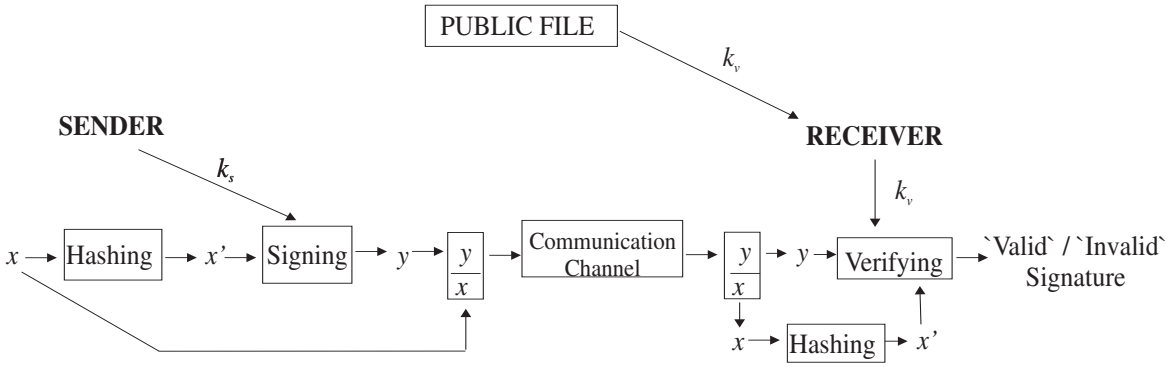

Figure 1: Digital signatures and hash functions

Indeed, if $\gamma = (\alpha^{r^{-1}} \bmod p) \bmod q$ and $\delta = r \cdot (x + a \cdot \delta) \bmod q$ then

$$\begin{aligned}(\alpha^{x\cdot\delta^{-1}} \cdot \beta^{\gamma\cdot\delta^{-1}} \bmod p) \bmod q &= (\alpha^{x\cdot\delta^{-1}} \cdot \alpha^{a\cdot\gamma\cdot\delta^{-1}} \bmod p) \bmod q \\ &= (\alpha^{\delta^{-1}(x+a\cdot\gamma)} \bmod p) \bmod q \\ &= (\alpha^{r^{-1}} \bmod p) \bmod q \\ &= \gamma.\end{aligned}$$

By choosing p and q such that $|p| = 1024$ and $|q| = 160$, the signatures will be only 320 bits long.

Hash Functions

The digital signatures presented in the previous paragraph may be as long as the corresponding messages. Moreover, in case of a long message, the signing and the verification process may be very time-consuming. The idea is to sign a smaller amount of information, without compromising the security. A *hash function* h maps arbitrary binary strings to strings of some fixed length m. Thus, before signing a long message, a hash function is applied. The result, referred to as the *message digest* is then signed. The combination of a digital signature scheme and hash functions is presented in Figure 1.

Depending on the intended application, a hash function $h : \cup_{i\geq 1}\mathbf{Z}_2^i \rightarrow \mathbf{Z}_2^m$ must satisfy certain security requirements:

1. Given $y \in \mathbf{Z}_2^m$, the problem of finding $x \in \cup_{i \geq 1} \mathbf{Z}_2^i$ such that $h(x) = y$ is intractable - functions that satisfy such requirements are called *one-way*[1] (or *preimage resistant*);

2. Given $x \in \cup_{i \geq 1} \mathbf{Z}_2^i$, the problem of finding $x' \in \cup_{i \geq 1} \mathbf{Z}_2^i$, $x' \neq x$, such that $h(x') = h(x)$ is intractable - functions that satisfy such requirements are called *weak collision resistant*;

3. The problem of finding a *collision* of h, i.e. a pair (x, x'), $x, x' \in \cup_{i \geq 1} \mathbf{Z}_2^i$, $x' \neq x$, such that $h(x') = h(x)$, is intractable - functions that satisfy such requirements are called *strong collision resistant*.

An implicit requirement is that $h(x)$ is easy to compute, for any x. In order to avoid the birthday attack, the size of a message digests has to be greater than 160 (bits).

There is a class of *keyed hash functions* (also referred to as *MACs* (message authentication codes)). MACs can be used for message origin authentication (as a symmetric technique) and data integrity.

[1] In general, the appellative *one-way* does not assume the compression property - an one-way function is a function $h : X \to Y$, such that given $y \in Y$, the problem of finding $x \in X$ such that $h(x) = y$ is intractable.