Chapter 5
Digital Signatures

Digital Signatures

A digital signature or digital signature scheme is a type of asymmetric cryptography used to simulate the security properties of a handwritten signature on paper. Digital signature schemes normally give two algorithms, one for signing which involves the user's secret or private key, and one for verifying signatures which involves the user's public key. The output of the signature process is called the "digital signature."

A signature provides authentication of a "message". Messages may be anything, from electronic mail to a contract, or even a message sent in a more complicated cryptographic protocol. Digital signatures are used to create public key infrastructure (PKI) schemes in which a user's public key (whether for public-key encryption, digital signatures, or any other purpose) is tied to a user by a digital identity certificate issued by a certificate authority. PKI schemes attempt to unbreakably bind user information (name, address, phone number, etc.) to a public key, so that public keys can be used as a form of identification.

A digital signature scheme typically consists of three algorithms:

A key generation algorithm G, that randomly produces a "key pair" (PK, SK) for the signer. PK is the verifying key, which is to be public, and SK is the signing key, to be kept private.

A signing algorithm S, that, on input of a message m and a signing key SK, produces a signature σ.

A signature verifying algorithm V, which on input a message m, a verifying key PK, and a signature σ, either accepts or rejects.

Required Properties of Digital Signature Schemes

Two main properties are required. First, signatures computed honestly should always verify. That is, V should accept (m, PK, S (m, SK)) where SK is the secret key related to PK, for any message m. Secondly, it should be hard for any adversary, knowing only PK, to create valid signature(s)

Generally hashes are used in digital signature scheme due to the following reasons:

For efficiency: The signature will be much shorter and thus save time since hashing is generally much faster than signing in practice.

For compatibility: Messages are typically bit strings, but some signature schemes operate on other domains (such as, in the RSA, numbers modulo a number N). A hash function can be used to convert an arbitrary input into the proper format.

For integrity: Without the hash function, the text to be signed may split in many blocks for the signature scheme to act on them. However, the receiver of the signed blocks is not able to recognize if all the blocks are not present and not in the appropriate order.

Benefits of digital signatures

- Authentication: Digital signatures can be used to authenticate the source of messages. When ownership of a digital signature secret key is bound to a specific user, a valid signature shows that the message was sent by that user. For example, suppose a bank's branch office sends instructions to the central office requesting a change in the balance of an account. If the central office is not convinced that such a message is truly sent from an authorized source, acting on such a request could be a grave mistake.

- Integrity: In many cases, the sender and receiver of a message may have a need for trust that the message has not been altered during transmission. Although encryption hides the contents of a message, it may be possible to change an encrypted message
without understanding it. However, if a message is digitally signed, any change in the message will invalidate the signature. Furthermore, there is no efficient way to modify a message and its signature to produce a new message with a valid signature, because this is still considered to be computationally infeasible by most cryptographic hash functions.

Drawbacks of digital signatures

-Trusted Time Stamping: Digital signature algorithms and protocols do not inherently provide certainty about the date and time at which the underlying document was signed. The signer might, or might not, have included a time stamp with the signature, or the document itself might have a date mentioned on it, but a later reader cannot be certain the signer did not, for instance, backdate the date or time of the signature.

- Non-repudiation: The word repudiation refers to any act of disclaiming responsibility for a message. A message's recipient may insist the sender attach a signature in order to make later repudiation more difficult, since the recipient can show the signed message to a third party (eg, a court) to reinforce a claim as to its signatories and integrity. However, loss of control over a user's private key will mean that all digital signatures using that key, and so ostensibly 'from' that user, are suspect. Nonetheless, a user cannot repudiate a signed message without repudiating their signature key. It is aggravated by the fact there is no trusted time stamp, so new documents (after the key compromise) cannot be separated from old ones, further complicating signature key invalidation. Certificate Authorities usually maintain a public repository of public-key so the association user-key is certified and signatures cannot be repudiated. Expired certificates are normally removed from the directory. It is a matter for the security policy and the responsibility of the authority to keep old certificates for a period of time if a non-repudiation of data service is provided.

Digital Signatures Schemes

- RSA

Basic RSA signatures are computed as follows:

- To generate RSA signature keys, one simply generates an RSA key pair. See RSA encryption fro Key generation mechanism.
- To sign a message m, the signer computes $\sigma = m^d \bmod n$. To verify, the receiver checks that $\sigma^e = m \bmod n$. where (e, n) is the public key and (d, n) is the private key.

This scheme is not very secure. To prevent attacks, one can first get a message digest of the message m and then apply the RSA algorithm described above to the result.

Signing Messages: Suppose Alice wishes to send a signed message (m) to Bob. She can use her own private key (d, n) to do so. She produces a hash value of the message (h(m)), find $(h(m))^d$ mod n (as she does when decrypting a message), and attaches it as a "signature" to the message. When Bob receives the signed message, he uses the same hash algorithm in conjunction with Alice's public key (e, n). He raises the $((h(m))^d \bmod n)^e \bmod n$ (as he does when encrypting a message), and compares the resulting hash value with the message's actual hash value. If the two agree, he knows that the author of the message was in possession of Alice's secret key, and that the message has not been tampered with since.

- ElGamal Signature Scheme

The ElGamal signature scheme is based on the difficulty of computing discrete logarithms. The ElGamal signature algorithm is rarely used in practice. A variant developed at NSA and known as the Digital Signature Algorithm is much more widely used. The ElGamal signature scheme allows that a verifier can confirm the authenticity of a message m sent by the signer sent to him over an insecure channel.

Description: This scheme requires the following parameters and procedures:
A long term public/private keys where public key is (g, p, T) and private key S such that g^S mod p = T.
New Different public/private key pair for each message being signed. If message is m, choose random number S_m and find g^{Sm} mod p = T_m.

Signing Process:

Take the well known message digest (hash) function, say H. Using this hash function calculate message digest d_m of $m|T_m$ i.e. find $d_m = H(m|T_m)$.

Calculate signature $X = S_m + d_mS \bmod (p-1)$.

Signer sends message m along with X and T_m. since the receiver knows m and T_m,

Verifying Process:

Calculate d_m using obtained m and T_m.

Check whether $g^X = T_mT^{dm} \bmod p$. If this is true the signature is valid, else not valid. This is true since $g^X = g^{Sm + dmS} = g^{Sm}\, g^{dmS} = T_mT^{dm} \bmod p$.

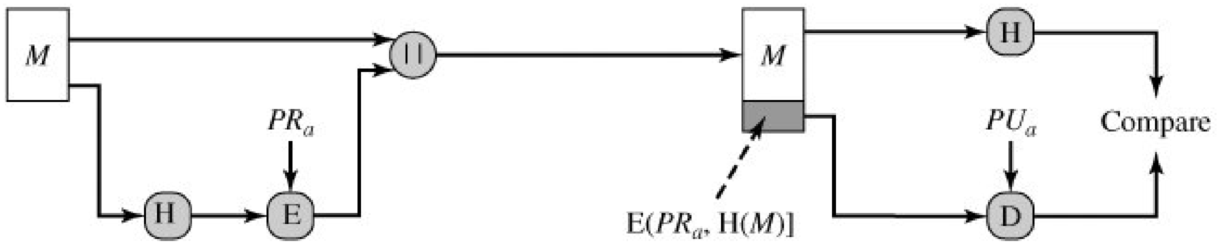
Security: A third party can forge signatures either by finding the signer's secret key x or by finding collisions in the hash function. Both problems are believed to be difficult. The signer must be careful to choose a different k uniformly at random for each signature and to be certain that k, or even partial information about k, is not leaked. Otherwise, an attacker may be able to deduce the secret key x with reduced difficulty, perhaps enough to allow a practical attack. In particular, if two messages are sent using the same value of k and the same key, then an attacker can compute x directly.

Digital Signature Standard

The National Institute of Standards and Technology (NIST) has published Federal Information Processing Standard FIPS 186, known as the Digital Signature Standard (DSS). The DSS makes use of the Secure Hash Algorithm (SHA) described in later chapter and presents a new digital signature technique, the Digital Signature Algorithm (DSA). The DSS was originally proposed in 1991 and revised in 1993 in response to public feedback concerning the security of the scheme. There was a further minor revision in 1996. In 2000, an expanded version of the standard was issued as FIPS 186-2. This latest version also incorporates digital signature algorithms based on RSA and on elliptic curve cryptography. In this section, we discuss the original DSS algorithm.

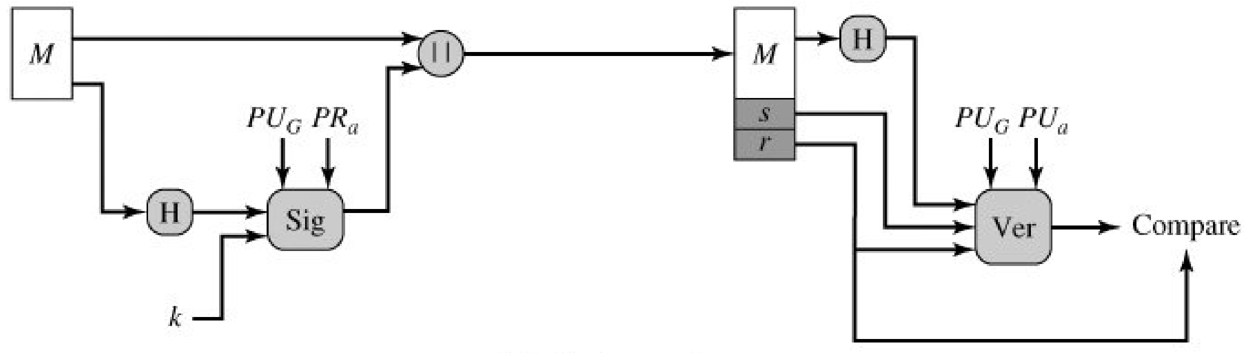
Two Approaches to Digital Signatures (Revisited)

In the RSA approach, the message to be signed is input to a hash function that produces a secure hash code of fixed length. This hash code is then encrypted using the sender's private key to form the signature. Both the message and the signature are then transmitted. The recipient takes the message and produces a hash code. The recipient also decrypts the signature using the sender's public key. If the calculated hash code matches the decrypted signature, the signature is accepted as valid. Because only the sender knows the private key, only the sender could have produced a valid signature.



(a) RSA approach

The DSS approach also makes use of a hash function. The hash code is provided as input to a signature function along with a random number k generated for this particular signature. The signature function also depends on the sender's private key (PR_a)and a set of parameters known to a group of communicating principals. We can consider this set to constitute a global public key (PU_G).The result is a signature consisting of two components, labeled s and r. At the receiving end, the hash code of the incoming message is generated. This plus the signature is input to a verification function. The verification function also depends on the global public key as well as the sender's public key (PU_a), which is paired with the sender's private key. The output of the verification function is a value that is equal to the signature component r if the signature is valid. The signature function is such that only the sender, with knowledge of the private key, could have produced the valid signature.



(b) DSS approach

The Digital Signature Algorithm

Global Public-Key Components

p	prime number where $2^{L\ 1} < p < 2^L$ for 512 L 1024 and L a multiple of 64; i.e., bit length of between 512 and 1024 bits in increments of 64 bits
q	prime divisor of (p 1), where $2^{159} < q < 2^{160}$; i.e., bit length of 160 bits
g	$= h^{(p\ 1)/q} \bmod p$, where h is any integer with $1 < h < (p\ 1)$ such that $h^{(p\ 1)/q} \bmod p > 1$

User's Private Key

x	random or pseudorandom integer with $0 < x < q$

User's Public Key

y	$= g^x \bmod p$

User's Per-Message Secret Number

k	= random or pseudorandom integer with $0 < k < q$

Signing

r	$= (g^k \bmod p) \bmod q$
s	$= [k^{-1} (H(M) + xr)] \bmod q$

Signature = (r, s)

Verifying

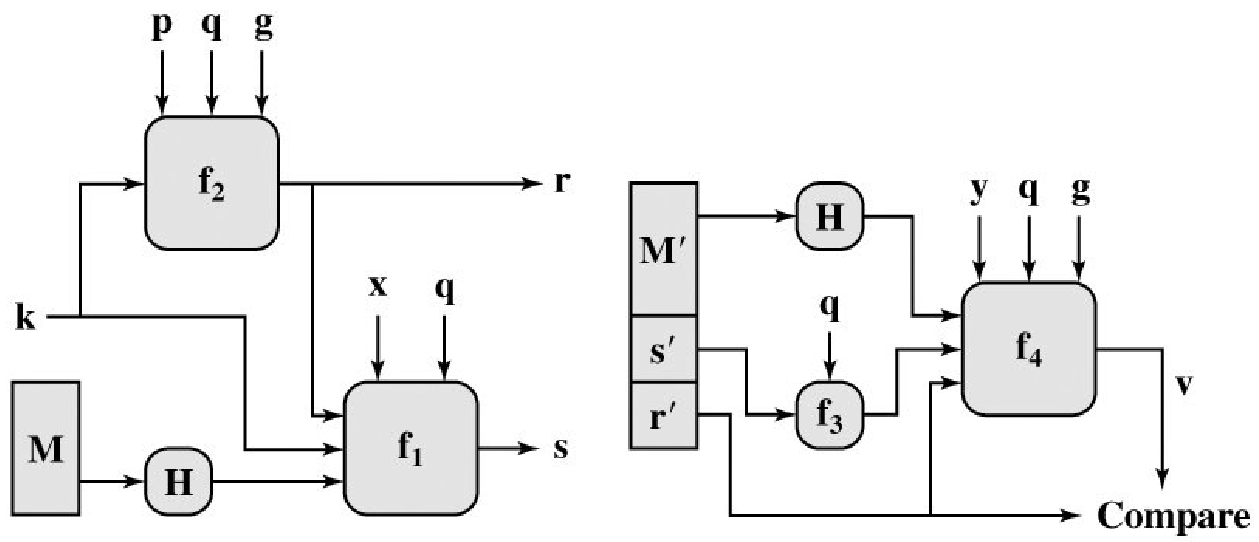
w	$= (s')^{-1} \bmod q$
u1	$= [H(M')w] \bmod q$

u2	=(r')w mod q
v	= [($g^{u\,1}$ $y^{u\,2}$) mod p] mod q

TEST: v = r'

M	= message to be signed
H(M)	= hash of M using SHA-1
M', r', s'	= received versions of M, r, s

The following figure depicts the functions of signing and verifying.



$s = f_1(H(M), k, x, r, q) = (k^{-1}(H(M) + xr)) \bmod q$

$r = f_2(k, p, q, g) = (g^k \bmod p) \bmod q$

(a) Signing

$w = f_3(s', q) = (s')^{-1} \bmod q$

$v = f_4(y, q, g, H(M'), w, r')$

$= ((g^{(H(M')w) \bmod q}\, y^{r'w \bmod q}) \bmod p) \bmod q$

(b) Verifying