Digital Signatures

Tecnologie Internet
a.a. 2022/2023

Summary

- Principles of Digital Signatures
- RSA Digital Signature Scheme
- Elgamal Digital Signature Scheme
- Schnorr Digital Signature Scheme
- NIST Digital Signature Standard (DSS)

Principles of Digital Signatures

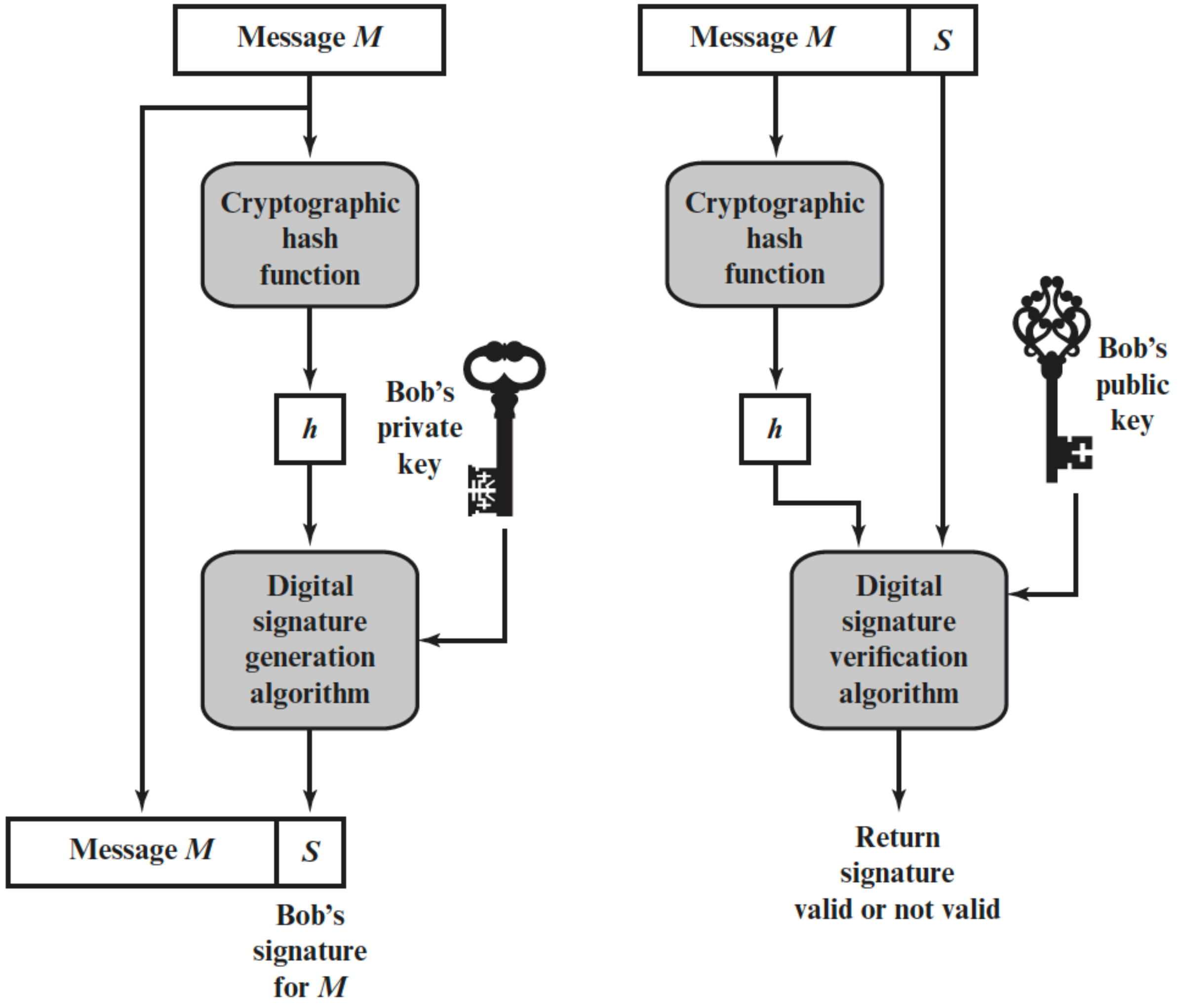
The most important development from the work on public-key cryptography is the digital signature.

The digital signature provides a set of security capabilities that would be difficult to implement in any other way.

Suppose that Bob wants to send a message to Alice. Although it is not important that the message be kept secret, **Bob wants Alice to be certain that the message is indeed from him**.

For this purpose, **Bob uses a secure hash function**, such as SHA-512, to generate a hash value for the message. That hash value, together with Bob's private key serves as input to a **digital signature generation algorithm**, which produces a short block that functions as a digital signature.

Principles of Digital Signatures


Message M
Cryptographic hash function
h
Bob's private key
Digital signature generation algorithm
Message M
S
Bob's signature for M
Message M
S
Cryptographic hash function
h
Bob's public key
Digital signature verification algorithm
Return signature valid or not valid

(a) Bob signs a message (b) Alice verifies the signature

Principles of Digital Signatures

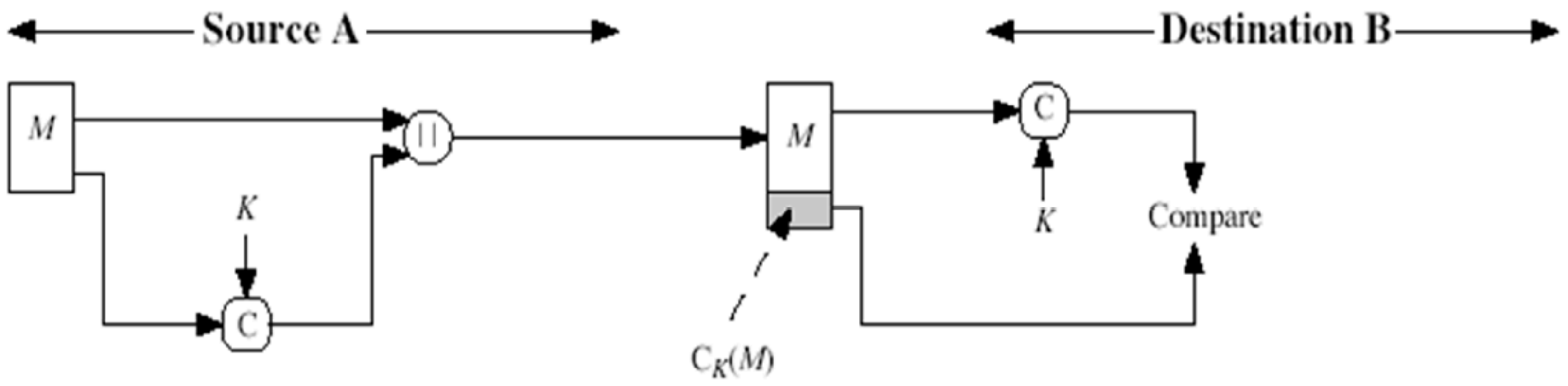
When Alice receives the message plus signature, she (1) calculates a hash value for the message; (2) provides the hash value and Bob’s public key as inputs to a **digital signature verification algorithm**.

If the algorithm returns the result that the signature is valid, Alice is assured that **the message must have been signed by Bob**. No one else has Bob’s private key and therefore no one else could have created a signature that could be verified for this message with Bob’s public key.

In addition, it is impossible to alter the message without access to Bob’s private key, so **the message is authenticated both in terms of data origin and in terms of data integrity**.

Principles of Digital Signatures

Remember the message authentication scheme based on shared secret key (MAC..):



Consider the following **disputes** that could arise.
1. Bob may forge a different message and claim that it came from Alice. Bob would simply have to create a message and append an authentication code using the key that Alice and Bob share.
2. Alice can deny sending the message. Because it is possible for Bob to forge a message, there is no way to prove that Alice did in fact send the message.

Principles of Digital Signatures

The digital signature must have the following properties:

- It must verify the author and the date and time of the signature.
- It must authenticate the contents at the time of the signature.
- It must be verifiable by third parties, to resolve disputes.

Thus, the digital signature function must include the **message authentication** function but also the **non-repudiation** function!

Message authentication alone protects two parties that exchange messages against malicious third parties.

Digital signature protects the two parties also against each other.

Types of Attacks Against Digital Signatures

A = user being attacked
C = attacker

- **Key-only attack**: C only knows A’s public key
- **Known message attack**: C has a set of messages and their signatures
- **Generic chosen message attack**: C obtains from A valid signatures for chosen messages that do not depend on PU_A
- **Directed chosen message attack**: similar to the generic one, but the list of messages to be signed is chosen based on PU_A
- **Adaptive chosen message attack**: C may request from A signatures of messages that depend on previously obtained message-signature pairs

Digital Signature Forgery

A = user being attacked
C = attacker

- **Total break**: C determines A's private key
- **Universal forgery**: C finds an algorithm that provides way of constructing valid signatures on arbitrary messages
- **Selective forgery**: C forges a signature for a particular message chosen by C
- **Existential forgery**: C forges a signature for at least one message; C has no control over the message

(with a non-negligible probability)

Digital Signature Requirements

- The signature must be a bit pattern **that depends on the message being signed**
- The signature **must use some information only known to the sender** to prevent both forgery and denial
- It must be relatively easy to produce the digital signature
- It must be relatively easy to recognize and verify the digital signature
- It must be computationally infeasible to forge a digital signature
- It must be practical to retain a copy of the digital signature in storage

Digital Signature + Confidentiality

Confidentiality can be provided by encrypting the entire message plus signature with a shared secret key (symmetric encryption).

Note that **it is important to perform the signature function first, and then an outer confidentiality function**.

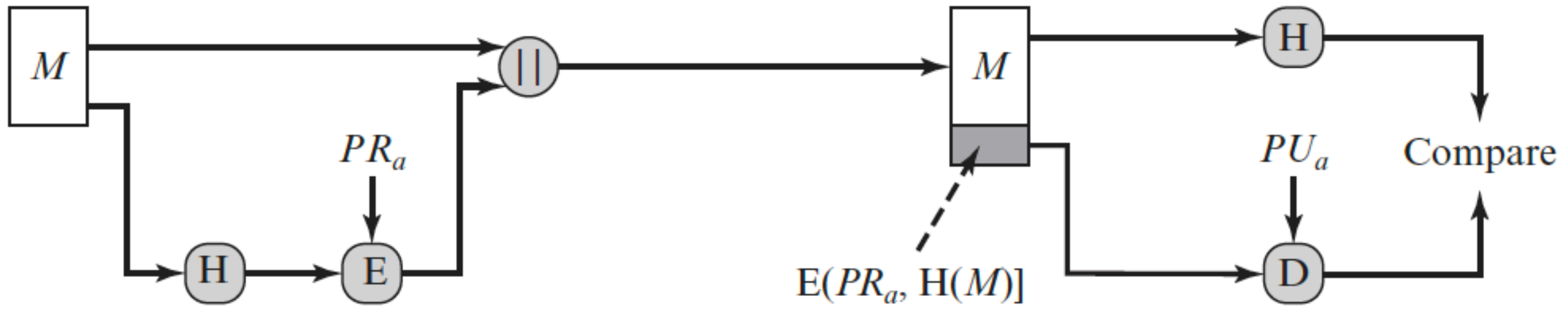
In case of dispute, some third party must view the message and its signature. If the signature is calculated on an encrypted message, then the third party also needs access to the decryption key to read the original message.

If the signature is the inner operation, then the recipient can store the plaintext message and its signature for later use in dispute resolution.

RSA Digital Signature Scheme

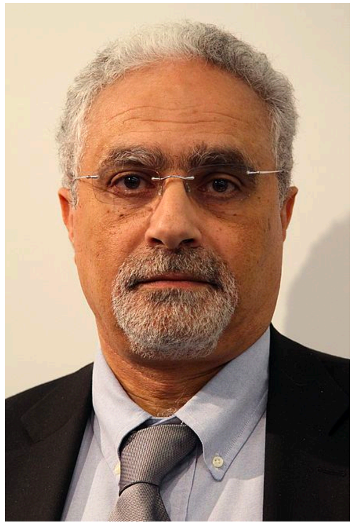
In the RSA approach, the message to be signed is input to a hash function that produces a secure hash code of fixed length. This hash code is then encrypted using the sender's private key to form the signature. Both the message and the signature are then transmitted.

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.



https://www.linkedin.com/in/taherelgamal/

Elgamal Digital Signature Scheme



Recall that if α is a primitive root of the prime number q, then the numbers

$$\alpha \bmod q,\ \alpha^2 \bmod q,\ \ldots,\ \alpha^{q-1} \bmod q$$

are distinct integers from 1 through q-1 in some permutation.

It can be shown that, if α is a primitive root of q, then

1. For any integer m, $\alpha^m \equiv 1 \pmod q$ iff $m \equiv 1 \pmod{q-1}$
2. For any integers i, j, $\alpha^i \equiv \alpha^j \pmod q$ iff $i \equiv j \pmod{q-1}$

Elgamal Digital Signature Scheme

User A generates a **private/public key** pair as follows.

1. Generate a random integer X_A, such that $1 < X_A < q - 1$
2. Compute $Y_A = \alpha^{X_A} \bmod q$
3. A's private key is X_A; A's public key is $\{q, \alpha, Y_A\}$

Computing X_A from Y_A is considered to be infeasible, as it requires to solve a **discrete logarithm problem**.

Elgamal Digital Signature Scheme

To **sign** a message M, user A first computes the hash m = H(M), such that m is an integer in the range $0 \leq m \leq q-1$.

A then forms a digital signature as follows:

1. Choose a random integer K such that $1 \leq K \leq q-1$ and GCD(K,q-1) = 1
2. Compute $S_1 = \alpha^K \bmod q$
3. Compute $K^{-1} \bmod (q-1)$
4. Compute $S_2 = K^{-1}(m - X_A S_1) \bmod (q-1)$
5. The digital signature consists of the pair (S_1,S_2)

Elgamal Digital Signature Scheme

Any user B can **verify** the signature as follows.

1. Compute $V_1 = \alpha^m \bmod q$
2. Compute $V_2 = (Y_A)^{S_1}(S_1)^{S_2} \bmod q$
3. The digital signature is valid if $V_1 = V_2$

Proof:

$\alpha^m \bmod q = (Y_A)^{S_1}(S_1)^{S_2} \bmod q$	assume $V_1 = V_2$
$\alpha^m \bmod q = \alpha^{X_A S_1}\alpha^{K S_2} \bmod q$	substituting for Y_A and S_1
$\alpha^{m - X_A S_1} \bmod q = \alpha^{K S_2} \bmod q$	rearranging terms
$m - X_A S_1 \equiv K S_2 \bmod (q - 1)$	property of primitive roots
$m - X_A S_1 \equiv K K^{-1}(m - X_A S_1) \bmod (q - 1)$	substituting for S_2

Elgamal Digital Signature Scheme

Example:

1. Alice chooses $X_A = 16$.
2. Then $Y_A = \alpha^{X_A} \bmod q = \alpha^{16} \bmod 19 = 4$.
3. Alice's private key is 16; Alice's public key is $\{q, \alpha, Y_A\} = \{19, 10, 4\}$.

Suppose Alice wants to sign a message with hash value $m = 14$.

1. Alice chooses $K = 5$, which is relatively prime to $q - 1 = 18$.
2. $S_1 = \alpha^K \bmod q = 10^5 \bmod 19 = 3$
3. $K^{-1} \bmod (q - 1) = 5^{-1} \bmod 18 = 11$.
4. $S_2 = K^{-1}(m - X_A S_1) \bmod (q - 1) = 11\,(14 - (16)(3)) \bmod 18 = -374 \bmod 18 = 4$.

Bob can verify the signature as follows.

1. $V_1 = \alpha^m \bmod q = 10^{14} \bmod 19 = 16$.
2. $V_2 = (Y_A)^{S_1}(S_1)^{S_2} \bmod q = (4^3)(3^4) \bmod 19 = 5184 \bmod 19 = 16$.

The signature is valid!

Schnorr Digital Signature Scheme

The Schnorr scheme minimizes the message-dependent amount of computation required to generate a signature.

The main work for signature generation does not depend on the message and can be done during the idle time of the processor.

The scheme is based on using a prime p, with p-1 having a prime factor q of appropriate size.

Typically, p is a 1024-bit number, and q is a 160-bit number.

Schnorr Digital Signature Scheme

User A generates a **private/public key** pair as follows.

1. Choose primes p and q, such that q is a prime factor of p-1
2. Choose an integer a, such that $a^q = 1 \bmod p$.
3. Choose a random integer s with $0 < s < q$; ***s* is the private key**
4. Calculate $v = a^{-s} \bmod p$; ***v* is the public key**

a, p, q are public

Schnorr Digital Signature Scheme

With s and v, user A generates a signature for message M as follows.

1. Choose a random integer r with $0 < r < q$; compute $x = a^r \bmod p$

2. Concatenate M with x and hash the result to compute the value e:

   $$e = \mathrm{H}(M \,||\, x)$$

3. Compute $y = (r + se) \bmod q$

4. The digital signature consists of the pair (e,y)

UNIVERSITÀ DI PARMA

Schnorr Digital Signature Scheme

Any other user can verify the signature as follows.

1. Compute $x' = a^y v^e \bmod p$
2. Verify that $e = H(M \,||\, x')$

Proof:

$$x' \equiv a^y v^e \equiv a^y a^{-se} \equiv a^{y-se} \equiv a^r \equiv x \pmod{p}$$

NIST Digital Signature Standard (DSS)

Proposed by NIST in 1991.

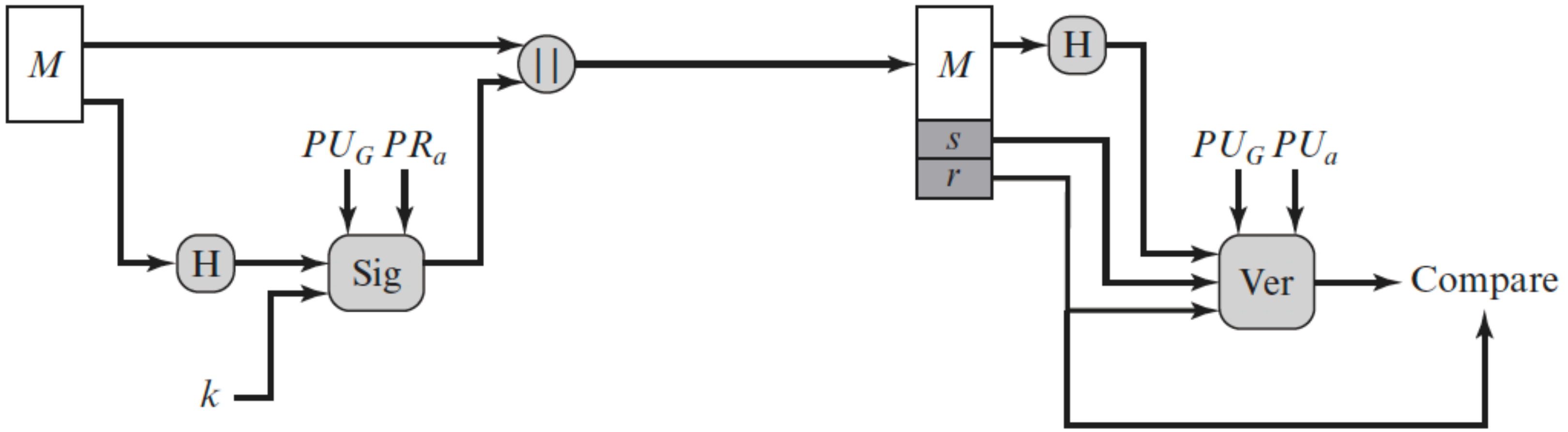
Based on an algorithm known as DSA (Digital Signature Algorithm) that derives from the Elgamal and Schnorr schemes.

Also in this case, security depends on difficulty of computing discrete logarithms.

NIST Digital Signature Standard (DSS)

The hash code is provided as input to a signature function along with a random number *k* generated for this particular signature.

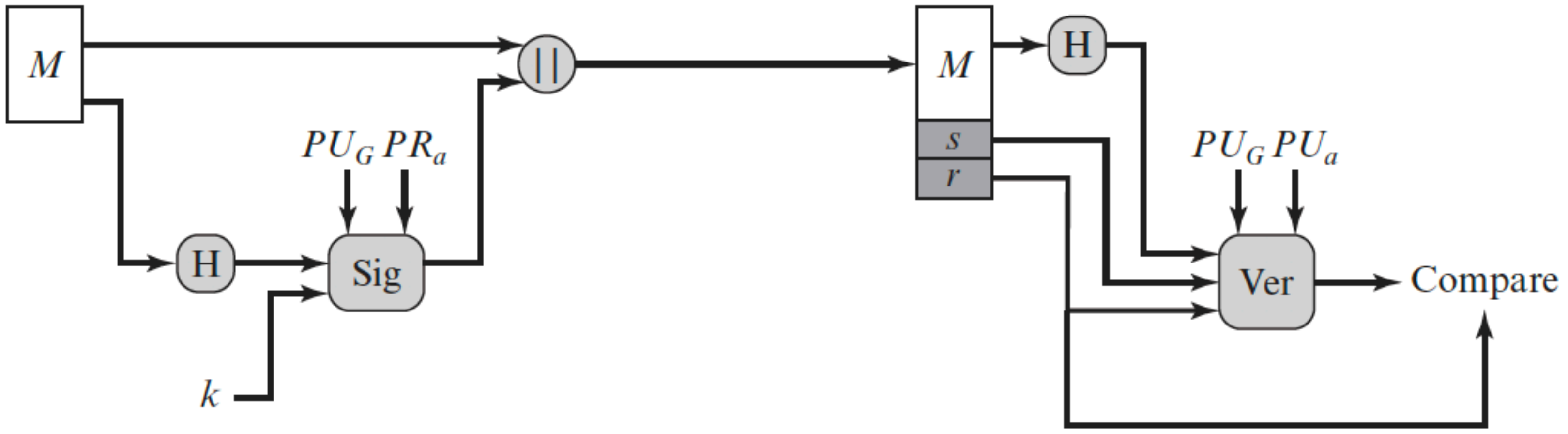
The signature function also depends on the sender's private key and a set of parameters that may be shared between different users of the system. We can consider this set to constitute a **global public key (*PU_G*)**. The result is a signature consisting of two components, labeled *s* and *r*.



NIST Digital Signature Standard (DSS)

At the receiving end, the hash code of the incoming message is generated. The hash code and the signature are inputs to a verification function.

The verification function also depends on the global public key as well as the sender's public key, 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.



Digital Signature Algorithm (DSA)

Global public-key components (PU_G):

p prime number where $2^{L-1} < p < 2^L$, for $512 \leq L \leq 1024$ and L a multiple of 64;

q prime divisor of $(p-1)$, N-bit long, such that $2^{N-1} < q < 2^N$

$g = h^{(p-1)/q} \bmod p > 1$, where h is an integer, $1 < h < (p-1)$

Digital Signature Algorithm (DSA)

Private key:

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

Public key:

$y = g^x \bmod p$

Per-message secret number:

k random or pseudorandom integer, with $0 < k < q$
(should be unique for each signing)

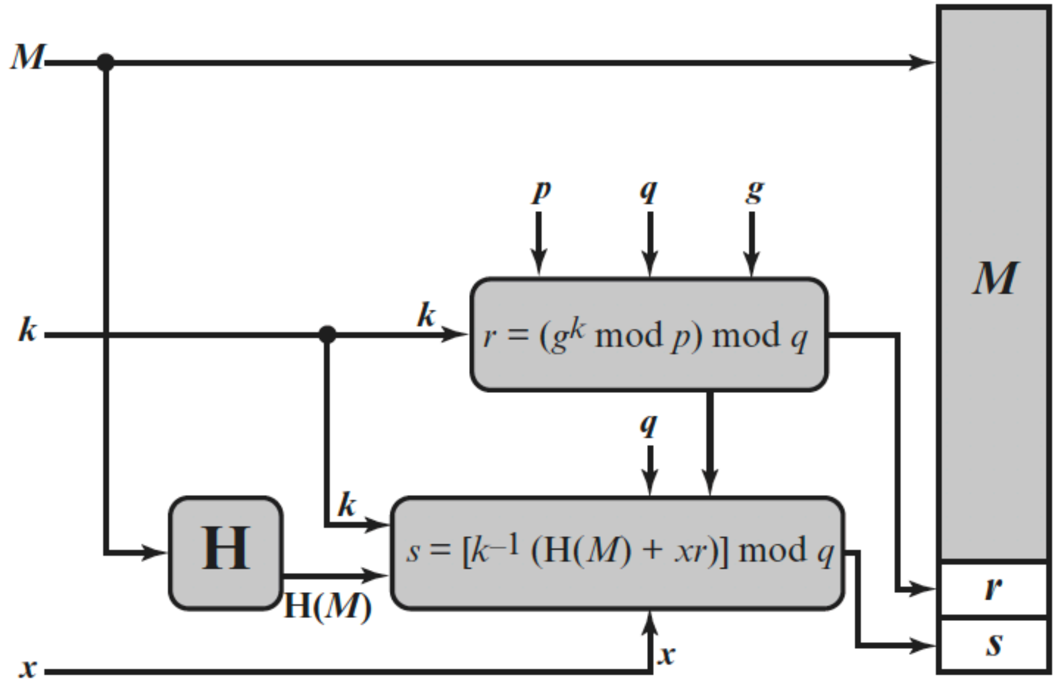
Digital Signature Algorithm (DSA)

Signing:

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

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

Signature = (r,s)



Digital Signature Algorithm (DSA)

Verifying:

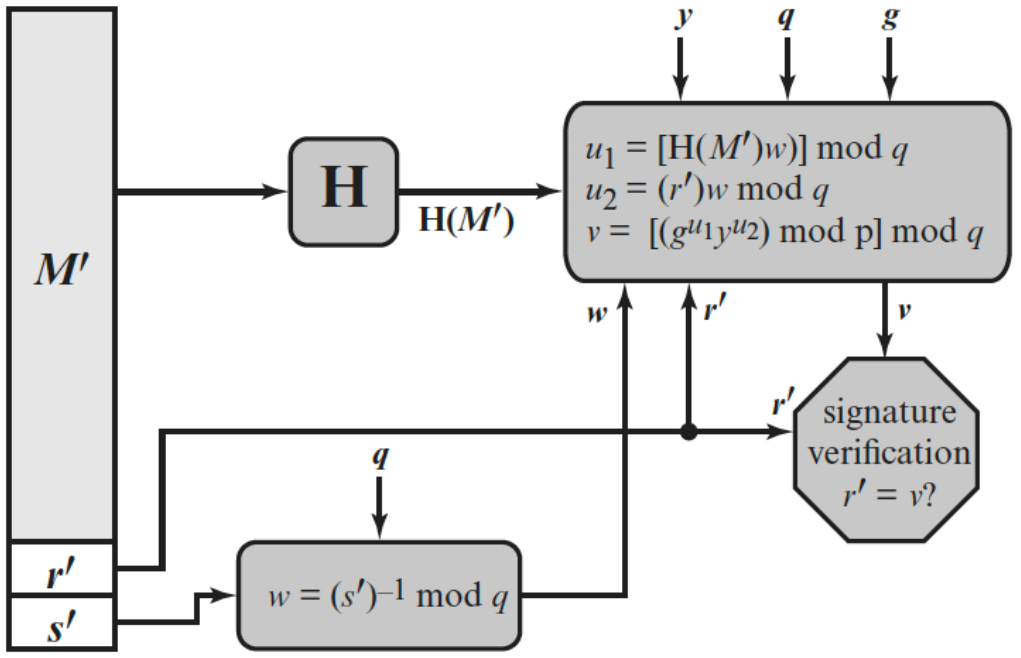
$w = s^{-1} \bmod q$

$u_1 = [H(M')w)] \bmod q$

$u_2 = r'w \bmod q$

$v = [(g^{u1}y^{u2}) \bmod p] \bmod q$

TEST: $v = r'$



where M', r', s' are received versions of M, r, s

References

William Stallings, ***Cryptography and Network Security - Principles and Practice***, 7th edition, Pearson 2017