Unit-IV
Cryptographic Hash Functions and Digital Signatures

Message Authentication

Message authentication is a mechanism or service used to verify the integrity of a message. It assures that data received are exactly as sent (i.e., there is no modification, insertion, deletion, or replay). Message authentication ensures that the message has been sent by a genuine identity and not by an imposter.

Message authentication is concerned with:

- Protecting the integrity of message
- Validating identity of originator
- Non-repudiation of origin

Message Authentication Function

Any message authentication mechanism has two levels of functionality. At the lower level, there must be some sort of function that produces an authenticator: a value to be used to authenticate a message. This lower-level function is then used as a primitive in a higher-level authentication protocol that enables a receiver to verify the authenticity of a message.

There are three types of functions that may be used to produce an authenticator.

- ***Hash function:*** A function that maps a message of any length into a fixed- length hash value, which serves as the authenticator.
- ***Message encryption:*** The ciphertext of the entire message serves as its authenticator.
- ***Message authentication code (MAC):*** A function of the message and a secret key that produces a fixed-length value that serves as the authenticator.

Message Authentication Code (MAC)

Message authentication code is a function of the message and a secret key that produces a fixed-length value that serves as the authenticator.
This technique assumes that the sender and a receiver share a common secret key k. When a sender has a message to send receiver it calculate the MAC as a function of the message and key.

$$MAC = C(M, k)$$

Where,

$C = MAC\ function,$

$M = Input\ messsage,$

$K = shared\ key$

$MAC = Message\ Authentication\ Code$

The message plus MAC are transmitted to the intended receiver. The receiver performs the some calculation on received message using the same secret key to generate a new MAC. Then the received MAC is compared to the calculated. If the received MAC matches the calculated MAC then the receiver is assumed that the message has not been altered.

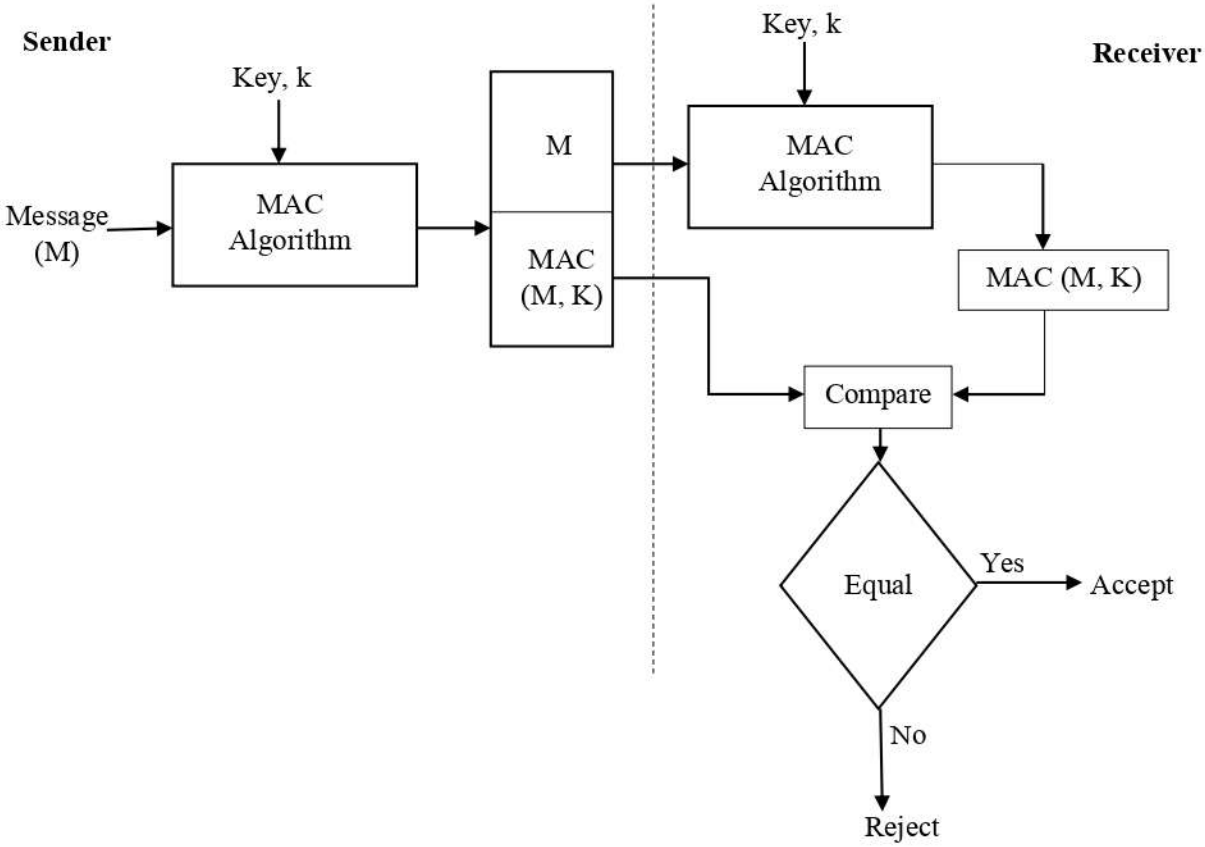
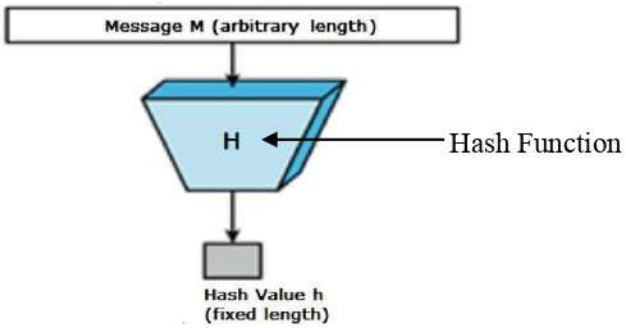


Fig: Message Authentication using a MAC

A MAC function is similar to encryption. One difference is that the MAC algorithm need not be reversible, as it must be for decryption. Because of mathematical properties of the authentication function, it is less vulnerable to being broken than encryption.

Cryptographic Hash Functions

Hash function is a function that maps data of arbitrary size (often called the "message") to a bit string of a fixed size (the "hash value", "hash", or "message digest") which serves as the authenticator and is a one-way function, that is, a function which is practically infeasible to invert.



Properties of Hash Function

1. ***Preimage Resistance (One-way property):***
   This property means that it should be computationally hard to find the input (message) from the output (hash value) i.e. for any given hash value h, it should be difficult to find any message m such that $H(m) = h$.
2. ***Second Preimage Resistance (Weak collision resistance):***
   This property means given an input and its hash, it should be hard to find a different input with the same hash i.e. for any given input m_1, it is computationally hard to find a different input m_2 such that $H(m_1) = H(m_2)$.
3. ***Collision Resistant (Strong collision resistance):***
   This property means it should be hard to find two different inputs of any length that result in the same hash. In other words, it is computationally hard to find two different massage m_1 and m_2 such that $H(m_1) = H(m_2)$.

Applications of Hash functions

- Digital signature
- Message authentication
- Password security
- Encryption
- Message digest

Message Digests

Message Digest Version 4 (MD4)

The MD4 function is a cryptographic algorithm that takes a message of arbitrary length as input and produces a 128 bit message digest or hash value or fingerprint as output.

Suppose a b-bit message as input and we need to find its message digest.It is assumed that the bits of the message are $m_0, m_1, \ldots\ldots, m_{b-1}$.

Step 1: Append padded bits

The message is padded so that its length is congruent to 448, modulo 512. A single '1' bit is append to the message and then '0' bits are appended so that the length in bits equals 448 modulo 512.

Message	100000	

(Message length + padded bits) % 512 = 448

Step 2: Append length

A 64-bit representation of b is appended to the result of the previous step. The resulting message has a length that is an exact multiple of 512 bits.

Message	100000	64 bits

(Message length + padded bits + 64 bits) % 512 = 0

Step 3: Initialize MD buffer

A 4-word buffer (A, B, C, D) is used to compute the message digest. Here each of A, B, C, D is a 32-bit register. These are initialized to the following values in hexadecimal, low-order byte first:

A: 01 23 45 67
B: 89 ab cd ef
C: fe dc ba 98
D: 76 54 32 10

Step 4: Process message in 16-word (512 bit) blocks

It contains ***three passes (rounds)*** with ***16 steps or operations*** each. We first define three auxiliary functions that each take as input three 32-bit words and produce as output one 32-bit word.

Pass 1: $F(X,Y,Z) = (X \wedge Y) \vee (\neg X \wedge Z)$ [step 0 to 15]
Pass 2: $G(X,Y,Z) = (X \wedge Y) \vee (X \wedge Z) \vee (Y \wedge Z)$ [step 16 to 31]
Pass 3: $H(X,Y,Z) = X \oplus Y \oplus Z$ [step 32 to 47]

One operation is illustrated in the figure below.

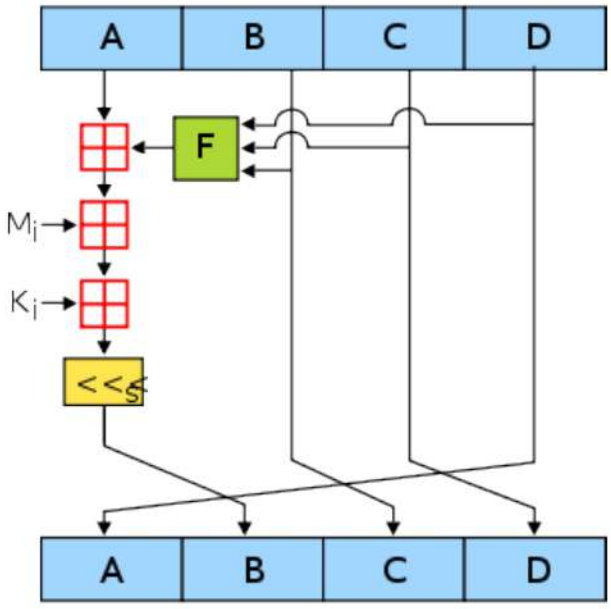


Fig: One MD4 operation

The figure shows how the auxiliary function F is applied to the four buffers (A, B, C and D), using message word M_i and constant K_i. The item "<<<s" denotes a binary left shift by s bits.

Step 5: output

After all rounds have been performed, the buffers A, B, C, D contains the MD4 output starting with lower bit A and ending with higher bit D.

Message Digest Version 5 (MD5)

The MD5-message digest algorithm is a widely used cryptographic hash function producing a 128-bit (16-byte) hash value, typically expressed in text format as a 32 digit hexadecimal number.

MD5 were invented by Ron Rivest as an improved version of MD4.

Operations:

Step 1 – step 3

Same as of MD4

Step 4: Process message in 16-word (512 bit) blocks

It contains ***four passes (rounds)*** with ***16 steps or operations*** each. We first define four auxiliary functions that each take as input three 32-bit words and produce as output one 32-bit word.

Pass 1: $F(X,Y,Z) = (X \wedge Y) \vee (\neg X \wedge Z)$ [step 0 to 15]
Pass 2: $G(X,Y,Z) = (X \wedge Z) \vee (Y \wedge \neg Z)$ [step 16 to 31]
Pass 3: $H(X,Y,Z) = X \oplus Y \oplus Z$ [step 32 to 47]
Pass 4: $H(X,Y,Z) = Y \oplus (X \wedge \neg Z)$ [step 48 to 64]

One operation is illustrated in the figure below.

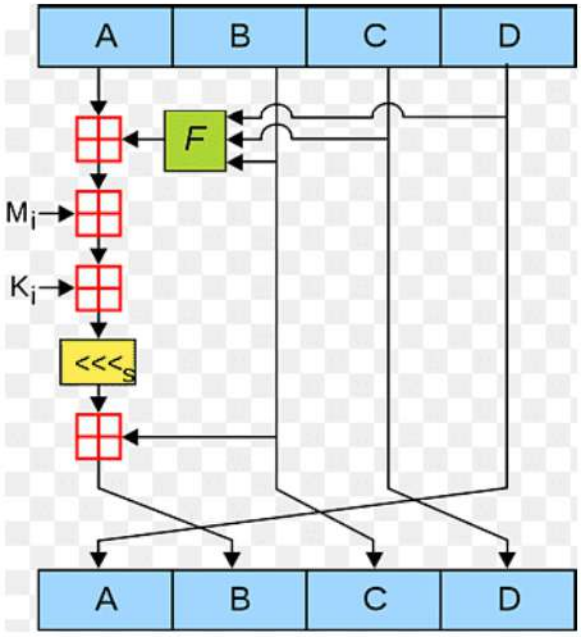


Fig: One MD5 operation

The figure shows how the auxiliary function F is applied to the four buffers (A, B, C and D), using message word M_i and constant K_i. The item "<<<s" denotes a binary left shift by s bits.

Step 5: output

After all rounds have been performed, the buffers A, B, C, D contains the MD5 output starting with lower bit A and ending with higher bit D.

MD4 vs MD5

The following are the differences between MD4 and MD5:

1. A fourth round has been added.
2. Each step now has a unique additive constant, so there are 64 constants.
3. The function G in round 2 was changed from $((X \wedge Y) \vee (X \wedge Z) \vee (Y \wedge Z))$ to $((X \wedge Z) \vee (Y \wedge \neg Z))$ to make G less symmetric.
4. Each step now adds in the result of the previous step. This promotes a faster "avalanche effect".
5. The order in which input words are accessed in rounds 2 and 3 is changed, to make these patterns less like each other.
6. The shift amounts in each pass have been approximately optimized, to yield a faster "avalanche effect." The shifts in different passes are distinct.

Secure Hash Algorithm (SHA)

The secure hash algorithm is a family of cryptographic hash functions published by the National Institute Standards and Technology (NIST) as a U.S. Federal Information Processing Standard (FIPS). It includes the following variations: SHA, SHA-0, SHA-1, SHA-2, SHA-3.

Secure Hash Algorithm-1 (SHA-1)

- SHA-1 is a cryptographic hash function which takes an input and produces the output of 160 bits hash value known as a message digest.
- It works for any input message that is less than 2^{16} bits.

Operations:

1. ***Message Padding:***

   Padding is performed as follows: a single '1' bit is append to the message, and then enough zero bits are append so that the length in bits of the padded message becomes congruent to 448, modulo 512.

2. ***SHA-1 Message digest computation:***

   SHA-1 consists of 80 iterations. Each time it process the 512 bits message and at last it produces the output of 160 bits hash value.

   Let the 160 bits hash value be $ABCDE$.
   Initially,

   $A = 67\ 45\ 23\ 01$
   $B = ef\ cd\ ab\ 89$
   $C = 98\ ba\ dc\ fe$
   $D = 10\ 32\ 54\ 76$
   $E = c3\ d2\ e1\ f0$

   In each iteration, these values are updated as

   $B = old\ A$
   $C = old\ B \lfloor\!\lfloor 30$
   $D = old\ C$
   $E = old\ D$
   $A = E + (A \lfloor\!\lfloor 5) + W_t + K_t + f(t, B, C, D)$

Where,
$W_t = W_{n-3} \oplus W_{n-8} \oplus W_{n-14} \oplus W_{n-16}$ is the t^{th} 32 bits word block and K_t is the constant depending upon the value of t given by following relations

$K_t = 5a827999 \quad if\ 0 \leq t \leq 19$
$K_t = 6ed9eba1 \quad if\ 20 \leq t \leq 39$
$K_t = 8f1bbcdc \quad if\ 40 \leq t \leq 59$
$K_t = ca62c1d6 \quad if\ 60 \leq t \leq 79$

Again, $f(t, B, C, D)$ is a function that varies according to the following relations:

$f(t,B,C,D) = (B \wedge C) \vee (\neg B \wedge D) \quad if\ 0 \leq t \leq 19$
$f(t,B,C,D) = B \oplus C \oplus D \quad if\ 20 \leq t \leq 39$
$f(t,B,C,D) = (B \wedge C) \vee (B \wedge D) \vee (C \wedge D) \quad if\ 40 \leq t \leq 59$
$f(t,B,C,D) = B \oplus C \oplus D \quad if\ 60 \leq t \leq 79$

Secure Hash Algorithm-2 (SHA-2)

SHA-2 is a family of two similar hash functions known as SHA-256 and SHA-512, with different block sizes. Both algorithm belongs to SHA-2. They differ in the word size. SHA-256 uses 32-bit words where SHA-512 uses 64-bit words. There are also truncated versions of each standard, known as SHA-224, SHA-384, SHA-512, SHA-512/224, SHA-512/256.

Parameters for various version of SHA

Algorithm	Message Digest Size	Message Size	Block Size	Word Size	No of Step
SHA-1	160	$<2^{64}$	512	32	80
SHA-224	224	$<2^{64}$	512	32	64
SHA-256	256	$<2^{64}$	512	32	64
SHA-384	384	$<2^{128}$	1024	64	80
SHA-512	512	$<2^{128}$	1024	64	80

Note: *All sizes are measured in bits.*

Digital Signature

Digital signature is an electronic signature that can be used to authenticate the identity of the sender of a message and to ensure that the original content of the message or document that has been sent is unchanged.

Digital signature schemes normally gives 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.

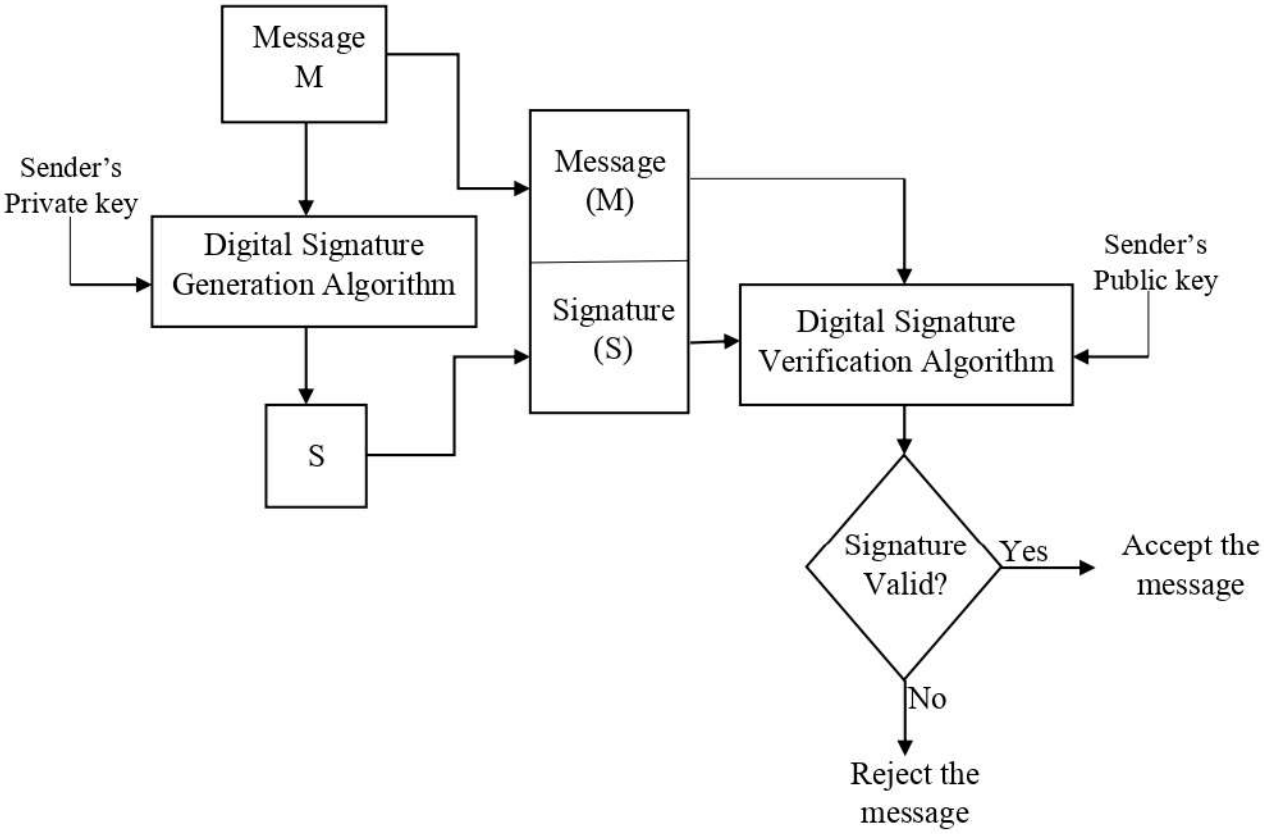


Fig: Generic model of Digital Signature

In a digital signature process, the sender uses a signing algorithm to sign the message. The message and the signature are sent to receiver. The receiver receives the message and the signature and applies the verifying algorithm to the combination. If the result is true, the message is accepted otherwise it is rejected.

Direct and Arbitrated Digital Signature

- A ***direct digital signature*** involves only two parties, a sender and a receiver. It is assumed that the receiver knows the public key of the sender. A digital signature may be formed by encrypting the entire message with the sender's private key or by encrypting a hash code of the message with the sender's private key.
  - There is a threat of forged digital signatures in direct digital signature.

- An ***arbitrated digital signature*** involves three parties, a sender, a receiver and a trusted arbiter. Every signed message from a sender X to a receiver Y goes first to an arbiter A, who subjects the message and its signature to a number of tests to check its origin and content. The message is then dated and sent to Y with an indication that it has been verified to the satisfaction of the arbiter.
  -There is no threat of forged signatures because an independent arbiter verifies the entire process with due dating and time-stamping.

Q. In arbitrated digital signature, if the signer doesn't want the arbiter to see the message, how is this signing performed?

Soln:

In order to address the issue of arbiter seeing the message, what the sender can do is, use an encrypted version of the message (encrypted using a key that is shared between X and Y) rather than the message itself while communicating with the arbiter A. This way, the arbiter will do "blind" signing rather than seeing the message.
If PKI is used, then the sender X can encrypt the message with Y's public key so that only Y can see the message and not the arbiter A.

Q. What are the properties a digital signature should have?

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

Digital Signature Requirements

The digital signature must have the following requirements:
- Must be a bit pattern depending on the message being signed.
- Signature must use some information unique to the sender to prevent forgery and denial.
- Must be computationally easy to produce a signature.
- Must be computationally easy to recognize and verify the signature.
- Must be computationally infeasible to forge a digital signature.
- Must be practical to retain a copy of the digital signature in storage.

Digital Signature Standard (DSS)

- Developed by the U.S. National Security Agency, the Digital Signature Standard (DSS) is a collection of procedures and standards for generating a digital signature used for authenticating electronic documents.
- Designed to provide only the digital signature function.
- Cannot be used for encryption or key exchange.
- Uses SHA for hashing the message.

RSA Approach for Digital Signature

When RSA is used for digital signature, 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 (PR_a) 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 (PU_a). 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.

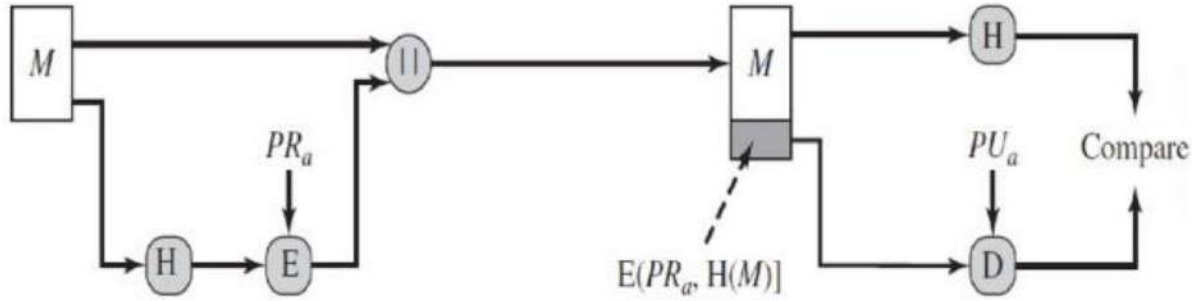


Fig: RSA Approach

DSS Approach for Digital Signature

It 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 (PR_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.

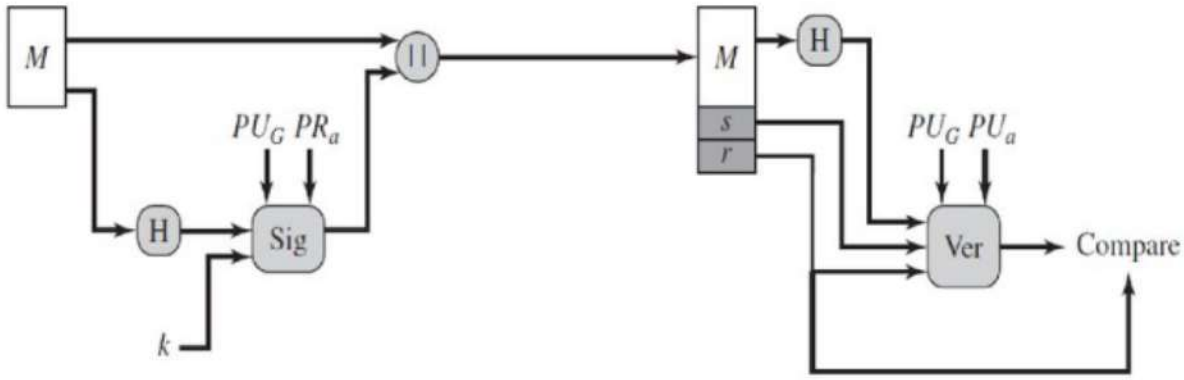
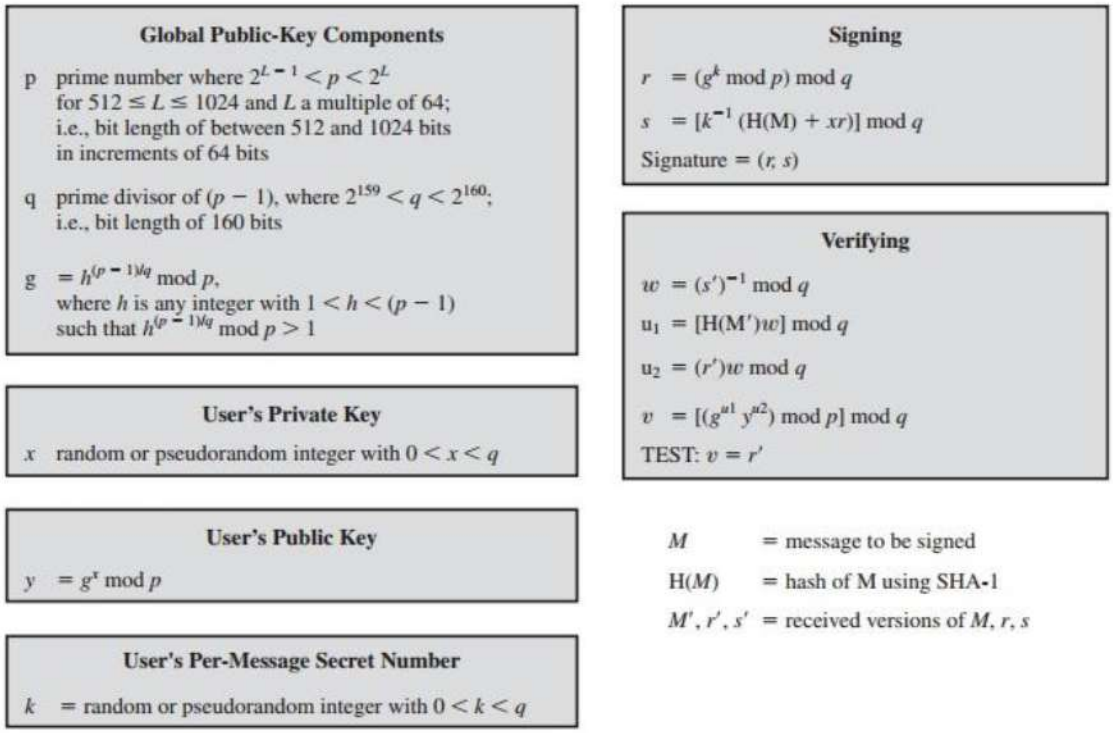


Fig: DSS Approach

Digital Signature Algorithm (DSA)

- It creates a 320 bit signature with 512-1024 bit security.
- Digital signature algorithm consists of 2 parts: generation of a pair of public key and private key; generation and verification of digital signature, which can be described as;



Comparison of Digital Signature, Hash Function and Message Authentication Code

Digital Signatures	Hash Functions	Authentication codes (MACs)
They deal with 1024 bit minimum key length [3] and have not been patented or licensed yet.	Has functions mostly deals with 128 to 256 bit key. examples: MD5, RFC, SHA1, RACE	MACs are able to reduce message into short length. It is typically 32 or 64 bit long and key is 56 bit long. MAC= hash value+ Public Key
Digital signatures rely on public key cryptography, where verification key is public and signing key is referred as secret or private key.		MACs are based on secret key or private key.
As a prerequisite, parties did not need to agree on shared key before starting encryption.	As a prerequisite, parties did not need to agree on shared key before starting encryption.	As a prerequisite, a secret key needs to be shared therefore both parties need to agree on same key before starting the encryption through authentication code.
Digital signatures are slow in processing than hash function and symmetric encryption scheme.	Fast in processing speed.	Quite fast in processing.
Did not provide confidentiality.	Did not provide confidentiality.	Did not provide confidentially.
Digital signatures provide origin authentication.	They did not provide origin authentication.	They provide origin authentication.
Digital signatures provide integrity.	They provide integrity.	They provide integrity.

Please let me know if I missed anything or anything is incorrect.

poudeljayanta99@gmail.com