SNS COLLEGE OF TECHNOLOGY

Coimbatore-35
An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A+' Grade
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

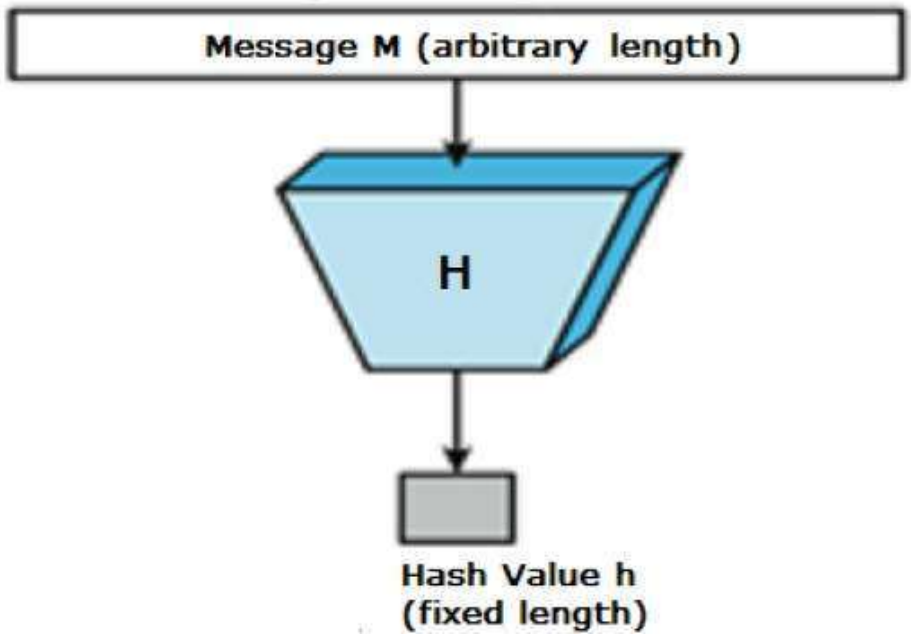
DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

19ITB302-Cryptography and Network Security

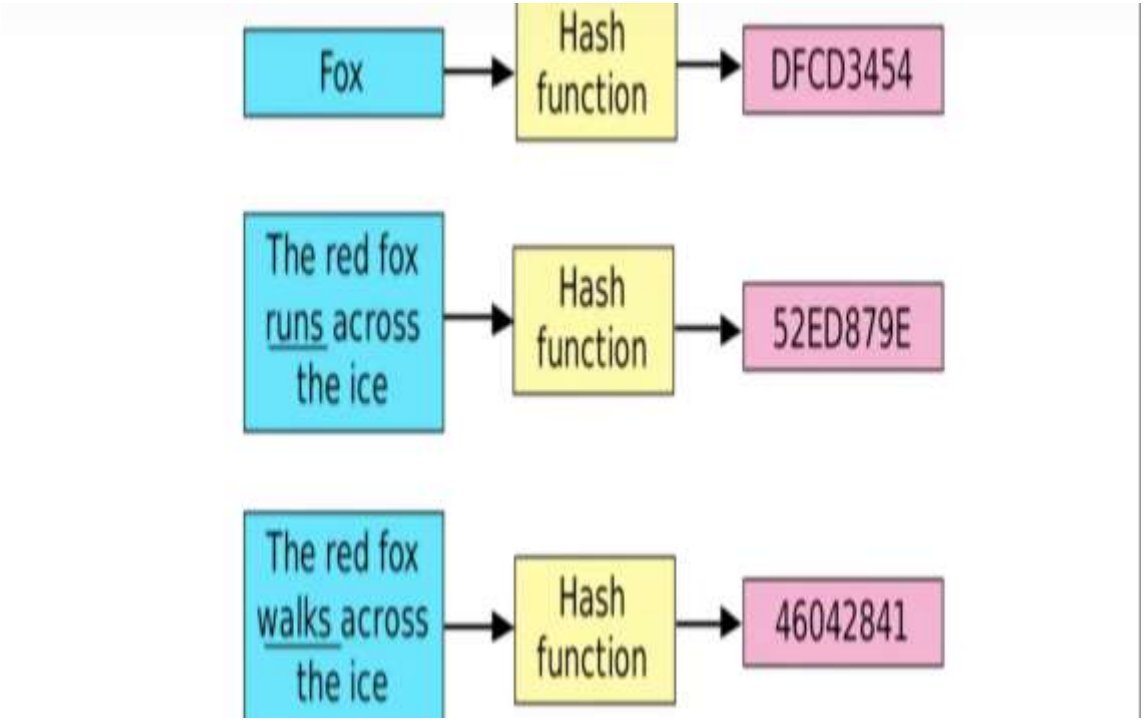
UNIT-3 HASH FUNCTION AND DIGITAL SIGNATURE

Cryptographic Hash Functions

- A **hash function** H accepts a variable-length block of data **M** as input and produces a fixed-size hash value **h = H(M)**
- Values returned by a hash function are called **message digest** or simply **hash values**.
- A change to any bit or bits in M results, with high probability, in a change to the hash code.
- The kind of hash function needed for security applications is referred to as a **cryptographic hash function.**



- A cryptographic hash function is an algorithm for which it is computationally infeasible to invert
- Because of these characteristics, hash functions are often used to determine whether or not data has changed.
- A small change in the input data will have the whole hash function output to be changed.



Properties of Hash function

- **Compression**:Output of the hash function is much smaller than the size of the input
- **Pre image resistance**: Its difficult to find the input from given hash function output, h=H(m) if h is given, it is infeasible to find m
- **Collision Resistance**: It is difficult to find m1 and m2 such that hash value H(m1)=H(m2)

Characteristics of Hash function

- It is quick to calculate hash value(h) for any given message
- Hash Function can be applied to variable length of data block
- A small Change in a message should change the hash value
- Hash function has one way property
- Hash function uses all the input data

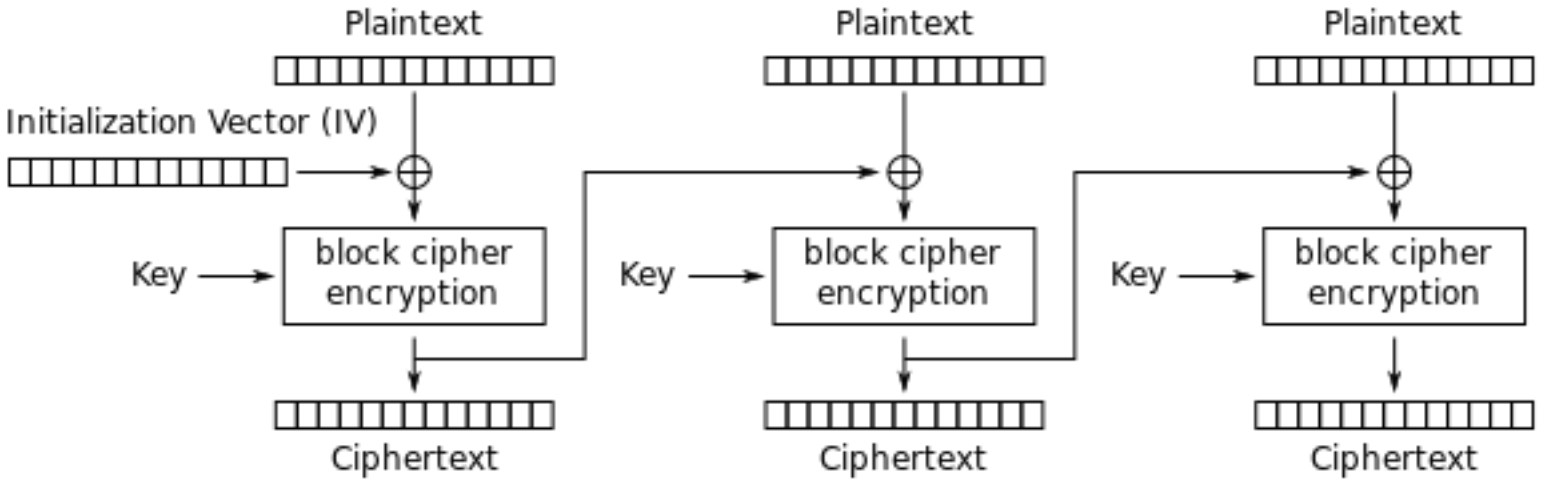
Simple Hash Functions

Bit by Bit XOR

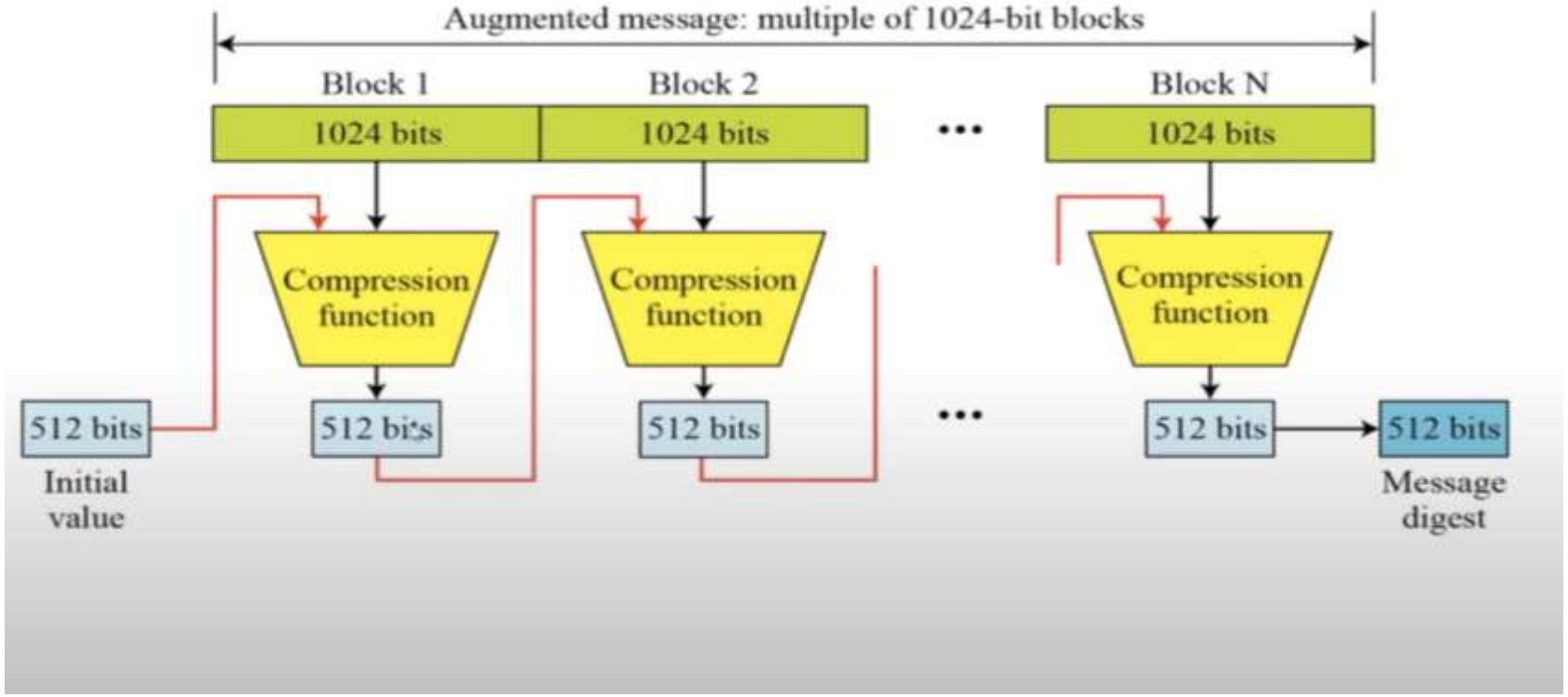
- The input (message, file, etc.) is viewed as a sequence of n-bit blocks. The input is processed one block at a time in an iterative fashion to produce an n-bit hash function.
- One of the simplest hash functions is the bit-by-bit exclusive-OR (XOR) of every block. This can be expressed as
- $Ci = bi1 \oplus bi2 \oplus \ldots \oplus bim$

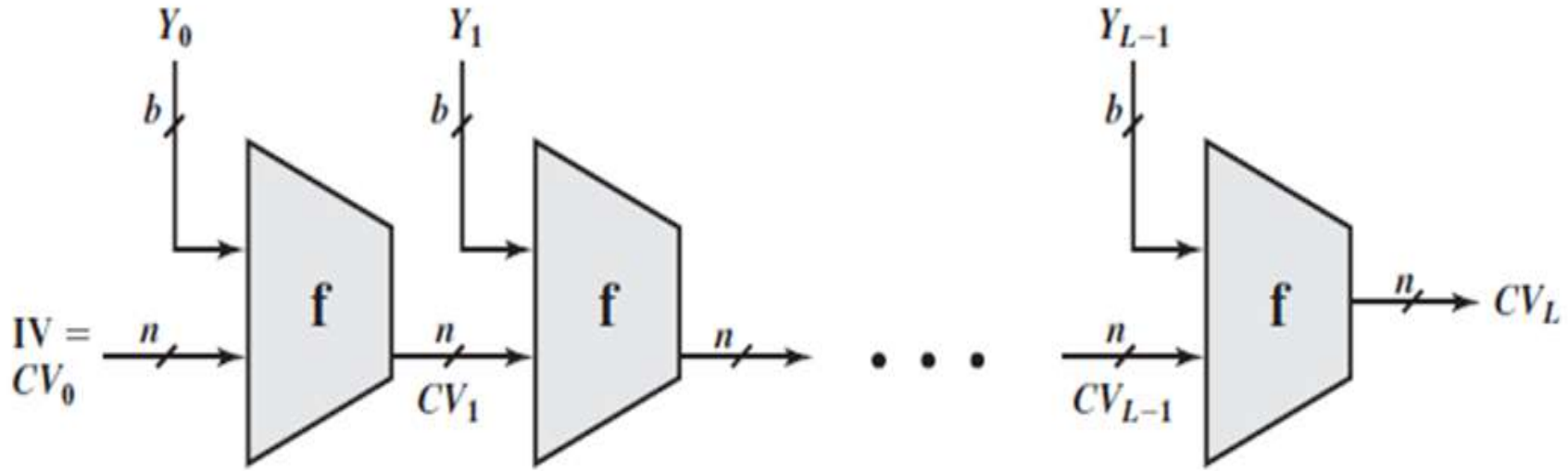
Hash Function based on CBC

Cipher Block Chain



Cipher Block Chaining (CBC) mode encryption


Augmented message: multiple of 1024-bit blocks
Block 1
Block 2
Block N
1024 bits
1024 bits
1024 bits
Compression function
Compression function
Compression function
512 bits
512 bits
512 bits
512 bits
512 bits
Initial value
Message digest



IV = Initial value
CV_i = Chaining variable
Y_i = ith input block
f = Compression algorithm

L = Number of input blocks
n = Length of hash code
b = Length of input block

Secure Hash Algorithm (SHA)

- SHA was developed by the National Institute of Standards and Technology (NIST) and published as a federal information processing standard (FIPS 180) in 1993.
- SHA-1 produces a hash value of 160 bits. In 2002, NIST produced a revised version of the standard, FIPS 180-2, that defined three new versions of SHA, with hash value lengths of 256, 384, and 512 bits, known as SHA-256, SHA-384, and SHA-512, respectively. Collectively, these hash algorithms are known as SHA-2
- The algorithm takes as input a message with a maximum length of less than 2128 bits and produces as output a 512-bit message digest. The input is processed in 1024-bit blocks

Processing of SHA

Step 1 Append padding bits.

- The message is padded so that its length is congruent to 896 modulo 1024 [length≡896(mod 1024)]. (Eg:24+872 mod 1024=896)
- Padding is always added, even if the message is already of the desired length.
- Thus, the number of padding bits is in the range of 1 to 1024.
- The padding consists of a single 1 bit followed by the necessary number of 0 bits.

Step 2 Append length.

- A block of 128 bits is appended to the message.
- The outcome of the first two steps yields a message that is an integer multiple of 1024 bits in length.

Step 3 Initialize hash buffer.

A 512-bit buffer is used to hold intermediate and final results of the hash function.

- a = 6A09E667F3BCC908
- b = BB67AE8584CAA73B
- c = 3C6EF372FE94F82B
- d = A54FF53A5F1D36F1
- e = 510E527FADE682D1
- f = 9B05688C2B3E6C1F
- g = 1F83D9ABFB41BD6B
- h = 5BE0CD19137E2179

Step 4 Process message in 1024-bit (128-word) blocks.

The heart of the algorithm is a module that consists of 80 rounds

Processing of SHA

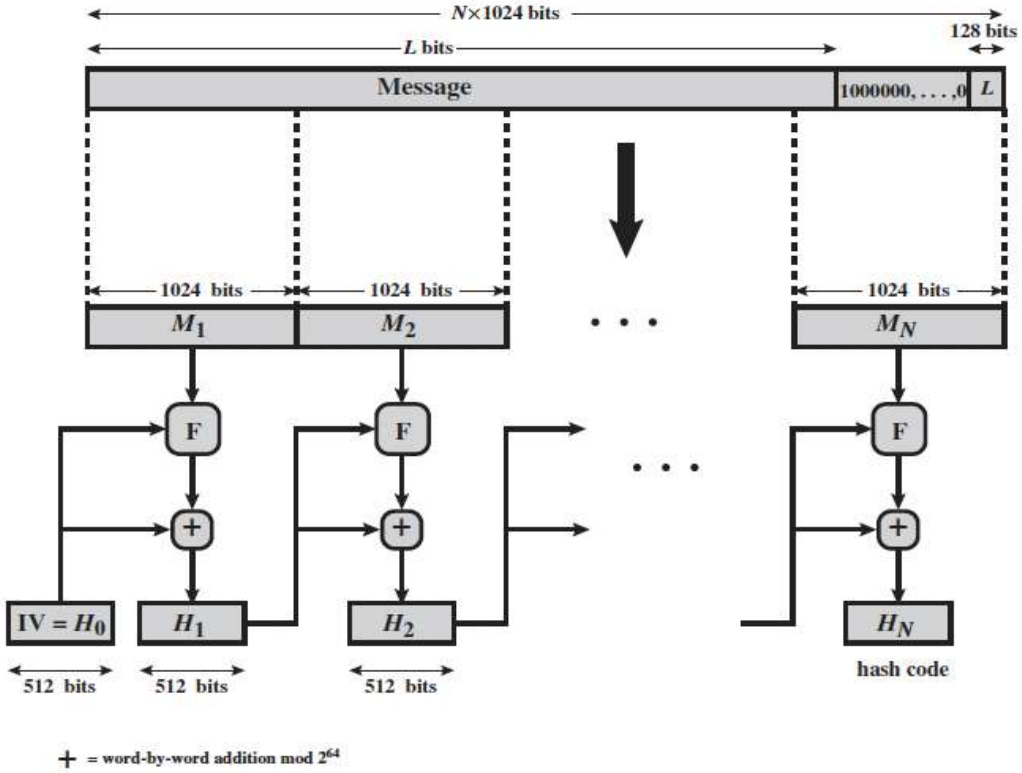


Figure 11.9 Message Digest Generation Using SHA-512

Processing of Single 1024 Bit Block

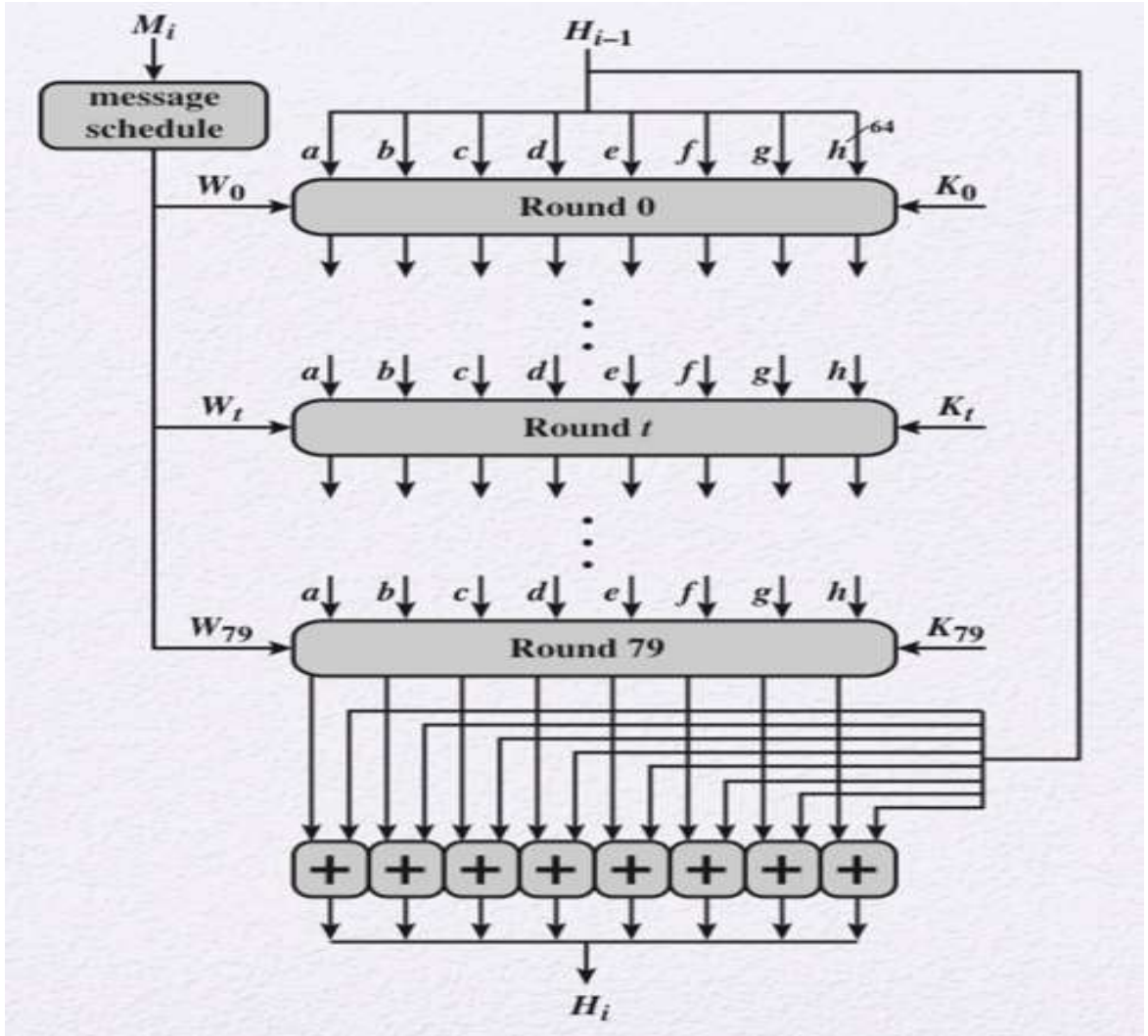
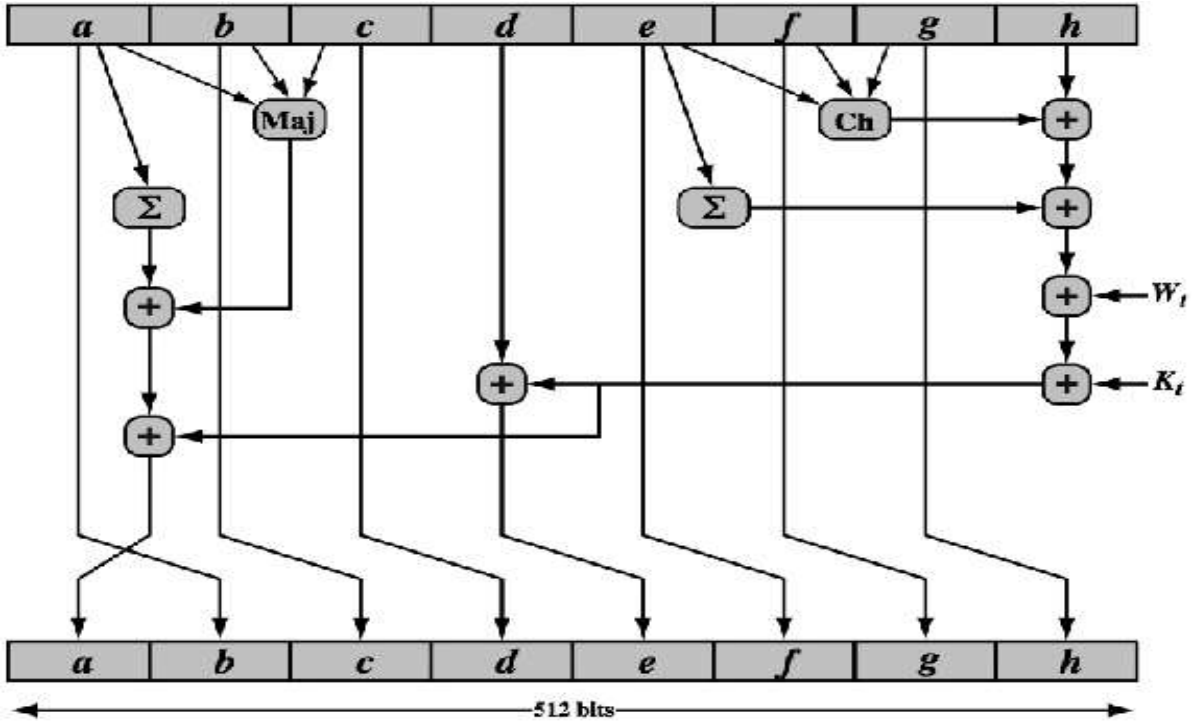


Table 11.4 SHA-512 Constants

428a2f98d728ae22	7137449123ef65cd	b5c0fbcfec4d3b2f	e9b5dba58189dbbc
3956c25bf348b538	59f111f1b605d019	923f82a4af194f9b	ab1c5ed5da6d8118
d807aa98a3030242	12835b0145706fbe	243185be4ee4b28c	550c7dc3d5ffb4e2
72be5d74f27b896f	80deb1fe3b1696b1	9bdc06a725c71235	c19bf174cf692694
e49b69c19ef14ad2	efbe4786384f25e3	0fc19dc68b8cd5b5	240ca1cc77ac9c65
2de92c6f592b0275	4a7484aa6ea6e483	5cb0a9dcbd41fbd4	76f988da831153b5
983e5152ee66dfab	a831c66d2db43210	b00327c898fb213f	bf597fc7beef0ee4
c6e00bf33da88fc2	d5a79147930aa725	06ca6351e003826f	142929670a0e6e70
27b70a8546d22ffc	2e1b21385c26c926	4d2c6dfc5ac42aed	53380d139d95b3df
650a73548baf63de	766a0abb3c77b2a8	81c2c92e47edaee6	92722c851482353b
a2bfe8a14cf10364	a81a664bbc423001	c24b8b70d0f89791	c76c51a30654be30
d192e819d6ef5218	d69906245565a910	f40e35855771202a	106aa07032bbd1b8
19a4c116b8d2d0c8	1e376c085141ab53	2748774cdf8eeb99	34b0bcb5e19b48a8
391c0cb3c5c95a63	4ed8aa4ae3418acb	5b9cca4f7763e373	682e6ff3d6b2b8a3
748f82ee5defb2fc	78a5636f43172f60	84c87814a1f0ab72	8cc702081a6439ec
90befffa23631e28	a4506cebde82bde9	bef9a3f7b2c67915	c67178f2e372532b
ca273eceea26619c	d186b8c721c0c207	eada7dd6cde0eb1e	f57d4f7fee6ed178
06f067aa72176fba	0a637dc5a2c898a6	113f9804bef90dae	1b710b35131c471b
28db77f523047d84	32caab7b40c72493	3c9ebe0a15c9bebc	431d67c49c100d4c
4cc5d4becb3e42b6	597f299cfc657e2a	5fcb6fab3ad6faec	6c44198c4a475817

SHA-512 Round Function



$$T_1 = h + \mathrm{Ch}(e,f,g) + \left(\sum_1^{512} e\right) + W_t + K_t$$

$$T_2 = \left(\sum_0^{512} a\right) + \mathrm{Maj}(a,b,c)$$

$$h = g$$

$$g = f$$

$$f = e$$

$$e = d + T_1$$

$$d = c$$

$$c = b$$

$$b = a$$

$$a = T_1 + T_2$$

where

t = step number; $0 \le t \le 79$

$\mathrm{Ch}(e,f,g)$ = $(e \text{ AND } f) \oplus (\text{NOT } e \text{ AND } g)$
the conditional function: If e then f else g

$\mathrm{Maj}(a,b,c)$ = $(a \text{ AND } b) \oplus (a \text{ AND } c) \oplus (b \text{ AND } c)$
the function is true only of the majority (two or three) of the arguments are true

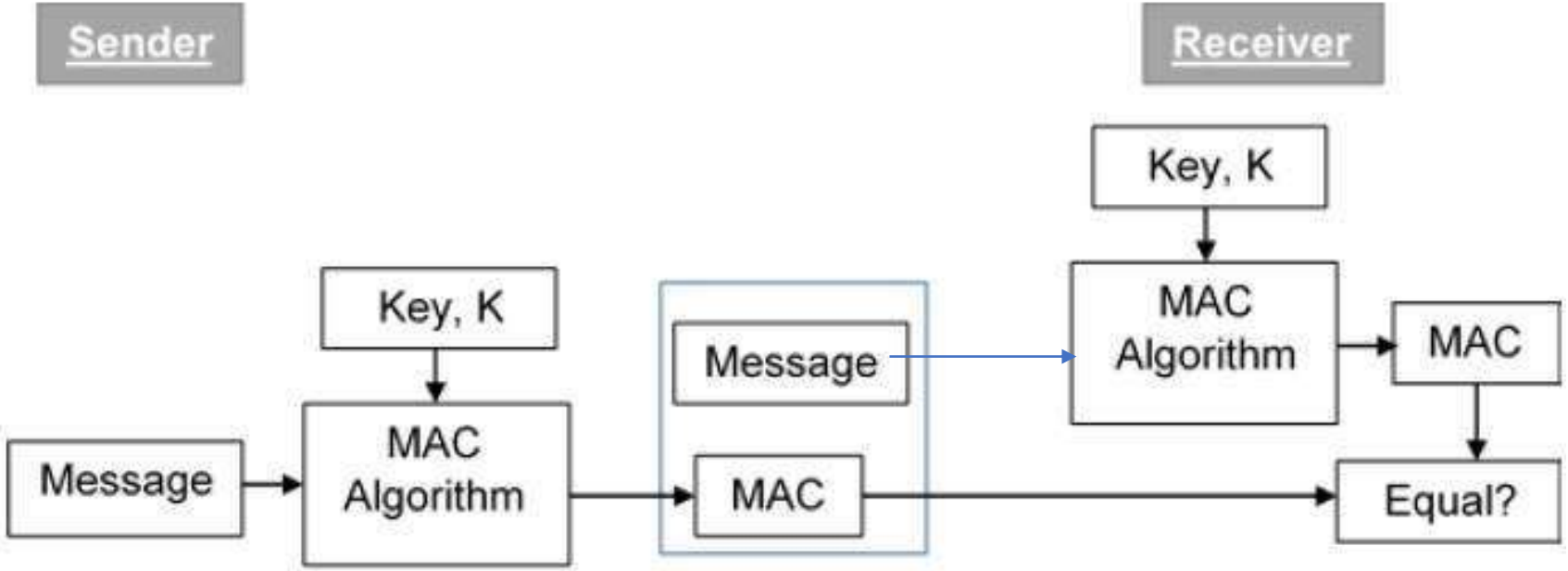
$\left(\sum_0^{512} a\right)$ = $\mathrm{ROTR}^{28}(a) \oplus \mathrm{ROTR}^{34}(a) \oplus \mathrm{ROTR}^{39}(a)$

$\left(\sum_1^{512} e\right)$ = $\mathrm{ROTR}^{14}(e) \oplus \mathrm{ROTR}^{18}(e) \oplus \mathrm{ROTR}^{41}(e)$

$\mathrm{ROTR}^n(x)$ = circular right shift (rotation) of the 64-bit argument x by n bits

Message Authentication Code

- A message authentication code (MAC) is an algorithm that requires the use of a secret key.
- A MAC takes a variable-length message and a secret key as input andproduces an authentication code.
- A recipient in possession of the secret key can generate an authentication code to verify the integrity of the message



Authentication Requirements

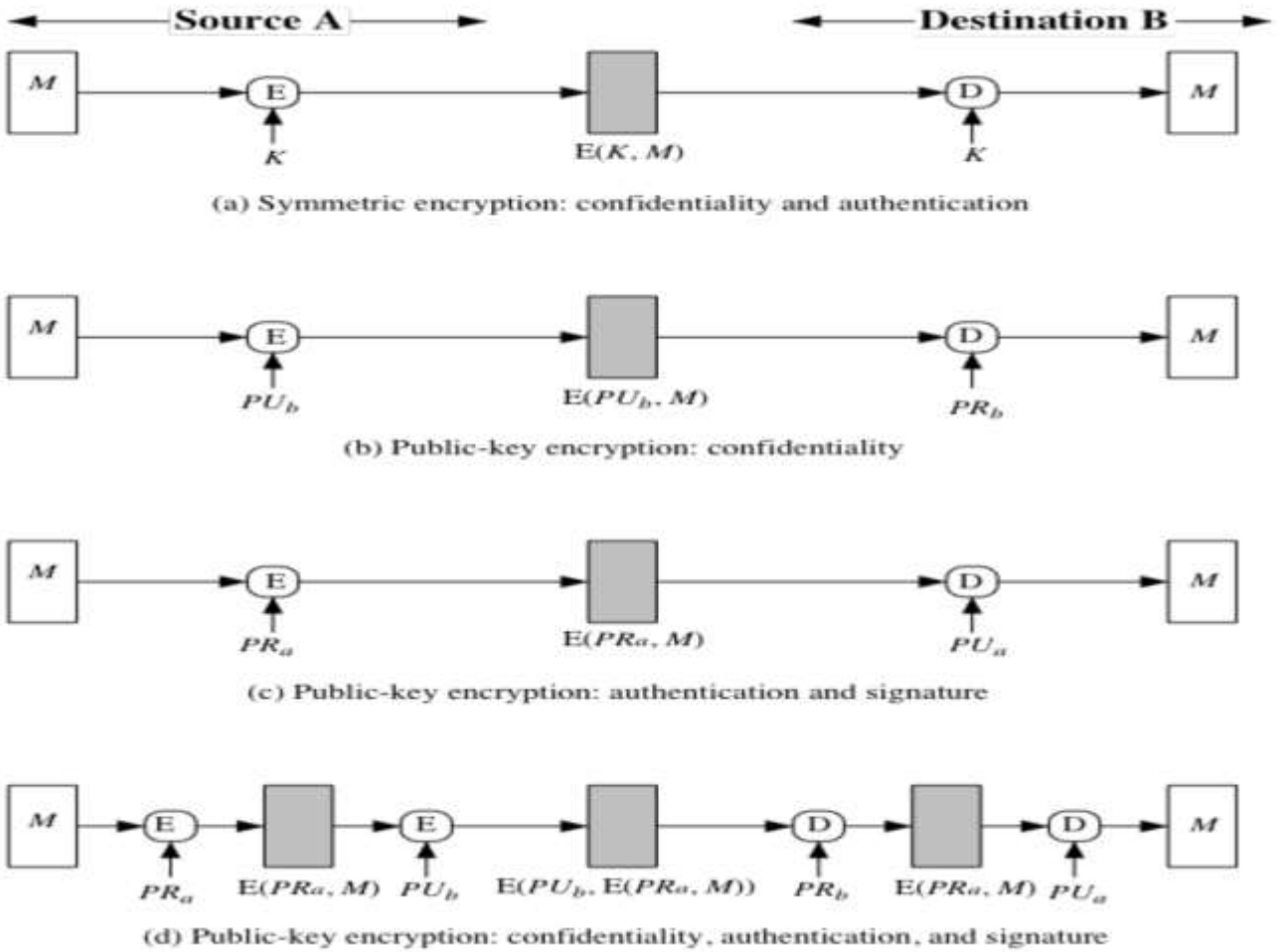
- **Disclosure:** Release of message contents to any person or process not possessing the appropriate cryptographic key.
- **Traffic analysis:** Discovery of the pattern of traffic between parties.
- **Masquerade:** Insertion of messages into the network from a fraudulent source.
- **Content modification:** Changes to the contents of a message, including insertion, deletion, transposition, and modification.
- **Sequence modification:** Any modification to a sequence of messages between parties, including insertion, deletion, and reordering.
- **Timing modification:** Delay or replay of messages. In a connection-oriented application, an entire session or sequence of messages could be a replay of some previous valid session, or individual messages in the sequence could be delayed or replayed.
- **Source repudiation:** Denial of transmission of message by source.
- **Destination repudiation:** Denial of receipt of message by destination.

1. **Hash function** - A function that maps a message of any length into a fixed length hash value, which serves as the authenticator

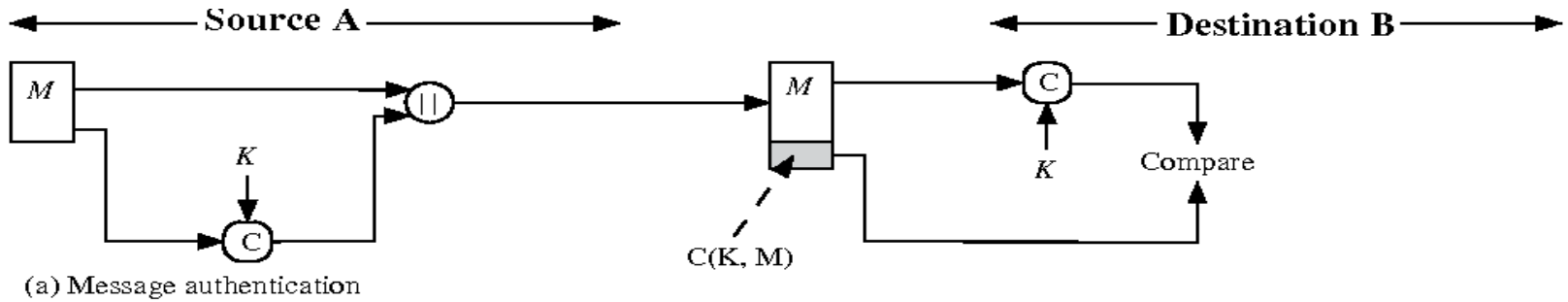
2. **Message encryption** - The ciphertext of the entire message serves as its authenticator

3. **Message Authentication Code (MAC) -** A function of the message and a secretkey that produces a fixed-length value that serves as the authenticator.

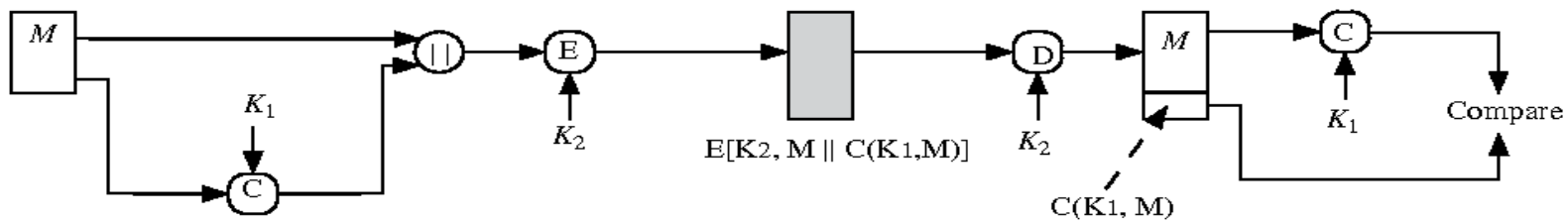
Message Encryption



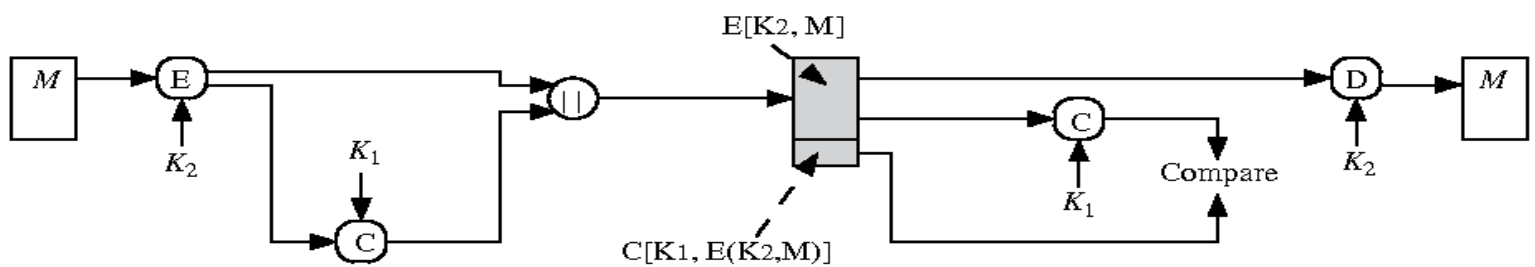
Message Authentication Code



(a) Message authentication



(b) Message authentication and confidentiality; authentication tied to plaintext



(c) Message authentication and confidentiality; authentication tied to ciphertext

REQUIREMENTS FOR MESSAGE AUTHENTICATION CODES

- The MAC is appended to the message at the source at a time when the message is assumed or known to be correct. The receiver authenticates that message by recomputing the MAC.
- If an opponent observes and, it should be computationally infeasible for the opponent to construct a message M' such that MAC (K, M') = MAC (K, M)
- MAC(K, M) should be uniformly distributed in the sense that for randomly chosen messages, M and M' , the probability that is MAC(K, M) = MAC(K, M') is 2^n, where n is the number of bits in the MAC

HMAC:

- Hash-based Message Authentication Code (HMAC) is a type of message authentication code (MAC) involving a cryptographic hash function and a secret cryptographic key. HMAC makes it possible to confirm the data integrity and authenticity of a message.
- HMAC is a great resistance towards cryptanalysis attacks as it uses the Hashing concept twice. HMAC consists of twin benefits of Hashing and MAC and thus is more secure than any other authentication code.

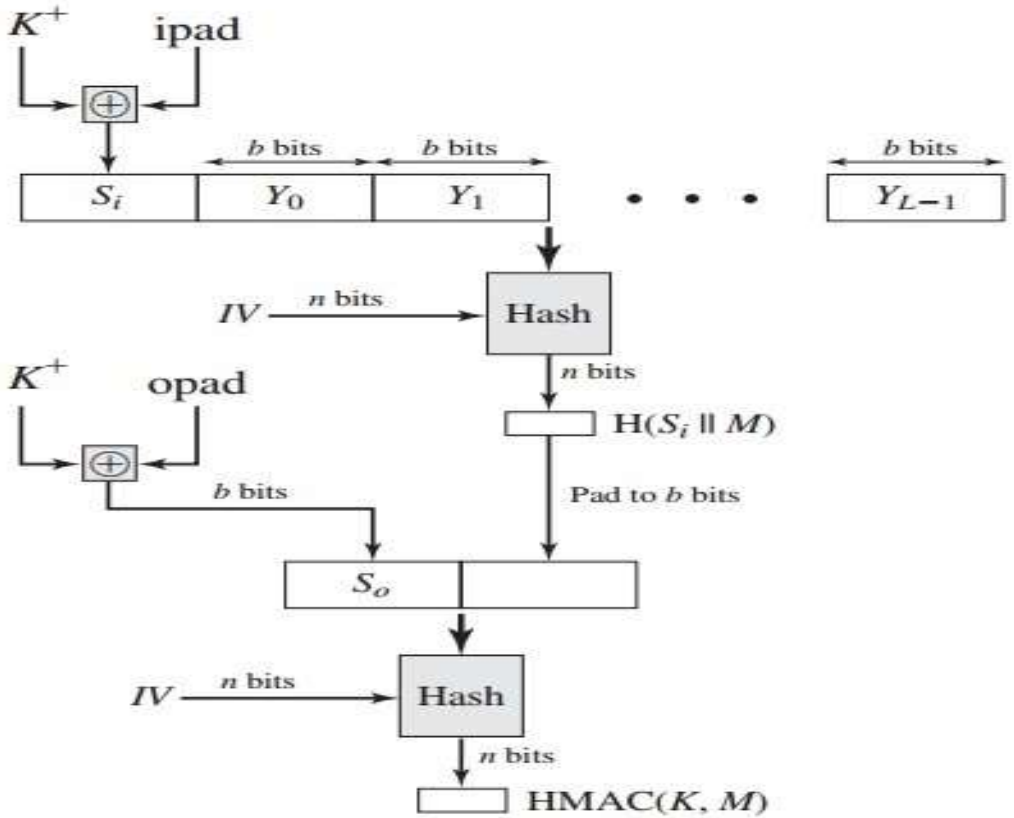


Figure 12.5 HMAC Structure

- H = embedded hash function (e.g.,MD5, SHA-1, RIPEMD-160)
- IV = initial value input to hash function
- M = message input to HMAC
- Yi = i th block of M, $0 \leq i \leq L - 1$
- L = number of blocks in M
- b = number of bits in a block
- n = length of hash code produced by embedded hash function
- K = secret key; recommended length is ≥ n; if key length is greater than b, the key is input to the hash function to produce an n-bit key.
- K+ = K padded with zeros on the left so that the result is b bits in length ipad = 00110110 (36 in hexadecimal) repeated b/8 times
- opad = 01011100 (5C in hexadecimal) repeated b/8 times

Digital Signature

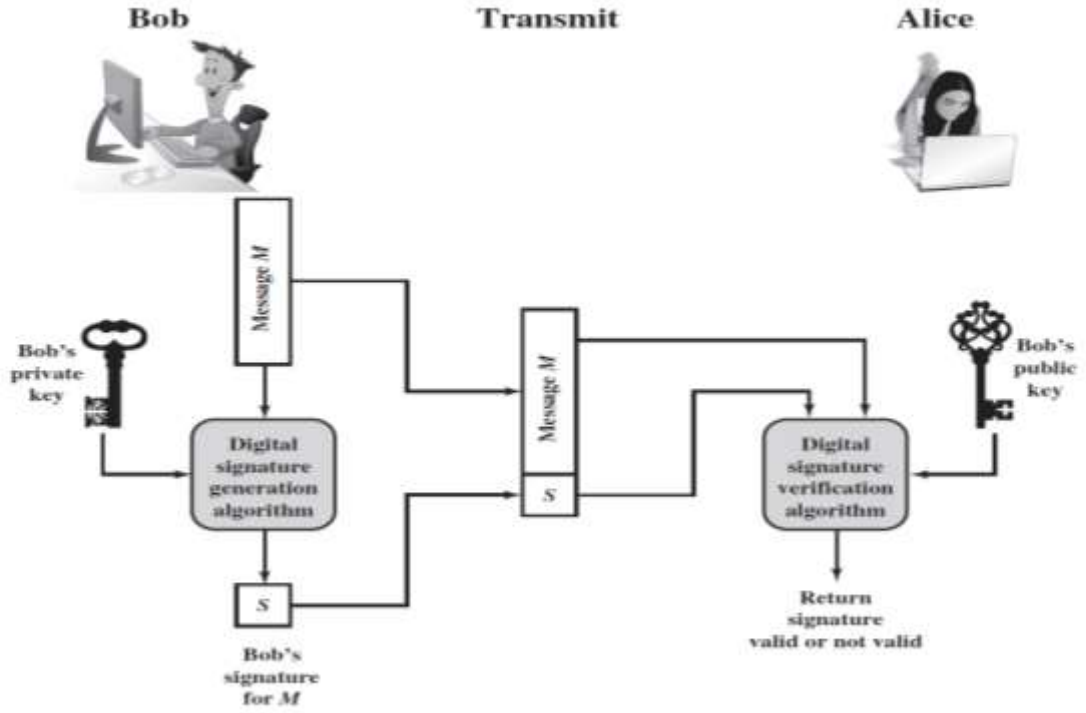


Figure 13.1 Generic Model of Digital Signature Process

Requirements

- The signature must be a bit pattern that depends on the message being signed.
- The signature must use some information unique 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.