Unit III – Syllabus

UNIT III HASH FUNCTIONS AND DIGITAL SIGNATURES (8 hours)

Authentication requirement

Authentication function

MAC – Hash function – Security of hash function and MAC

- MD5
- SHA
- HMAC
- CMAC –

Digital signature and authentication protocols –

- DSS –
- EI Gamal –
- Schnorr.

Day 19 & 20

Authentication requirement

Authentication function

MAC – Hash function – Security of hash function and MAC

- MD5

Message Authentication Requirements

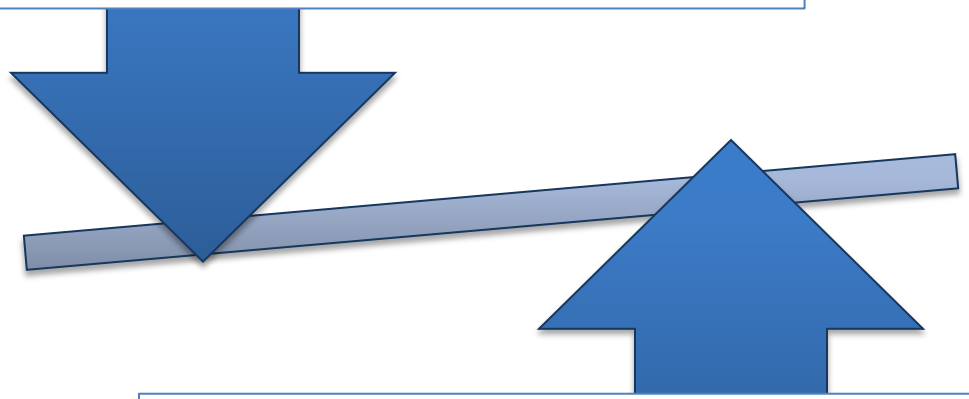
- Disclosure
  - Release of message contents to any person or process not possessing the appropriate cryptographic key
- Traffic analysis
  - Discovery of the pattern of traffic between parties
- Masquerade
  - Insertion of messages into the network from a fraudulent source
- Content modification
  - 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
- Source repudiation
  - Denial of transmission of message by source
- Destination repudiation
  - Denial of receipt of message by destination

Message Authentication Functions

- Two levels of functionality:

Lower level

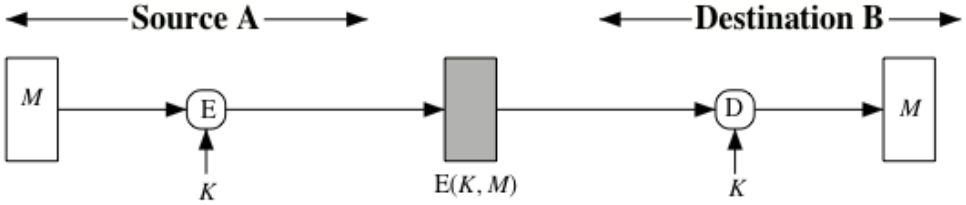
- There must be some sort of function that produces an authenticator



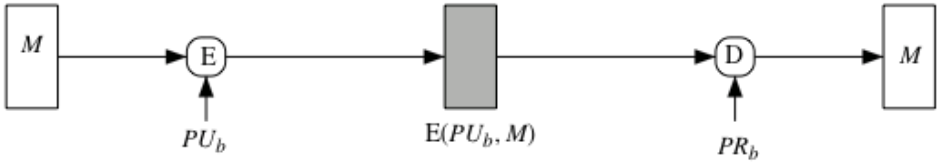
Higher-level

- Uses the lower-level function as a primitive in an authentication protocol that enables a receiver to verify the authenticity of a message

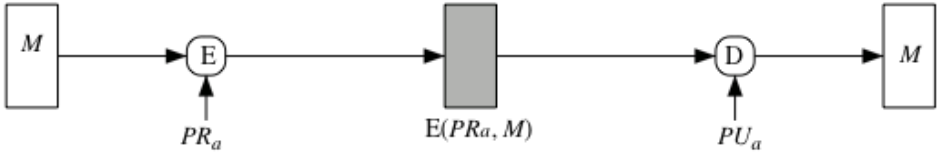
- Hash function
  - A function that maps a message of any length into a fixed-length hash value which serves as the authenticator
- Message encryption
  - The ciphertext of the entire message serves as its authenticator
- Message authentication code (MAC)
  - A function of the message and a secret key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



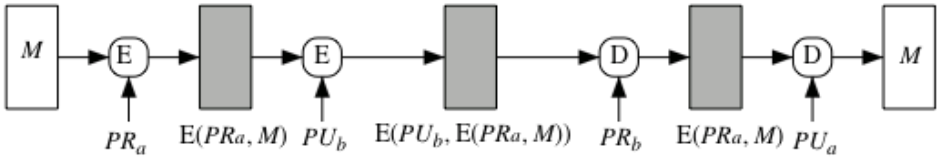
(a) Symmetric encryption: confidentiality and authentication



(b) Public-key encryption: confidentiality



(c) Public-key encryption: authentication and signature



(d) Public-key encryption: confidentiality, authentication, and signature

Figure 12.1 Basic Uses of Message Encryption

Public-Key Encryption

- The straightforward use of public-key encryption provides confidentiality but not authentication
- To provide both confidentiality and authentication, A can encrypt *M* first using its private key which provides the digital signature, and then using B's public key, which provides confidentiality
- Disadvantage is that the public-key algorithm must be exercised four times rather than two in each communication

MAC concept

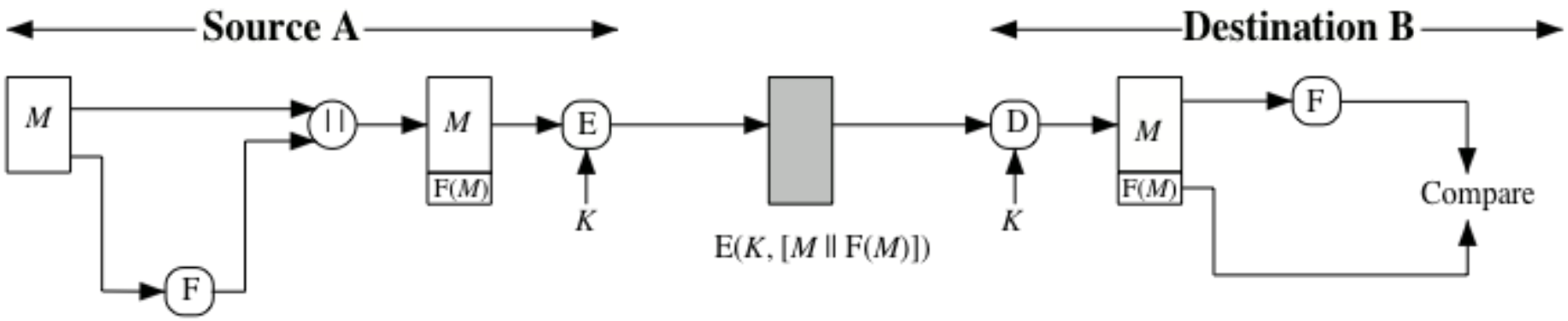
Requirements for MACs

Taking into account the types of attacks, the MAC needs to satisfy the following:

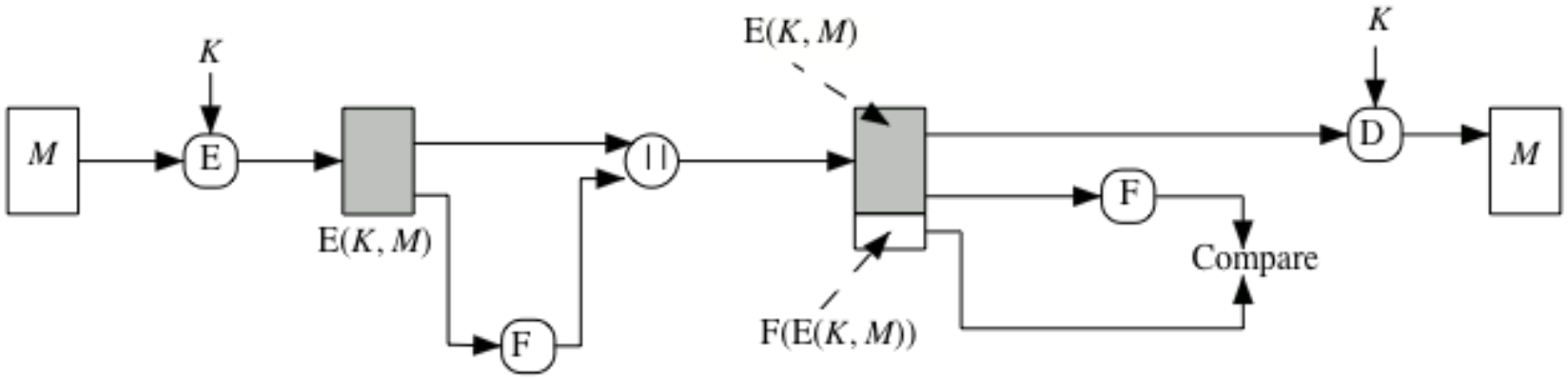
The first requirement deals with message replacement attacks, in which an opponent is able to construct a new message to match a given MAC, even though the opponent does not know and does not learn the key

The second requirement deals with the need to thwart a brute-force attack based on chosen plaintext

The final requirement dictates that the authentication algorithm should not be weaker with respect to certain parts or bits of the message than others



(a) Internal error control



(b) External error control

Figure 12.2 Internal and External Error Control

Message Digest 5 – (MD-5)

Input : blocks of 512 bits

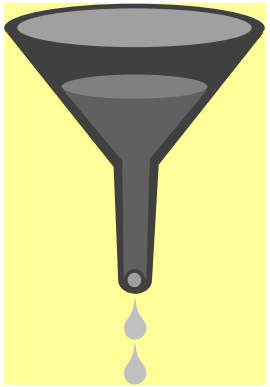
Initial Vector: 128 bits

Output: 128 bits

For each 512 bits input: 4 rounds performed

MD5: Message Digest Version 5

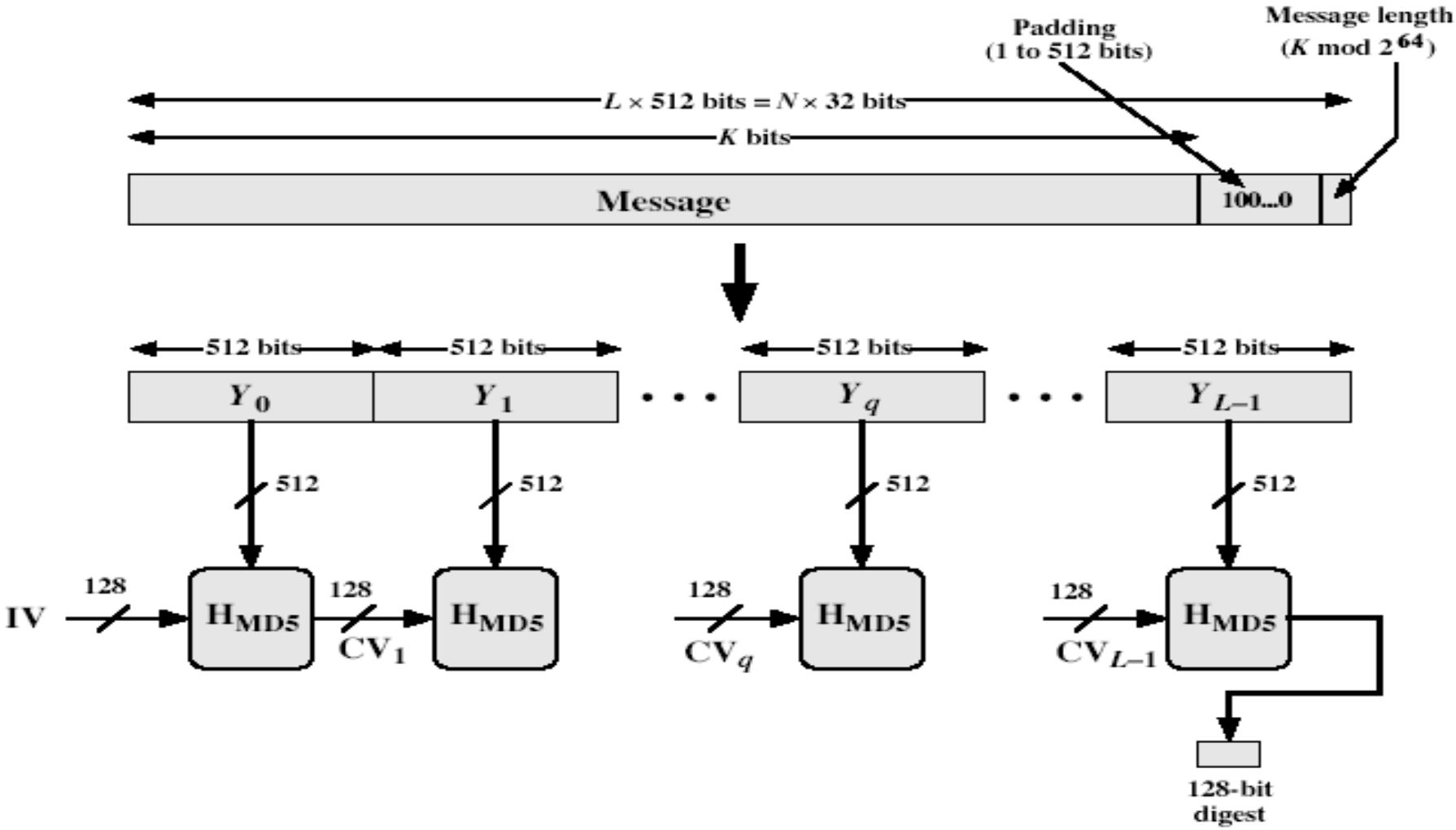
input Message



Output 128 bits Digest

- Until recently the most widely used hash algorithm
  - in recent times have both brute-force & cryptanalytic concerns
- Specified as Internet standard RFC1321

MD5 Overview



MD5 Overview

1. Pad message so its length is 448 mod 512
2. Append a 64-bit original length value to message
3. Initialise 4-word (128-bit) MD buffer (A,B,C,D)
4. Process message in 16-word (512-bit) blocks:
   - Using 4 rounds of 16 bit operations on message block & buffer
   - Add output to buffer input to form new buffer value
5. Output hash value is the final buffer value

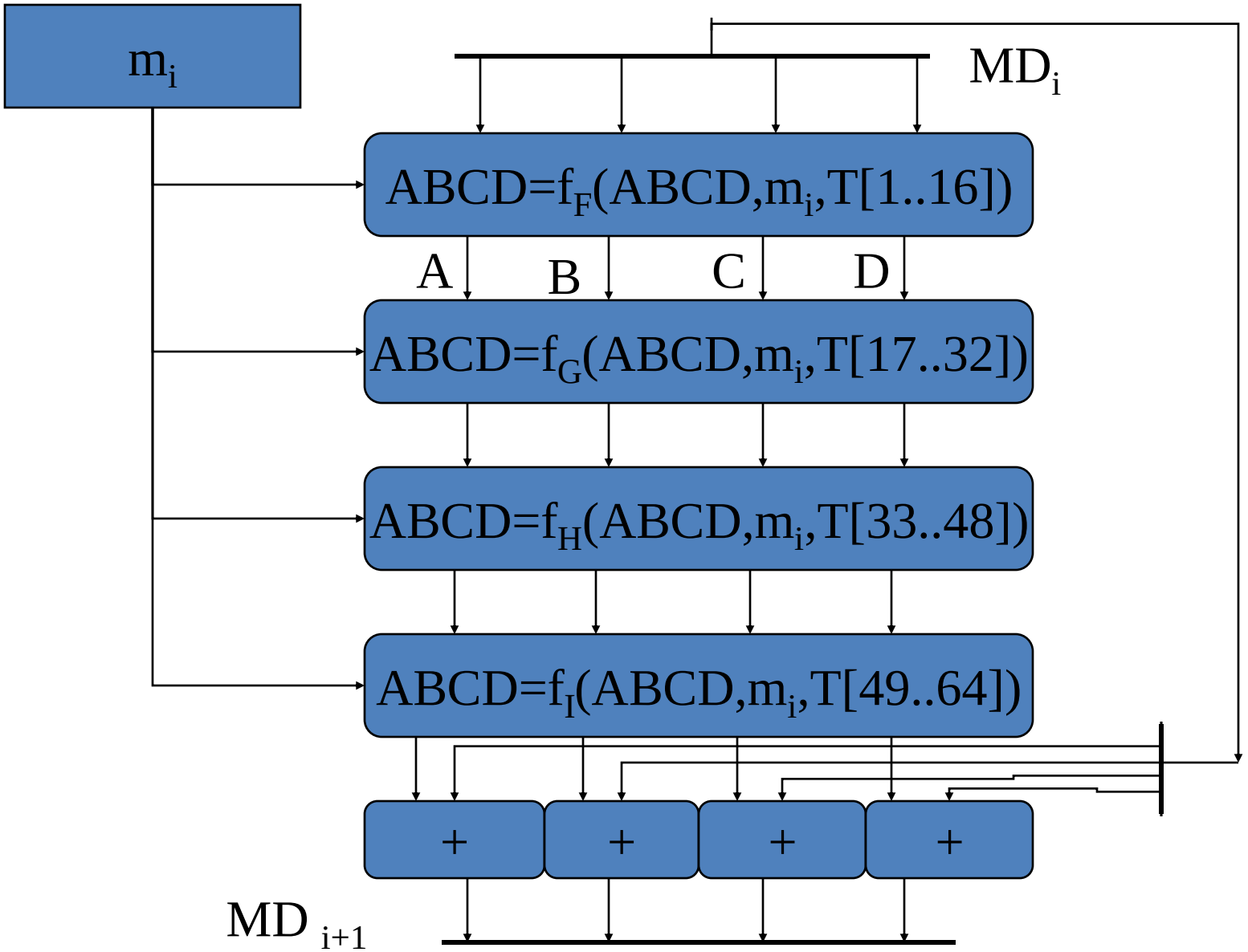
Padding Twist

- Given original message M, add padding bits "10^*" such that resulting length is 64 bits less than a multiple of 512 bits.
- Append (*original length in bits* mod 2^{64}), represented in 64 bits to the padded message
- Final message is chopped 512 bits a block

MD5 Process

- As many stages as the number of 512-bit blocks in the final padded message
- Digest: 4 32-bit words: MD=A|B|C|D
- Every message block contains 16 32-bit words: $m_0|m_1|m_2\ldots|m_{15}$
  - Digest MD_0 initialized to: A=01234567,B=89abcdef,C=fedcba98, D=76543210
  - Every stage consists of 4 passes over the message block, each modifying MD
- Each block 4 rounds, each round 16 steps

Processing of Block m_i - *4 Passes*



Different Passes...

Each step *t* (0 <= *t* <= 79):

- Input:
  - m_t – a 32-bit word from the message

    With different shift every round
  - T_t – int(2^{32} * abs(sin(i))), 0<i<65

    Provided a randomized set of 32-bit patterns, which eliminate any regularities in the input data
  - ABCD: current MD
- Output:
  - ABCD: new MD

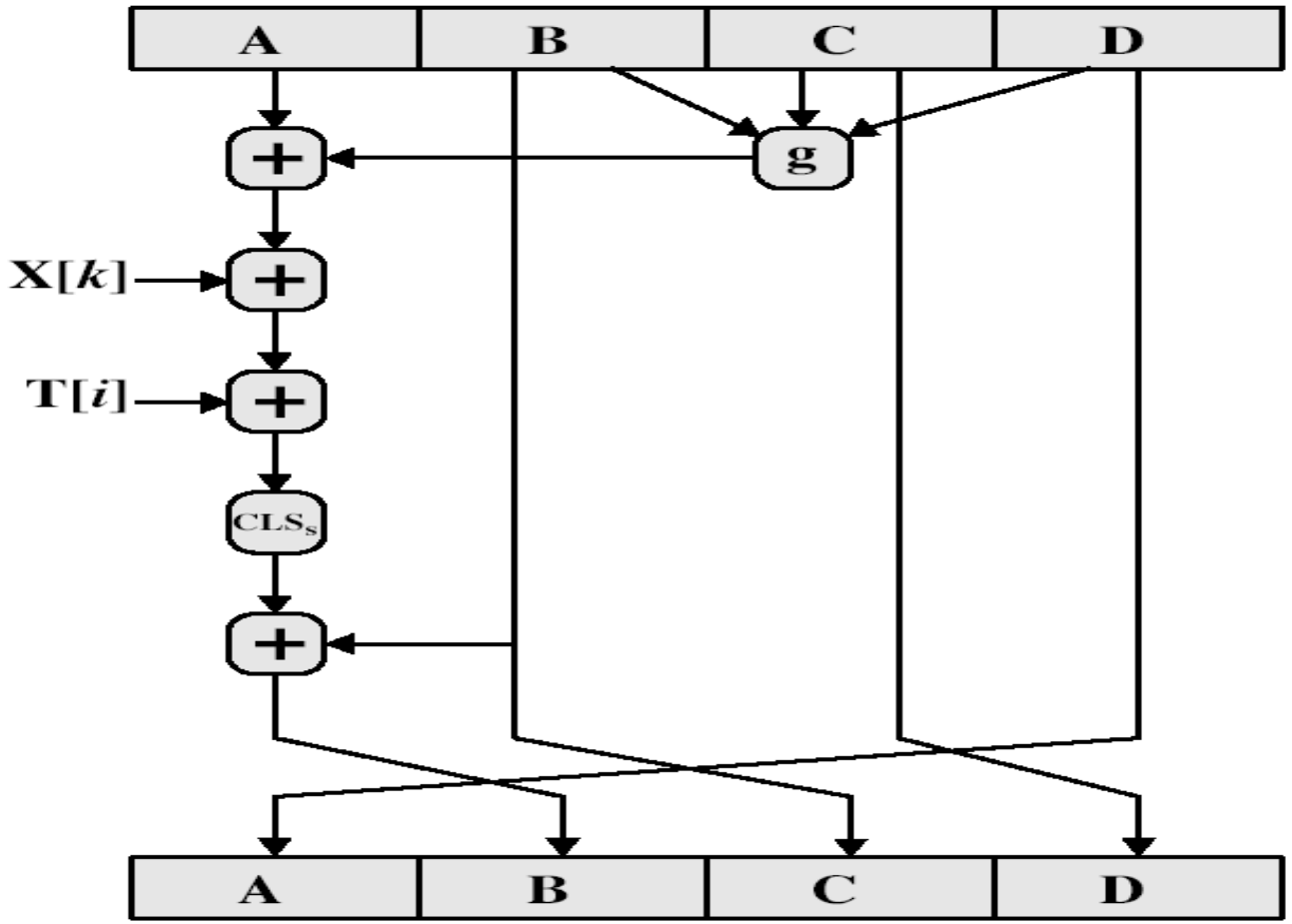
MD5 Compression Function

- Each round has 16 steps of the form:

  a = b+((a+g(b,c,d)+X[k]+T[i])<<<s)

- a,b,c,d refer to the 4 words of the buffer, but used in varying permutations
  - note this updates 1 word only of the buffer
  - after 16 steps each word is updated 4 times
- where g(b,c,d) is a different nonlinear function in each round (F,G,H,I)

MD5 Compression Function



Functions and Random Numbers

- F(B,C,D) == (B$\wedge$C)$\vee$(~B $\wedge$ D)
  - selection function
- G(B,C,D) == (B $\wedge$ D) $\vee$(C $\wedge$~ D)
- H(B,C,D) == B$\oplus$C$\oplus$D
- I(B,C,D) == C$\oplus$(B $\wedge$ ~D)