UNIT I INTRODUCTION

Basics of cryptography, conventional and public-key cryptography, hash functions, authentication, and digital signatures.

1.1 BASICS OF CRYPTOGRAPHY

Cryptography is the science of using mathematics to encrypt and decrypt data. Cryptography enables you to store sensitive information or transmit it across insecure networks (like the Internet) so that it cannot be read by anyone except the intended recipient.

Important Terminologies

Plain text: An original message is known as the **plaintext**.

Cipher text: The coded message is called the **cipher text**.

Encryption: The process of converting from plaintext to cipher text is known as enciphering or encryption.

Decryption: The process of converting from cipher text in to plain text is known as deciphering or decryption.

Cryptography The many schemes used for encryption constitute the area of study known as cryptography. Such a scheme is known as a cryptographic system or a cipher.

Cryptanalysis: Techniques used for deciphering a message without any knowledge of the enciphering details fall into the area of cryptanalysis. Cryptanalysis is what the layperson calls "breaking the code."

Cryptology: The areas of cryptography and cryptanalysis together are called cryptology.

History of Cryptography (Classical Encryption Techniques, Conventional Cryptography)

Hieroglyph

The first known evidence of cryptography can be traced to the use of 'hieroglyph'. Some 4000 years ago, the Egyptians used to communicate by messages written in hieroglyph

Conventionally, there are two types of ciphers. They are the following:

1. Substitution Ciphers

2. Transposition Ciphers

SUBSTITUTION TECHNIQUES

A substitution technique is one in which the letters of plaintext are replaced by other letters or by numbers or symbols.

Substitution ciphers can be categorized as either

i) Mono alphabetic ciphers or ii) polyalphabetic ciphers.

Various substitution ciphers are

(i) Caesar Cipher

(ii) Mono alphabetic cipher

(iii) Playfair cipher

(iv) Hill cipher

(v) Poly alphabetic cipher

(vi) Vignere cipher

CAESAR CIPHER (OR) SHIFT CIPHER

Caeser cipher was proposed by Julius Caesar. The Caesar cipher involves replacing each letter of the alphabet with the letter standing 3 places further down the alphabet.

PLAINTEXT : internet

CYPHERTEXT : lqwhuqhw

MONOALPHABETIC CIPHER

- Each plaintext letter maps to a different random cipher text letter
- Here, 26! Possible keys are used to eliminate brute force attack

PLAYFAIR CIPHER

The best known multiple letter encryption cipher is the playfair, which treats digrams in the plaintext as single units and translates these units into cipher text digrams. The playfair algorithm is based on the use of 5x5 matrix of letters constructed using a keyword.

Rules for encryption

- Repeating plaintext letters that would fall in the same pair are separated with a filler letter such as ‘x’.

- Two plaintext letters that fall in the same row of the matrix are each replaced by the letter to the right, with the first element of the row circularly following the last. For example, ar is encrypted as RM.
- Two plaintext letters that fall in the same column are each replaced by the letter beneath, with the top element of the column circularly following the last. For example, mu is encrypted as CM.
- Otherwise, each plaintext letter in a pair is replaced by the letter that lies in its own row and the column occupied by the other plaintext letter. Thus, hs becomes BP and ea becomes IM (or JM, as the encipherer wishes).

HILL CIPHER

It is a multi-letter cipher. It is developed by Lester Hill. The encryption algorithm takes m successive plaintext letters and substitutes for them m cipher text letters.

$C=KP \bmod 26$

$P=K^{-1}C \bmod 26$

POLYALPHABETIC CIPHERS

Poly alphabetic cipher is a simple technique to improve mono-alphabetic technique.

The features are

- A set of related mono-alphabetic substitution rules are used
- A key determines which particular rule is chosen for a given transformation.

Example: Vigenere Cipher

TRANSPOSITION TECHNIQUES

A very different kind of mapping is achieved by performing some sort of permutation on the plaintext letters. This technique is referred to as a transposition cipher.

Examples

- Rail Fence Cipher
- Row transposition ciphers

Goal and Services

Goal: The primary goal of cryptography is to secure important data on the hard disk or as it passes through a medium that may not be secure itself. Usually, that medium is a computer network.

Services: Cryptography can provide the following services:

- Confidentiality (secrecy)
- Integrity (anti-tampering)
- Authentication
- Non-repudiation.

Types of Cryptography

- Symmetric Key Cryptography
- Asymmetric Key Cryptography
- Hash Functions

1.2 CONVENTIONAL AND PUBLIC-KEY CRYPTOGRAPHY

Conventional Cryptography (Symmetric Cryptography) (Private key Crypto Systems)

Symmetric encryption is also referred to as conventional encryption or single-key encryption.

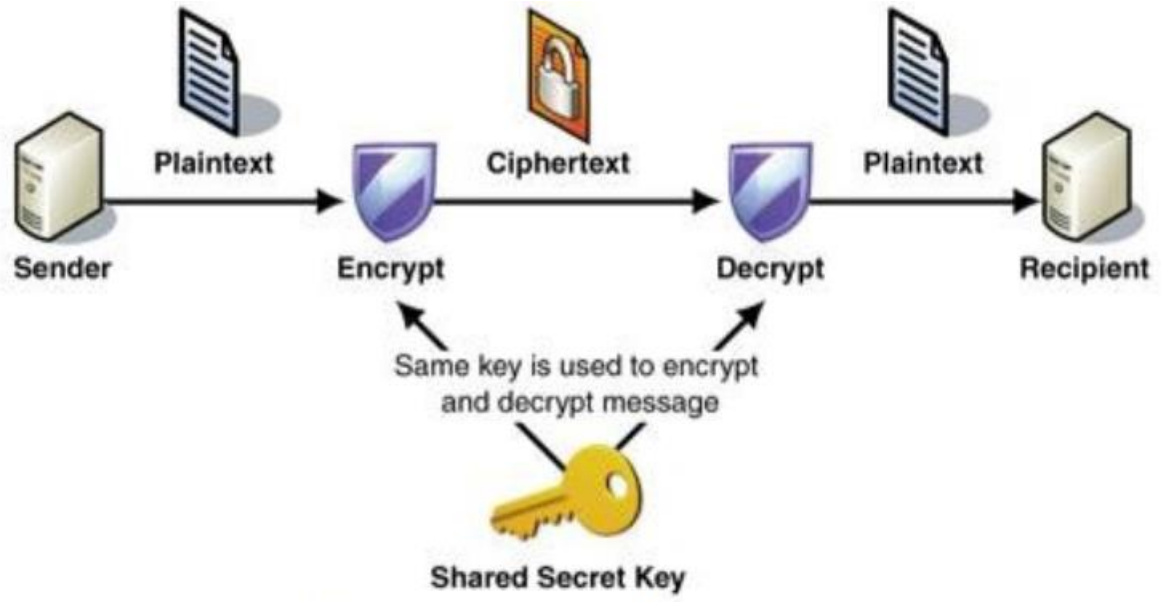
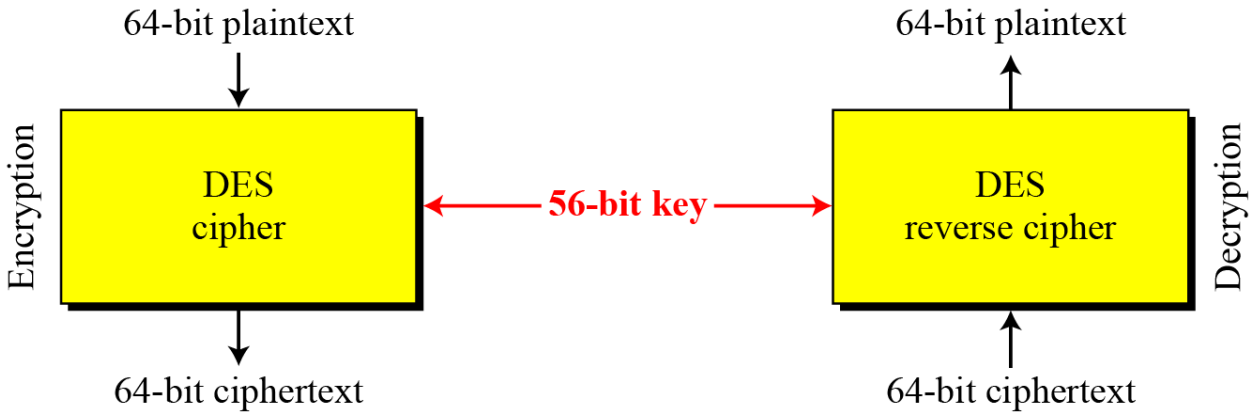


Fig Symmetric Key cryptography

DATA ENCRYPTION STANDARD

- The Data Encryption Standard (DES) is a symmetric-key block cipher published by the National Institute of Standards and Technology (NIST).
- It follows Feistel Cipher Structure.
- The most widely used encryption scheme is based on the Data Encryption Standard (DES) adopted in 1977.
- The algorithm itself is referred to as the Data Encryption Algorithm (DEA).

For DES, data are encrypted in 64-bit blocks using a 56-bit key. The algorithm transforms 64-bit input in a series of steps into a 64-bit output.



DES Encryption

The overall scheme for DES encryption is illustrated in the Figure. There are two inputs to the encryption function: the plaintext to be encrypted and the key. The plaintext must be 64 bits in length and the key is 56 bits in length.

General Depiction of DES Encryption Algorithm

Phase 1

- Looking at the left-hand side of the figure, we can see that the processing of the plaintext proceeds in three phases.
- First, the 64-bit plaintext passes through an initial permutation (IP) that rearranges the bits to produce the permuted input.

Phase 2: This is followed by a phase consisting of 16 rounds of the same function, which involves both permutation and substitution functions.

- The output of the last (sixteenth) round consists of 64 bits that are a function of the input plaintext and the key. The left and right halves of the output are swapped to produce the preoutput.

Phase 3:

Finally, the preoutput is passed through a permutation (IP-1) that is the inverse of the initial permutation function, to produce the 64-bit ciphertext. The right-hand portion of Figure shows the way in which the 56-bit key is used.

Operation on key:

- Initially, the key is passed through a permutation function.
- Then, for each of the 16 rounds, a subkey (Ki) is produced by the combination of a left circular shift and a permutation.
- The permutation function is the same for each round, but a different subkey is produced because of the repeated shifts of the key bits.

Initial Permutation

- The input to a table consists of 64 bits numbered from 1 to 64.
- The 64 entries in the permutation table contain a permutation of the numbers from 1 to 64.
- Each entry in the permutation table indicates the position of a numbered input bit in the output, which also consists of 64 bits

Details of Single Round

The below figure shows the internal structure of a single round. The left and right halves of each 64-bit intermediate value are treated as separate 32-bit quantities, labeled L (left) and R (right). The overall processing at each round can be summarized in the following formulas:

$L_i = R_{i-1}$

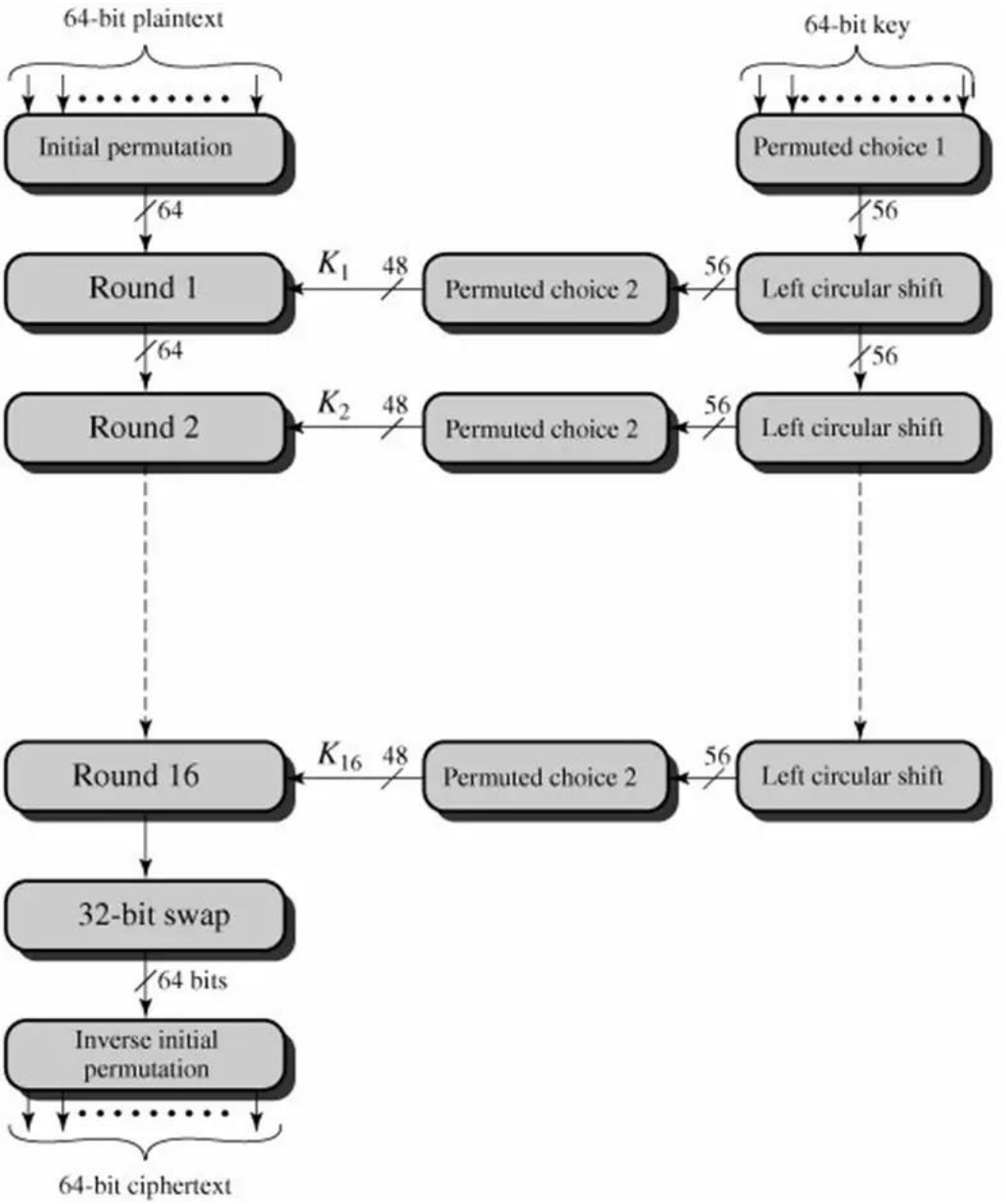
$R_i = L_{i-1} \times F(R_{i-1}, K_i)$

The round key Ki is 48 bits. The R input is 32 bits. This R input is first expanded to 48 bits by using a table that defines a permutation plus an expansion that involves duplication of 16 of the R bits. The resulting 48 bits are XORed with Ki. This 48-bit result passes through a substitution function that produces a 32-bit output, which is then permuted.

Definition of S-Boxes

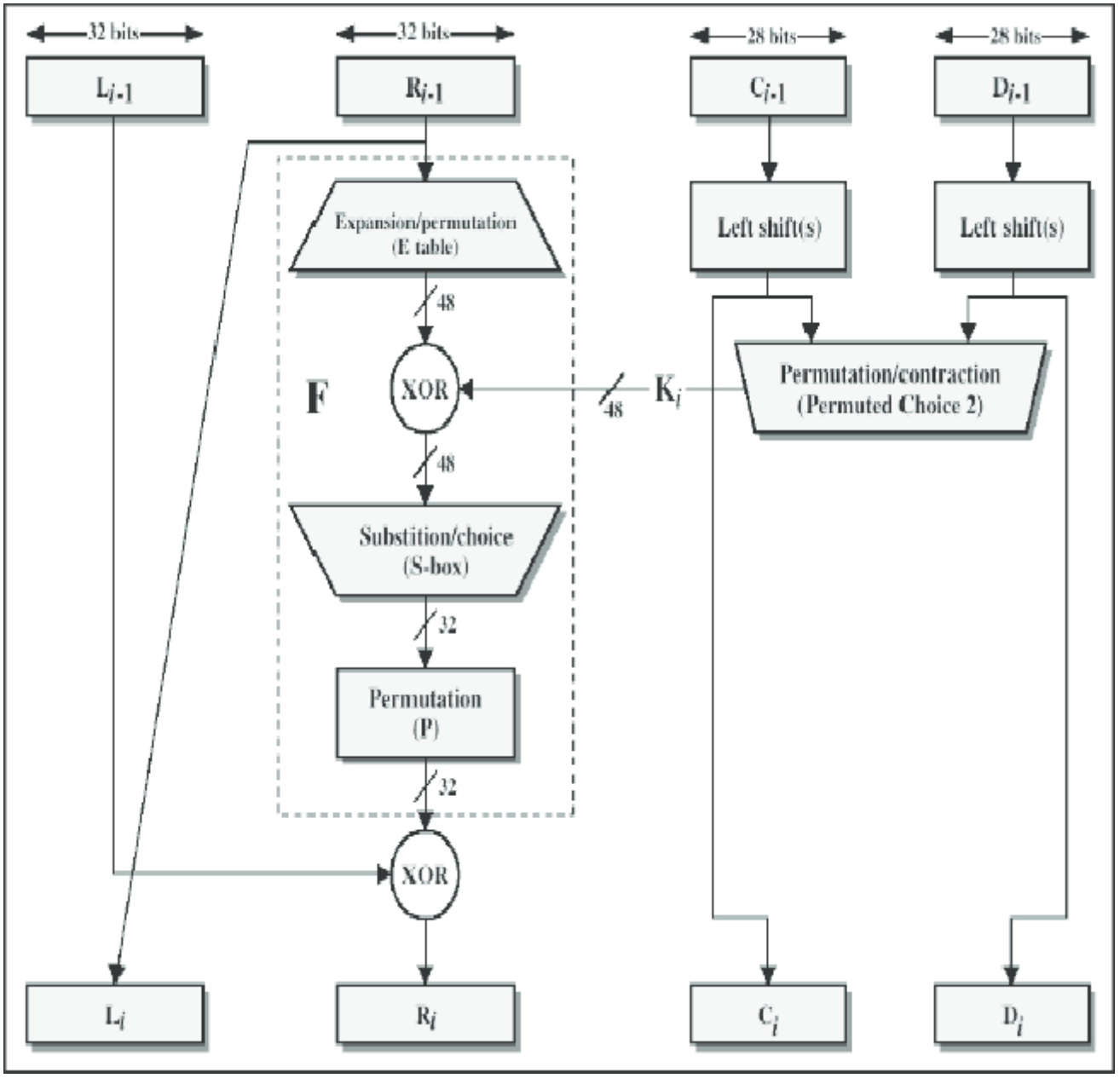
The substitution consists of a set of eight S-boxes, each of which accepts 6 bits as input and produces 4 bits as output. The first and last bits of the input to box Si form a 2-bit binary number to select one of four substitutions defined by the four rows in the table for Si. The middle four bits select one of the sixteen columns

For example, in S1 for input 011001, the row is 01 (row 1) and the column is 1100 (column 12). The value in row 1, column 12 is 9, so the output is 1001.



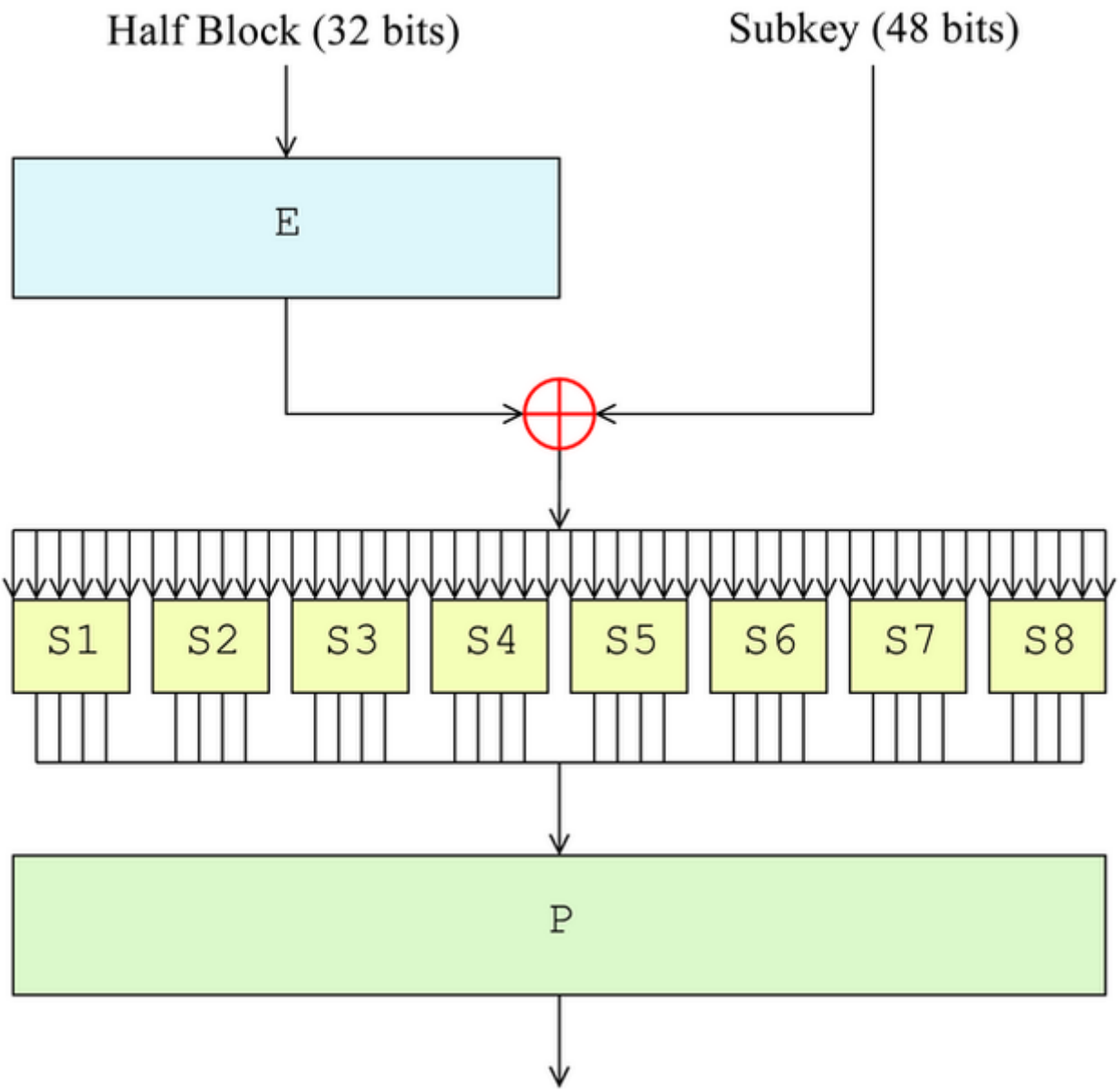
DES Encryption Algorithm

Initial Permutation								*Final Permutation*							
58	50	42	34	26	18	10	02	40	08	48	16	56	24	64	32
60	52	44	36	28	20	12	04	39	07	47	15	55	23	63	31
62	54	46	38	30	22	14	06	38	06	46	14	54	22	62	30
64	56	48	40	32	24	16	08	37	05	45	13	53	21	61	29
57	49	41	33	25	17	09	01	36	04	44	12	52	20	60	28
59	51	43	35	27	19	11	03	35	03	43	11	51	19	59	27
61	53	45	37	29	21	13	05	34	02	42	10	50	18	58	26
63	55	47	39	31	23	15	07	33	01	41	09	49	17	57	25



Key Generation

The 64-bit key is used as input to the algorithm. The bits of the key are numbered from 1 through 64; every eighth bit is ignored. The key is first subjected to a permutation governed by a table labeled Permuted Choice One. The resulting 56-bit key is then treated as two 28-bit quantities, labeled C0 and D0. At each round, Ci-1 and Di-1 are separately subjected to a circular left shift, or rotation, of 1 or 2 bits. These shifted values serve as input to the next round. They also serve as input to Permuted Choice 2, which produces a 48-bit output that serves as input to the function F(Ri-1, Ki)

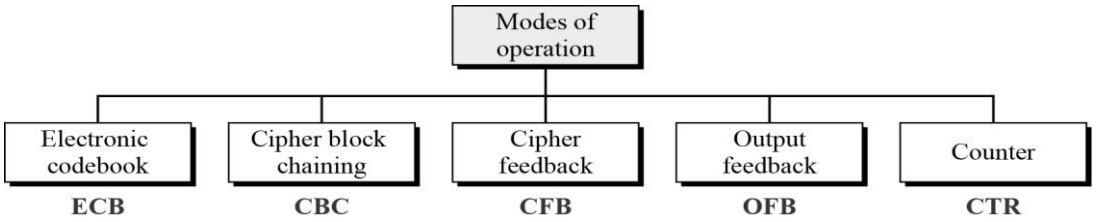


DES Decryption:

As with any Feistel cipher, decryption uses the same algorithm as encryption, except that the application of the sub keys is reversed. Additionally, the initial and final permutations are reversed.

Block cipher mode of operation

- To apply a block cipher in a variety of applications, four "modes of operation" have been defined by NIST.
- A mode of operation is a technique for enhancing the effect of a cryptographic algorithm or adapting the algorithm for an application, such as applying a block cipher to a sequence of data blocks or a data stream



1. Electronic Code Book

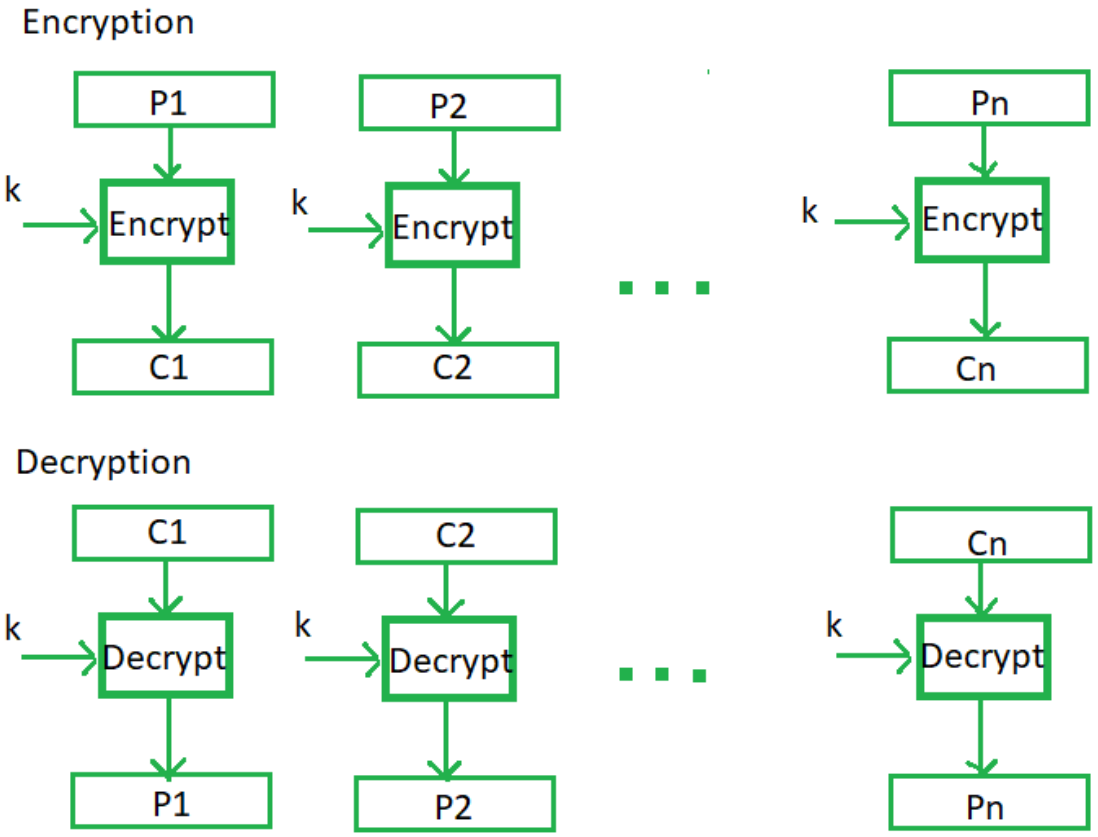
- The simplest mode is the electronic codebook (ECB) mode shown in figure.
- Here plaintext is handled one block at a time and each block of plaintext is encrypted using the same key.
- The term codebook is used because, for a given key, there is a unique cipher text for every b-bit block of plaintext.
- When the message longer than b bits, to break the message into b-bit blocks. For the last block when the no of bits is less than b, padding the last block if necessary.
- Decryption is performed one block at a time, always using the same key.

Uses: The ECB method is ideal for a short amount of data, such as an encryption key.

Disadvantage:

- When b -bit block of plaintext appears more than once in the message, it always produces the same cipher text output.
- For lengthy messages, the ECB mode may not be secure.

- If the message is highly structured, it may be possible for a cryptanalyst to exploit these regularities.

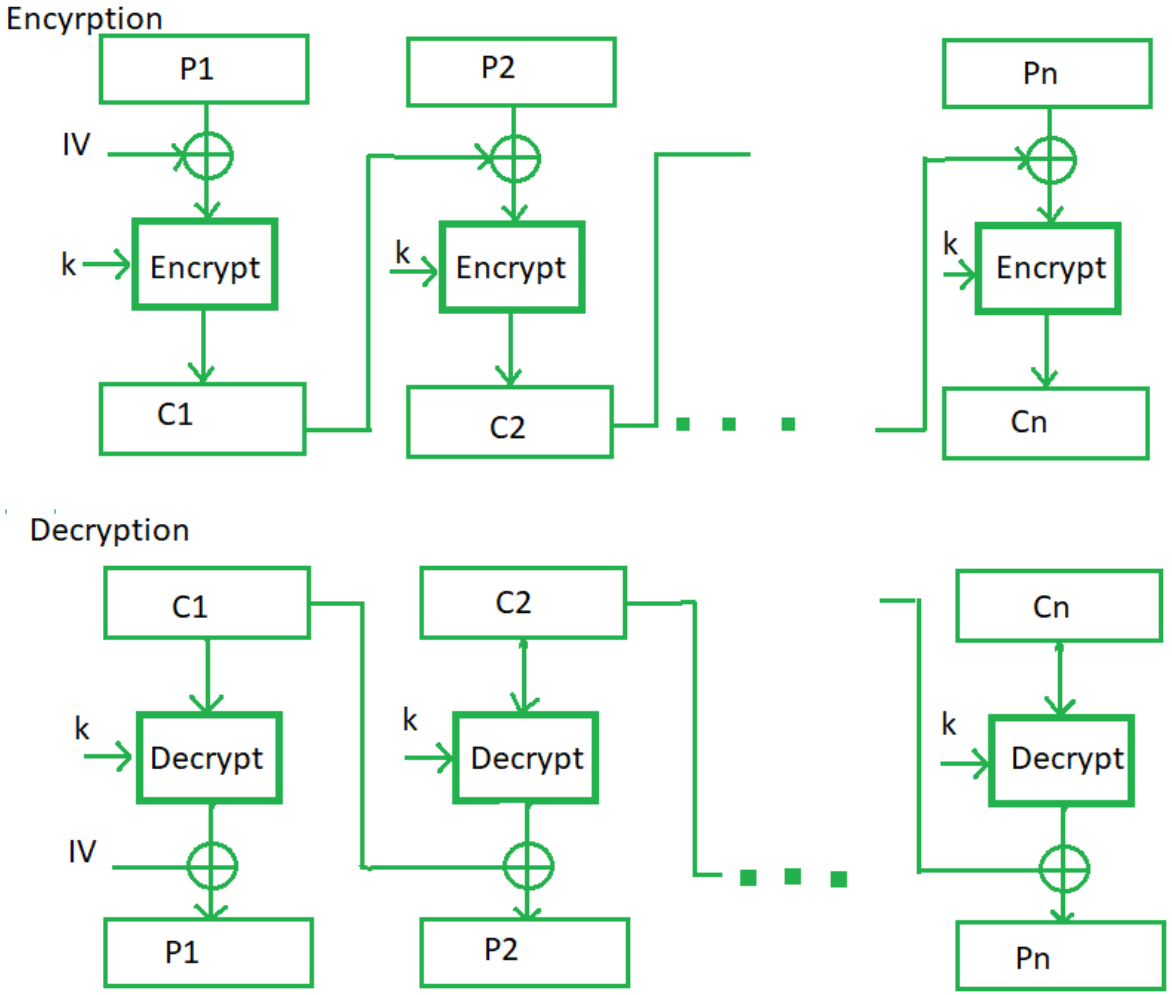


2. Cipher Block Chaining Mode (CBC)

I/P= current plaintext block XOR preceding cipher text block

In this scheme, the input to the encryption algorithm is the XOR of the current plaintext block and the preceding cipher text block; the same key is used for each block. There is no relationship between plaintext block.

- Load the n-bit Initialization Vector (IV) in the top register.
- XOR the n-bit plaintext block with data value in top register.
- Encrypt the result of XOR operation with underlying block cipher with key K.
- For decryption, IV data is XORed with first ciphertext block decrypted.
- The first ciphertext block is also fed into to register replacing IV for decrypting next ciphertext block.



Advantages of CBC

- CBC works well for input greater than b bits.
- CBC is a good authentication mechanism.
- Better resistive nature towards cryptanalsis than ECB.

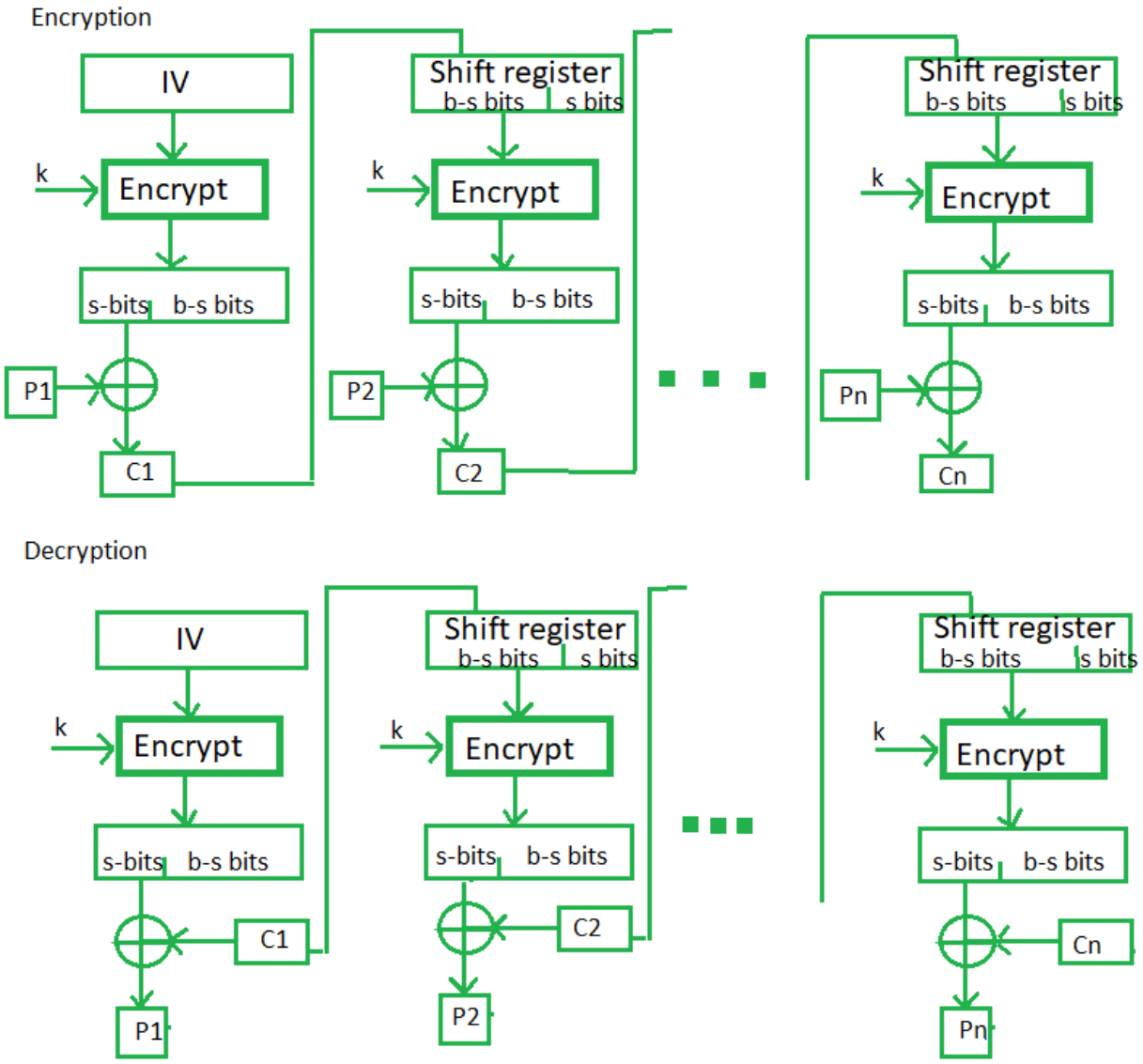
Disadvantages of CBC –

- Parallel encryption is not possible since every encryption requires previous cipher.

3. Cipher feedback mode

- A stream cipher eliminates the need to pad a message to be an integral number of blocks. It also can operate in real time.
- Thus, if a character stream is being transmitted, each character can be encrypted and transmitted immediately using a character-oriented stream cipher.
- The input to the encryption function is a b-bit shift register that is initially set to some initialization vector (IV).

- The leftmost (most significant) s bits of the output of the encryption function are XORed with the first segment of plaintext P1 to produce the first unit of ciphertext C1.
- The contents of the shift register are shifted left by s bits and C1 is placed in the rightmost.
- This process continues until all plaintext units have been encrypted.



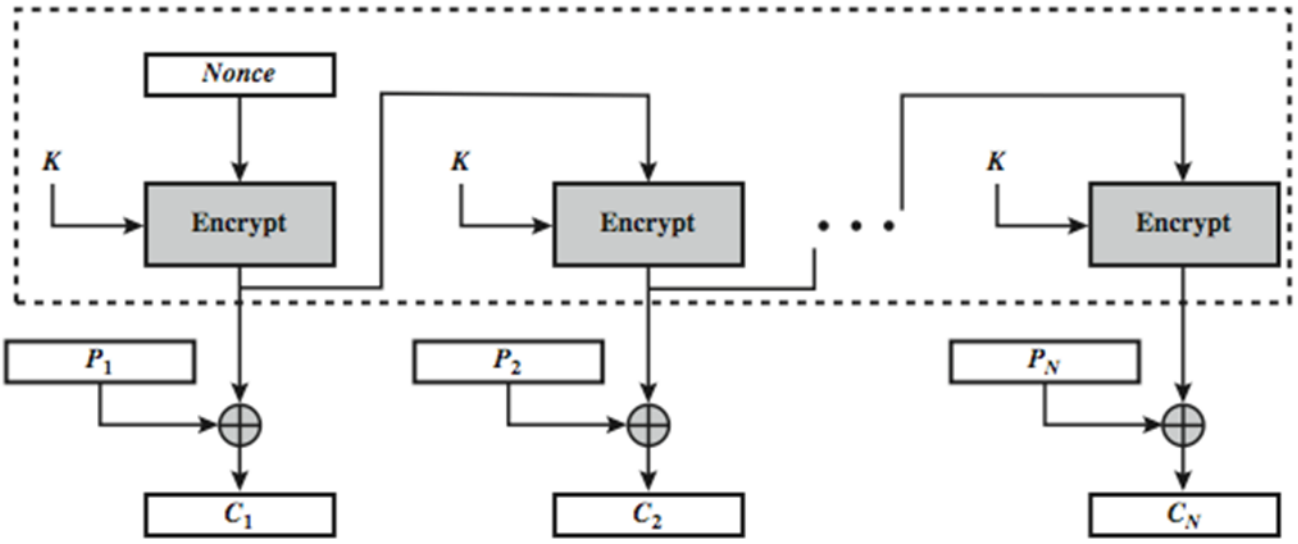
$$C_i = E_K(C_{i-1}) \oplus P_i,$$
$$P_i = E_K(C_{i-1}) \oplus C_i,$$
$$C_0 = \mathrm{IV}.$$

For decryption, the same scheme is used, except that the received ciphertext unit is XORed with the output of the encryption function to produce the plaintext unit.

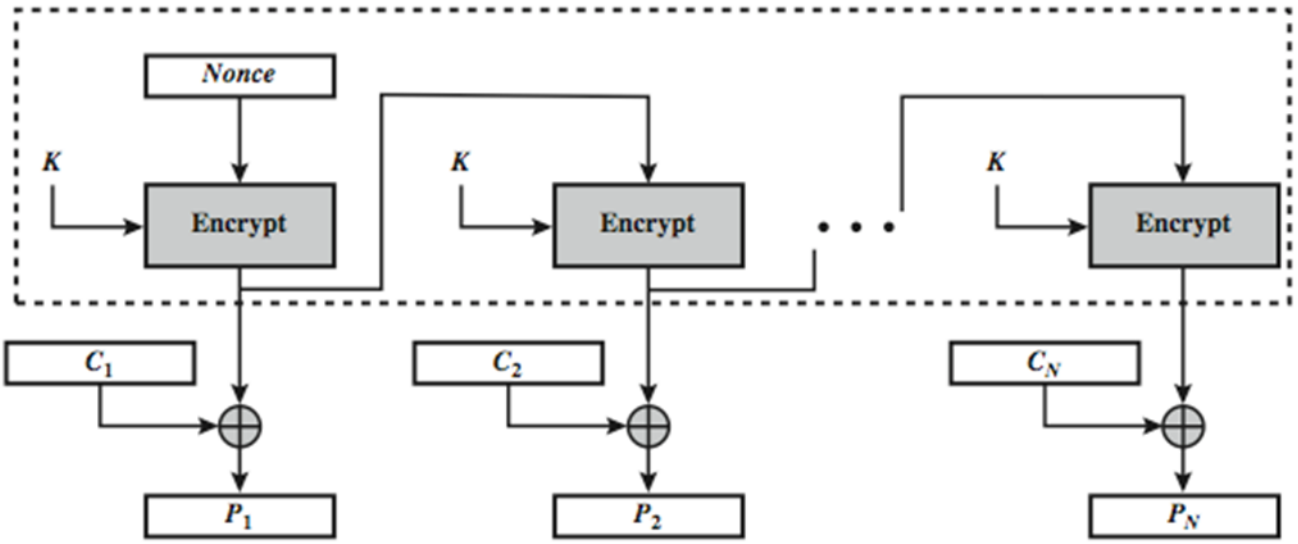
Note that it is the encryption function that is used, not the decryption function. This is easily explained. Let Ss(X) be defined as the most significant s bits of X.

4. Output feedback mode

The output feedback (OFB) mode is similar in structure to that of CFB. The output of the encryption function that is fed back to the shift register in OFB, whereas in CFB the cipher text unit is fed back to the shift register.



(a) Encryption



(b) Decryption

Advantage:

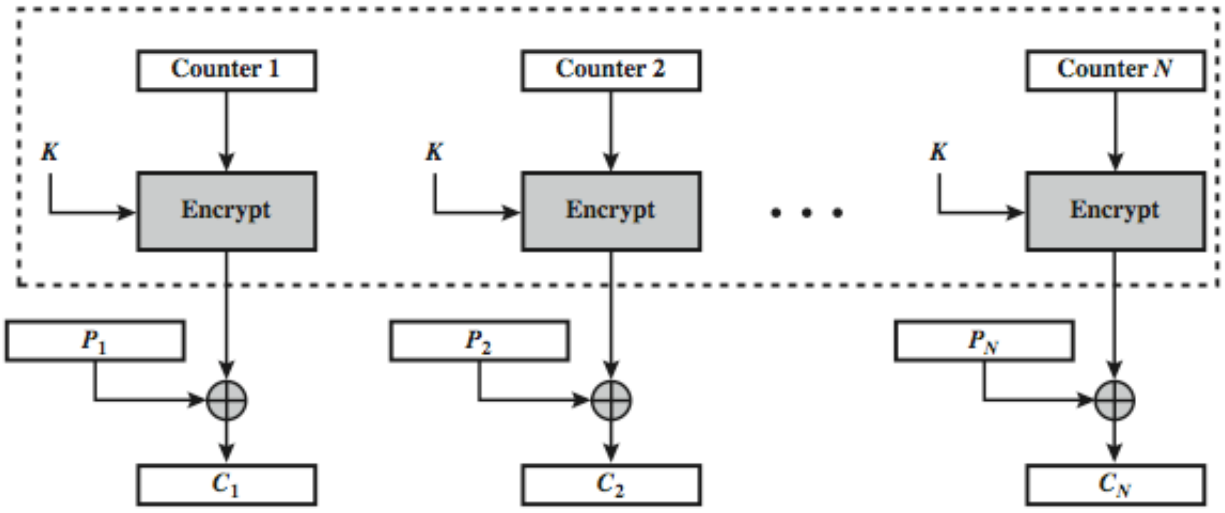
One advantage of the OFB method is that bit errors in transmission do not propagate.

Disadvantage:

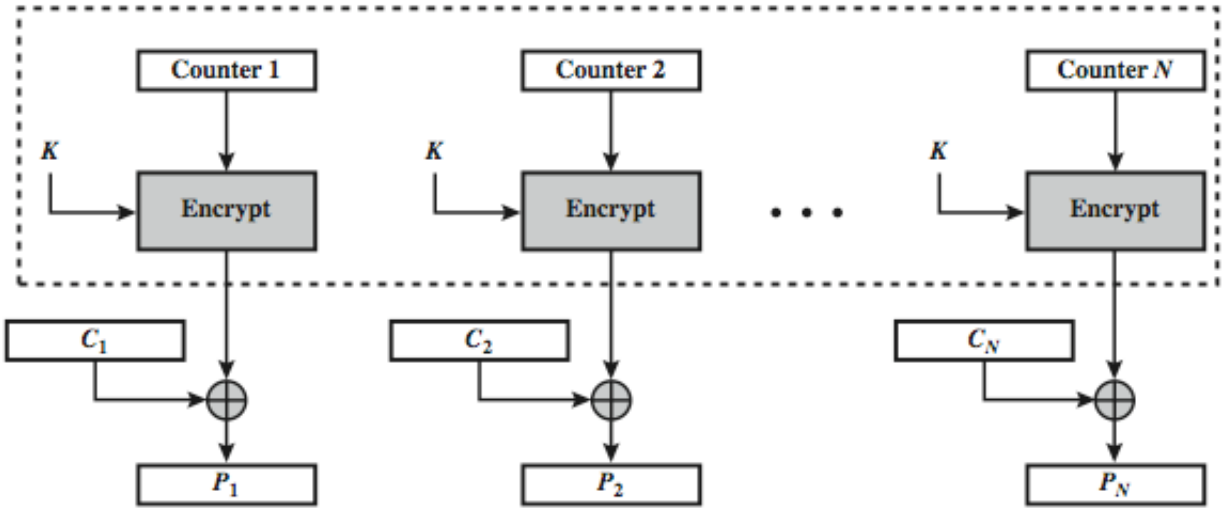
The disadvantage of OFB is that it is more vulnerable to a message stream modification attack than is CFB.

5. Counter Mode – (CTR)

A counter, equal to the plaintext block size is used. The counter is initialized to some value and then incremented by 1 for each subsequent block. For encryption, the counter is encrypted and then XORed with the plaintext block to produce the cipher text block; there is no chaining. For decryption, the same sequence of counter values is used, with each encrypted counter XORed with a cipher text block to recover the corresponding plaintext block.



(a) Encryption



(b) Decryption

Advantages:

- Hardware efficiency
- Software efficiency
- Preprocessing
- Random access

Advanced Encryption Standard (AES)

- AES is a non-Feistel cipher that encrypts and decrypts a data block of 128 bits. It uses 10, 12, or 14 rounds.
- The key size, which can be 128, 192, or 256 bits, depends on the number of rounds.

Key size (words/bytes/bits)	4/16/128	6/24/192	8/32/256
Plaintext block size (words/bytes/bits)	4/16/128	4/16/128	4/16/128
Number of rounds	10	12	14
Round key size (words/bytes/bits)	4/16/128	4/16/128	4/16/128
Expanded key size (words/bytes)	44/176	52/208	60/240

AES – Overall structure

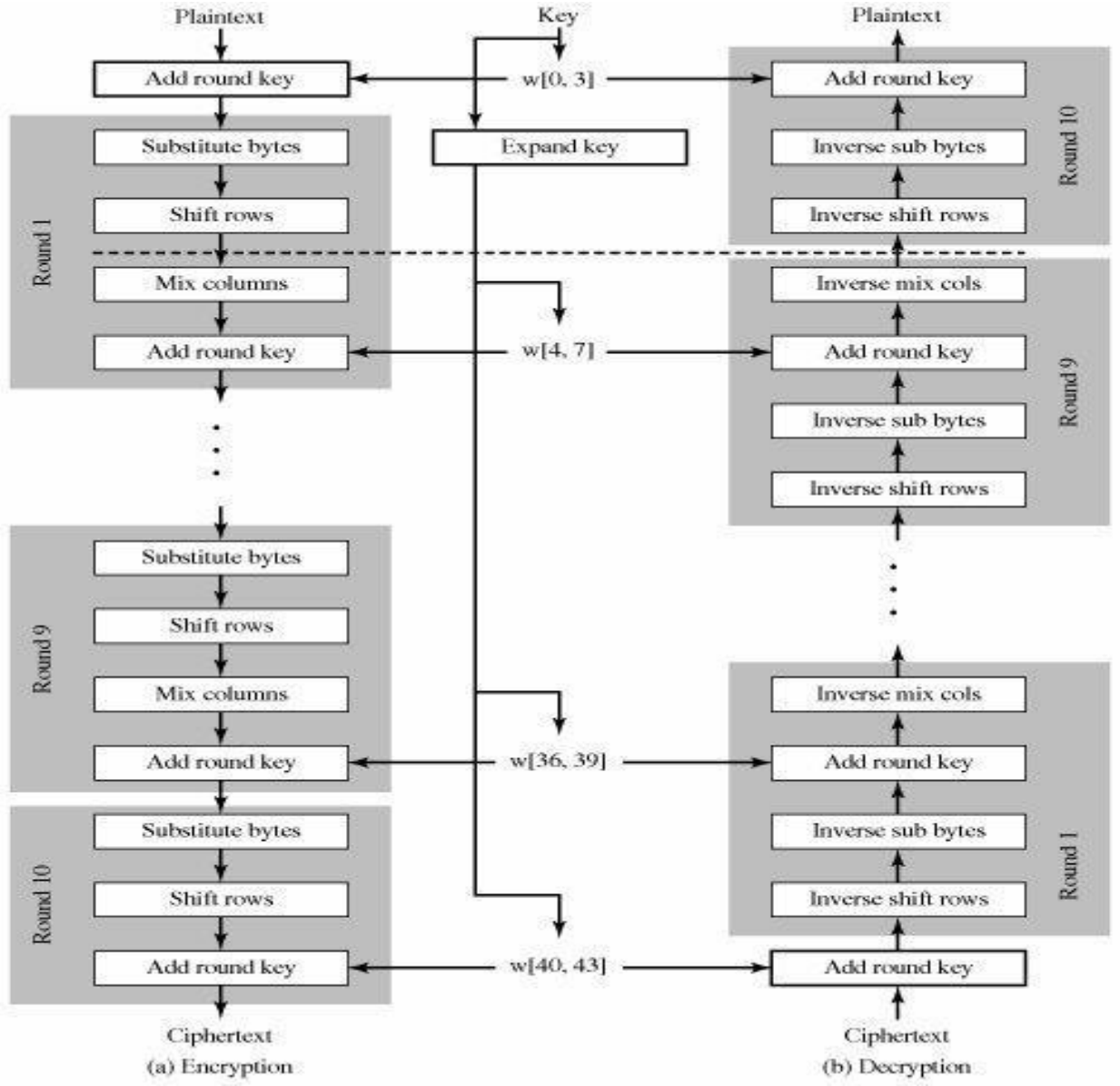
- The input to the encryption and decryption algorithms is a single 128-bit block. This block is copied into the State array, which is modified at each stage of encryption or decryption. After the final stage, State is copied to an output matrix.
- The 128-bit key is depicted as a square matrix of bytes. This key is then expanded into an array of key schedule words; each word is four bytes and the total key schedule is 44 words for the 128-bit key.

AES – Data structures: Input , state array and output Key , expanded key

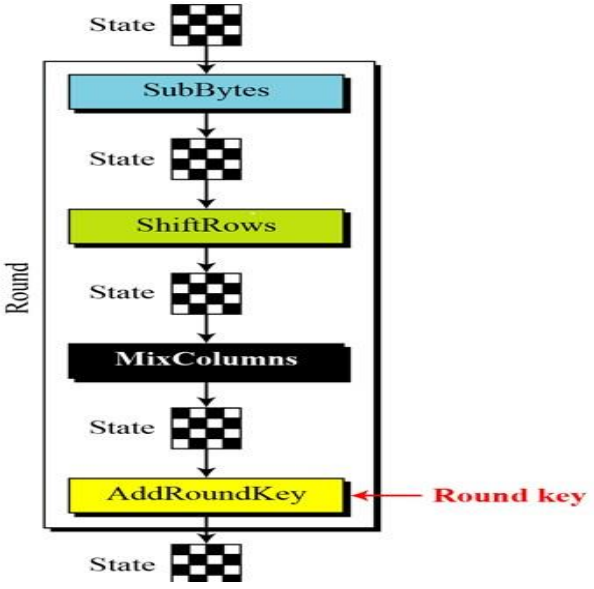
Four different stages are used, one of permutation and three of substitution:

- Substitute bytes: Uses an S-box to perform a byte-by-byte substitution of the block
- ShiftRows: A simple permutation
- MixColumns: A substitution that makes use of arithmetic over GF(2^8)
- AddRoundKey: A simple bitwise XOR of the current block with a portion of the expanded key

- For both encryption and decryption, the cipher begins with an AddRoundKey stage, followed by nine rounds that each includes all four stages, followed by a tenth round of three stages.



AES Round Structure



Substitute byte formation :

- The forward substitute byte transformation, called SubBytes, is a simple table lookup.
- AES defines a 16 x 16 matrix of byte values, called an S-box (Table 5.4a), that contains a permutation of all possible 256 8-bit values.
- Each individual byte of State is mapped into a new byte in the following way: The leftmost 4 bits of the byte are used as a row value and the rightmost 4 bits are used as a column value. These row and column values serve as indexes into the S-box to select a unique 8-bit output value.

ShiftRows Transformation

- Forward and Inverse Transformations
- The forward shift row transformation, called ShiftRows, is depicted in Figure . The first row of
- State is not altered. For the second row, a 1-byte circular left shift is performed. For the third row, a 2- byte circular left shift is performed. For the fourth row, a 3-byte circular left shift is performed. The following is an example of ShiftRows

87	F2	4D	97
EC	6E	4C	90
4A	C3	46	E7
8C	D8	95	A6

→

87	F2	4D	97
6E	4C	90	EC
46	E7	4A	C3
A6	8C	D8	95

Mix columns transformation :

Forward and Inverse Transformations The forward mix column transformation, called MixColumns, operates on each column individually. Each byte of a column is mapped into a new value that is a function of all four bytes in that column.

Add Round key

Forward and Inverse Transformations

In the forward add round key transformation, called AddRoundKey, the 128 bits of State are bitwise XORed with the 128 bits of the round key.

PUBLIC KEY CRYPTOGRAPHY (Asymmetric Cryptography)

Difficulties in Symmetric encryption

According to Diffie-Hellman

i) Key distribution is a serious issue.

ii) Symmetric encryption is not applicable for Digital signatures

Public key encryption scheme :

Asymmetric algorithms rely on one key for encryption and a different but related key for decryption.

A public-key encryption scheme has six ingredients

- Plaintext: This is the readable message or data given as input to the algorithm.
- Encryption algorithm: The encryption algorithm performs various transformations on the plaintext.
- Public and private keys: This is a pair of keys that have been selected so that if one is used for encryption, the other is used for decryption.
- Cipher text: This is the scrambled message produced as output. It depends on the plaintext and the key. For a given message, two different keys will produce two different cipher texts.
- Decryption algorithm: This algorithm accepts the cipher text and the matching key and produces the original plaintext.

Differences between conventional and asymmetric

Conventional / symmetric / private	**Asymmetric / public key encryption**
The same algorithm with the same key is used for encryption and decryption.	One algorithm is used for encryption and decryption with a pair of keys, one for encryption and one for decryption.
The sender and receiver must share the algorithm and the key.	The sender and receiver must each have one of the matched pair of keys

Applications for Public-Key Cryptosystems

Three categories of applications are

Encryption/decryption: The sender encrypts a message with the recipient's public key.

Digital signature: The sender "signs" a message with its private key. Signing is achieved by a cryptographic algorithm applied to the message or to a small block of data that is a function of the message.

Key exchange: Two sides cooperate to exchange a session key. Several different approaches are possible, involving the private key(s) of one or both parties.

RSA – Algorithm

- It was developed by Rivest, Shamir and Adleman. This algorithm makes use of an expression with exponentials. Plaintext is encrypted in blocks, with each block having a binary value less than some number n.
- The RSA Algorithm: It is a public key cryptography algorithm. RSA can be used for key exchange, digital signatures and the encryption of small blocks of data.
- The RSA scheme is a cipher in which the plaintext and cipher text are integers between 0 and n - 1 for some n. A typical size for n is 1024 bits, or 309 decimal digits. That is, n is less than 2^{1024}

Encryption and decryption are of the following form, for some plaintext block M and cipher text block C:

$$C = M^e \bmod n$$

$$M = C^d \bmod n$$

Both the sender and receiver know the value of n. the sender knows the value of e and only the receiver knows the value of d. thus, this is a public key encryption algorithm with a public key of KU = {e, n) and a private key of KR = {d, n}.

For this algorithm to be satisfactory for public key encryption, the following requirements must be met:

1. It is possible to find values of e, d, n such that Med = M mod n for all M < n.
2. It is relatively easy to calculate Me and Cd for all values of M < n.
3. It is infeasible to determine d given e and n.

- It is the best known & widely used public-key scheme and based on exponentiation in a finite (Galois) field over integers modulo a prime.
- Security due to cost of factoring large numbers

- Plaintext and cipher text are integers between 0 and n – 1 for some n. (eg . 1024 bits) Ingredients of RSA Algorithm

The ingredients are the following:

p, q, two prime numbers	(private, chosen)
$n = pq$	(public, calculated)
e, with gcd(ϕ (n),e) = 1; $1 < e < \phi(n)$	(public, chosen)
$d \equiv e^{-1} (\mathrm{mod}\ \phi(n))$	(private, calculated)

RSA Key Setup

- This key setup is done once (rarely) when a user establishes (or replaces) their public key.
- Each user generates a public/private key pair by:
- Selecting two large primes at random - p, q
- Computing their system modulus N=p .q
    - ø (N)= (p-1) (q-1)
- Selecting at random the encryption key e where 1< e < ø(N), gcd (e ,ø (N)) =1
- Solve following equation to find decryption key d
    - e. d=1 mod ø(N) and 0 ≤ d ≤ N

$$ed = 1 \bmod \emptyset(n)$$

OR

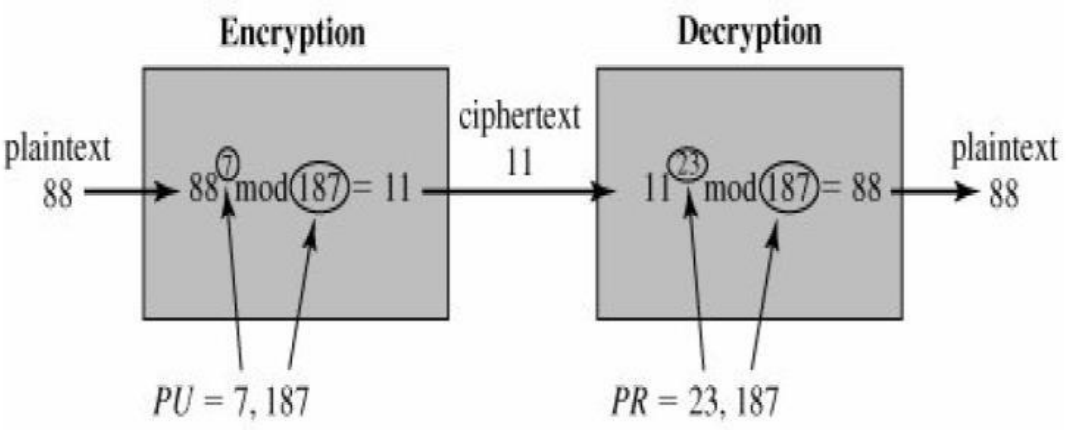
$$d = \frac{1 + k\,\emptyset(n)}{e}$$

- Publish their public encryption key:
    - KU = {e, N}
- Keep secret private decryption key:
    - KR = {d, p, q} RSA Use

- To encrypt a message M the sender: obtains **public key** of recipient KU={e ,N} computes C=Me mod N
- To decrypt the ciphertext C the owner:
    - uses their private key KR={d,p,q}
    - computes: **$M=C^d$ mod N**
- note that the message M must be smaller than the modulus N (block if needed)

RSA Example

1. Select primes: p=17 & q=11
2. Compute $n = pq$ =17×11=187
3. Compute ø(n)=(p–1)(q-1)=16×10=160
4. Select e *:* gcd(e,160)=1; choose e=7
5. Determine d*:* de=1 mod 160 and d < 160 Value is d=23 since 23×7=161
6. Publish public key KU={7,187}
7. Keep secret private key KR={23,17,11}
8. Given message M = 88 (88<187)
9. Encryption: C = 88^7 mod 187 = 11



88^7 mod 187 = [(88^4 mod 187) x (88^2 mod 187) x (88^1 mod 187)] mod 187

88^1 mod 187 = 8888^2 mod 187 = 7744 mod 187 = 77

$88^4 \bmod 187 = 59969536 \bmod 187 = 132$

$88^7 \bmod 187 = (88 \times 77 \times 132) \bmod 187 = 894432 \bmod 187 = 11$

10. Decryption: $M = 11^{23} \bmod 187 = 88$

Key Generation Alice	
Select p, q	p and q both prime, $p \neq q$
Calculate $n = p \times q$	
Calcuate $\phi(n) = (p-1)(q-1)$	
Select integer e	$\gcd(\phi(n), e) = 1; 1 < e < \phi(n)$
Calculate d	$d \equiv e^{-1} \pmod{\phi(n)}$
Public key	$PU = \{e, n\}$
Private key	$PR = \{d, n\}$

Encryption by Bob with Alice's Public Key	
Plaintext:	$M < n$
Ciphertext:	$C = M^e \bmod n$

Decryption by Alice with Alice's Public Key	
Ciphertext:	C
Plaintext:	$M = C^d \bmod n$

Figure 9.5 The RSA Algorithm

The security of RSA:

Five possible approaches to attacking the RSA algorithm are

■ Brute force: This involves trying all possible private keys.

■ Mathematical attacks: There are several approaches, all equivalent in effort to factoring the product of two primes.

■ Timing attacks: These depend on the running time of the decryption algorithm.

■ Hardware fault-based attack: This involves inducing hardware faults in the processor that is generating digital signatures.

■ Chosen ciphertext attacks: This type of attack exploits properties of the RSA algorithm.

Diffie – Hellman Key Exchange

- The purpose of the algorithm is to enable two users to securely exchange a key that can then be used for subsequent encryption of messages.
- The algorithm itself is limited to the exchange of secret values.
- The Diffie-Hellman algorithm depends for its effectiveness on the difficulty of computing discrete logarithms.
- The Diffie-Hellman algorithm uses exponentiation in a finite (Galois) field (modulo a prime or a polynomial)

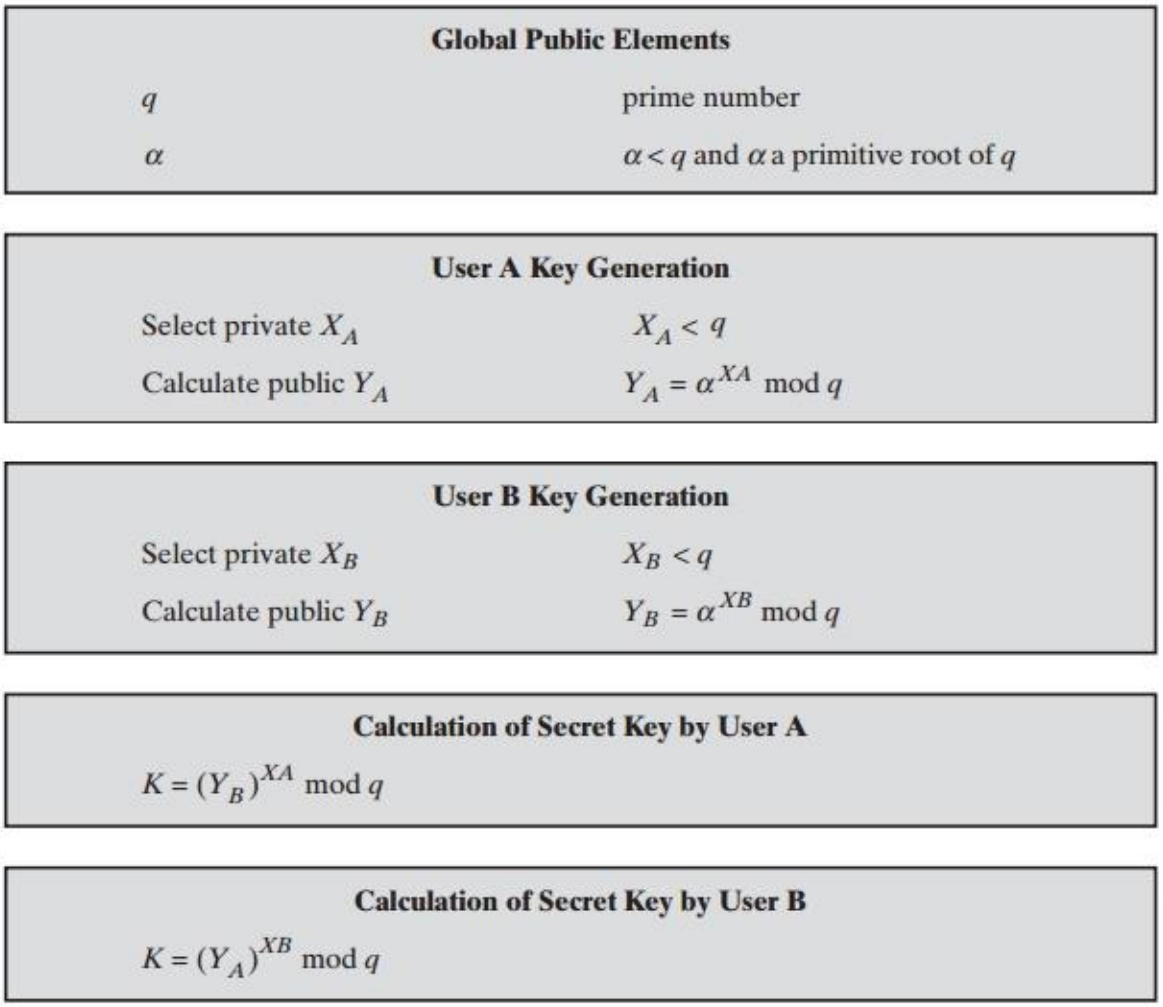


Figure 10.1 The Diffie-Hellman Key Exchange Algorithm

The result is that the two sides have exchanged a secret value.

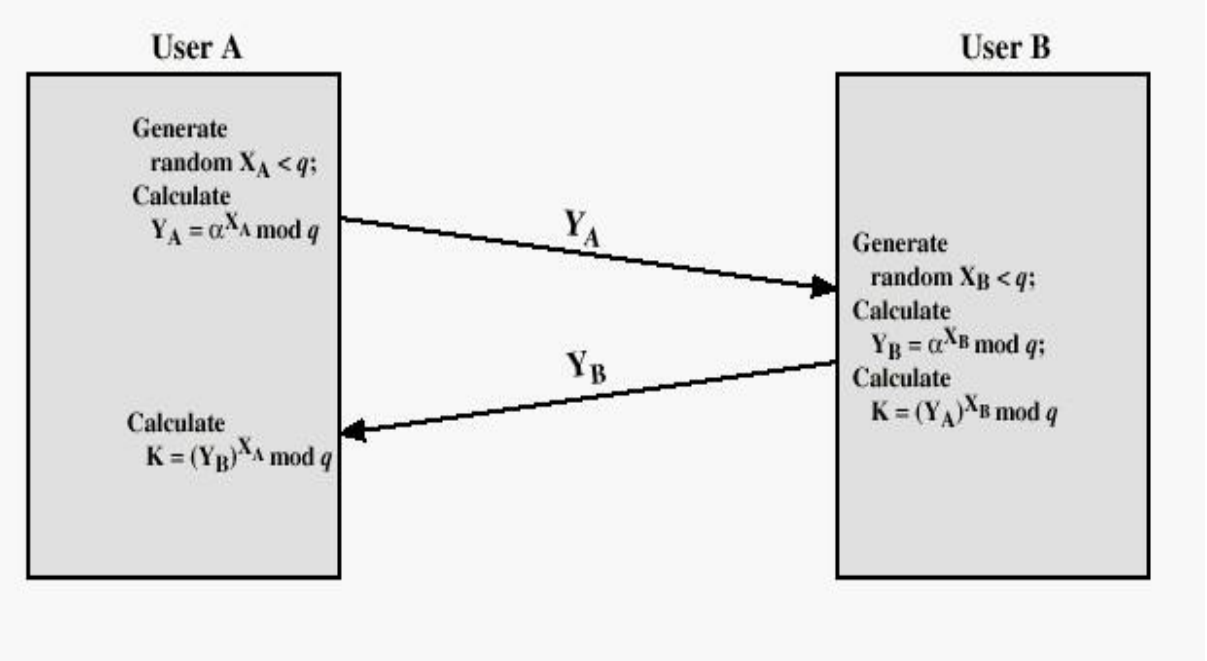
Ex : ὰ = 3 XA = 97 and XB = 233

A computes YA = 397 mod 353 = 40.

B computes YB = 3233 mod 353 = 248.

After they exchange public keys, each can compute the common secret key: A computes K = (YB)XA mod 353 = 24897 mod 353 =160.

B computes K = (YA)XB mod 353 = 40233 mod 353 = 160.



Man-in-the-Middle Attack

Suppose Alice and Bob wish to exchange keys, and Darth is the adversary. The attack proceeds as follows:

1. Alice sends an encrypted message M: E(K2, M).
2. Darth intercepts the encrypted message and decrypts it, to recover M.
3. Darth sends Bob E(K1, M) or E(K1, M'), where M' is any message. In the first case, Darth simply wants to eavesdrop on the communication without altering it. In the second case, Darth wants to modify the message going to Bob.

This vulnerability can be overcome with the use of digital signatures and public-key Certificates.

1.3 Hash functions

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

Hash function accepts a variable size message M as input and produces a fixed size output, referred to as hash code H(M). **The hash code does not use any key.** The hash code is also referred to as message digest or hash value.

The message plus concatenated hash code is encrypted using symmetric encryption. The encryption is applied to the entire message plus hash code,

confidentiality is also provided.

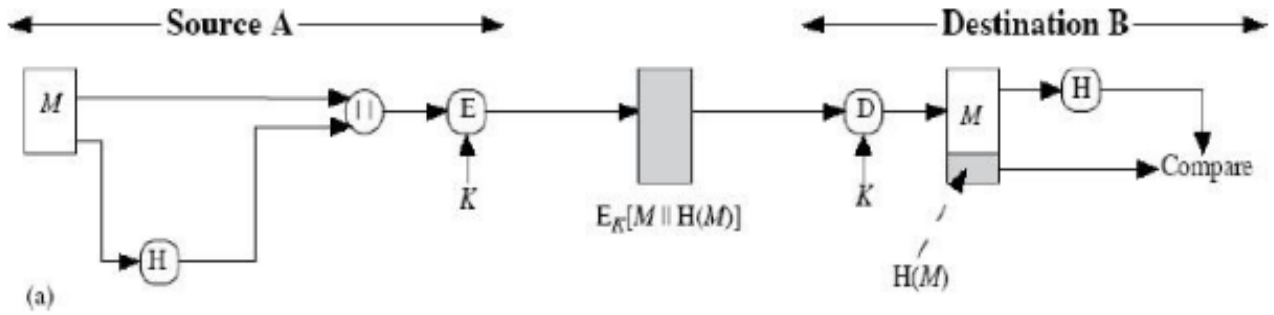


Figure (a) Hash Function

Requirements for a Hash Function

1. H can be applied to a block of data of any size.
2. H produces a fixed-length output.
3. H(x) is relatively easy to compute for any given x, making both hardware and software implementations practical.
4. For any given value h, it is computationally infeasible to find x such that H(x) = h. This is sometimes referred to in the literature as the one-way property.
5. For any given block x, it is computationally infeasible to find y x such that H(y) = H(x). This is sometimes referred to as weak collision resistance.
6. It is computationally infeasible to find any pair (x, y) such that H(x) = H(y). This is sometimes referred to as strong collision resistance.

SHA (Secure Hash Algorithm)

The most widely used hash function has been the Secure Hash Algorithm (SHA). SHA was developed by the National Institute of Standards and Technology. SHA is based on the hash function MD4, and its design closely models MD4. 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.

SHA-512 Logic

Message Digest Generation Using SHA-512

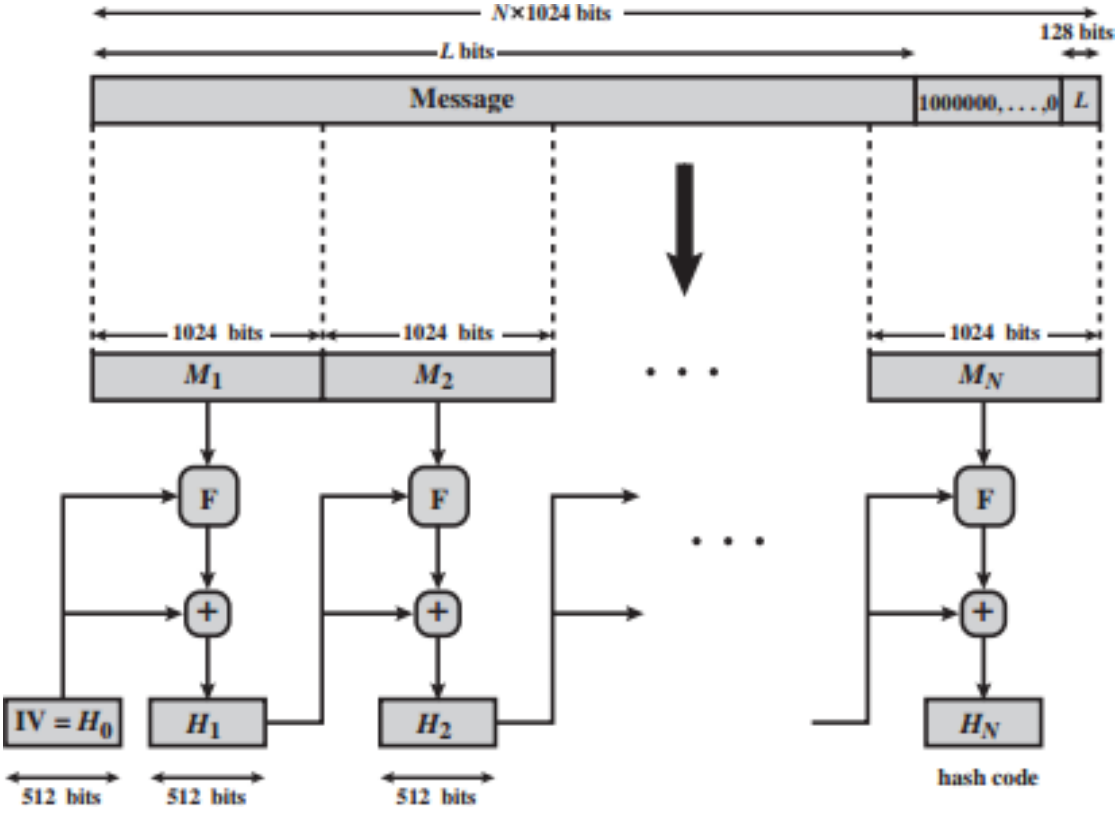
The algorithm takes as input a message with a maximum length of less than bits 2^{128} bits and produces as output a 512-bit message digest. The input is processed in 1024-bit blocks. The processing consists of the following steps.

Step 1 Append padding bits. The message is padded so that its length is congruent to 896 modulo 1024 of 1 to 1024. The padding consists of a single 1 bit followed by the necessary number of 0 bits.

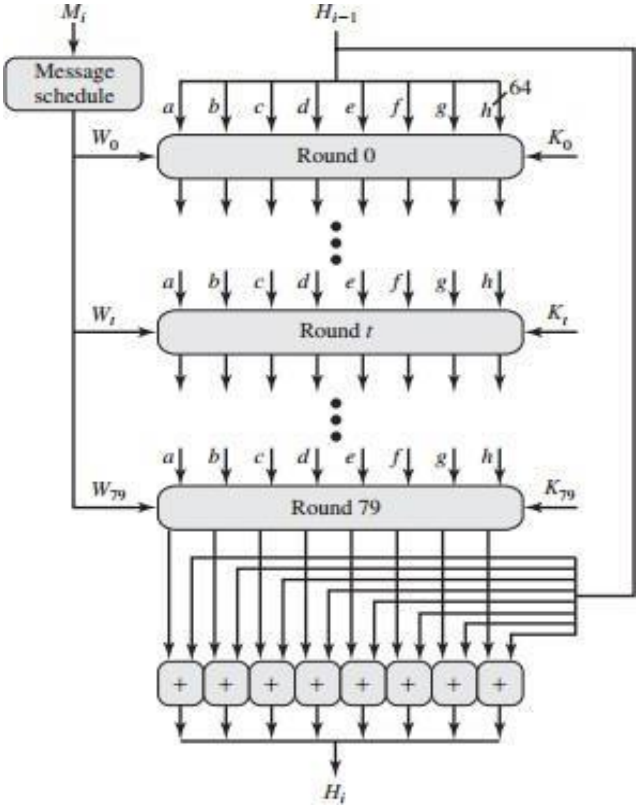
Step 2 Append length. A block of 128 bits is appended to the message. This block is treated as an unsigned 128-bit integer (most significant byte first) and contains the length of the original message (before the padding).
The outcome of the first two steps yields a message that is an integer multiple of 1024 bits in length. The expanded message is represented as the sequence of 1024-bit blocks *M*1, *M*2, Á , *MN* , so that the total length of the expanded message is *N* * 1024 bits

Step 3 Initialize hash buffer. A 512-bit buffer is used to hold intermediate and final results of the hash function. The buffer can be represented as eight 64-bit registers (a, b, c, d, e, f, g, h).These registers are initialized to the following 64-bit integers (hexadecimal values) These values are stored in **big-endian** format, which is the most significant byte of a word in the low-address (leftmost) byte position.

Step 4 Process message in 1024-bit (128-word) blocks. The heart of the algorithm is a module that consists of 80 rounds; Each round takes as input the 512-bit buffer value, ABCDEFGH, and updates the contents of the buffer. At input to the first round, the buffer has the value of the intermediate hash value, *Hi*-1).
Step 5 Output. After all 1024-bit blocks have been processed, the output from the N th stage is the 512- bit message digest.

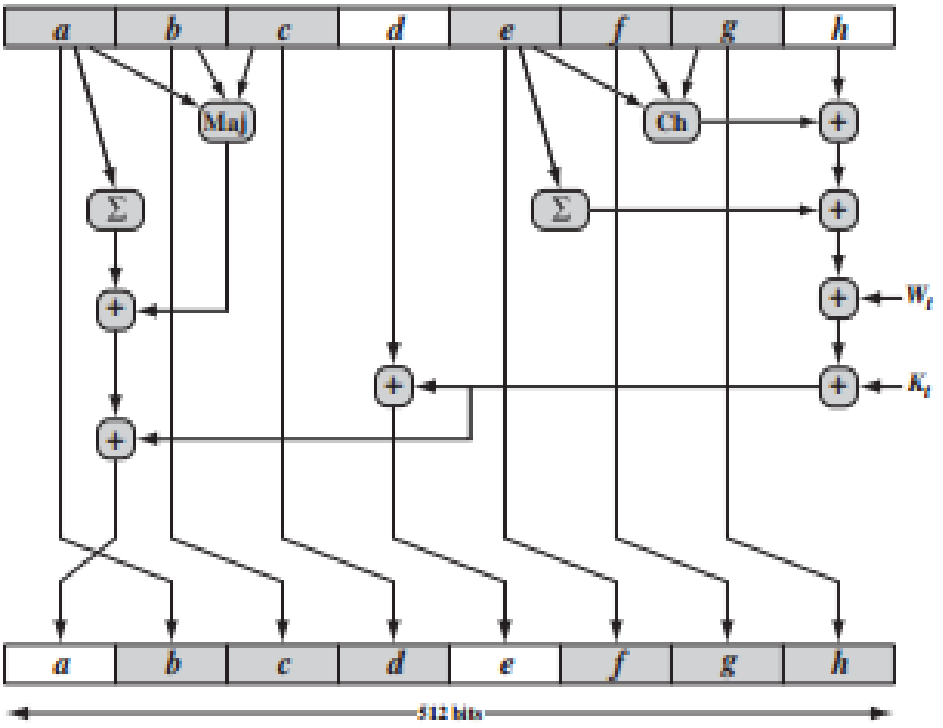


Message Digest Generation Using SHA-512



SHA-512 Processing of a Single 1024-Bit Block

1.



Compression Function

1.4 Authentication

Authentication is the process of verifying the identity of a user or information.

User authentication is the process of verifying the identity of a user when that user logs in to a computer system

In most computer security contexts, user authentication is the fundamental building block and the primary line of defense. User authentication is the basis for most types of access control and for user accountability. RFC 2828 defines user authentication.

REMOTE USER-AUTHENTICATION PRINCIPLES

The process of verifying an identity claimed by or for a system entity.

An authentication process consists of two steps:

Identification step: Presenting an identifier to the security system.

Verification step: Presenting or generating authentication information that corroborates the binding between the entity and the identifier.

There are four general means of authenticating a user's identity, which can be used alone or in combination:

Something the individual knows: Examples include a password, a personal

identification number (PIN), or answers to a prearranged set of questions.

Something the individual possesses: Examples include cryptographic keys, electronic keycards, smart cards, and physical keys. This type of authenticator is referred to as a token.

Something the individual is (static biometrics): Examples include recognition by fingerprint, retina, and face.

Something the individual does (dynamic biometrics): Examples include recognition by voice pattern, handwriting characteristics, and typing rhythm.

Message Authentication

Message authentication is a procedure to verify that received messages come from the assumed source and have not been altered.

MESSAGE AUTHENTICATION FUNCTION

Message Authentication Function can be grouped into three classes.

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

1.5 Digital Signature

The most important development from the work on public-key cryptography is the digital signature. Message authentication protects two parties who exchange messages from any third party. However, it does not protect the two parties against each other. A digital signature is analogous to the handwritten signature.

It must have the following properties:

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

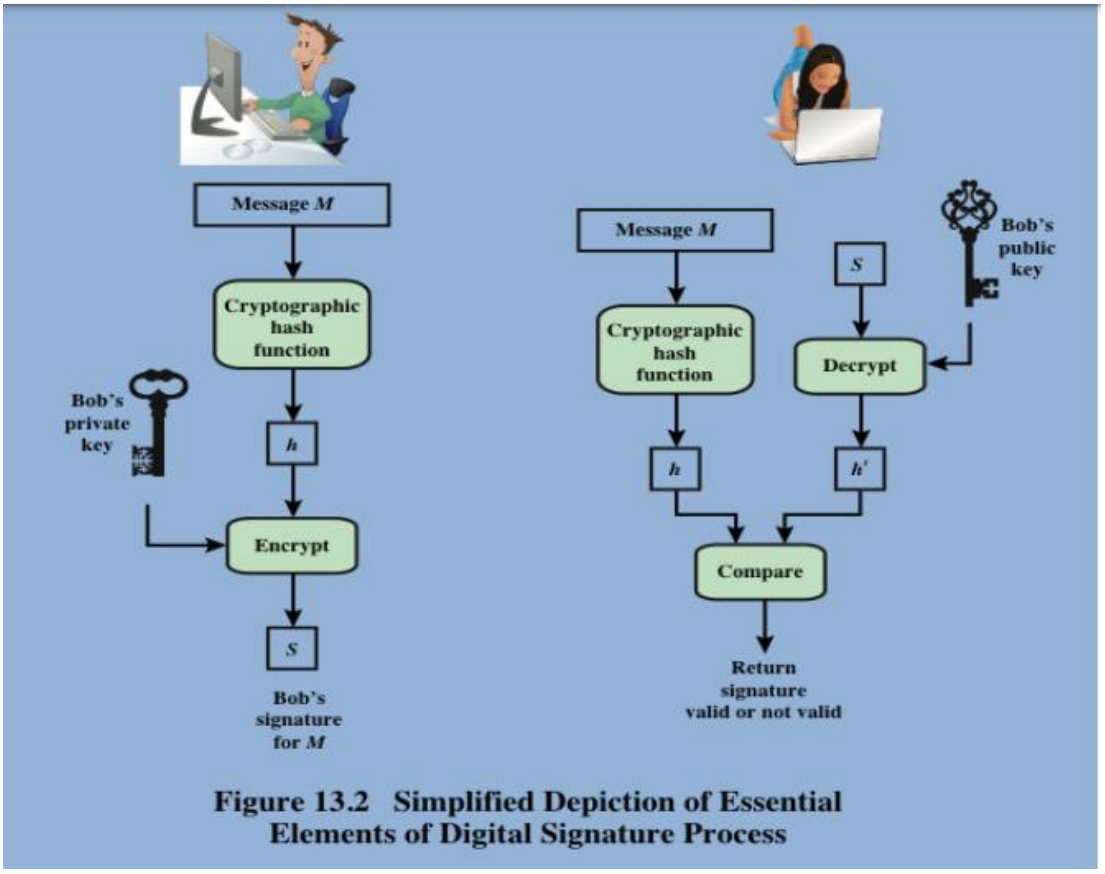


Figure 13.2 Simplified Depiction of Essential Elements of Digital Signature Process

Requirements for a digital signature:

- 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.

Types: Direct and arbitrated

Direct Digital Signatures

- involve only sender & receiver
- assumed receiver has sender's public-key
- digital signature made by sender signing entire message or hash with private-key
- security depends on sender's private-key
- Drawback: Forgery

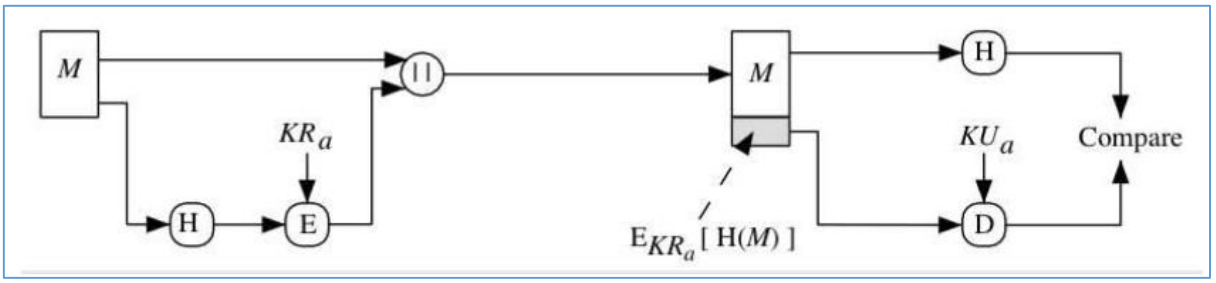
Arbitrated Digital Signatures

- involve use of arbiter A
  - validates any signed message
  - then dated and sent to recipient
- requires a great deal of trust in arbiter
- can be implemented with either private or public-key algorithms
- arbiter may or may not see message

Digital Signature Scheme

RSA APPROACH

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.

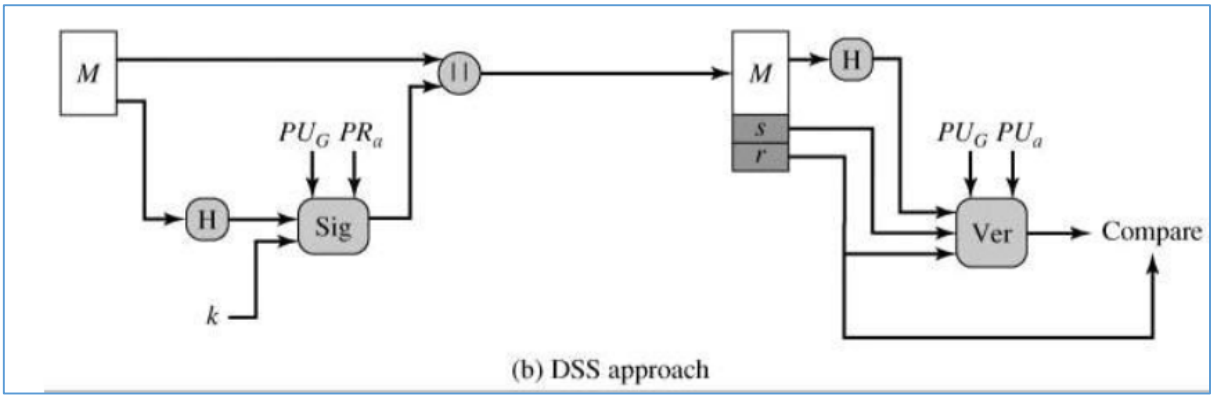


DSS APPROACH

- DSS uses an algorithm that is designed to provide only digital signature function. Unlike RSA, it cannot be used for encryption or key exchange. The DSS approach 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 (PRa) and the global public key (PUG)
- 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 (PUa), 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 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

1. Global Public key Components

p- prime no. where $2^{L-1} < p < 2^{L}$ for $512 \leq L \leq 1024$

q – prime divisor of (p-1)

Choose $g = h^{(p-1)/q} \bmod p$

where h is any integer with $1 < h < (p-1)$ such that $h^{(p-1)/q} \bmod p > 1$

2. User's Private key

x - random or pseudo random integer with $0 < x < q$

3. User's Public key

$y = g^{x} \bmod p$

4. User's Per Message Secret Number

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

5. DSA Signature Creation

To sign a message M the sender: the sender generates a random signature key k, $k<q$

Computes signature pair:

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

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

Signature = (r,s)

6. DSA Signature Verification

After received M & signature (r,s)

Verify a signature, recipient computes:

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

$u1 = [H(M')w] \bmod q$

$u2 = (r'w) \bmod q$

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

If v=r then signature is verified.

Schnorr Digital Signature Scheme

- As with the ElGamal digital signature scheme, the Schnorr signature scheme is based on discrete logarithms.
- The Schnorr scheme minimizes the message dependent amount of computation required to generate a signature.

Steps

1.Key Generation

Schnorr Key Setup: Choose suitable primes p , q

- Choose a such that $a^q = 1 \bmod p$
- (a,p,q) are global parameters for all
- Each user (e.g., A) generates a key
- Chooses a secret key (number): $0 < s < q$
- Computes his public key: $v = a^{-s} \bmod q$

2. Signature Generation

User signs message by

- Choosing random r with $0<r<q$ and
- computing $x = a^r \bmod p$
- Concatenating message with x and hashing:

  $e = H(M \| x)$
- Computing: $y = (r + se) \bmod q$
- Signature is pair (e, y)

3. Signature Verification

Any other user can verify the signature as follows:

- Computing: $x' = a^y v^e \bmod p$
- Verifying that: $e = H(M \| x')$
- $x' = a^y v^e = a^y a^{-se} = a^{y-se} = a^r = x \bmod p$

ElGamal cryptosystem

- In 1984, T. Elgamal announced a public-key scheme based on discrete logarithms, closely related to the Diffie-Hellman technique. The ElGamal cryptosystem is used in some form in a number of standards including the digital signature standard (DSS) and the S/MIME email standard.
- As with Diffie-Hellman, the global elements of ElGamal are a prime number q and a, which is a primitive root of q.
- User A generates a private/public key pair.
- The security of ElGamal is based on the difficulty of computing discrete logarithms, to recover either x given y, or k given K.

1. Global Public key Components

q - prime no.

a – primitive root of q

2. User A signs a message M to B by computing

- Generate a random integer X_A, such that $1 < X_A < q-1$
- Compute $\mathbf{Y}_A = a^{X_A} \bmod q$
- A's Private key is X_A
- A's Public key is $\{q, a, Y_A\}$

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

3. Any user B that has access to A's public key can encrypt a message as follows:

1. Represent the message as an integer M in the range $0 \ldots M \ldots q - 1$. Longer messages are sent as a sequence of blocks, with each block being an integer less than q.

2. Choose a random integer k such that $1 \ldots k \ldots q - 1$.

3. Compute a one-time key $K = (YA)^k \bmod q$.

4. Encrypt M as the pair of integers $(C1, C2)$ where

$C1 = a^k \bmod q$; $C2 = KM \bmod q$

4. User A recovers the plaintext as follows:

1. Recover the key by computing $K = (C1)^{XA} \bmod q$.

2. Compute $M = (C2^{K-1}) \bmod q$.

Example

For example, let us start with the prime field GF(19); that is, $q = 19$. It has primitive roots {2, 3, 10, 13, 14, 15}, We choose $a = 10$.

Alice generates a key pair as follows:

1. Alice chooses $XA = 5$.

2. Then $YA = a^{XA} \bmod q = a^5 \bmod 19 = 3$

3. Alice's private key is 5; Alice's pubic key is $\{q, a, YA\} = \{19, 10, 3\}$.

Suppose Bob wants to send the message with the value $M = 17$. Then,

1. Bob chooses $k = 6$.

2. Then $K = (YA)^k \bmod q = 3^6 \bmod 19 = 729 \bmod 19 = 7$.

3. So

$C1 = a^k \bmod q = 10^6 \bmod 19 = 11$

$C2 = KM \bmod q = 7 * 17 \bmod 19 = 119 \bmod 19 = 5$

4. Bob sends the ciphertext (11, 5). For decryption:

1. Alice calculates $K = (C1)^{XA} \bmod q = 11^5 \bmod 19 = 161051 \bmod 19 = 7$.

2. Then K^{-1} in GF(19) is $7^{-1} \bmod 19 = 11$.

3. Finally, $M = (C2^{K-1}) \bmod q = 5 * 11 \bmod 19 = 55 \bmod 19 = 17$.