**Computer Sciences**
**University of Technology**

/ / 2012
**Date**:
**Time**: 3 hours
**Lecturer**: DR.Alaa K.

**Final Exam. 2011-2012**
**Term**

**Subject** :Public key and Block cipher cryptography
**Class**: 3rd
**Branch**: information security

**حلول أسئلة النموذج الأول**

**Q1/ Write a complete program for encryption and decryption using ELGamal public key method.**

*Program in vb6*

```
t1 = Second(Time())

 beta = fast(alfa, k, p)

   Print #2, beta

    pb = fast(alfa, a, p)

      zz = fast(pb, k, p)

        While Not (EOF(1))

            xx = Asc(Input(1, #1))

             sekma = (xx * zz Mod p)

              Print #2, sekma;

         Wend

        t2 = Second(Time())

      Text3.Text = t2 - t1

    s = MsgBox("compelet encryption", 14, "Note")

                End If
```

**Q2/ E**ncrypt the message "computer" using the AES block cipher algorithm ,the secret key is "science".

Q2/

Sol/ The plain text : Computer

The Key : Science

The ASCII for plain and Key

| | | |
|---|---|---|
| C | 99 | 01100011 |
| O | 111 | 01101111 |
| M | 109 | 01101101 |
| P | 112 | 01110000 |
| U | 117 | 01110101 |
| t | 116 | 01110100 |
| e | 101 | 01100101 |
| r | 114 | 01110010 |

| | | |
|---|---|---|
| S | 115 | 01110011 |
| C | 99 | 01100011 |
| i | 105 | 11101001 |
| e | 101 | 01100101 |
| n | 110 | 11011110 |
| C | 99 | 01100011 |
| e | 101 | 01100101 |
| # | 32 | 00100000 |

Convert the Ascii from decimal to Hexdecimal

C ~ 99 = 63
O ~ 111 = 6F
M ~ 109 = 6D
P ~ 112 = 70
U ~ 117 = 75
t ~ 116 = 74
e ~ 101 = 65
r ~ 114 = 72

S ~ 115 = 73
C ~ 99 = 63
i ~ 105 = 69
e ~ 101 = 65
n ~ 110 =
C ~ 99 = 63
e ~ 101 = 6E
# ~ 32 = 20

* The Round R0 is ADD (plain, key)

| | | | |
|---|---|---|---|
| 63 | 6F | 6D | 70 |
| 75 | 74 | 65 | 72 |

+

| | | | |
|---|---|---|---|
| 73 | 63 | 69 | 65 |
| 6E | 63 | 6E | 20 |

=

| | | | |
|---|---|---|---|
| D6 | ED | D6 | D5 |
| E3 | D7 | D3 | 92 |

The new Transformation

* The Round n (substitute, Shift Row, ADD key)

1- substitute

D6 ED D6 D5
E3 D7 D3 92

each value as D6 where D is row and 6 is Col.
in S-Box:

D6 = F6
ED = 55
D6 = F6
D5 = 03

E3 = 11
D7 = 0E
D3 = 66
92 = 4F

then

| | | | |
|---|---|---|---|
| F6 | 55 | F6 | 03 |
| 11 | 0E | 66 | 4F |

2 - shift Row:-

F6 55 F6 03
11 0E 66 4F

- The Row(1) shift to left Zero location
- The Row(2) shift to left 1 location

11 0E 66 4F ⟹

| | | | |
|---|---|---|---|
| 0E | 66 | 4F | 11 |

then the new Transformation is:

| | | | |
|---|---|---|---|
| F6 | 55 | F6 | 03 |
| 0E | 66 | 4F | 11 |

3 - Add key

| | | | |
|---|---|---|---|
| F6 | 55 | F6 | 03 |
| 0E | 66 | 4F | 11 |

+

| | | | |
|---|---|---|---|
| 73 | 63 | 69 | 65 |
| 6E | 63 | 6E | 20 |

E2 B8 D8 68
= 7C C9 BD 31

Then the new Transformation is:

= E2 B8 D8 68
7C C9 BD 31

**Q3/E**xplain the following in detail

**a/**

Q3/ Compare between the S-Box (DES, Blowfish)

| DES | Blowfish |
|---|---|
| 1- 8 S-Box number | 1- 4 S-Box number |
| 2- input (6) and (4) output | 2- input (8) and output 32 |
| 3- 2D | 3- 1-D |
| 4- 4 bits on each cell | 4- S-Box array of 1-D 32-bits cell |
| 5- replacement | 5- replacement |
| 6- the X are 2-bit Y 4-bits as index | 6- 8-bits as index |
| 7- Static value | 7- init to replace value |
| 8- full off line | 8- full on run program. |

**b/ D**igital signature in RSA

The RSA public-key cryptosystem can be used for both encryption and signatures. Each user has three integers e, d and n, n = pq with p and q large primes. For the key pair (e, d), ed ≡ 1 (mod φ(n)) must be satisfied. If sender **A** wants to send signed message c corresponding to message m to receiver **B**, A signs it using A's private key, computing

$c \equiv m^{d}_{A} \pmod{n_A}$.

First A computes

$\phi(n_A) \equiv lcm\ (p_A - 1, q_A - 1)$

where **lcm** stands for the least common multiple. The sender **A** selects his own key pair $(e_A, d_A)$ such that

$e_A \bullet d_A \equiv 1 \pmod{\phi(n_A)}$

The modulus $n_A$ and the public key $e_A$ are published .The following Figure illustrates the RSA signature scheme.

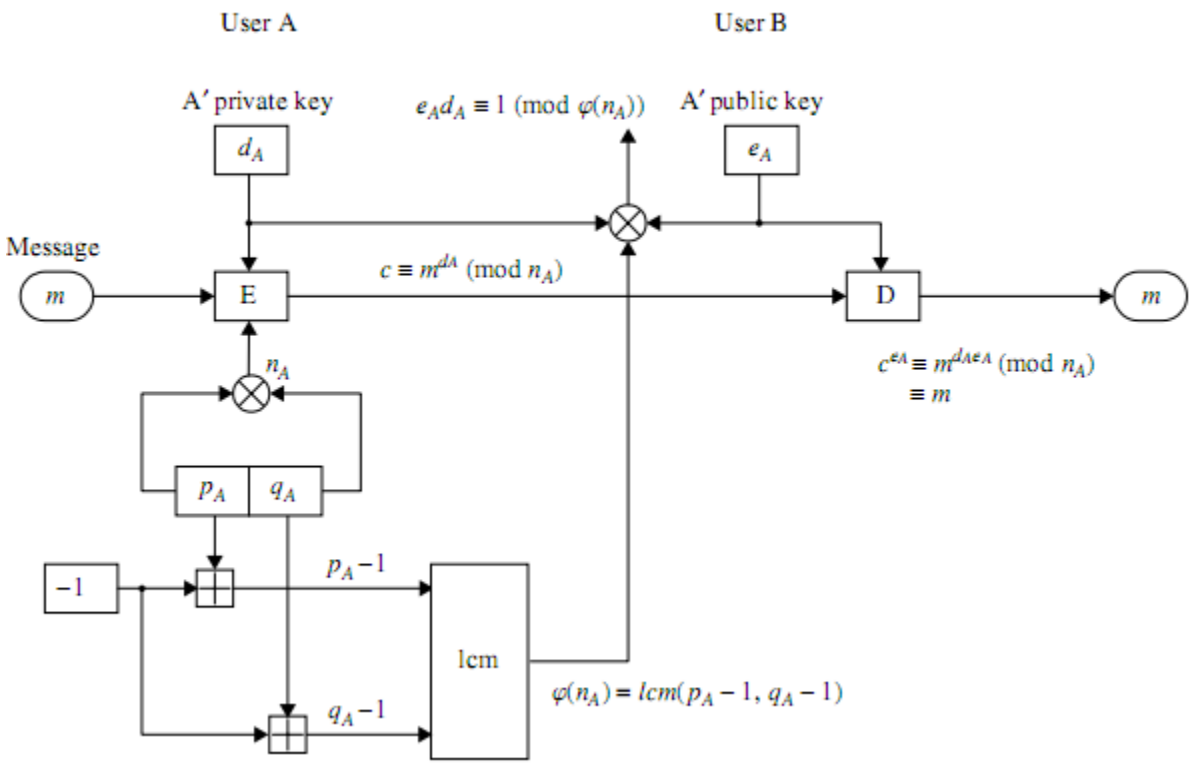


Figure :The RSA signature scheme.

**<u>Example</u>** 1:

Choose p = 11 and q = 17.Then n = p*q = 187.

Compute $\phi(n) = 1cm\ (p - 1, q - 1)$

$= 1cm\ (10, 16) = 80$

Select $e_A = 27$.Then $e_A d_A \equiv 1 \pmod{\phi(nA)}$

$27 d_A \equiv 1 \pmod{80}$

$d_A = 3$

Suppose **m = 55**. Then the signed message is

$c \equiv m^{dA} \pmod{187}$

$\equiv 553 \pmod{187} \equiv 132$

The message will be recreated as:

$m \equiv c^{eA} \pmod{n}$

$\equiv 13227 \pmod{187} \equiv$ **55**

Thus, the message m is accepted as authentic.

c/Encryption algorithm for McElice.

1- Integers **K,N** and **T** are fixed as common system parameter.
2- Each entity A should

   2.1 Choose a **K*N** generator (**G**) for a binary (**N,K**) liner code which can correct (**T**) errors and for which an efficient decoding algorithm is know.

   2.2 Select a random **K*K** binary non-singular matrix **S**

2.3 Select a random **N*N** permutation matrix **P**

3- Compute **K*N** matrix **Ǧ=SGP**

4- Alice public key is (**Ǧ,T**) and private key (**S,G,P**)

- **<u>Algorithm: Encryption</u>**

  Encryption **Bob** should do the following

1- Obtain Bob authentic public key(**Ǧ**,**T**)
2- Representation the message as binary string **M** of length **K**.
3- Choose a random binary vector (**Z**) of the length (**N**) have at most (**T**)
4- Compute the binary vector **C=M Ǧ +Z**
5- Send the cipher text **C** to Alice

**<u>Algorithm: Decryption</u>**

To recover the plain text (**M**) from **C** ,Alice should do the following:

1- Compute $\check{C} = CP^{-1}$ ,where $P^{-1}$ is the inverse of the matrix P
2- Compute $M^{\wedge} = \check{C}[\check{C}_1, \check{C}_2, \ldots \check{C}_n]$

   If $\check{C}_i > 0$ then $M_i^{\wedge} = 1$ else $M_i^{\wedge} = 0$

3- Compute $M = M^{\wedge} S^{-1}$

**Q4/Rind the Result :**

**1-**

$$\Phi(M^r) = M^{r-1}(M-1)$$

Find $\Phi(3^3)$ the number 9 is not prime then:

$\Phi(3^3) = 3^2(3-1) = 9*2 = 18$

{2,4,5,7,8,10,11,13,14,16,17,19,20,22,23,25,26}

2- 48 = 2 × 2 × 2 × 2 × 3,

$180 = 2 \times 2 \times 3 \times 3 \times 5$.

What they share in common is two "2"s and a "3"

Least common multiple = $2 \times 2 \times 2 \times 2 \times 3 \times 3 \times 5 = 720$

Greatest common divisor = $2 \times 2 \times 3 = 12$.

$3 - (-4) \bmod 7 = 3 \bmod 7 = 3$

$4 \ (-5^2) \bmod 3$

$25 \bmod 3 = 1$

**Q5/**

Q5/

1- design IP and $IP^{-1}$ for DES (16 input, 16 output)

The array of input and output is 4X4 cell and the index (1 to 16)

| | | | |
|---|---|---|---|
| 10 | 16 | 2 | 5 |
| 8 | 13 | 6 | 14 |
| 12 | 15 | 9 | 3 |
| 11 | 1 | 4 | 7 |

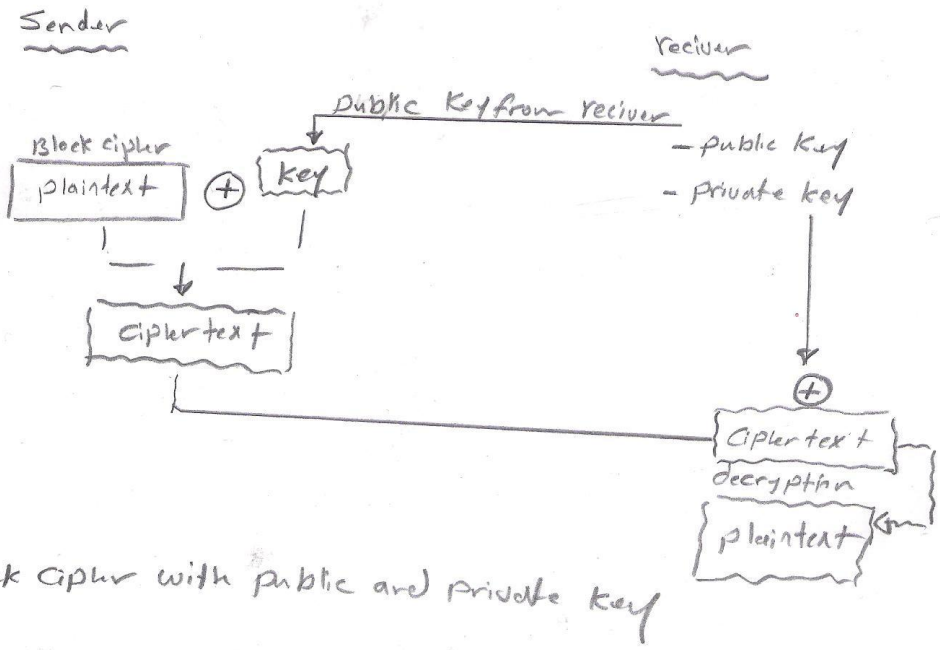
IP 4X4

| | | | |
|---|---|---|---|
| 14 | 3 | 12 | 15 |
| 4 | 7 | 16 | 5 |
| 11 | 1 | 13 | 9 |
| 6 | 8 | 10 | 2 |

$IP^{-1}$ 4 X 4

Q5/

2- Can enhance [illegible] Block




Block cipher with public and private key

**Q6/**

**a/ The Feistel (F) Function**

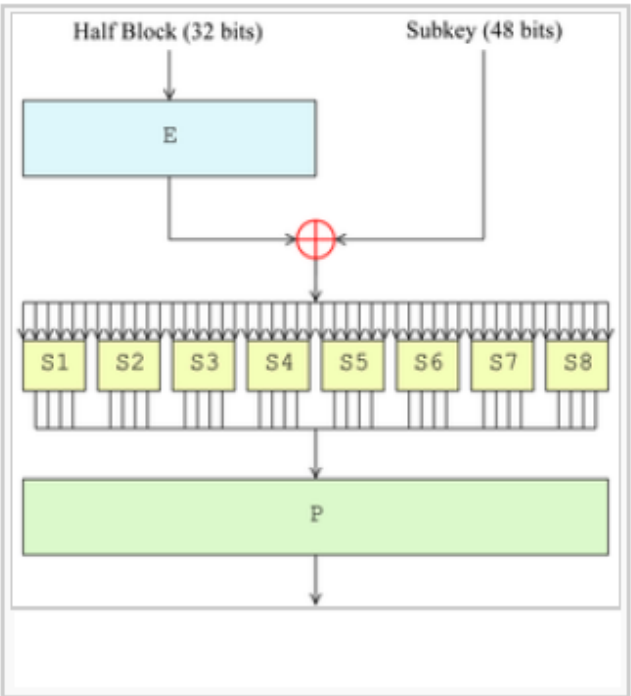
The F-function, depicted in Figure (2-2), operates on half a block (32 bits) at a time and consists of four stages:

**1.** *Expansion* the 32-bit half-block is expanded to 48 bits using the *expansion permutation*, denoted *E* in the diagram, by duplicating some of the bits.

**2.** *Key mixing* the result is combined with a *subkey* using an XOR operation. Sixteen 48-bit subkeys  one for each round are derived from the main key using the *key schedule* (described below).

**3.** *Substitution* after mixing in the subkey, the block is divided into eight 6-bit pieces before processing by the *S-boxes*, or *substitution boxes*. Each of the eight S-boxes replaces its six input bits with four output bits according to a non-linear transformation, provided in the form of a look up table. The S-boxes provide the core of the security of DES without them; the cipher would be linear, and trivially breakable.

**4.** *Permutation* finally, the 32 outputs from the S-boxes are rearranged according to a fixed permutation, the *P-box*. The alternation of substitution from the S-boxes, and permutation of bits from the P-box and Expansion provides so-called "confusion and diffusion" respectively, a concept identified by Claude Shannon in the 1940s as a necessary condition for a secure yet practical cipher.




**b/** The subkeys are calculated using the **Blowfish algorithm**. The exact method follows.

**(1)** Initialize first the P-array and then the four S-boxes, in order, with a fixed string. This string consists of the hexadecimal digits of p.

**(2)** XOR $P_1$ with the first 32 bits of the key, XOR $P_2$ with the second 32-bits of the key, and so on for all bits of the key (up to $P_{18}$). Repeatedly cycle through the key bits until the entire P-array has been XORed with key bits.

**(3)** Encrypt the all-zero string with the Blowfish algorithm, using the subkeys described in steps (1) and (2).

**(4)** Replace $P_1$ and $P_2$ with the output of step (3).

**(5)** Encrypt the output of step (3) using the Blowfish algorithm with the modified subkeys.

**(6)** Replace $P_3$ and $P_4$ with the output of step (5).

**(7)** Continue the process, replacing all elements of the P-array, and then all four S-boxes in order, with the output of the continuously changing Blowfish algorithm.

In total, 521 iterations are required to generate all required subkeys. Applications can store the subkeys—there's no need to execute this derivation process multiple times.

*NOTE:*

P=18 32-bit to byte (div 8)

P=18*4=72

S=256 32-bit to byte (div 8)

S=256*4(byte)*4(number s-box)=4096

The total= 4168

In each round we replace 2 p then 8 round to replace all p

Then

521(number Round)* 8(Number P-)=4168 byte as subkey