[Q1](#Q1)[Q2](#Q2)[Q3](#Q3)[Q4](#Q4)[Q5](#Q5)[Q6](#Q6)[Q7](#Q7)

**Eastern Mediterranean University**

**Computer Engineering Department**

**CMSE-353 Security of Software Systems**

 **Midterm Exam**

**Five A4 sheets of handwritten paper may be used for your help. Photocopies, printouts, etc. are not allowed! Electronic devices are not allowed**

**Duration: 110 Minutes November 28, 2018**

**Std Id\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Std Name\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Instructor Alexander Chefranov**

**Totally 6 questions, 9 pages**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | **7.** | **Total** |
| **10** | **15** | **15** | **15** | **15** | **15** | **15** | **100** |
|  |  |  |  |  |  |  |  |

**Q1.** **(10 points).** Consider a system with the users Emre, John, Simon, and Ahmed, objects O1, O2, O3, O4, O5. Emre can read O1 and owns O3, John is an owner of O2 and can read O1 and O4, Simon can read and write O2 and execute O4, Ahmed is an owner of O1, O4, and O5. Construct an Access Control List to keep the privileges described.

# Q2. (15 points). Encrypt and decrypt back the following English language message “Software Security” using English alphabet substitution cipher with a key phrase: “MPs to vote on May's Brexit deal on 11 December”. Construct a substitution table. Give necessary explanations.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 | 4 | 5 |
| A | B | c | d | e | f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z |
| m | p | s | t | o | v | e | n | a | y | b | r | x | i | d | l | c | f | g | h | j | k | q | u | w | Z |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Plaintext | s | o | f | t | w | a | r | e | s | e | c | u | r | i | t | Y |
| Ciphertext | g | d | v | h | q | m | f | o | g | o | s | j | f | a | h | W |
| Decrypted | s | o | f | t | w | a | r | e | s | e | c | u | r | i | t | y |

**Q3.** **(15 points).** Using Hill cipher with size 2 block,

1. Encrypt the **first** block of the following message “Hello, World!” preserving blanks, commas, and exclamation marks.

What numerical codes of the symbols you use? Construct an appropriate key matrix. What conditions must be satisfied by the key matrix? What modulo value shall be used? Show that the matrix you construct satisfies the conditions. Calculate inverse of the key matrix.

1. Decrypt back the **first** ciphertext block encrypted in a)

Give necessary explanations.

Code table

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| a | b | c | d | e | f | g | h | ı | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z | ‘ ‘ | , | ! |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 |

N=29

Encryption matrix; ; Conditions:

The first block of the plaintext is “he”= [7 4], ”pi”

Apply Extended Euclid algorithm to find :

1. A=(1,0,29), B=(0,1,27)

q=floor(29/27)=1

T=A-qB=(1,-1,2)

1. A=(0,1,27), B=(1,-1,2)

q=floor(27/2)=13

T=A-qB=(-13, 14,1), hence, . Check it:

Thus,

Let us check that the inverse matrix is calculated correctly:

Let us decrypt back now the 1st ciphertext block

=”he”=

**Q4.** **(15 points).** DES round key generation procedure is described in the text below and Figure 3.8 from the Lecture notes:

**“KEY GENERATION**

Input key has 64 bits. But each 8th bit is not used: bits 8,16,24,32,40,48,56,64 are not further used. The 56-bit key is first subjected to permutation Permuted Choice 1:

|  |
| --- |
| Permuted Choice 1 (PC-1) |
| 57 49 41 33 25 17 91 58 50 42 34 26 1810 2 59 51 43 35 2719 11 3 60 52 44 36 |
| 63 55 47 39 31 23 157 62 54 46 38 30 2214 6 61 53 45 37 2921 13 5 28 20 12 4 |

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 as governed by the following:

|  |
| --- |
| Schedule of Left Shifts |
| Round number 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16Bits rotated 1 1 2 2 2 2 2 2 1 2 2 2 2 2 2 1 |

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.

|  |
| --- |
| Permuted Choice 2 (PC-2) |
| 14 17 11 24 1 5 3 2815 6 21 10 23 19 12 426 8 16 7 27 20 13 241 52 31 37 47 55 30 4051 45 33 48 44 49 39 5634 53 46 42 50 36 29 32 |



“

The general DES schema is



A master 64-bit key, *K*, specified by a user in hexadecimal form is as follows, K=0x12bcde8932781121. Get the first round-key, *K1*. Give necessary explanations

**,**

K1=0x 04 55 aa a4 34 78

**Q5.** **(15 points).** Assume that 48-bit result of XOR with the round-key, Ki, in Figure 3.8, is as follows, Res=0x12dcbd560abc, in hexadecimal. What are the 4 bits output by S-box S2. Expansion/permutation table from Figure 3.8

|  |
| --- |
| Expansion/Permutation (E table) |
| 32 | 1 2 3 4 | 5 |
| 4 | 5 6 7 8 | 9 |
| 8 | 9 10 11 12 | 13 |
| 12 | 13 14 15 16 | 17 |
| 16 | 17 18 19 20  | 21 |
| 20 | 21 22 23 24 | 25 |
| 24 | 25 26 27 28 | 29 |
| 28 | 29 30 31 32 | 1 |

Figure 3.9 illustrating work of S-boxes



and Table 3.3 showing each S-box



In binary,

Input to S2 is (101101), Row#=(11)=3, Col#=(0110)=6. On the cross of the Row=3 and Col=6, we find 4=(0100) that are the bits output by S2.

**Q.6.** **(15 points)** Write the formula defining a multiplicative inverse in Zn? Find 131-1 mod 200 using Extended Euclid’s algorithm. Show your calculations. Check correctness of your calculations using the multiplicative inverse defining formula.

Hint:

EXTENDED EUCLID(m,b)

1. (A1,A2,A3):=(1,0,m); (B1,B2,B3):=(0,1,b);
2. if B3=0 return A3=gcd(m,b); no inverse
3. if B3=1 return B3 = gcd(m,b); B2= b-1 mod m
4. Q=
5. (T1,T2,T3):=(A1-QB1, A2-QB2, A3-QB3)
6. (A1,A2,A3):= (B1,B2,B3)
7. (B1,B2,B3):= (T1,T2,T3)
8. goto 2
9. A=(1,0,200), B=(0,1,131)

q=floor(200/131)=1

T=A-qB=(1,-1,69)

1. A=(0,1,131), B=(1,-1,69)

q=floor(131/69)=1

T=A-qB=(-1, 2, 62)

1. A=(1,-1,69), B=(-1,2,62)

q=floor(69/62)=1

T=A-qB=(2,-3,7)

1. A=(-1,2,62), B=(2,-3,7)

q=floor(62/7)=8

T=A-qB=(-17,26,6)

1. A=(2,-3,7), B=(-17,26,6)

q=floor(7/6)=1

T=A-qB=(19,-29,1), hence, 131-1 mod 200 = -29=171

Let us check it:

**Q.7.** **(15 points)** Consider the materials below on the AES S-box construction

“



The S-box is constructed in the following fashion:

1. Initialize the S-box with the byte values in ascending order row by row. Thus, the value of the byte at row x, column y is {xy}
2. Map each byte in the S-box to its multiplicative inverse in the finite field GF(28), ; the value {00} is mapped to itself.
3. Consider that each byte in the S-box consists of 8 bits labeled (b7,b6,b5,b4,b3,b2,b1,b0). Apply the following transformation to each bit of each byte in the S-box:

 (5.1)

where ci is the i-th bit of byte c with the value {63}, that is, (c7c7c5c4c3c2c1c0)=(01100011). The prime  indicates that the variable is to be updated by the value on the right. The AES standard depicts this transformation in matrix form as follows:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| B0’ |  | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |  | B0 |  | 1 |  |
| B1’ |  | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 |  | B1 |  | 1 |  |
| B2’ |  | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 |  | B2 |  | 0 |  |
| B3’ | = | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | x | B3 | + | 0 | (5.2) |
| B4’ |  | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |  | B4 |  | 0 |  |
| B5’ |  | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |  | B5 |  | 1 |  |
| B6’ |  | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 |  | B6 |  | 1 |  |
| B7’ |  | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |  | B7 |  | 0 |  |

Each element in the product matrix is the bitwise XOR of elements of one row and one column. Further, the final addition, shown in (5.2), is a bitwise XOR.

As an example, consider the input value {95}. The multiplicative inverse in GF(28) is {95}-1 ={8a}, which is 10001010 in binary. Using equation (5.2),

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |  | 0 |  | 1 |  | 1 |  | 1 |  | 0 |
| 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 |  | 1 |  | 1 |  | 0 |  | 1 |  | 1 |
| 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 |  | 0 |  | 0 |  | 0 |  | 0 |  | 0 |
| 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | x | 1 | + | 0 | = | 1 | + | 0 | = | 1 |
| 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |  | 0 |  | 0 |  | 0 |  | 0 |  | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |  | 0 |  | 1 |  | 0 |  | 1 |  | 1 |
| 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 |  | 0 |  | 1 |  | 1 |  | 1 |  | 0 |
| 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |  | 1 |  | 0 |  | 0 |  | 0 |  | 0 |

The result is {2a}, which should appear in row {09} column {05} of the S-box. This is verified by checking Table 5.4a.”

Calculate according to the description above S-box entry (2,2).

(xy)=(22)=(0010 0010)= and we need inverting it. Use Extended Euclid algorithm

1. A=(1,0,), B=(0,1,)

T=A-qB=(1,, )

1. A=(0,1,), B=(1,, )

T=A-qB=(, 1+, ),

1. A=(1,, ), B=(, , )

T=A-qB=(, , 1)= (, , 1),

hence,

Let us check it:

Hence,

Using matrix multiplication and addition:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |  | 0 |  | 1 |  | 0 |  | 1 |  | 1 |
| 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 |  | 1 |  | 1 |  | 0 |  | 1 |  | 1 |
| 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 |  | 0 |  | 0 |  | 0 |  | 0 |  | 0 |
| 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | x | 1 | + | 0 | = | 0 | + | 0 | = | 0 |
| 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |  | 1 |  | 0 |  | 1 |  | 0 |  | 1 |
| 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |  | 0 |  | 1 |  | 1 |  | 1 |  | 0 |
| 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 |  | 1 |  | 1 |  | 1 |  | 1 |  | 0 |
| 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |  | 0 |  | 0 |  | 1 |  | 0 |  | 1 |

Thus S-box(2,2)=(1001 0011)=(93) that is actually seen in the S-box.