CSS 322 – ASSIGNMENT 1 ANSWERS

(See the Excel spreadsheet for individual answers to the encryption questions)

Question 1(a)

Lets use an example student ID of: 4722772961

Then the key is calculated as:

Key	=	dec2bin(ID mod 1024)
	=	dec2bin(993)
	=	1111100001

The Key should be 10 digits (for example, make sure enough 0's are on the left hand side).

Since we are using Electronic Code Book, then we can simply treat each block separately. S-DES operates on 8-bit blocks of plaintext so the first 8 bits are:

P = 0101 0011

Lets generate the two sub-keys, K1 and K2.

Apply P10 on Key, which re-arranges the 10-bits so the output order is:

3 5 2 7 4 10 1 9 8 6

So we have: 11101 11000

Now do a left shift of one bit on both the left and right half:

11011 10001

Now apply P8 which picks 8 bits and re-arranges according to:

6 3 7 4 8 5 10 9

So we have: 10010110

This is the value of K1: 10010110

Now continuing from the previous 10 bits, we do a left shift by 2 positions to get:

01111 00110

And now apply P8 again to get K2:

01011101

Now we can do the encryption of P = 0101 0011

The Initial Permutation IP rearranges the plaintext according to:

2 6 3 1 4 8 5 7

So we get: 1000 1101

Now operating on the right half (1101) we expand and permute using:

4 1 2 3 2 3 4 1

To get: 11101011

Now XOR with the key K1 to get:

```
      11101011
XOR   10010110
=     01111101
```

And split into left and right halves and input into S-Boxes S0 and S1

Input to S0 is 0111. We look up row 01 and column 11 to get 00
Input to S1 is 1101. We look up row 11 and column 10 to get 00

The output 4 bits of the S-Boxes are rearranged according to P4:

2 4 3 1

To get: 0000 (easy!)

Now we XOR with the left half of the IP (1000) to get:

```
      0000
XOR   1000
=     1000
```

And we join with the original right half from IP to get 1000 1101

And swap the halves and sue as input to the second round: 1101 1000

Expand and permute on right half (1000) gives:
01000001

XOR with key K2

```
      01000001
XOR   01011101
=     00011100
```

Feed the two halves into the S-Boxes:

S-Box S0: 0001 gives row 01 and column 00: 11
S-Box S1: 1100 gives row 10 and column 10: 01

Apply P4 gives: 1101

XOR with left half (1101):

	1101
XOR	1101
=	0000

And the left half (0000) and right half (1000) are input to the inverse IP:

4 1 3 5 7 2 8 6

To give the Ciphertext of:
00010000

In summary the S-DES encryption of 01010011 with Key 1111100001 produces the Ciphertext 00010000

A test using the software at http://buzzard.ups.edu/sdes/sdes.html confirms the results.

Question 1(b)

Using ECB, to encrypt the remaining parts we simply have to apply the same S-DES algorithm on each plaintext block using the same input key 1111100001. Then the final output ciphertext will be the concatenation of the ciphertext output from each block.

Question 1(c)

There are 4 b-bit blocks in the original plaintext so we would apply S-DES four times, all using the Key = 1111100001.

Plaintext 1: 01010011	Ciperhtext1: 00010000
Plaintext 2: 01001001	Ciphertext2: 11110110
Plaintext 3: 01001001	Ciphertext3: 11110110
Plaintext 4: 01010100	Ciphertext4: 00001110

Final Ciphertext: 00010000 11110110 11110110 00001110

Question 1(d)
If using the counter mode of operation (rather than ECB) then you encrypt a counter value with S-DES and then XOR the result with the block of plaintext to obtain the ciphertext.

Hence our inputs to S-DES are the Key and P1 to P4 being the counter values. The outputs are also shown:

Key = 1111100001

P1	=	00000000	O1	=	10011111
P2	=	00000001	O2	=	11010000

P3	=	00000010	O3	=	10011010
P4	=	00000011	O4	=	00100011

So now we XOR each O with the input plaintext block

O1:		10011111
	XOR	01010011
	=	11001100

O2:		11010000
	XOR	01001001
	=	10011001

O3:		10011010
	XOR	01001001
	=	11010011

O4:		00100011
	XOR	01010100
	=	01110111

So the end ciphertext is: 11001100 10011001 11010011 01110111

Question 1(e)

ECB is not suitable for encrypting long messages because if a block of plaintext is repeated in the input, then the same ciphertext will be obtained. That is, you will get repetitions of the ciphertext matching repetitions of the plaintext. This makes it easier for the attacker to analyse. Whereas ounter Mode produces a different ciphertext block, even if the plaintext block is the same. Blocks 2 and 3 are the same in the plaintext. Note that ECB produces the same output ciphertext block (11110110) where as Counter Mode produces different ciphertext.

Question 2(a)

Lets use an example student ID of: 4722772961

Then the key is calculated as:

Key	=	dec2bin(ID mod 65536)
	=	dec2bin(52193)
	=	1100 1011 1110 0001

The Key should be 16 digits (for example, make sure enough 0's are on the left hand side).

The ciphertext is: 1000 1011 0111 1010

The first step is to generate the round keys:

K0 = original key = 1100 1011 1110 0001

W0	=	1100 1011
W1	=	1110 0001
W2	=	W0 XOR 1000 0000 XOR SubNib(RotNib(W1))
	=	1100 1011 XOR 1000 0000 XOR SubNib(0001 1110)

From S-Box for decryption:

0001: row 00, col 01 = 0101
1110: row 11, col 10 = 1101

	=	1100 1011 XOR 1000 0000 XOR 0101 1101
	=	0100 1011 XOR 0101 1101
	=	0001 0110
W3	=	W2 XOR W1
	=	0001 0110 XOR
		1110 0001
	=	1111 0111
W4	=	W2 XOR 0011 0000 XOR SubNib(RotNib(W3))
	=	0001 0110 XOR 0011 0000 XOR SubNib(0111 1111)
	=	0010 0110 XOR 1111 1110
	=	1101 1000
W5	=	W4 XOR W3
	=	1101 1000 XOR 1111 0111
	=	0010 1111
K1	=	W2, W3
	=	0001 0110 1111 0111
K2	=	W4, W5
	=	1101 1000 0010 1111

So the first step of the encryption is to add the round key K2 to the Ciphertext:

```
       1000 1011 0111 1010 XOR
       1101 1000 0010 1111
=      0101 0011 0101 0101
```

Now do the inverse shift row, which is same as normal shift row: swap the second nibble with the fourth nibble to get:

```
       0101 0101 0101 0011
```

Now an inverse nibble substation, which uses the inverse (decryption S-Box):

```
       0101 -> row 01, col 01 -> 0111
       0011 -> row 00, col 11 -> 1011

       0111 0111 0111 1011
```

Now add the round key K1:

```
       0111 0111 0111 1011
XOR    0001 0110 1111 0111
=      0110 0001 1000 1100
```

Now do the inverse mix columns. Assuming the nibbles are n1, n2, n3, n4, then according to the mix ciolumn for decryption function we get:

(First convert nibbles to Hex for GF multiplcaition: 6, 1, 8, C)

```
N1'    =     9n1 XOR 2n2
       =     9 x 6 XOR 2 x 1
       =     3 XOR 2              or 0011 XOR 0010
       =     1                    or 0001

N2'    =     2n1 XOR 9n2
       =     2 x 6 + 9 x 1
       =     C + 9
       =     5                    or 0101

N3'    =     9n3 XOR 2n4
       =     9 x 8 + 2 x C
       =     4 + B
       =     F
       =     1111

N4'    =     2n3 XOR 9n4
       =     2 x 8 + 9 x C
       =     3 + 6
       =     5
       =     0101
```

So the result is: 0001 0101 1111 0101

Now we are into the second round, so do the inverse shift row to get:

0001 0101 1111 0101 (not that the 2nd and 4th nibbles are the same)

Now the inverse nibble substitution using the S-Box:

0001 -> row 00, col 01 -> 0101
0101 -> row 01, col 01 -> 0111
1111 -> row 11, col 11 -> 1110

0101 0111 1110 0111

Now the final adding of the round key K0

	0101 0111 1110 0111
XOR	1100 1011 1110 0001
=	1001 1100 0000 0110

And so the final plaintext is: 1001 1100 0000 0110

Question 2(b)

(Question and answers from Question 5.6 in Stallings textbook)

a. AddRoundKey
b. The MixColumn step, because this is where the different bytes interact with each other.
c. The ByteSub step, because it contributes nonlinearity to AES.
d. The ShiftRow step, because it permutes the bytes.
e. There is no wholesale swapping of rows or columns. AES does not require this step because: The MixColumn step causes every byte in a column to alter every other byte in the column, so there is not need to swap rows; The ShiftRow step moves bytes from one column to another, so there is no need to swap columns

Question 3

example1.txt

DES/ECB using Key: 13 F6 A7 83 E9 1B 33 4B

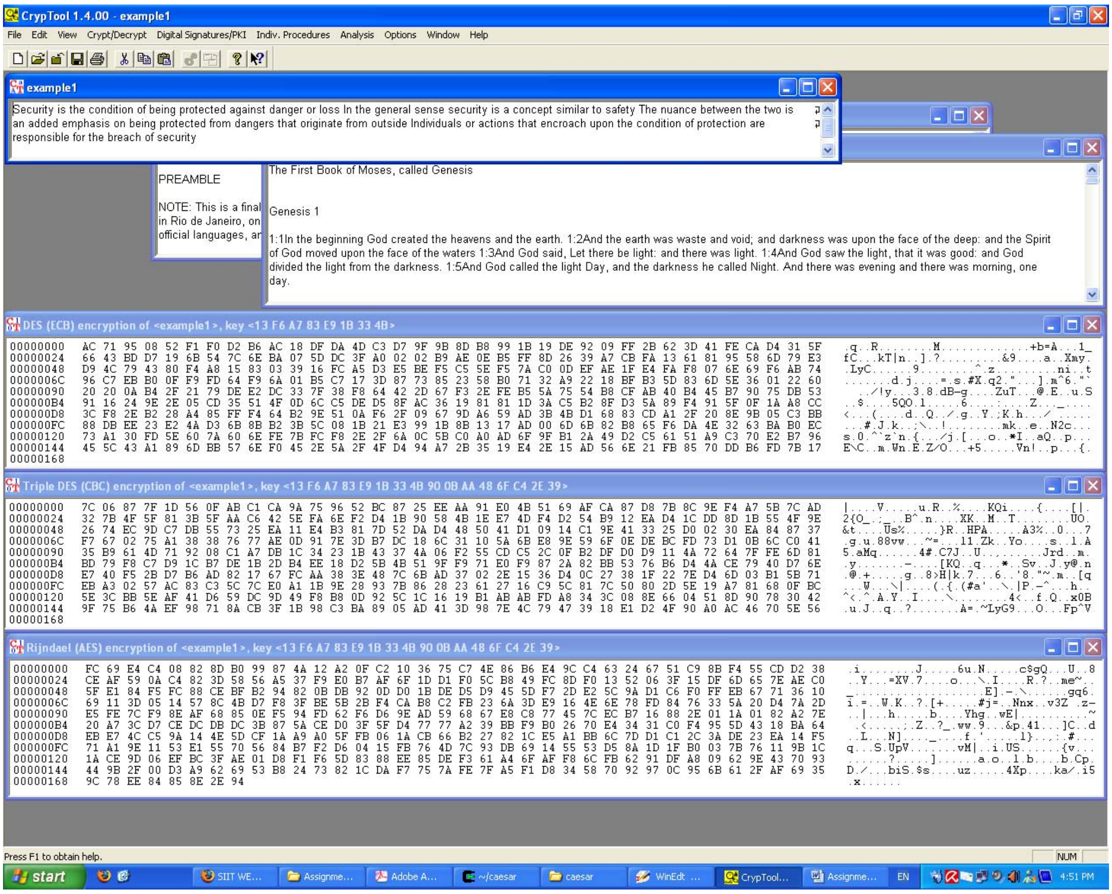
- Encryption was immediate (e.g. less than 1 second)
- Decryption was immediate
- Note that the decrypted version contains NULL characters when viewed as ASCII text. These are added for padding.

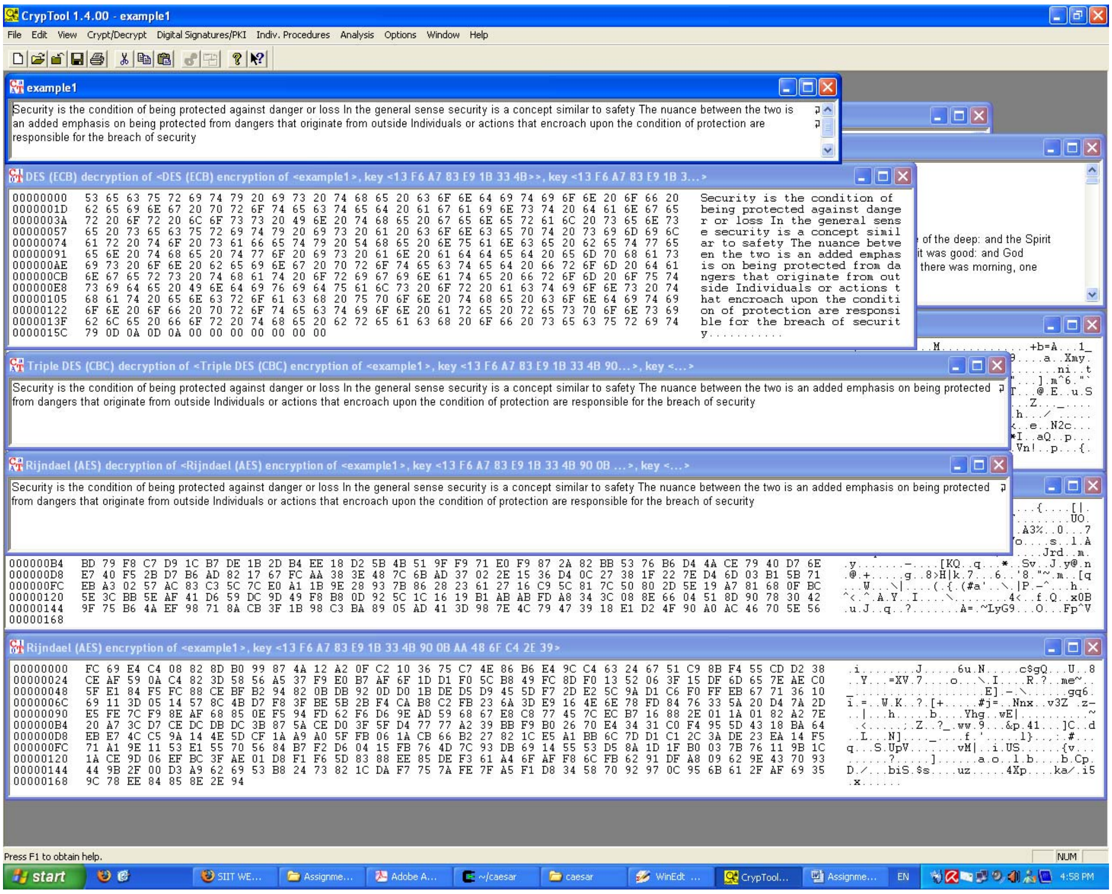
3DES/CBC using Key: 13 F6 A7 83 E9 1B 33 4B 90 0B AA 48 6F C4 2E 39

- Encryption was immediate
- Decryption was immediate

AES using Key: 13 F6 A7 83 E9 1B 33 4B 90 0B AA 48 6F C4 2E 39

- Encryption was immediate
- Decryption was immediate




CrypTool 1.4.00 - example1
File Edit View Crypt/Decrypt Digital Signatures/PKI Indiv. Procedures Analysis Options Window Help
example1
Security is the condition of being protected against danger or loss In the general sense security is a concept similar to safety The nuance between the two is an added emphasis on being protected from dangers that originate from outside Individuals or actions that encroach upon the condition of protection are responsible for the breach of security
Triple DES (CBC) decryption of <Triple DES (CBC) encryption of <example1>, key <13 F6 A7 83 E9 1B 33 4B 90...>, key <...>
Rijndael (AES) decryption of <Rijndael (AES) encryption of <example1>, key <13 F6 A7 83 E9 1B 33 4B 90 0B ...>, key <...>
Rijndael (AES) encryption of <example1>, key <13 F6 A7 83 E9 1B 33 4B 90 0B AA 48 6F C4 2E 39>
Press F1 to obtain help.
NUM
start
4:58 PM