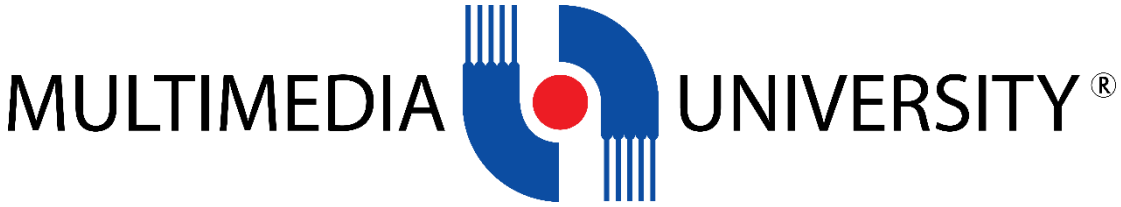

TDS3551

Data Management

NAME	ID
Wong Zi Xiang	1142701102
Andy Yong Jun Jie	1142700565
Lee Boon Ping	1142701239
Fang Siaw Tung	1141328079

Table of Contents

Abstract

Numerous type of ciphers had been created since ancient times until now. In this assignment, we will be implementing both classic symmetric ciphers and modern symmetric cipher. Classic symmetric ciphers we implemented are Vigenère Cipher (Poly-alphabetic) and Additive Cipher (Mono-alphabetic). Modern symmetric cipher implemented is Advanced Encryption Standard (AES). We will be comparing time complexity between Vigenère Cipher and Additive Cipher. Also, we will inject bit error into AES ciphertext to investigate effects of bit errors on transmitted ciphertext.

Introduction

In cryptography, a cipher is an algorithm for performing encryption or decryption—a series of well-defined steps that can be followed as a procedure. There are two types of ciphers, classical and modern ciphers.

A *classical cipher* is a type of cipher that was invented before the computer era. Most classical cipher can be solved by hand and simple to break with modern computers. There are two types of classical ciphers—substitution cipher and transposition cipher. In a substitution cipher, letters are systematically replaced throughout the message for other letters. Whereas in transposition cipher, the letters themselves are kept unchanged, but their order within the message is scrambled according to some well-defined scheme.

A *modern cipher* is a type of cipher that was invented and designed to be run on computers. There are two types of modern cipher, which are symmetric and asymmetric cipher. The main difference between symmetric and asymmetric cipher is the number of keys. In symmetric ciphers, one key is shared on encryption and decryption; In asymmetric ciphers, public key of intended recipient is used to encrypt messages, then private key of recipient is used to decrypt the secret message.

Member's Contribution

NAME	Contribution
Wong Zi Xiang	AES+CBC, Abstract, Introduction, Background Study AES, Implementation AES+CBC
Andy Yong Jun Jie	Caesar & Vinegere Cipher, Background Study Caesar & Vinegere Cipher, Implementation Caesar & Vinegere
Lee Boon Ping	Report
Fang Siaw Tung	Report

Background Study

Caesar Cipher

One of the simplest and most widely known encryption techniques. It is a type of monoalphabetic substitution. During encryption, the ciphertext is plaintext shifted left or right by a certain number of positions. Decryption is done is reverse, if plaintext is shifted left 3 times during encryption, decrypt it by shifting right the cipher text 3 times. Caesar cipher also can be represented using modular arithmetic by transforming the alphabets into numbers. For example, alphabet *Z* can be represented as 25, *A* can be represented as 0. During encryption, plaintext value is added with shift value and then perform modulo operation. During encryption, plaintext value is subtracted by shift value and then perform modulo operation. The algorithm is shown below:

$$E(x) = (x + n)\ mod\ 26$$
$$D(x) = (x - n)\ mod\ 26$$

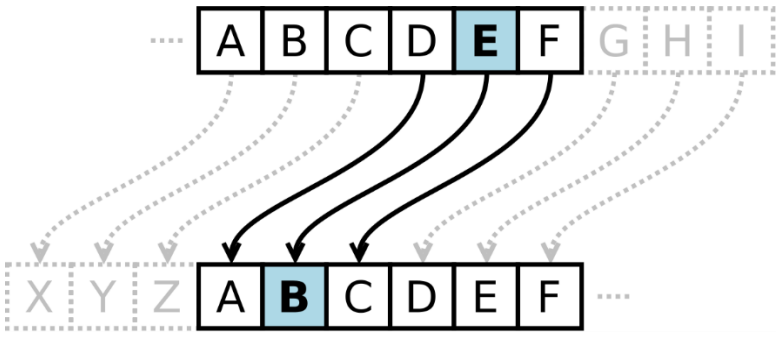

Figure 1.1 Example of Caesar's Cipher

Vigenère Cipher

Vigenère Cipher is a type of polyalphabetic substitution cipher originally proposed by Glovan Battista Bellaso. However, the cipher was later credited incorrectly to Blaise de Vigenère in the 19th century, thus the cipher is now widely known as Vigenère Cipher.

Vigenère Cipher is similar to Caesar Cipher. Both uses modulo operation on alphabets. But Vigenère Cipher requires a key in the form of alphabets. In encryption, the key alphabets are firstly repeated until it matches the length of the plaintext, then the plaintext alphabets and key alphabets is converted to numbers, then they are added together and perform modulo operation. During decryption, ciphertext alphabets and key alphabets are subtracted and perform modulo operation.

Advanced Encryption Standard

AES is a subset of the Rjindael cipher developed by Vincent Rijmen and Joan Daemen. For AES, NIST selected three members of Rjindael family, each with fixed block size 16 bytes and three key sizes: 128, 192 and 256 bits. AES is adopted by United States government and used globally.

AES is designed based on principle known as substitution-permutation network, which mean a composition of substitution and permutation. The key size of the AES determines the number of transformation rounds that convert the input. The number of round, *Nr*:

- 128-bits is 10 rounds
- 192-bits is 12 rounds
- 256-bits is 14 rounds

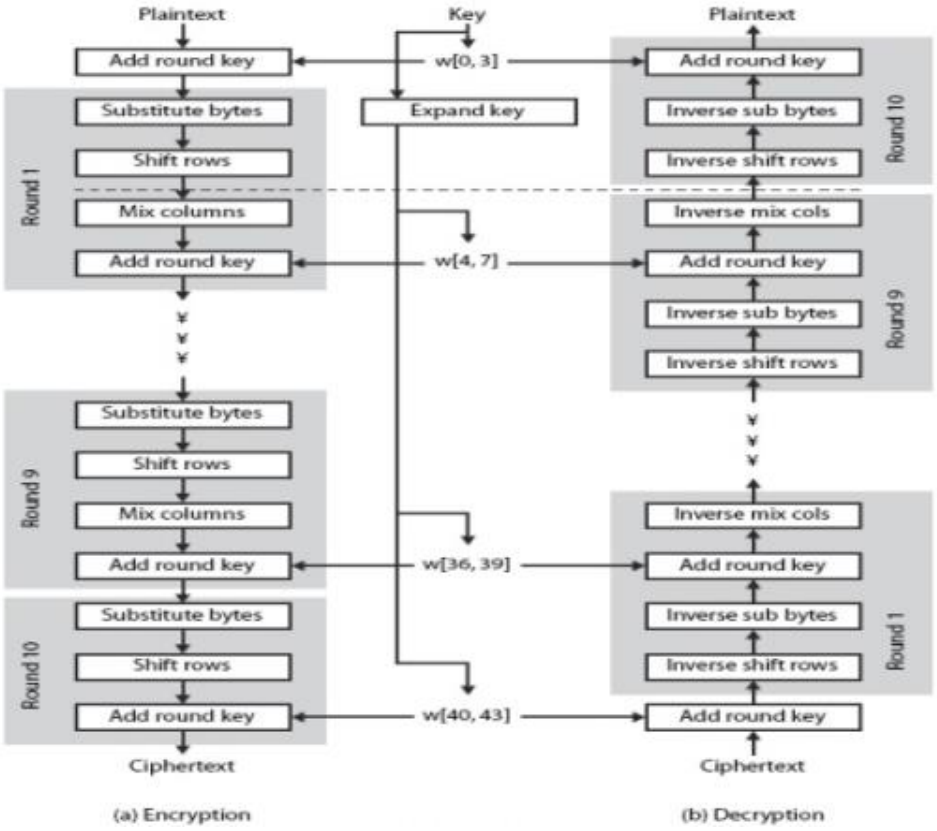

Figure 1.2 Advanced Encryption Standard

Each round is made up of 4 processing steps. Reverse rounds are applied to decrypt the ciphertext back to plaintext using the same key. The high-level description of the algorithm is:

1. KeyExpansions — generates round keys
2. InitialRound
    1. AddRoundKey — each byte of the state is XOR-ed with round keys block
3. Rounds
    1. SubBytes — each byte is replaced with another according to S-box
    2. ShiftRows — last 3 rows of state shifted a certain number of steps
    3. MixColumns — four bytes in each column in state is combined
    4. AddRoundKey
4. FinalRound (no MixColumns)
    1. SubBytes
    2. ShiftRows
    3. AddRoundKey

Implementation Details
Caesar Cipher

Refer back to Background Study for Caesar Cipher. We will go into details of the implementation of Caeser Cipher. The algorithm is shown below:

$$E(x) = (x + n)\ mod\ 26$$
$$D(x) = (x - n)\ mod\ 26$$

X is value of depending on the position in the alphabet of the plaintext or ciphertext and n is the key for encryption and decryption. Key for encryption and decryption must between 0 until 25. For example, to encrypt a plaintext, first letter of plaintext and the key that between 0 until 25 is adding together and mod 26 will output position of the ciphertext letter. In decryption, first letter of ciphertext and the key that between 0 until 25 to subtract their value. If the value of the letter after subtract is negative, add 26 and the result will show the position of plaintext. From the position will know the letter.

A	B	C	D	E	F	G	H	I	J	K	L	M
↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕
0	1	2	3	4	5	6	7	8	9	10	11	12
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕
13	14	15	16	17	18	19	20	21	22	23	24	25

Figure 2.1 Table of alphabet code for encrypt and decrypt

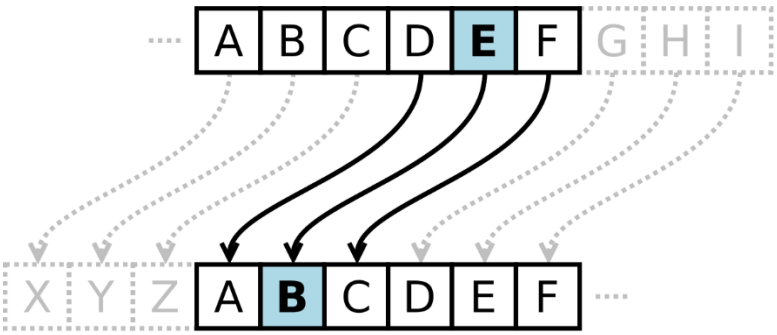

Figure 2.2 Example of Caeser Cipher

Plaintext: h → 07	Encryption: (07 + 15) mod 26	Ciphertext: 22 → W
Plaintext: e → 04	Encryption: (04 + 15) mod 26	Ciphertext: 19 → T
Plaintext: l → 11	Encryption: (11 + 15) mod 26	Ciphertext: 00 → A
Plaintext: l → 11	Encryption: (11 + 15) mod 26	Ciphertext: 00 → A
Plaintext: o → 14	Encryption: (14 + 15) mod 26	Ciphertext: 03 → D

Figure 2.3 Example of Caeser Cipher

In order to break Caeser Cipher, exhaustive key searches and brute-force attacks to are used to break the ciphertext without the key because the key domain only has 26 keys which only 25 keys are of any use. Besides that, it also could simply try each in turn of key from 0 until 25.

Vigenère Cipher

Refer back to Background Study for Vigenere Cipher. We will go into details of the implementation of Vigenere Cipher. The algorithm is shown below:

$\boldsymbol{M = M_1 \ldots M_n}$ is the message, $\boldsymbol{C = C_1 \ldots C_n}$ is the ciphertext and $\boldsymbol{K = K_1 \ldots K_n}$ is the key.

Encryption:

$$C_i = E_K(M_i) = (M_i + K_i)\ mod\ 26$$

Decryption:

$$M_i = D_K(C_i) = (C_i - K_i)\ mod\ 26$$

M is message or plaintext, C is ciphertext and K is key for encryption and decryption. Letters have a value depending position in the alphabet which starting from 0. For example, to encrypt a text, first letter of message and the first letter of the key is adding together to the value and mod 26, resulted will give the position of the ciphertext letter. Decrypting a ciphertext, first letter of ciphertext and first letter of the key to subtract their value. If the value is negative, add 26 and the result will show the position of plaintext. From the position will know the letter. According to the position of alphabet table below:

A	B	C	D	E	F	G	H	I	J	K	L	M
↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕
0	1	2	3	4	5	6	7	8	9	10	11	12
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕
13	14	15	16	17	18	19	20	21	22	23	24	25

Figure 2.4 Table of alphabet code for encrypt and decrypt

Plaintext:	s	h	e	i	s	l	i	s	t	e	n	i	n	g
P's values:	18	07	04	08	18	11	08	18	19	04	13	08	13	06
Key stream:	***15***	***00***	***18***	***02***	***00***	***11***	***15***	***00***	***18***	***02***	***00***	***11***	***15***	***00***
C's values:	**07**	**07**	**22**	**10**	**18**	**22**	**23**	**18**	**11**	**6**	**13**	**19**	**02**	**06**
Ciphertext:	**H**	**H**	**W**	**K**	**S**	**W**	**X**	**S**	**L**	**G**	**N**	**T**	**C**	**G**

Figure 2.5 Example of Vigenere Cipher Encryption

	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	v	v	w	x	y	z
A	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
B	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	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	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

Figure 2.6 Vigenere Tableau

The method to encrypt by this table is locate the first letter of plaintext message in the first row of the table and first letter of the key from the left column. The ciphertext is located at the intersection. For example, given a key letter M and a plaintext letter t, the ciphertext letter is at the intersection of the row labelled M and the column labelled t, so the ciphertext is F. For decryption, locate first letter of the key in the left column and locates the row of first letter of cipher text then go up in the column to read first letter which is

plaintext. For example, given a key letter M and ciphertext letter F, the plaintext letter is at the top of column of the ciphertext letter F, so the plaintext letter is t.

By breaking Vigenere Cipher, it should determine length of the key or the key. Primary method to find the length of the key which is Kasiski. Further cryptanalysis based on frequency analysis of ciphertext letter after knowing the key size. Ciphertext is encrypted with different key should be analysed separately.

Advanced Encryption Standard with Cipher Feedback

Refer back to Background Study for AES. We will go into details of the implementation of AES.

KeyExpansion

A key schedule is used to expand a short key into number of separate round keys. It produces the needed round keys from the initial cipher key.

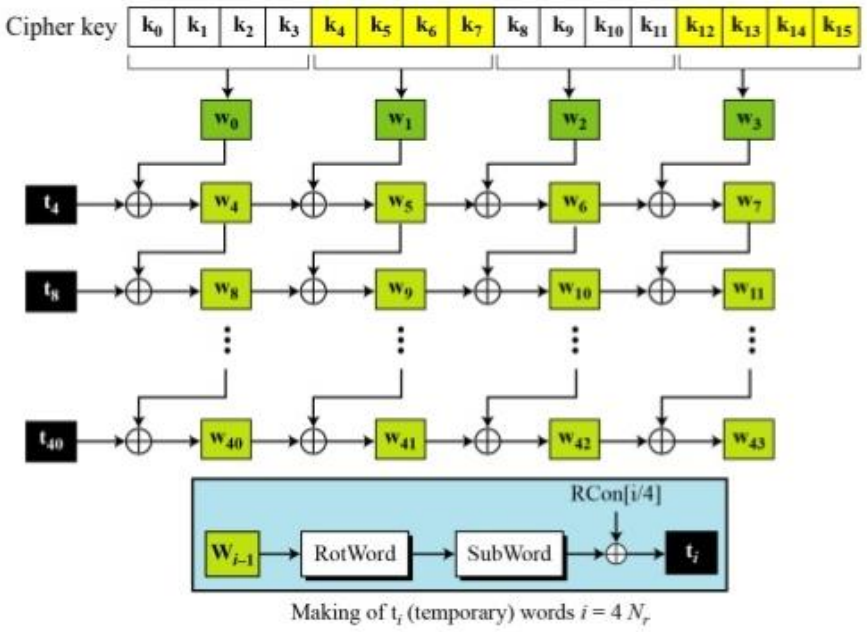

Figure 2.7 Key Expansion Diagram

AddRoundKey

In this step, each byte of the plaintext is combined with respective byte of the round subkey using bitwise XOR.

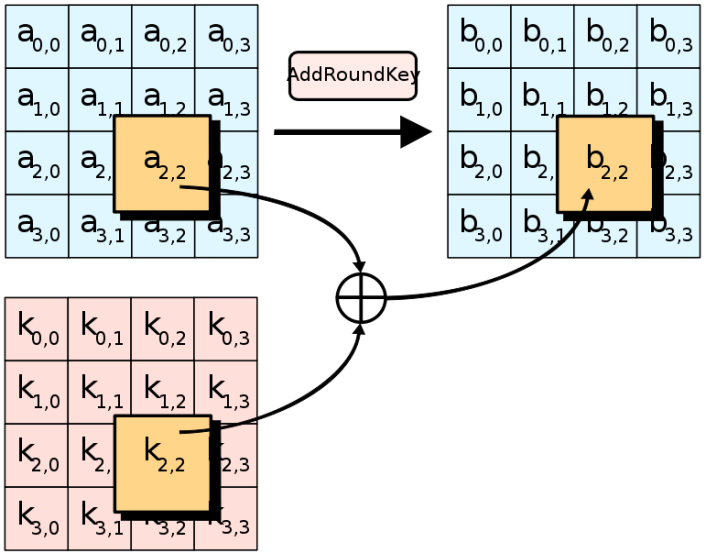

Figure 2.8 Add Round Key Diagram

SubBytes

In this step, each byte in the state matrix is replaced with SubByte using an 8-bit substitution box (S-box).

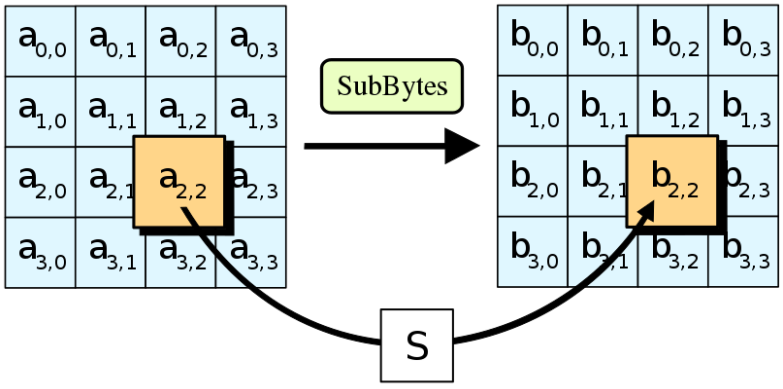

Figure 2.9 Substitute Bytes Diagram

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

Figure 2.10 S-Box

ShiftRows

This step operates on the rows of the state. Bytes in each row is cyclically shifted by a certain offset. The first row is unchanged. The second row is shifted one to the left. Third and fourth row is shifted two and three to the left respectively.

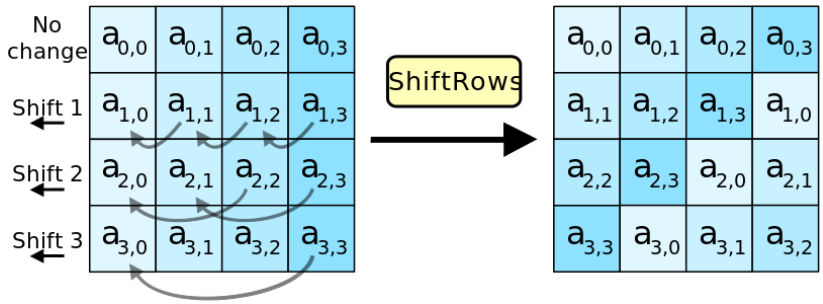

Figure 2.11 Shift Rows Diagram

MixColumns

In this step, each column is transformed using a fixed matrix. Each column of the state is multiplied with the fixed matrix

$$\begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix}$$

Figure 3.6 Fixed Matrix

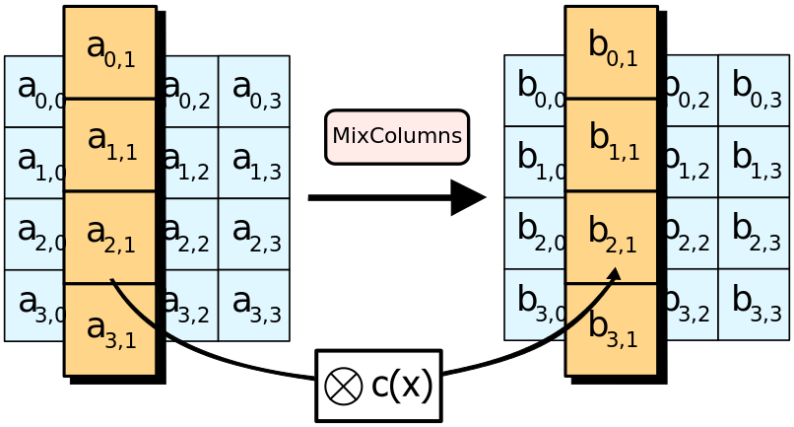

Figure 2.12 Mix Columns Diagram

Cipher Feedback Mode

AES utilizes mode of operation is needed to encrypt multiple block of data. In CFB, an initialization vector (IV) is used in the first block and encrypted. Part of the output is selected (s-bits) and the first plaintext block is XOR-ed with the selected bits. The result is the first ciphertext block. The contents in the shift register are shifted left by s-bits and first ciphertext block is placed in the right most s-bits of the shift register. Encryption and decryption using OFB is the same.

$$C_i = head(E_K(S_{i-1}), x) \oplus P_i$$
$$P_i = head(E_K(S_{i-1}), x) \oplus C_i$$
$$S_i = \left((S_{i-1} << x) + C_i\right) mod\ 2^n$$
$$S_0 = IV$$

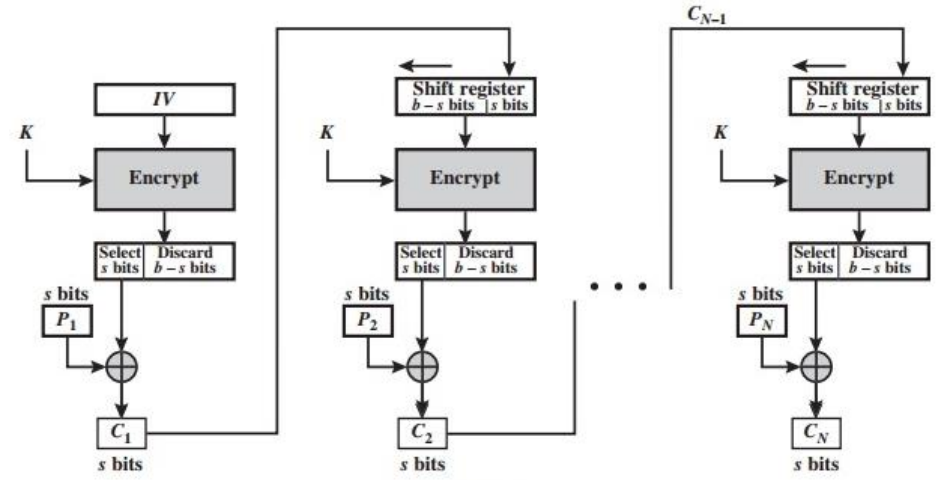

Figure 2.13 Cipher Feedback Mode Encryption

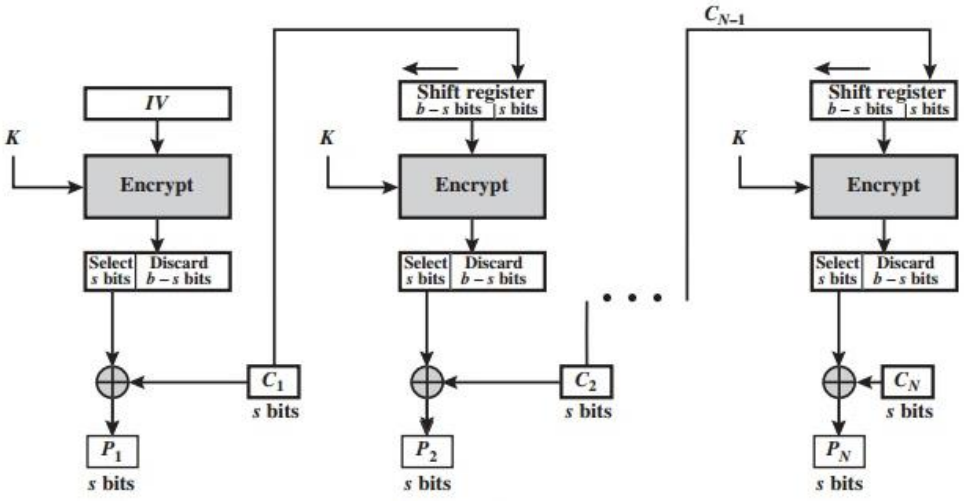

Figure 2.14 Cipher Feedback Mode Decryption

Results

Cipher Feedback Mode Bit-error Propagation

We have injected one bit-error in the ciphertext. Results of decryption are shown below:

Plaintext : Computer security is fun!

Original Ciphertext : IûfÅ5[00][1A][00][95]fÑ°ñ;³![00][83].sŽbÆa[00][08][00][85]+ÐÉ

Error Ciphertext : IûfÅ7[00][1A][00][95]fÑ°ñ;³![00][83].sŽbÆa[00][08][00][85]+ÐÉ

Decrypted ciphertext: Compvter securitv is fun!

As we can see from the result above, the error is contained in one-bit only and is not propagated to other blocks.

Comparison of results (Caeser Cipher and Vigenere Cipher)

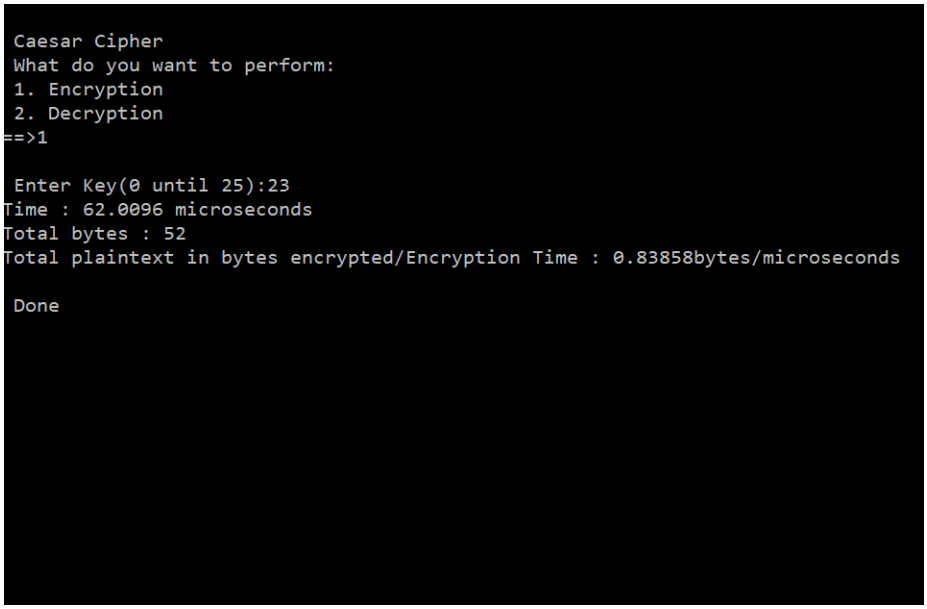

Figure 3.1 Results of Caeser Cipher Encryption

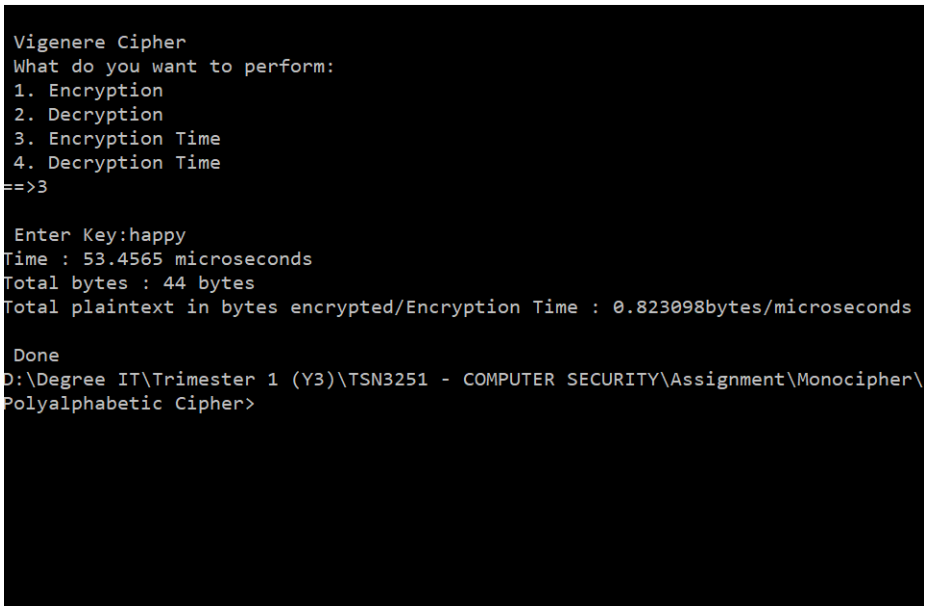

Figure 3.2 Results of Vigenere Cipher Encryption

The comparison results show that caeser cipher and vigenere cipher has the most same result of time in encryption due to error of vigenere cipher using the string to declare the spacing and special characters like (,.?”).

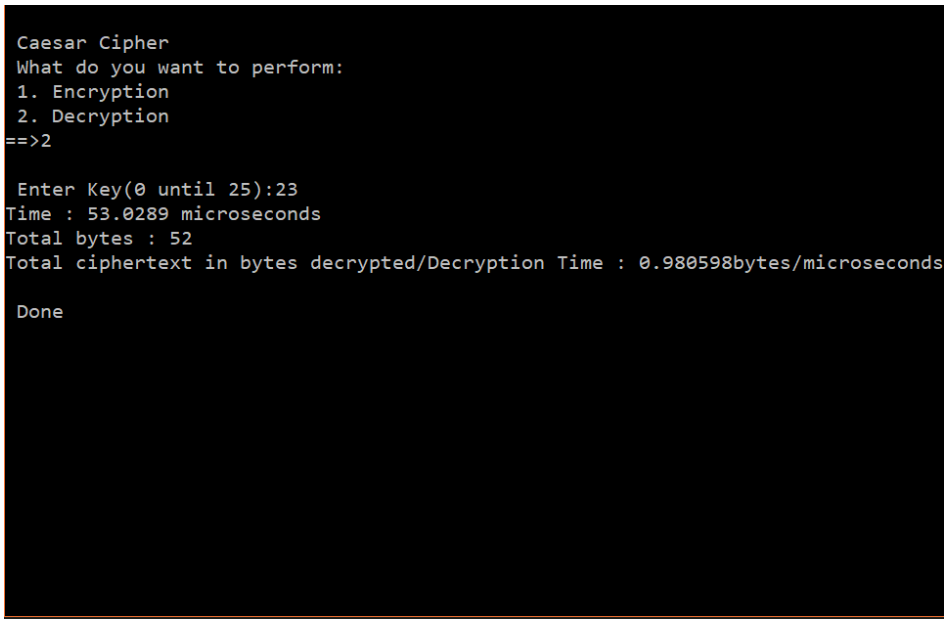

Figure 3.3 Results of Caeser Cipher Decryption

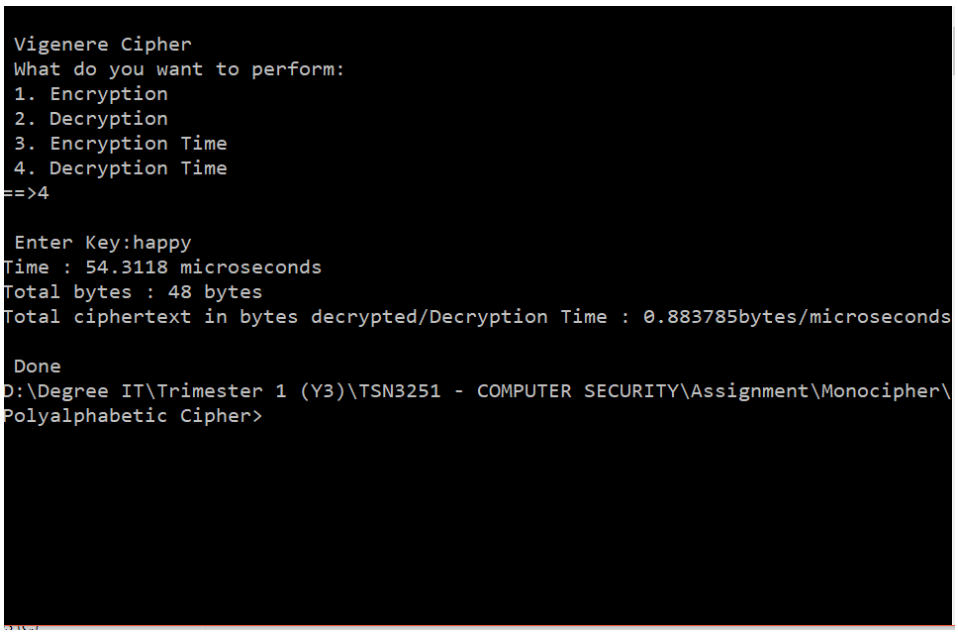

Figure 3.4 Results of Vigenere Cipher Decryption

The comparison results show that caeser cipher and vigenere cipher has the most same result of time in decryption due to error of vigenere cipher using the string to declare the spacing and special characters like („.?”).

Discussion with critical comments

From what we have researched so far, there is a very good reason why we stopped using classical cipher like Caesar Cipher and Vigenere Cipher. Caesar Cipher is cipher that easy break by brute-force attack because the key domain of the Caesar Cipher is 26 only, so brute-force attack can break cipher without know the key by trying all the possibilities of 25 keys from 0 to 25. Vigenere Cipher is easy to break if we find out the length of the key by using Kasiski. Time complexity of the Vigenere Cipher is O(n), so the time needed to break the Vigenere Cipher is very fast. Attacks have been published but none are computationally feasible as of now. To recover the key for AES-128 using biclique attacks, it requires a computational complexity of $2^{126.1}$, $2^{189.7}$ for AES-192 and $2^{126.1}$ for AES-256. So, it is not rational to use classical symmetric cipher instead of modern symmetric cipher.

Conclusion

Through this assignment, we have understood about the concept of cryptography. We have learned a few classical and modern symmetric ciphers such as Caesar Cipher, Vigenère Cipher and Advanced Encryption Standard. Together with mode of operation such as Cipher Feedback mode. We also successfully implemented those ciphers by writing programs.

References

1. Butler W.Lampson (2004). Computer Security in the Real World. Retrieved from web.mit.edu/6.826/www/notes/HO31.pdf
2. Margaret Rouse (2014). Advanced Encryption Standard. Retrieved from http://searchsecurity.techtarget.com/definition/Advanced-Encryption-Standard
3. Federal Information Processing Standards Publication 197. Announcing the Advanced Encryption Standard (AES). Retrieved from http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf
4. Daniel Rodriguez-Clark (2013). Polyalphabetic Substitution Ciphers. Retrieved from http://crypto.interactive-maths.com/polyalphabetic-substitution-ciphers.html
5. Chris Kowalczyk (2013). Caesar Cipher. Retrieved from http://www.crypto-it.net/eng/simple/caesar-cipher.html?tab=0
6. Chris Kowalczyk (2013). Vigenere Cipher. Retrieved from http://www.crypto-it.net/eng/simple/vigenere-cipher.html?tab=1

Appendices

-Source code with comments

-Manual describing the algorithm and running procedure with sample outputs

-Presentation slides