Cryptography and Key Management (Chapter 3, BCIS)

3.1 Introduction

Key terms:

cryptanalysis*: The process of obtaining the plaintext message from a ciphertext message without knowing the keys used to perform the encryption.*

cryptography*: The process of making and using codes to secure information.*

cryptology*: The field of science that encompasses cryptography and cryptanalysis.*

Algorithm*: The mathematical formula or method used to convert an unencrypted message into an encrypted message. This sometimes refers to the programs that enable the cryptographic processes.*

Bit stream cipher*: An encryption method that involves converting plaintext to ciphertext one bit at a time.*

Block cipher*: An encryption method that involves dividing the plaintext into blocks or sets of bits and then converting the plaintext to ciphertext one block at a time.*

Cipher*: When used as a verb, the transformation of the individual components (characters, bytes, or bits) of an unencrypted message into encrypted components or vice versa (see decipher and encipher); when used as a noun, the process of encryption or the algorithm used in encryption, and a term synonymous with cryptosystem.*

Ciphertext or cryptogram*: The unintelligible encrypted or encoded message resulting from an encryption.*

Code*: The process of converting components (words or phrases) of an unencrypted message into encrypted components.*

Decipher/Decryption*: The process of converting an encoded or enciphered message (ciphertext) back to its original readable form (plaintext). Also referred to as deciphering.*

Encipher/Encryption*: The process of converting an original message (plaintext) into a form that cannot be used by unauthorized individuals (ciphertext). Also referred to as enciphering.*

Key or cryptovariable*: The information used in conjunction with the algorithm to create the ciphertext from the plaintext; it can be a series of bits used in a mathematical algorithm or the knowledge of how to manipulate the plaintext. Sometimes called a cryptovariable.*

Keyspace*: The entire range of values that can be used to construct an individual key.*

Link encryption*: A series of encryptions and decryptions between a number of systems, wherein each system in a network decrypts the message sent to it and then reencrypts the message using different keys and sends it to the next neighbor. This process continues until the message reaches the final destination.*

Plaintext or cleartext*: The original unencrypted message that is encrypted and is the result of successful decryption.*

Steganography*: The process of hiding messages; for example, hiding a message within the digital encoding of a picture or graphic so that it is almost impossible to detect that the hidden message even exists.*

Work factor*: The amount of effort (usually expressed in units of time) required to perform cryptanalysis on an encoded message.*

3.2 Cipher Methods

There are two methods of encrypting plaintext: the bit stream method or the block cipher method, as defined in the previous section. In the bit stream method, each bit in the plaintext is transformed into a cipher bit one bit at a time. In the block cipher method, the message is divided into blocks—for example, sets of 8-, 16-, 32-, or 64-bit blocks—and then each block of plaintext bits is transformed into an encrypted block of cipher bits using an algorithm and a key.

3.2.1 Substitution Cipher

A substitution cipher exchanges one value for another—for example, it might exchange a letter in the alphabet with the letter three values to the right, or it might substitute one bit for another bit four places to its left.

Initial alphabet: **ABCDEFGHIJKLMNOPQRSTUVWXYZ**
Encryption alphabet: **DEFGHIJKLMNOPQRSTUVWXYZABC**

The previous example of substitution is based on a single alphabet and thus is known as a **monoalphabetic** substitution.

More advanced substitution ciphers use two or more alphabets, and are referred to as **polyalphabetic** substitutions.

To extend the previous example, consider the following block of text:

Plaintext: ABCDEFGHIJKLMNOPQRSTUVWXYZ
Substitution cipher 1: **D**EFGHIJKLMNOPQRSTUVWXYZABC
Substitution cipher 2: GHIJKLMNOPQRSTUVWXYZABC**D**EF
Substitution cipher 3: **J**KLMNOPQRSTUVWXYZABCDEFGHI
Substitution cipher 4: MNOPQRSTUVWXYZABCDEFGHIJ**K**L

The word “**AXAY**” will be substituted by “**DDJK**”. The rule here is; for the first letter we will be referencing cipher 1, for second letter cipher 2 and so on.

One must understand that ciphering is an art, any rule is valid till both the ciphering and deciphering parties have mutual understanding.

The second example can be further improved to create the **vigenere table**. The Vigenere table is shown below. A ciphering keyword is used along with the table. Lets understand the concept with an example.
Lets take “**NEPAL"** as the keyword and we want to encrypt “**GREEN CITY"**. These two phrases are arranged as follows:

N	E	P	A	L	N	E	P	A
G	R	E	E	N	C	I	T	Y

Now taking two characters from each row we determine the cipher character with the help of the Vigenere table.
For Column “N” and row “G”, we have “T”. Similarly we get the cipher text as

T	V	T	E	Y	P	M	I	Y

Hence, we will get the secret/cipher text as **“TVTEYPMIY”** for “**GREEN CITY**”.

	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
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

Fig: The Vigenere Table

3.2.2 Transposition/Permutation Cipher

This is a kind of cipher in which we **transpose a bit from a predefined position to another predefined position.** A cryptographic operation that involves simply rearranging the values within a block based on an established pattern. Let's say we have a transposition cipher table as below:

Initial position	Final Position
1	5
2	2
3	3
4	6
5	1
6	7
7	4
8	8

Fig: Transposition Table

Now let's see an example: let's encrypt “NEPAL " with this rule. Lets count the position of ‘N’ as 5, ‘E’ as 4 and so on. And finally we are padding three asterisks (or any other visible character for ease) in front to make it the string of 8 characters. Hence, we have “*****NEPAL**”.

According to rule ‘*’ at position 8 goes to position 8, ‘N’ at position 5 to position 1 and so on.

Following the rule we will get ‘****EL*PAN**’. We took an example of a string of length 8 characters, it’s mainly because in digital signal processing 8 bits signal make one byte. So, instead of those English characters we will have bits (0 s and 1s) getting transposed for every byte.

3.2.2 Exclusive OR

A function within Boolean algebra used as an encryption function in which two bits are compared. If the two bits are identical, the result is a binary 0; otherwise, the result is a binary 1. The X-OR operation table is as shown below:

First bit	Second bit	XOR result
0	0	0
0	1	1
1	0	1
1	1	0

Binary representation of some of the English alphabets are as follows:

Letter	ASCII Code	Binary
A	065	01000001
B	066	01000010
C	067	01000011
D	068	01000100
E	069	01000101

Lets say we want to encrypt a message “ABC”. It’s binary representation will be 01000001 01000010 01000011.

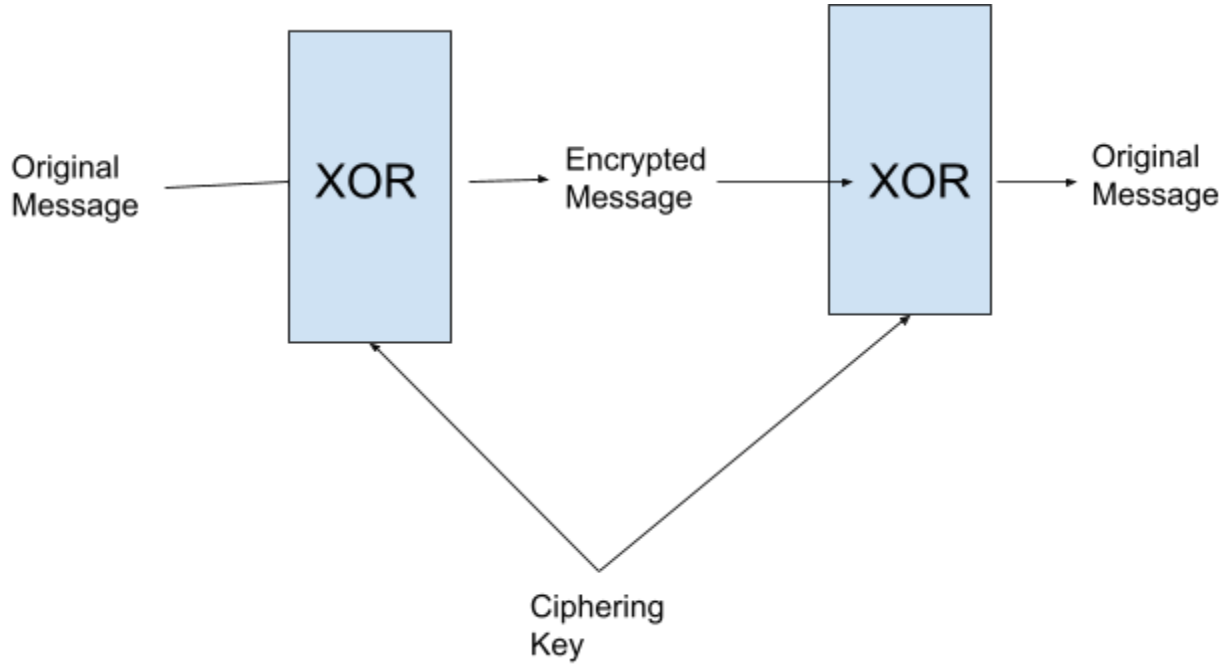
For the purpose of doing XOR we will need a ciphering character, for the sake of study, we will take ‘E’ (you may choose any), whose binary value is 01000101.

We will be using 3 Es since we got 3 characters in our plain text or message.

Message	0	1	0	0	0	0	0	1		0	1	0	0	0	0	1	0		0	1	0	0	0	0	1	1
Ciphering Key	0	1	0	0	0	1	0	1		0	1	0	0	0	1	0	1		0	1	0	0	0	1	0	1
XOR result	0	0	0	0	0	1	0	0		0	0	0	0	0	1	1	1		0	0	0	0	0	1	1	0

Table: Message encryption using XOR

By looking at the table above it is easy to understand that the XOR operation between message and ciphering key can be reversed by XOR operation between the result and the ciphering key which is made clear in the following image.



3.2.3 Vernam Cipher

It is another kind of Ciphering technique in which alphabets are converted into numeric values (for eg A=1, B=2, and so on). Original message is converted into the string of correspondingly assigned numerical values and added with the numerical values of pad string (or simply key string) having equal length. If we want to send “HELLO” and choose “PHONE” as pad text, it will be processed as follows:

Original Msg	H	E	L	L	O
Numerical value	8	5	12	12	15
One-time pad text	P	H	O	N	E
One-time pad text value	16	8	15	14	5
Sum of Msg and pad text	24	13	27	26	20
Modulo (It is necessary when sum is greater than 26)	-	-	1	-	
Ciphertext	X	M	A	Z	T

Hence, ciphertext is “XMAZT”.

Let's decipher it. ‘X’ is 24. Pad value for the first alphabet is 16. Subtract 16 from 28, results 8, whose corresponding alphabet is ‘H’. Similar approach works for all except the third alphabet. For the third alphabet ‘A’ whose value is 1, subtract pad value ‘15’ from 1, results -14. Since it is a negative value we have to add 26 with negative 14. Hence we get 12 which is letter ‘O’.

3.2.4 Book-Based Cipher

3.2.4.1 Book Cipher: In a book cipher, the ciphertext consists of a list of codes representing the page number, line number, and word number of the plaintext word. The algorithm is the mechanical process of looking up the references from the ciphertext and converting each reference to a word by using the ciphertext’s value and the key (the book). For example, from a copy of a particular popular novel, one may send the message 259,19,8; 22,3,8; 375,7,4; 394,17,2. Although almost any book can be used, dictionaries and thesauruses are typically the most popular sources, as they are likely to contain almost any word that might be needed. The recipient of a running key cipher must first know which book is used.

3.2.4.2 Running Key Cipher: In classical cryptography, the running key cipher is a type of polyalphabetic substitution cipher in which a text, typically from a book, is used to provide a very long keystream. Usually, the book to be used would be agreed ahead of time, while the passage to be used would be chosen randomly for each message and secretly indicated somewhere in the message.

The text used is *The C Programming Language* (1978 edition), and the *tabula recta* (refer to the table used in Vigenere Cipher) is the tableau.

The plaintext is **"Flee at once"**.

Page 63, line 1 is selected as the **running key**: **"*errors can occur in several places. A label has…*"**

The running key is then written under the plaintext:

Plaintext	f	l	e	e	a	t	o	n	c	e
Running key	E	R	R	O	R	S	C	A	N	O
Ciphertext	J	C	V	S	R	L	Q	N	P	S

The message is then sent as "JCVSR LQNPS". However, unlike a Vigenère cipher, if the message is extended, the key is not repeated; the key text itself is used as the key. If the message is extended, such as, "Flee at once. We are discovered", then the running key continues as before:

Plaintext	f	l	e	e	a	t	o	n	c	e	w	e	a	r	e	d	i	s	c	o	v	e	r	e	d
Running key	E	R	R	O	R	S	C	A	N	O	C	C	U	R	I	N	S	E	V	E	R	A	L	P	L
Ciphertext	J	C	V	S	R	L	Q	N	P	S	Y	G	U	I	M	Q	A	W	X	S	M	E	C	T	O

To determine where to find the running key, a fake block of five ciphertext characters is subsequently added, with three denoting the page number, and two the line number, using A=0, B=1 etc. to encode digits. Such a block is called an indicator block. The indicator block will be inserted as the second last of each message. (Many other schemes are possible for hiding indicator blocks.) Thus page 63, line 1 encodes as "AGDAB" (06301) where A=0, B=1, and so on.

This yields a final message of "JCVSR LQNPS YGUIM QAWXS **AGDAB** MECTO".

3.2.4.3 Template Cipher: The template cipher or perforated page cipher is not strictly an encryption cipher, but more of an example of steganography. The template cipher involves the use of a hidden message in a book, letter, or other message. The receiver must use a page with a specific number of holes cut into it and place it over the book page or letter to extract the

hidden message. Commonly shown in movies where an inmate sends coded messages from prison, this cipher is both difficult to execute and easy to detect, provided either party is physically searched. The presence of the perforated page is a clear indicator that some form of hidden message communication is occurring.

3.2.5 Hash Functions

Key Terms

hash algorithms: Public functions that create a hash value, also known as a message digest, by converting variable-length messages into a single fixed-length value.
hash functions: Mathematical algorithms that generate a message summary or digest (sometimes called a fingerprint) to confirm message identity and integrity.
hash value: See message digest.
message authentication code (MAC): A key-dependent, one-way hash function that allows only specific recipients (symmetric key holders) to access the message digest.
message digest: A value representing the application of a hash algorithm on a message that is transmitted with the message so it can be compared with the recipient's locally calculated hash of the same message. If both hashes are identical after transmission, the message has arrived without modification. Also known as a hash value.
Secure Hash Standard (SHS): A standard issued by the National Institute of Standards and Technology (NIST) that specifies secure algorithms, such as SHA-1, for computing a condensed representation of a message or data file.

In addition to ciphers, another important encryption technique that is often incorporated into cryptosystems is the hash function. Hash functions are mathematical algorithms used to confirm the identity of a specific message and confirm that the content has not been changed. While they do not create ciphertext, hash functions confirm message identity and integrity, both of which are critical functions in e-commerce.
Message or plain text is converted to hash value also known as **message digest** by the use of hash function. Hashing is a **one way** algorithm. **No method can convert hash value into plain text.** Since hashing is one way it is very **useful in storing passwords**. Or we can say that hash values are the standard for saving passwords in a database.

Features of Hash Functions

The typical features of hash functions are −

- Fixed Length Output (Hash Value)
    - Hash function converts data of arbitrary/any length to a fixed length. This process is often referred to as hashing the data.
    - In general, the hash is much smaller than the input data, hence hash functions are sometimes called compression functions.
    - Since a hash is a smaller representation of a larger data, it is also referred to as a digest.

    - Hash function with n bit output is referred to as an n-bit hash function. Popular hash functions generate values between 160 and 512 bits.
- Efficiency of Operation
    - Generally for any hash function h with input x, computation of h(x) is a fast operation.
    - Computationally hash functions are much faster than a symmetric encryption.

Properties of Hash Functions

In order to be an effective cryptographic tool, the hash function is desired to possess following properties −

- Pre-Image Resistance
    - This property means that it should be **computationally hard to reverse** a hash function.
    - In other words, if a hash function h produces a hash value z, then it should be a difficult process to find any input value x that hashes to z.
    - **This property protects against an attacker who only has a hash value and is trying to find the input.**
- Second Pre-Image Resistance
    - This property means given an input and its hash, it should be **hard to find a different input with the same hash**.
    - In other words, if a hash function h for an input **x produces hash value h(x)**, then it should be **difficult to find any other input value y such that h(y) = h(x)**.
    - This property of the hash function protects against an attacker who has an input value and its hash, and wants to substitute a different value as legitimate value in place of original input value.
- Collision Resistance
    - This property means it should be hard to find two different inputs of any length that result in the same hash. This property is also referred to as a collision free hash function.
    - In other words, for a hash function h, it is hard to find any two different inputs x and y such that h(x) = h(y).
    - Since, a hash function is a compressing function with fixed hash length, it is impossible for a hash function not to have collisions. This property of collision free only confirms that these collisions should be hard to find.
    - This property makes it very difficult for an attacker to find two input values with the same hash.

- **Also, if a hash function is collision-resistant then it is second pre-image resistant.**
- One must understand that there is a slight difference between second pre-image resistance and collision resistance. For the case prior, a message x is available to the attacker for which s/he has to find the y for which hashes are the same. In the latter case, attacker can take any two messages (x,y) for which the hashes are the same.

In general hashes are very tough, time and resource consuming to reverse. Yet, rainbow table attack can be catastrophic if someone has access to the password table. Refer to the previous chapter for rainbow table attack.

3.3 Symmetric Cryptography

Symmetric cryptography, known also as secret key cryptography, is the use of a single shared secret to share encrypted data between parties. Ciphers in this category are called symmetric because you use the same key to encrypt and to decrypt the data. In simple terms, the sender encrypts data using a password, and the recipient must know that password to access the data.

Symmetric encryption is a two-way process. With a block of plaintext and a given key, symmetric ciphers will always produce the same ciphertext. Likewise, using that same key on that block of ciphertext will always produce the original plaintext. Symmetric encryption is useful for protecting data between parties with an established shared key and is also frequently used to store confidential data. For example, ASP.NET uses 3DES to encrypt cookie data for a forms authentication ticket.

There are two types of symmetric encryption algorithms:

1. **Block algorithms**. Set lengths of bits are encrypted in blocks of electronic data with the use of a specific secret key. As the data is being encrypted, the system holds the data in its memory as it waits for complete blocks.
2. **Stream algorithms**. Data is encrypted as it streams instead of being retained in the system's memory.

Some examples of **symmetric** encryption algorithms include:

AES (Advanced Encryption Standard)
DES (Data Encryption Standard)
IDEA (International Data Encryption Algorithm)
Blowfish (Drop-in replacement for DES or IDEA)
RC4 (Rivest Cipher 4)
RC5 (Rivest Cipher 5)
RC6 (Rivest Cipher 6)

AES, DES, IDEA, Blowfish, RC5 and RC6 are block ciphers. RC4 is a stream cipher.

Some examples of where symmetric cryptography is used are:

Payment applications, such as card transactions where PII needs to be protected to prevent identity theft or fraudulent charges
Validations to confirm that the sender of a message is who he claims to be
Random number generation or hashing

3.3.1 DES (Data Encryption Standard)

The DES (Data Encryption Standard) algorithm is a symmetric-key block cipher created in the early 1970s by an IBM team and adopted by the National Institute of Standards and Technology (NIST). **The algorithm takes the plain text in 64-bit blocks and converts them into ciphertext using 16 different 48-bit keys which are generated from a 56 bits key (Note: Actually 64 bits key is used; but every 8th bit in a byte is used as parity bit, hence, only 56 bits will take part in the encryption process)**.

Since it's a symmetric-key algorithm, it employs the same key in both encrypting and decrypting the data. If it were an asymmetrical algorithm, it would use different keys for encryption and decryption.

DES uses 16 rounds of the Feistel structure, using a different key for each round.

DES became the approved federal encryption standard in November 1976 and was subsequently reaffirmed as the standard in 1983, 1988, and 1999.

DES's dominance came to an end in 2002, when the Advanced Encryption Standard (AES) replaced the DES encryption algorithm as the accepted standard, following a public competition to find a replacement. The NIST officially withdrew FIPS 46-3 (the 1999 reaffirmation) in May 2005, although Triple DES (3DES), remains approved for sensitive government information.

Note:

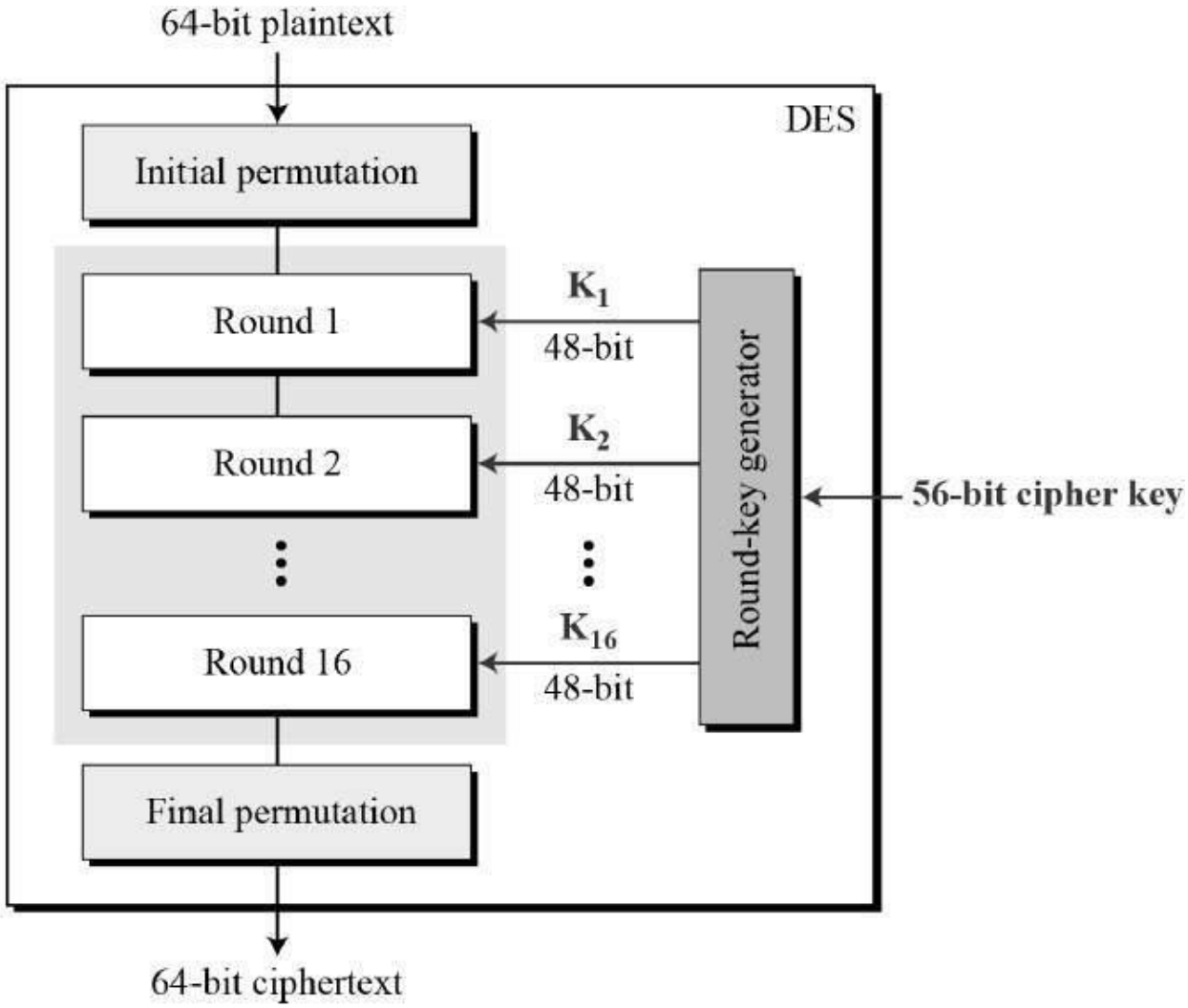
The **NIST had to replace the DES algorithm because its 56-bit key lengths were too small**, considering the increased processing power of newer computers. Encryption strength is related to the key size, and DES found itself a victim of the ongoing technological advances in computing. It reached a point where 56-bit was no longer good enough to handle the new challenges to encryption.

Note that just because DES is no longer the NIST federal standard, it doesn't mean that it's no longer in use. **Triple DES is still used today**, but it's considered a legacy encryption algorithm. Note that NIST plans to disallow all forms of Triple-DES from 2024 onward.

(The following content is totally referenced from tutorialspoint.com for educational purpose.)

The Data Encryption Standard (DES) is a symmetric-key block cipher published by the National Institute of Standards and Technology (NIST).

DES is an implementation of a Feistel Cipher. It uses a 16 round Feistel structure. The block size is 64-bit. Though, key length is 64-bit, DES has an effective key length of 56 bits, since 8 of the 64 bits of the key are not used by the encryption algorithm (function as check bits only). General Structure of DES is depicted in the following illustration −

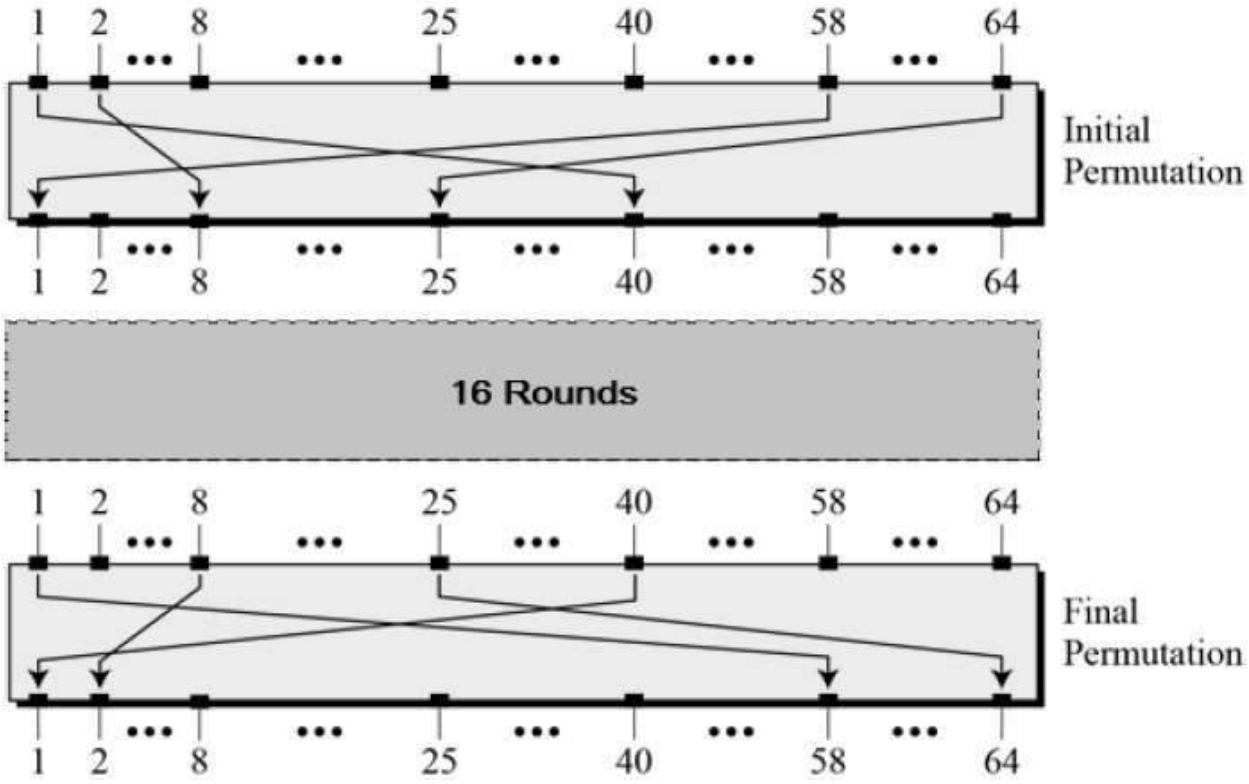


Since DES is based on the Feistel Cipher, all that is required to specify DES is −

- Round function
- Key schedule
- Any additional processing − Initial and final permutation

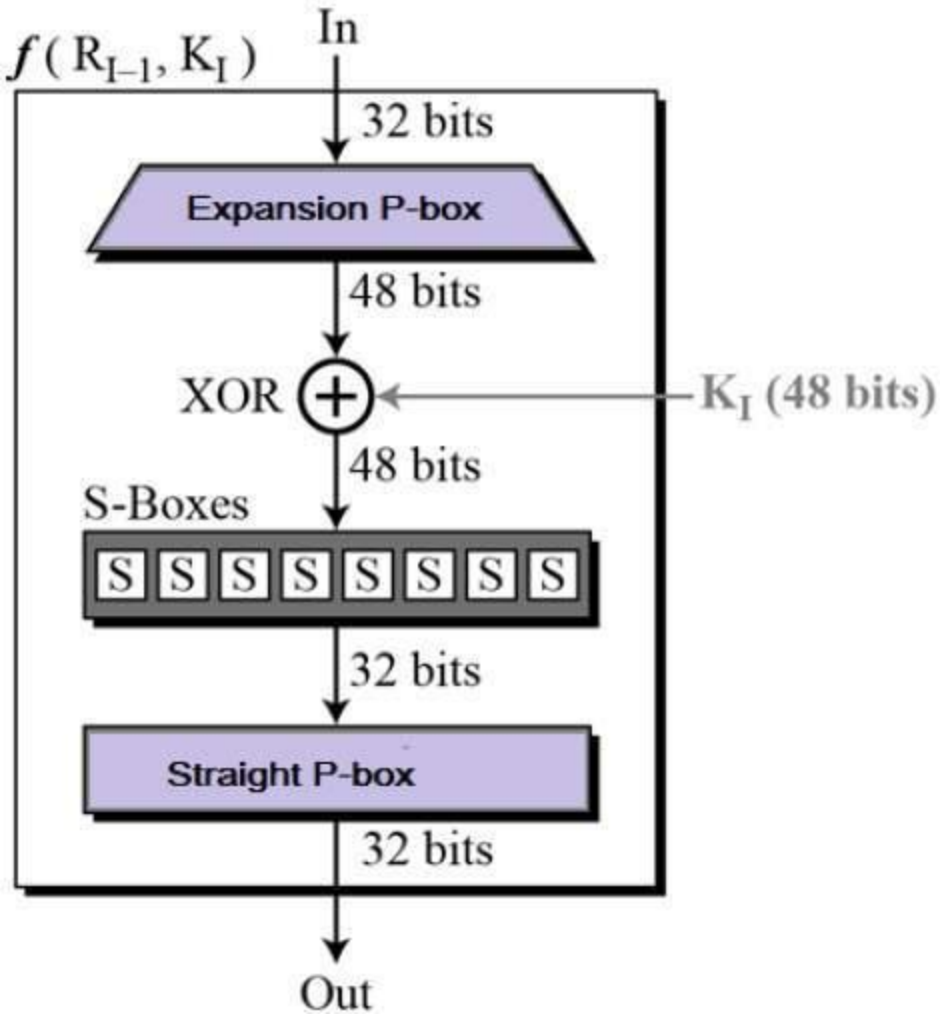
Initial and Final Permutation

The initial and final permutations are straight Permutation boxes (P-boxes) that are inverses of each other. They have no cryptography significance in DES. The initial and final permutations are shown as follows −

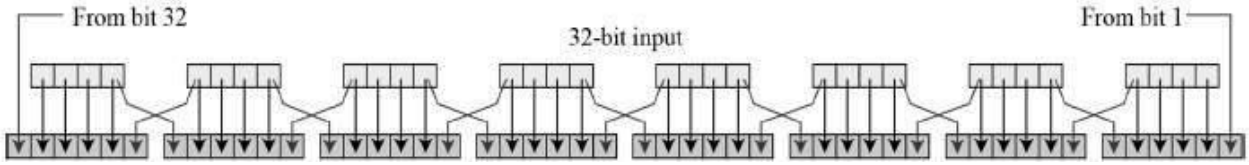


Round Function

The heart of this cipher is the DES function, *f*. The DES function applies a 48-bit key to the rightmost 32 bits to produce a 32-bit output.



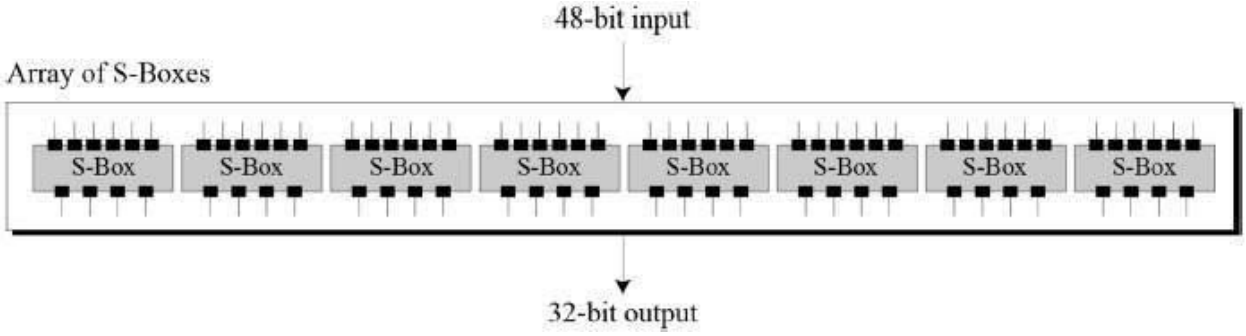
- Expansion Permutation Box − Since right input is 32-bit and round key is a 48-bit, we first need to expand right input to 48 bits. Permutation logic is graphically depicted in the following illustration −



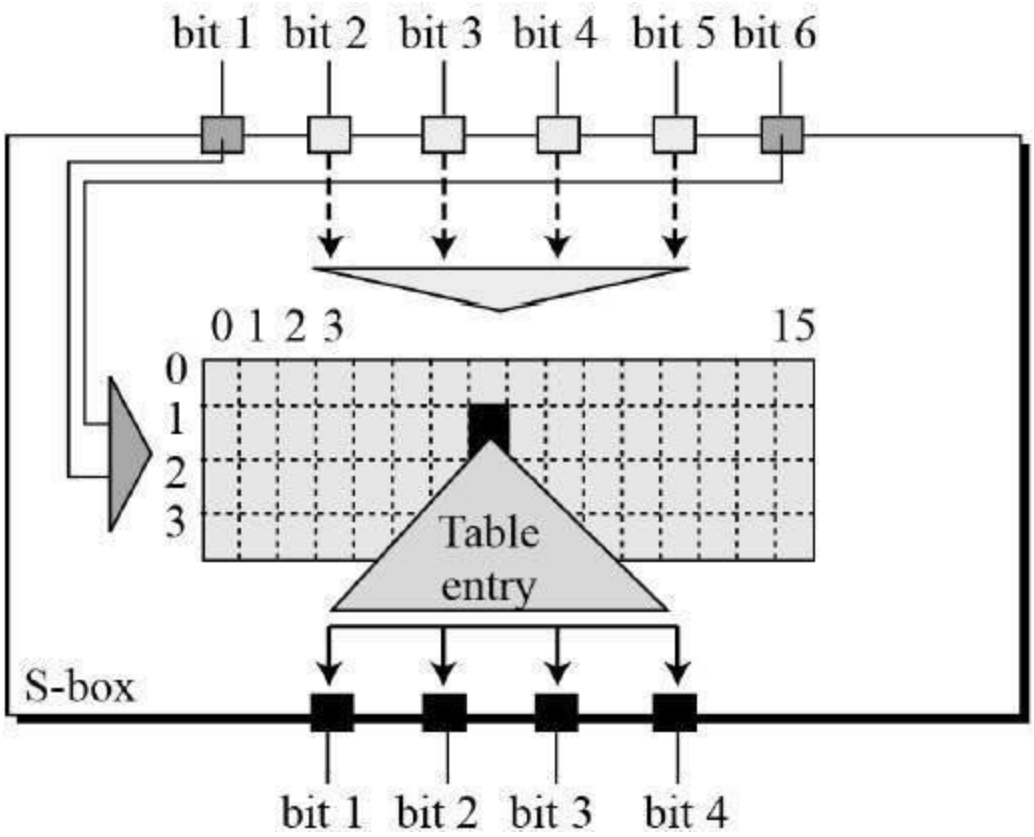
- The graphically depicted permutation logic is generally described as table in DES specification illustrated as shown −

32	01	02	03	04	05
04	05	06	07	08	09
08	09	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	31	31	32	01

- XOR (Whitener). − After the expansion permutation, DES does XOR operation on the expanded right section and the round key. The round key is used only in this operation.
- Substitution Boxes. − The S-boxes carry out the real mixing (confusion). DES uses 8 S-boxes, each with a 6-bit input and a 4-bit output. Refer the following illustration −



- The S-box rule is illustrated below −

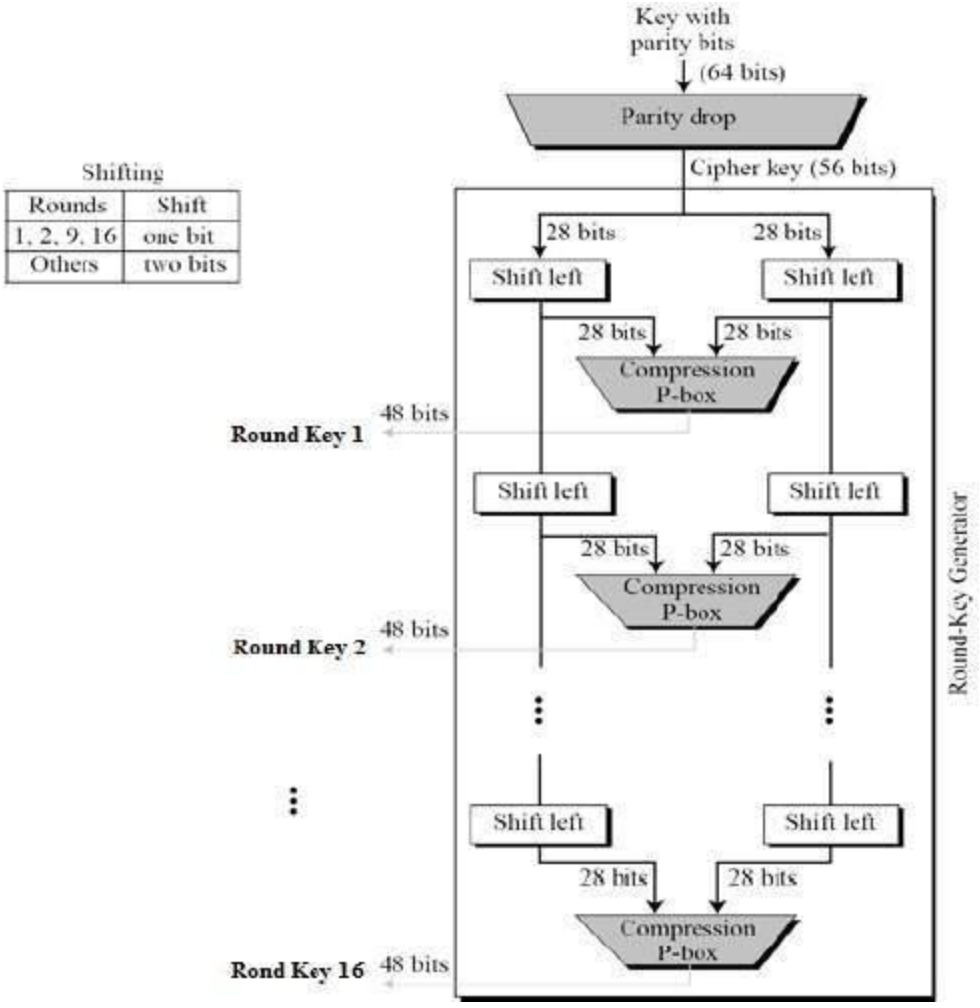


- There are a total of eight S-box tables. The output of all eight s-boxes is then combined in to 32 bit section.
- Straight Permutation − The 32 bit output of S-boxes is then subjected to the straight permutation with rule shown in the following illustration:

16	07	20	21	29	12	28	17
01	15	23	26	05	18	31	10
02	08	24	14	32	27	03	09
19	13	30	06	22	11	04	25

Key Generation

The round-key generator creates sixteen 48-bit keys out of a 56-bit cipher key. The process of key generation is depicted in the following illustration −



The logic for Parity drop, shifting, and Compression P-box is given in the DES description.

DES Analysis

The DES satisfies both the desired properties of block cipher. These two properties make cipher very strong.

- Avalanche effect − A small change in plaintext results in the very great change in the ciphertext.
- Completeness − Each bit of ciphertext depends on many bits of plaintext.

During the last few years, cryptanalysis have found some weaknesses in DES when keys selected are weak keys. These keys shall be avoided.

DES has proved to be a very well designed block cipher. There have been no significant cryptanalytic attacks on DES other than exhaustive key search.1

3.3.2 Advanced Encryption Standard (AES)

For detail study please refer to this link:
https://cybernews.com/resources/what-is-aes-encryption/

3.3.2 Difference Between DES and AES

Both AES and DES Ciphers are symmetric ciphers.

Following are the important differences between AES and DES Ciphers.

S. No.	Key	AES	DES
1	Definition	AES stands for Advanced Encryption Standard.	DES stands for Data Encryption Standard.
2	Key Length	Key length varies from 128 bits, 192 bits to 256 bits.	Key length is 56 bits.
3	Rounds of Operations	Rounds per key length: 128 bits - 10; 192 bits - 12; 256 bits - 14.	16 rounds of identical operations.
4	Network	AES structure is based on substitution-permutation network.	DES structure is based on feistel network.

5	Security	AES is de-facto world standard and is more secure than DES.	DES is weak and 3DES(Triple DES) is more secure than DES.
6	Rounds or steps of operations	Byte substitution, Shift Row, Mix Column and Key Addition.	Expansion, XOR operation with round key, Substitution and Permutation.
7	Size	AES can encrypt in blocks of 128 bits of plain text.	DES can encrypt in blocks of 64 bits of plain text.
8	Derived from	AES derives from Square cipher.	DES derives from the Lucifer cipher.
9	Desiged By	AES was designed by Vincent Rijmen and Joan Daemen.	DES was designed by IBM.
10	Known attacks	No known attack.	Brute-force, Linear crypt-analysis and Differential crypt-analysis.

References:

1. Principles of Information Security by Michael E. Whitman & Herbert J. Mattord
2. https://cybernews.com
3. https://www.techtarget.com/
4. https://www.simplilearn.com/
5. https://www.studypool.com/
6. https://usemynotes.com