Cryptography and Network Security

Unit-I Syllabus

Security goals, security attacks, security services and Mechanisms, Techniques, Mathematics of Cryptography

Chapter1 Introduction

Types of Security

1. Computer Security: Measures which ensures confidentiality, integrity and availability of the information system includes H/W, S/W, firmware etc.,
2. Network Security: It is the protection of access of files in a computer networks against hacking, misuse and unauthorized changes to the system.
3. Cyber Security: It is the protection of files transferred in network of networks.

General Terminology

1. Vulnerability: A defect or weakness in the feasibility, design, implementation, operation of a system
2. Threat: An advisory who is capable of and motivated to exploit the vulnerability
3. Attack: The use or exploitation of vulnerability. A bad guy may attack the system where as a good guy attacks the problem.
4. Attacker: The person or a process who initiates the attack.
5. Target: The person, company or a system which is vulnerable and effected.
6. Defender: The person or a process which prevents the attacks
7. Compromise: The successful exploitation of the target by the attacker

1.1 Security Goals

The three security goals defined in the CIA Triad are

Confidentiality: The first goal of Network Security is "Confidentiality". The function of "Confidentiality" is in protecting precious business data (in storage or in motion) from unauthorized persons. Confidentiality part of Network Security makes sure that the data is available only to intended and authorized persons. Access to business data should be only for those individuals who are permitted to use that data. Example in industry hiding some information from competitors is crucial for the operation of organization, in military concealment of sensitive information is a major concern. In banking customer accounts, credit cards etc are major concern.

Confidentiality not only applies to the storage of information but also applies to the transmission of data. when we send a piece of information from a remote computer, we need to conceal it during transmission.

Integrity: The second goal of Network Security is "Integrity". Integrity means changes need to be done only by authorized entities and through authorized mechanisms. Integrity aims at maintaining and assuring the accuracy and consistency of data. The function of Integrity is to make sure that the date is accurate and reliable and is not changed by unauthorized persons or hackers. The data received by the recipient must be exactly same as the data sent from the sender, without change in even single bit of data. Integrity violation is not necessarily the result of a malicious act, an interruption in the system, such as a power surge may also create unwanted changes in some information

Availability: The third goal of network security is "Availability". The function of "Availability" in Network Security is to make sure that the Data, Network Resources or Network Services are continuously available to the legitimate users, whenever they require it. Information is useless if it is not available. Information needs to be constantly changed, which means it must be accessible to authorized entities. The unavailability of information is just as harmful for an organization as the lack of confidentiality or integrity. Imagine what would happen to a bank if the customers could not access their accounts for verification.

1.2 Security Attacks

Security Attack are of two types

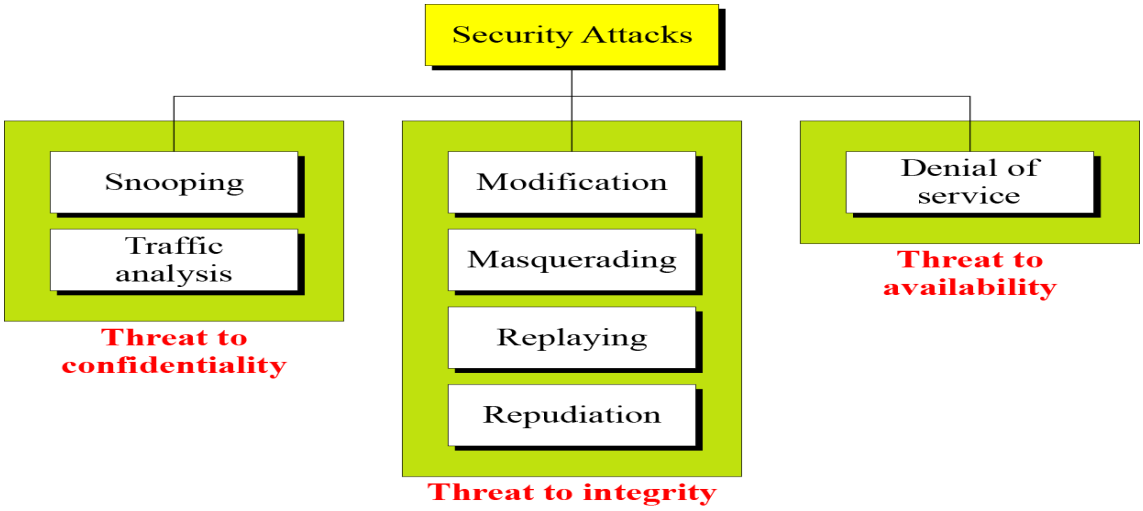
Passive Attacks

In this type of attacks, the attacker goal is just to obtain information. The attacker does not modify the data or harm the system. The system works normally. However the attack may harm the sender or receiver of the message. Attacks threatening the confidentiality come under this category. The revealing of the information may not harm the sender or the receiver of the message, there fore it is difficult to find the attack. They can be prevented by encipherment of the data.

Active Attacks

An active attack may change the data or harm the system. Attacks threatening the integrity and availability comes under this category. These attacks are normally detected easily than to prevent, because an attacker can launch them in a variety of ways.

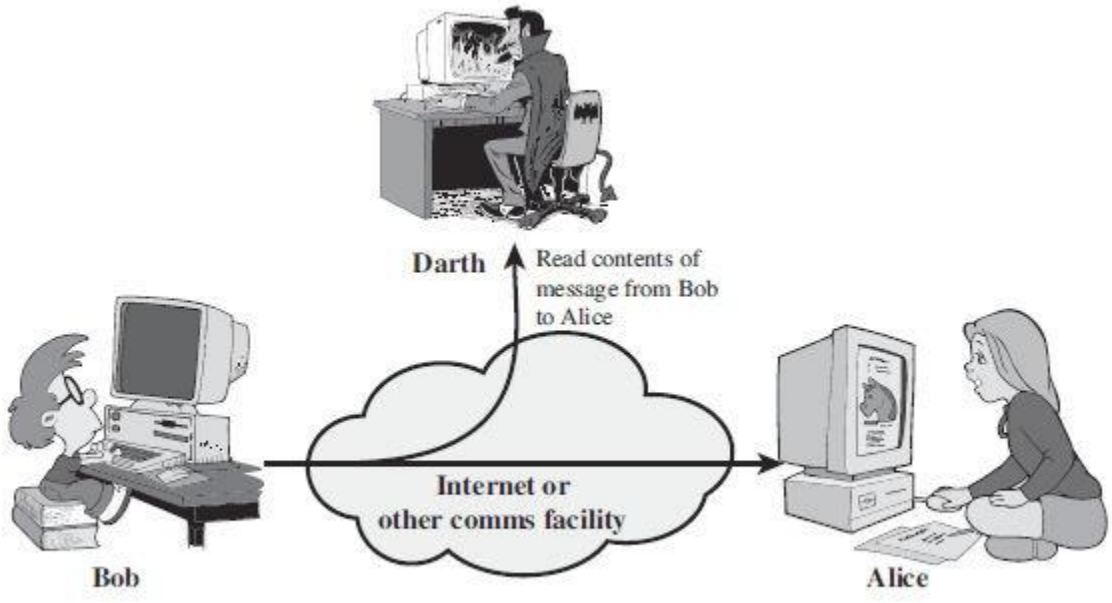
The attacks are classified in to following categories



1.2.1 Threat to confidentiality

1. Snooping or Release of message contents:

Snooping refers to unauthorized access to or interception of data.**The release of message contents is easily understood. A telephone conversation, an electronic mail message, and a transferred file may contain sensitive or confidential information. We would like to prevent an opponent from learning the contents of these transmissions.**

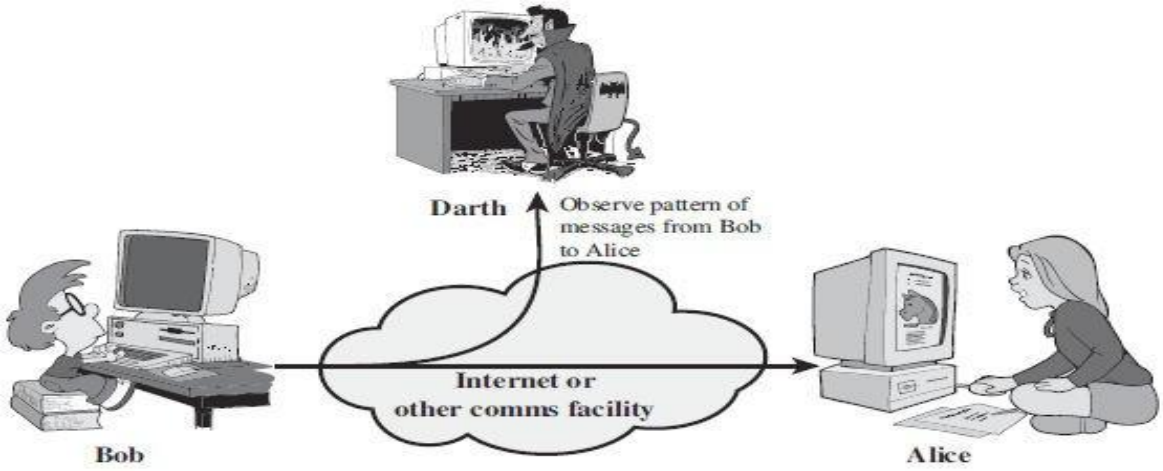


(a) Release of message contents

2. Traffic Analysis or Subtler

Snooping refers to unauthorized access to or interception of data.

A second type of passive attack, traffic analysis, is subtler . Suppose that we had a way of masking the contents of messages or other information traffic so that opponents, even if they captured the message, could not extract the information from the message. The common technique for masking contents is encryption. If we had encryption protection in place, an opponent might still be able to observe the pattern of these messages. The opponent could determine the location and identity of communicating hosts and could observe the frequency and length of messages being exchanged. This information might be useful in guessing the nature of the communication that was taking place. Passive attacks are very difficult to detect, because they do not involve any alteration of the data. Typically, the message traffic is sent and received in an apparently normal fashion, and neither the sender nor receiver is aware that a third party has read the messages or observed the traffic pattern. However, it is feasible to prevent the success of these attacks, usually by means of encryption. Thus, the emphasis in dealing with passive attacks is on prevention rather than detection.



(b) Traffic analysis

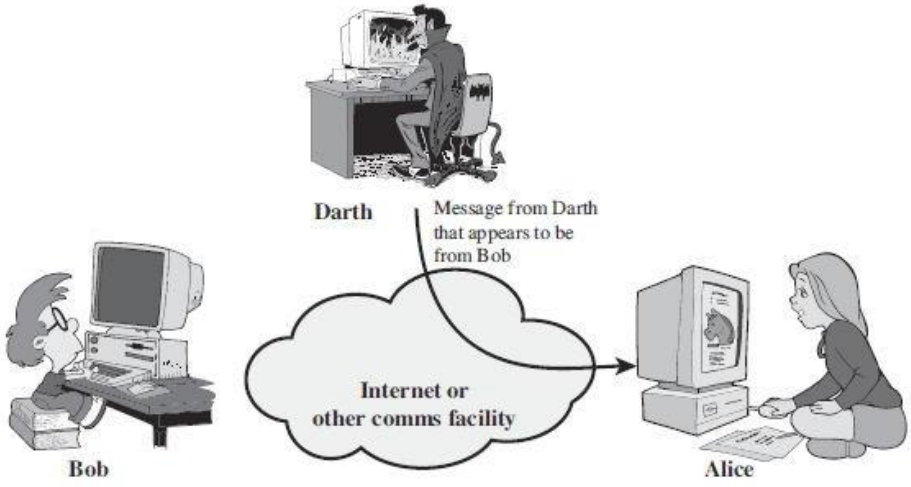
Attacks threatening Integrity

Active attacks involve some modification of the data stream or the creation of a false stream and can be subdivided into four categories:

- masquerade
- replay
- modification of messages
- repudiation

1. Masquerade

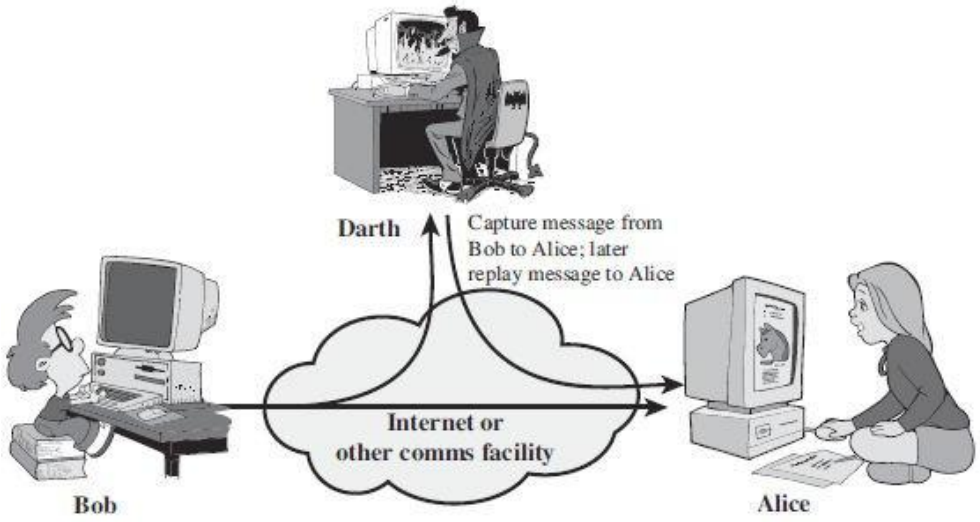
A masquerade or spoofing takes place when one entity pretends to be a different entity. A masquerade attack usually includes one of the other forms of active attack. For example, authentication sequences can be captured and replayed after a valid authentication sequence has taken place, thus enabling an authorized entity with few privileges to obtain extra privileges by impersonating an entity that has those privileges. The attacker might steal the bank card and PIN of a bank customer and pretend that she is that customer. Some times he may behave as a receiver. Example a user tries to contact a bank, but another site pretends that it is the bank and obtains some information from the user.



(a) Masquerade

2. Replay:

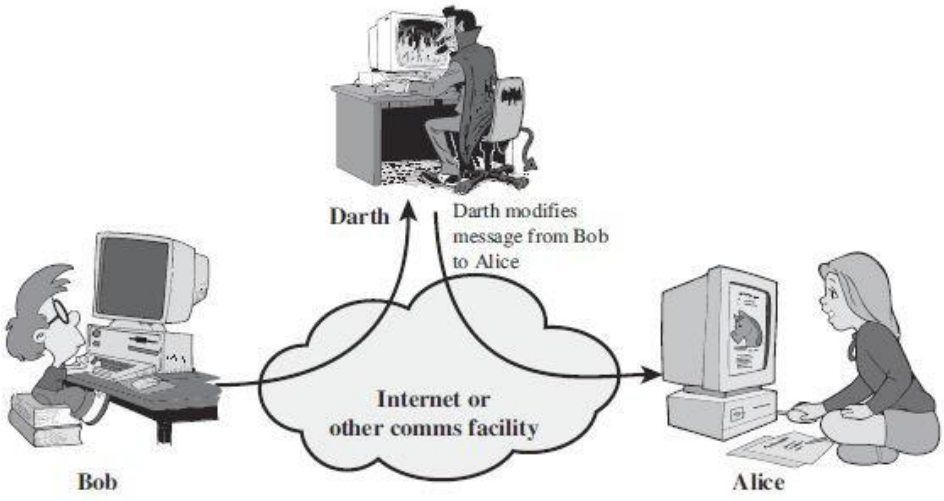
Replay involves the passive capture of a data unit and its subsequent retransmission to produce an unauthorized effect . For example a person sends a request to her bank to ask for payment to the attacker, who has done a job for her. The attacker intercepts the message and sends it again to receive another payment from the bank.



(b) Replay

3. Modification of Messages

Modification of messages simply means that some portion of a legitimate message is altered, or that messages are delayed , deleted or reordered, to produce an unauthorized effect. For example, a customer sends a message to do some transaction. The attacker intercepts the message and changes the type of transaction to benefit herself.



(c) Modification of messages

4. **Repudiation**:

The type of the attack is different from others because it is performed by one of the two parties in the communication, the sender or receiver. Repudiation means that sender of the message might later deny that she has sent the message; the receiver of the message might later deny that he has received the

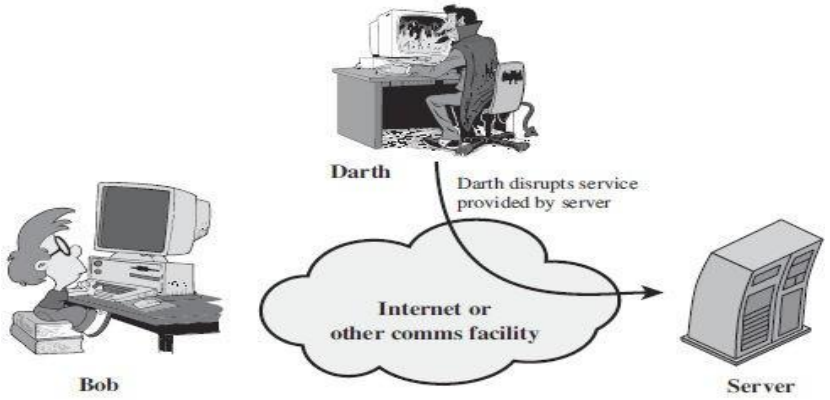
message.

An example of denial by the sender would be a bank customer asking the bank to send money to a third party but later denying hat he has made such a request. An example of denial by the receiver could occur when a person buys a product from a manufacturer and pays for it electronically, but the manufacturer later denies having received the payment and asks to pay once again.

Attacks Threatening Availability

Denial of Service

The denial of service prevents or inhibits the normal use or management of communications facilities. This attack may have a specific target; for example, an entity may suppress all messages directed to a particular destination. It may slow down or totally interrupt the service of a system. The attacker might send so many dummy requests to the server that the server crashes because of the heavy load. The attacker might intercept and delete a servers response to a client, making the client to believe that the server is not responding. The attacker may also intercept request from the clients causing the clients to send requests multiple times and overload the system.



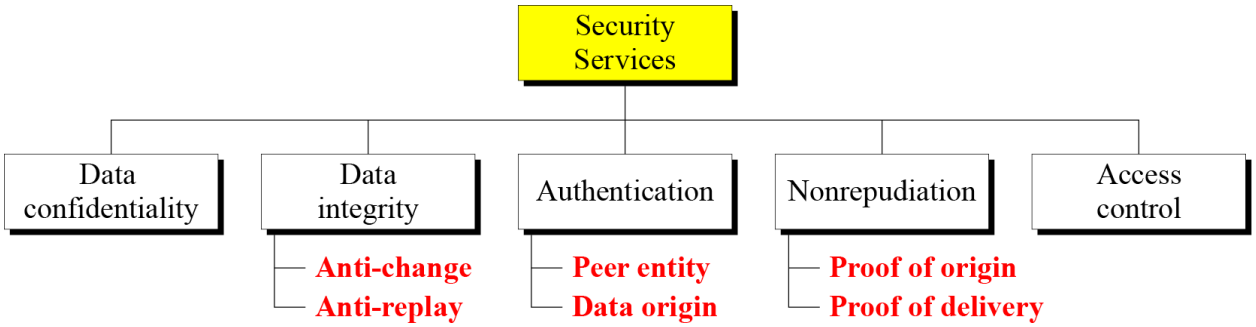
(d) Denial of service

Passive versus active attacks

Attacks	*Passive/Active*	*Threatening*
Snooping Traffic analysis	Passive	Confidentiality
Modification Masquerading Replaying Repudiation	Active	Integrity
Denial of service	Active	Availability

1.3 Security Services

Security services include data confidentiality, data integrity, authentication, nonrepudiation, and access control.



1.Data confidentiality

• It means that the content of a message when transmitted across a network must remain confidential, *i.e.* only the intended receiver and no one else should be able to read the message.

• The users therefore, want to encrypt the message they send so that an eavesdropper on the network will not be able to read the contents of the message.

It is designed to prevent snooping and traffic analysis attacks.

2. Data Integrity

• It means the data must reach the destination without any adulteration *i.e. modification, insertion, deletion and replaying the message*.

• There must be no changes during transmission, neither accidentally nor maliciously.

• Integrity of a message is ensured by attaching a checksum to the message.

• The algorithm for generating the checksum ensures that an intruder cannot alter the checksum or the message.

3. Authentication

• **In** message authentication the receiver needs to be .sure of the sender's identity *i.e.* the receiver has to make sure that the actual sender is the same as claimed to be.

• There are different methods to check the genuineness of the sender :

1. The two parties share a common secret code word. A party is required to show the secret code word to the other for authentication.

2. Authentication can be done by sending digital signature.

3. A trusted third party verifies the authenticity. One such way is to use digital certificates issued by a recognized certification authority.

4. Message non-repudiation

• Non-repudiation means that a sender must not be able to deny sending a message that it actually sent.

• The burden of proof falls on the receiver.

• Non-reproduction is not only in respect of the ownership of the message; the receiver must prove that the contents of the message are also the same as the sender sent.

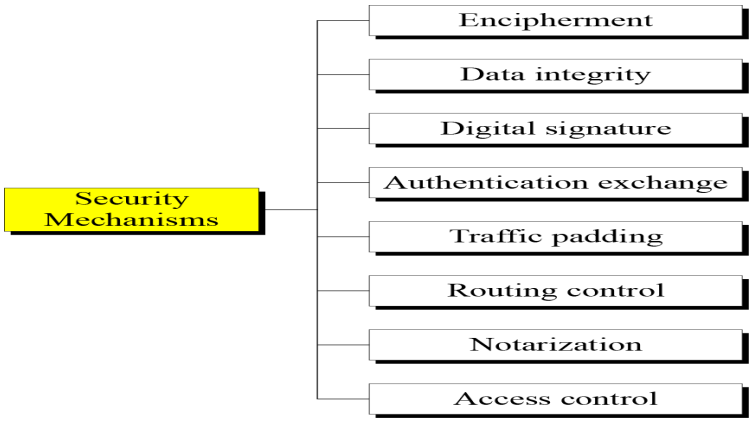
• Non-repudiation is achieved by authentication and integrity mechanisms.

5. Access Control

• In entity authentication (or user identification) the entity or user is verified prior to access to the system resources for reading, writing and modifying

1.4Security Mechanisms

A security mechanism is any process (or a device incorporating such a process) that is designed to detect,prevent,or recover from a security attack. Examples of mechanisms are encryption algorithms, digital signatures, and authentication protocols.



The various security mechanisms to provide security are as follows-

1. Encipherment:

This is hiding or covering of data which provides confidentiality. It is also used to complement other mechanisms to provide other services. Cryptography and Steganography are used for enciphering

2. Digital Integrity:

The data integrity mechanism appends to the data a short check value that has been created by a specific process from the data itself. Data integrity is preserved by comparing check value received to the check value generated.

3. Digital Signature:

A digital signature is a means by which the sender can electronically sign the data and the receiver can electronically verify the signature. The sender uses a process that involves showing that she owns a private key related to the public key that she announced publicly. The receiver uses the senders public key to prove that the message is indeed signed by the sender who claims to have sent the message.

4. Authentication Exchange:

In this two entities exchange some messages to prove their identity to each other.

5. Traffic Padding:

Traffic padding means inserting some bogus data into the data traffic to thwart the adversary's attempt to use the traffic analysis.

6. Routing Control:

Routing control means selecting and continuously changing different available routes between sender and receiver to prevent the opponent from eavesdropping on a particular route.

7. Notarization:

Notarization means selecting a third trusted party to control the communication between two entities. The receiver can involve a trusted third party to store the sender request in order to prevent the sender from later denying that she has made a request.

8. Access Control:

Access control used methods to prove that a user has access right to the data or resources owned by a system. Examples of proofs are passwords and PINs.

Relation between security services and mechanisms

Security Service	*Security Mechanism*
Data confidentiality	Encipherment and routing control
Data integrity	Encipherment, digital signature, data integrity
Authentication	Encipherment, digital signature, authentication exchanges
Nonrepudiation	Digital signature, data integrity, and notarization
Access control	Access control mechanism

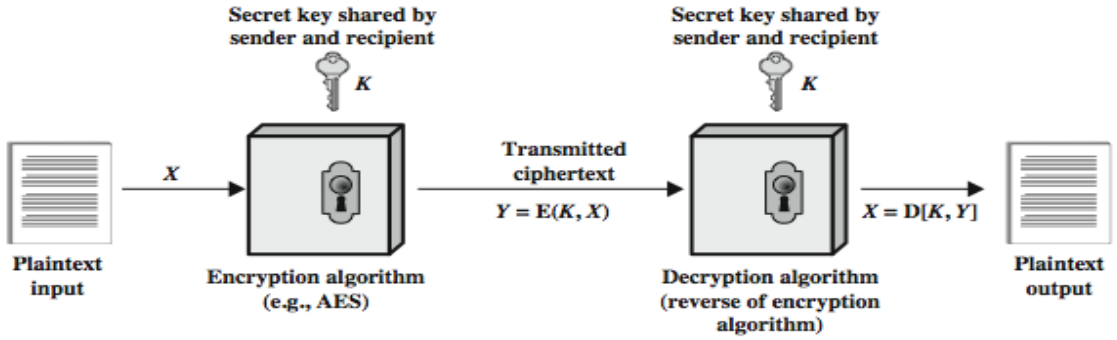
1.4 Techniques for Implementation of Security Goals

Cryptography

Cryptography, a word with Greek origins, means "secret writing". It refers to the encryption and decryption of messages using secret keys. It involves three distinct mechanisms

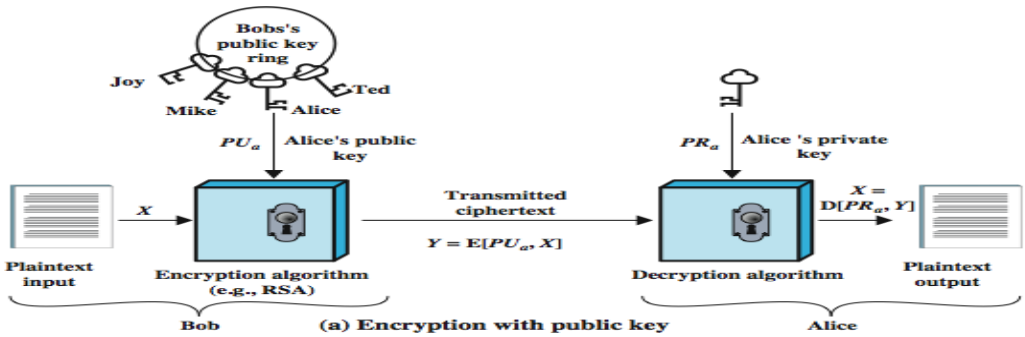
1. **Symmetric Key Encipherment**

   An entity say Alice can send a message to another entity Bob over an insecure channel with the assumption that an attacker Eve cannot understand the contents of the message by evesdropping over the channel. Alice encrypts the message using encryption algorithm and Bob decrypts the cipher text using decryption algorithm.



2. Assymmetric Key Encipherment

   In Assymetric key encipherment which is also called public key cryptography two keys are used, public key and private key. To send a secured message to bob, alice encrypts the message using bobs public key. To decrypt the message bob uses his own private key.



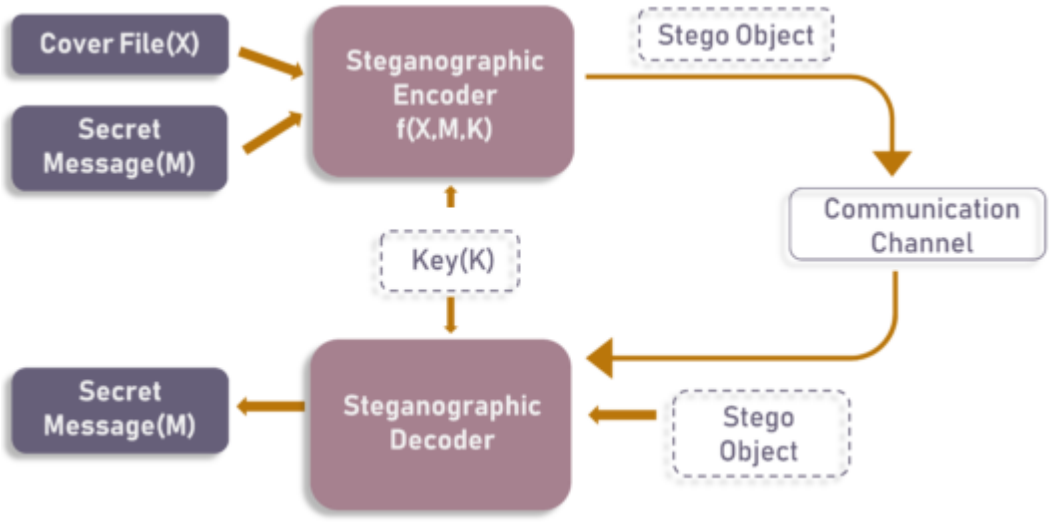
3. Hashing
   In hashing, a fixed length message digest is created out of a variable length message. The digest must be sent to bob. Hashing is used to provide check values.

Steganography

What is Steganography?

Steganography is the art and science of embedding secret messages in a cover message in such a way that no one, apart from the sender and intended recipient, suspects the existence of the message

The diagram below depicts a basic steganographic model.



As the image depicts, both cover file(X) and secret message(M) are fed into steganographic encoder as input. Steganographic Encoder function, f(X,M,K) embeds the secret message into a cover file. Resulting Stego Object looks very similar to your cover file, with no visible changes. This completes encoding. To retrieve the secret message, Stego Object is fed into Steganographic Decoder.

Steganography : Historical Background

Steganography is the practice of concealing a secret message behind a normal message. It stems from two Greek words, which are *steganos,* means covered and *graphia,* means writing. Steganography is an ancient practice, being practiced in various forms for thousands of years to keep communications private. For Example:

- The first use of steganography can be traced back to 440 BC when ancient Greece, people wrote messages on wood and covered it with wax, that acted as a covering medium
- Romans used various forms of Invisible Inks, to decipher those hidden messages light or heat were used
- During World War II the Germans introduced microdots, which were complete documents, pictures, and plans reduced in size to the size of a dot and were attached to normal paperwork
- Null Ciphers were also used to hide unencrypted secret messages in an innocent looking normal message

How is Steganography different from Cryptography?

	STEGANOGRAPHY	CRYPTOGRAPHY
Definition	It is a technique to hide the existence of communication	It's a technique to convert data into an incomprehensible form
Purpose	Keep communication secure	Provide data protection
Data Visibility	Never	Always
Data Structure	Doesn't alter the overall structure of data	Alters the overall structure of data
Key	Optional, but offers more security if used	Necessary requirement
Failure	Once the presence of a secret message is discovered, anyone can use the secret data	If you possess the decryption key, then you can figure out original message from the ciphertext

So, in other words, steganography is more discreet than cryptography when we want to send confidential information. The downside being, the hidden message is easier to extract if the presence of secret is discovered. For the remainder of this steganography tutorial, we will learn about different steganography techniques and tools.

Steganography Techniques

Depending on the nature of the cover object(actual object in which secret data is embedded), steganography can be divided into five types:

Text Steganography

Text Steganography is hiding information inside the text files. It involves things like changing the format of existing text, changing words within a text, generating random character sequences or using context-free grammars to generate readable texts.

Image Steganography

Hiding the data by taking the cover object as the image is known as image steganography. In digital steganography, images are widely used cover source because there are a huge number of bits present in the digital representation of an image. There are a lot of ways to hide information inside an image.

Audio Steganography

In audio steganography, the secret message is embedded into an audio signal which alters the binary sequence of the corresponding audio file. Hiding secret messages in digital sound is a much more difficult process when compared to others, such as Image Steganography.

This method hides the data in WAV, AU, and even MP3 sound files.

Video Steganography

In Video Steganography you can hide kind of data into digital video format. The advantage of this type is a large amount of data can be hidden inside and the fact that it is a moving stream of images and sounds. You can think of this as the combination of Image Steganography and Audio Steganography. Two main classes of Video Steganography include:

- Embedding data in uncompressed raw video and compressing it later
- Embedding data directly into the compressed data stream

Network Steganography (Protocol Steganography)

It is the technique of embedding information within network control protocols used in data transmission such TCP, UDP, ICMP etc.

1.5 The Model for Network Security

A message is to be transferred from one party to another across some sort of Internet service. The two parties, who are the *principals* in this transaction, must cooperate for the exchange to take place. A logical information channel is established by defining a route through the Internet from source to destination and by the cooperative use of communication protocols (e.g., TCP/IP) by the two principals.

Security aspects come into play when it is necessary or desirable to protect the information transmission from an opponent who may present a threat to confidentiality, authenticity, and so on. All the techniques for providing security have two components:

- A security-related transformation on the information to be sent.
- Some secret information shared by the two principals and, it is hoped, unknown to the opponent.

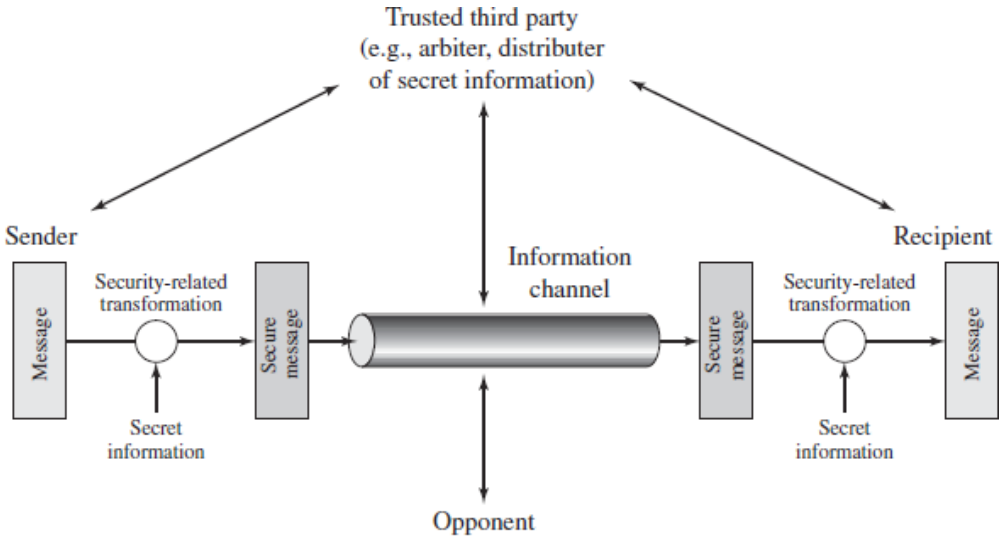


Figure 1.4 Model for Network Security

Question from Old question papers

1. Explain security services and mechanisms
2. What is a security attack? Explain the taxonomy of attacks related to the security goals
3. Explain the cryptography and steganography techniques.
4. Define three security goals? Explain the actual implementation techniques for security goals

Mathematics of Cryptography

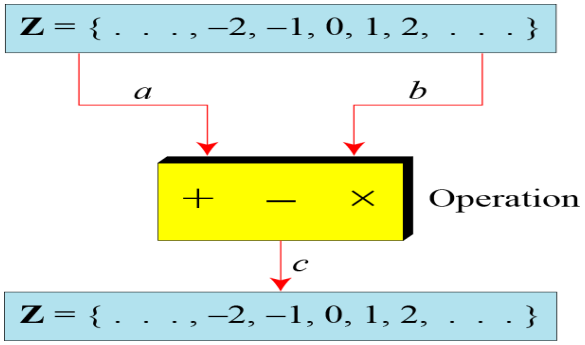
2.1 Integer Arithmetic

The set of integers, denoted by Z, contains all integral numbers (with no fraction) from negative infinity to positive infinity

Z={…….-3,-2,-1,0,1,2,3…….}

2.1.1 Binary Operations

In cryptography, we are interested in three binary operations applied to the set of integers. A binary operation takes two inputs and creates one output.



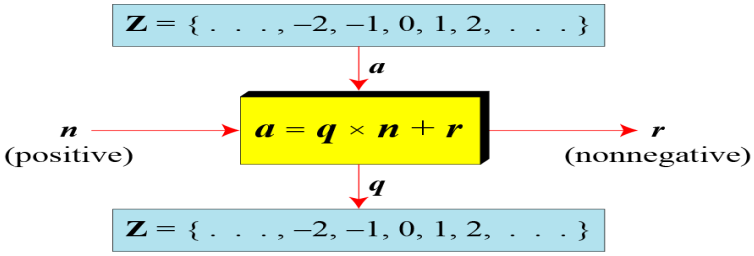
The following shows the results of the three binary operations on two integers. Because each input can be either positive or negative, we can have four cases for each operation.

Add:	$5 + 9 = 14$	$(-5) + 9 = 4$	$5 + (-9) = -4$	$(-5) + (-9) = -14$
Subtract:	$5 - 9 = -4$	$(-5) - 9 = -14$	$5 - (-9) = 14$	$(-5) - (-9) = +4$
Multiply:	$5 \times 9 = 45$	$(-5) \times 9 = -45$	$5 \times (-9) = -45$	$(-5) \times (-9) = 45$

2.1.2 Integer Division

In integer arithmetic, if we divide a by n, we can get q and r . The relationship between these four integers can be shown as

a=q * n+r



The two restrictions for division relationship in cryptography are the divisor is a positive integer(n>0) and remainder be a nonnegative integer(r>=0)

$$-255 = (-23 \times 11) + (-2) \quad \leftrightarrow \quad -255 = (-24 \times 11) + 9$$

Divisibility

If a is not zero and we let r = 0 in the division relation, we get

a=q x n

if the remainder is 0 then n|a

if the remainder is non zero n|a

Ex:

a. The integer 4 divides the integer 32 because 32 = 8 × 4. We show this as 4|32
b. The number 8 does not divide the number 42 because 42 = 5 × 8 + 2. There is a remainder, the number 2, in the equation. We show this as $8 \nmid 42$

Properties of Divisibility

Property 1: if $a|1$, then $a = \pm 1$.
Property 2: if $a|b$ and $b|a$, then $a = \pm b$.
Property 3: if $a|b$ and $b|c$, then $a|c$.
Property 4: if $a|b$ and $a|c$, then $a|(m \times b + n \times c)$, where m and n are arbitrary integers.

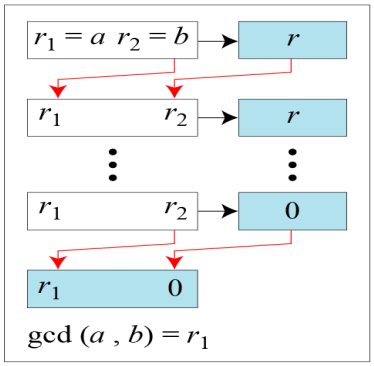
2.1.3 Euclidian Algorithm

Euclidian algorithm is used to find the GCD of two numbers. This algorithm is based on two numbers are

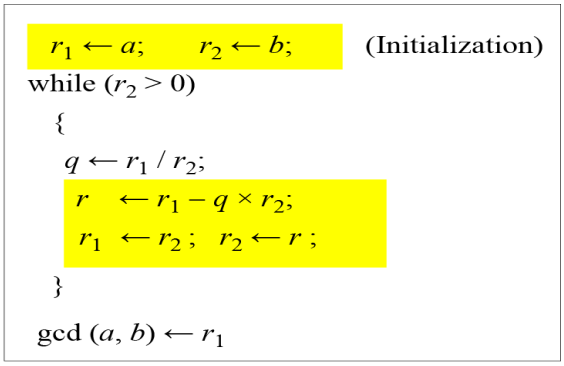
Fact1:gcd(a,0)=a

Fact2:gcd(a,b)=gcd(b,r)where r is the remainder of dividing a by b.

The algorithm and process of Euclidian algorithm are



a. Process



b. Algorithm

We use two variables r1,r2 for holding the changing values of reduction. In each step we calculate the value q is the quotient when r1 is divided by r2 and r is the remainder when r1 is divided by r2. In the next step r1 is replaced with r2 and r2 is replaced with r. this process is repeated until r2 > 0. If r2 is 0 then r1 is the gcd of two numbers a and b

Example problems on Euclidian Algorithm

1. **Find the greatest common divisor of 2740,1760 using Euclidian algorithm**

Let a=2740, b=1760

Initialize r1=a, r2=b, q=r1/r2, r = r1 – q x r2

q	r_1	r_2	r
1	2740	1760	980
1	1760	980	780
1	980	780	200
3	780	200	180
1	200	180	20
9	180	20	0
	20	0	

gcd(2740,1760)=r1=20

2. Find the greatest common divisor of 25,60 using Euclidian algorithm

Let a=25, b=60

Initialize r1=a, r2=b, q=r1/r2, r = r1 – q x r2

q	r_1	r_2	r
0	25	60	25
2	60	25	10
2	25	10	5
2	10	5	0
	5	0	

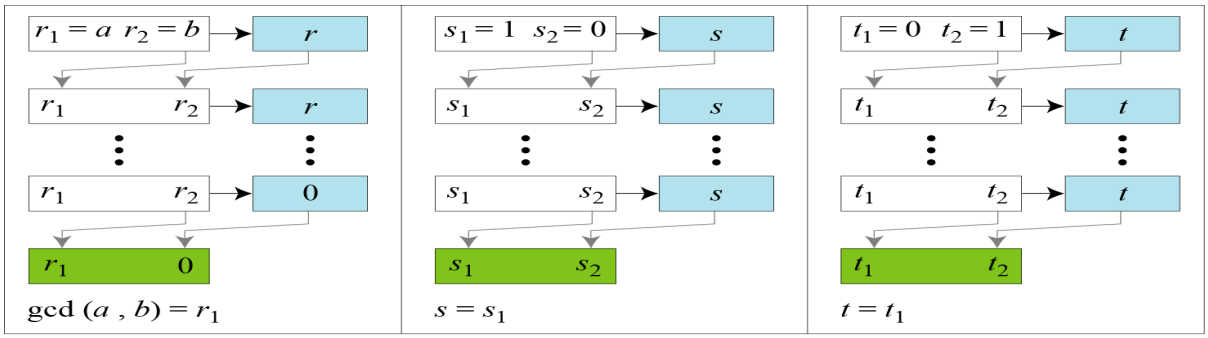
gcd(25,60)=5

2.1.4 Extended Euclidian Algorithm.

Given two integers a, b we need to find two numbers s and t such that

s x a + t x b=gcd(a,b)

The extended Euclidean algorithm can calculate the gcd (*a*, *b*) and at the same time calculate the value of *s* and *t*.



a. Process

$r_1 \leftarrow a; \quad r_2 \leftarrow b;$
$s_1 \leftarrow 1; \quad s_2 \leftarrow 0;$
$t_1 \leftarrow 0; \quad t_2 \leftarrow 1;$
(Initialization)

while $(r_2 > 0)$
{
$q \leftarrow r_1 / r_2;$

$r \leftarrow r_1 - q \times r_2;$
$r_1 \leftarrow r_2; \; r_2 \leftarrow r;$
(Updating r's)

$s \leftarrow s_1 - q \times s_2;$
$s_1 \leftarrow s_2; \; s_2 \leftarrow s;$
(Updating s's)

$t \leftarrow t_1 - q \times t_2;$
$t_1 \leftarrow t_2; \; t_2 \leftarrow t;$
(Updating t's)

}
gcd $(a, b) \leftarrow r_1; \; s \leftarrow s_1; \; t \leftarrow t_1$

b. Algorithm

In each step we calculate three sets of r,s,t. r1,r2 are initialized to a,b. s1,s2 are initialized to 1,0 and t1,t2 are initialized to 0,1. In each step we calculate q,r,s,t using the formulas

q=r1/r2

r= r1 - q x r2

s= s1 - q x s2

t= t1 - q x t2

In each step r1,r2 are assigned with r2,r of previous step. Similarly s1,s2 ,t1,t2 are assigned with s2,s,t2,t of previous step. This process is repeated while r2>0. Once r2 is 0, gcd of two numbers is r1.

Problems

1. **Given *a* = 161 and *b* = 28, find gcd (*a*, *b*) and the values of *s* and *t*.**

q	r_1	r_2	r	s_1	s_2	s	t_1	t_2	t
5	161	28	21	1	0	1	0	1	−5
1	28	21	7	0	1	−1	1	−5	6
3	21	7	0	1	−1	4	−5	6	−23
	7	0		**−1**	4		**6**	−23	

We get gcd (161, 28) = 7, *s* = −1 and *t* = 6.

-1*161+6*28=7

2. **Given *a* = 17 and *b* = 0, find gcd (*a*, *b*) and the values of *s* and *t*.**

We get gcd (17, 0) = 17, *s* = 1, and *t* = 0.

q	r_1	r_2	r	s_1	s_2	s	t_1	t_2	t
	17	0		**1**	**0**		**0**	**1**	

1*17+0*0=17

3.Given *a* = 0 and *b* = 45, find gcd (*a*, *b*) and the values of *s* and *t*.

We get gcd (0, 45) = 45, *s* = 0, and *t* = 1.

q	r_1	r_2	r	s_1	s_2	s	t_1	t_2	t
0	0	45	0	1	0	1	0	1	0
	45	0		0	1		**1**	0	

2.1.5 Linear Diophantine Equations

Linear Diophantine equations of two variables is an equation of type ax+by=c. we have to find integer values for x and y .This type of equation has either no solution or an infinite number of solutions.

Let d=gcd(a,b), if d does not divide c, then the equation has no solution. If d|c, then we have an infinite

number of solutions. One of them is called the particular solution and the remaining are general solutions.

Particular Solution

If d|c a particular solution to the equation can be found using the following steps.

1. Reduce the equation to $a_1x+b_1y=c_1$, by dividing both sides of the equation by d.
2. Solve for s and t in the relation $a_1 s+b_1 t=1$ using extended Euclidean algorithm
3. Find the particular solution

$x_0=(c/d)*s$

$y_0=(c/d)*t$

General Solution

$x = x_0 + k(b/d)$ and $y = y_0 - k(a/d)$

where k is an integer

Problems

1.Find the particular and general solutions to the equation 21*x* + 14*y* = 35.

Q	r1	r2	R	s1	s2	S	t1	t2	t
1	21	14	7	1	0	1	0	1	-1
2	14	7	0	0	1	-2	1	-1	3
	7	0		1	-2		-1	3	
	gcd			S			T		

21x+14y=35

Divide the equation with gcd(21,14)=7

21/7 x+14/7 y=35/7

3x+2y=5

Particular solution

x0=(c/d)*s y0=(c/d)* t

x0=(35/7)*1=5 y0=(35/7)*-1= -5

general solution

x1=x0 + (b/d)*k y1=y0 - (a/d)*k

x1=5+2*k y1=-5 – 3*k

K=0 x=5 y=-5

K=1 x=7 y=-8

K=2 x=7 y=-11

.

.

2. imagine we want to cash a $100 check and get some $20 and some $5 bills. Find the denominations of the cash.

Q	r1	r2	R	s1	s2	S	t1	t2	t
4	20	5	0	1	0	1	0	1	-4
	5	0		0	-2		1	-4	
	gcd			S			T		

Since *d* = gcd (20, 5) = 5 and 5 | 100, the equation has an infinite number of solutions, but only a few of them are acceptable in this case

20x+5y=100

Divide the equation with gcd(20,5)=5

20/5 x+5/5 y=100/5

4x+y=20

Particular solution

x0=(c/d)*0 y0=(c/d)* t

x0=(100/5)*0=0 y0=(100/5)*1= 20

general solution

x1=x0 + (b/d)*k y1=y0 - (a/d)*k

x1=0+1*k y1=20 – 4*k

K=0 x=0 y=20

K=1 x=1 y=16

K=2 x=2 y=12

K=3 x=3 y=8

K=4 x=4 y=4

K=5 x=5 y=0

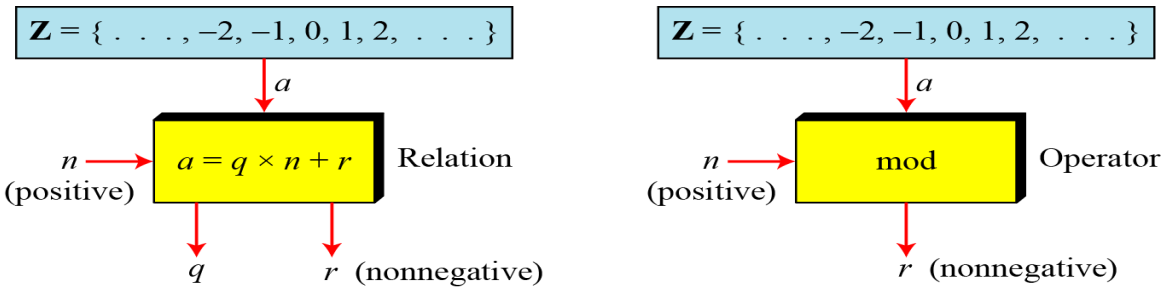
The general solutions with *x* and *y* nonnegative are (0, 20), (1, 16), (2, 12), (3, 8), (4, 4), (5, 0).

2.2 Modular Arithmetic

In Modular arithmetic for division operation a=q*n+r, we are concerned about the remainder of the division and not the quotient

2.2.1 Modulo Operator

The **mod** is a binary operator which takes two inputs a number(a) and a positive divisor(n) which is called modulus and it generates a remainder ® called residue.



Find the result of the following operations:

a. 27 mod 5 **b. 36 mod 12**

c. –18 mod 14 **d. –7 mod 10**

Solution:

a. **Dividing 27 by 5 results in *r* = 2**

b. **Dividing 36 by 12 results in *r* = 0.**

c. Dividing –18 by 14 results in *r* = –4. If output is negative add modulus 14 to -1. *r* = 10

d. Dividing –7 by 10 results in r = –7. After adding the modulus to –7, r = 3.

2.2.2 Set of Residues

The result of the modulo operation with modulus n is always an integer between o and n-1. The modulus operator creates a ser, which in modular arithmetic is called set of least residues modulo n or Z_n. We have infinite number of set of residues for each value of n. Ex: Z_2 is set of residues for modulo 2, Z_{11} is set of residues for modulo 11.

$$\mathbf{Z}_n = \{ 0, 1, 2, 3, \ldots, (n-1) \}$$

$$\mathbf{Z}_2 = \{ 0, 1 \} \qquad \mathbf{Z}_6 = \{ 0, 1, 2, 3, 4, 5 \} \qquad \mathbf{Z}_{11} = \{ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 \}$$

2.2.3 Congruence

In cryptography we use congruence instead of equality. Mapping from Z to Zn is not one to one. Infinite members of Z are mapped to Zn.

Ex: 2 mod 10=2,12 mod 10=2,22 mod 10=2……..

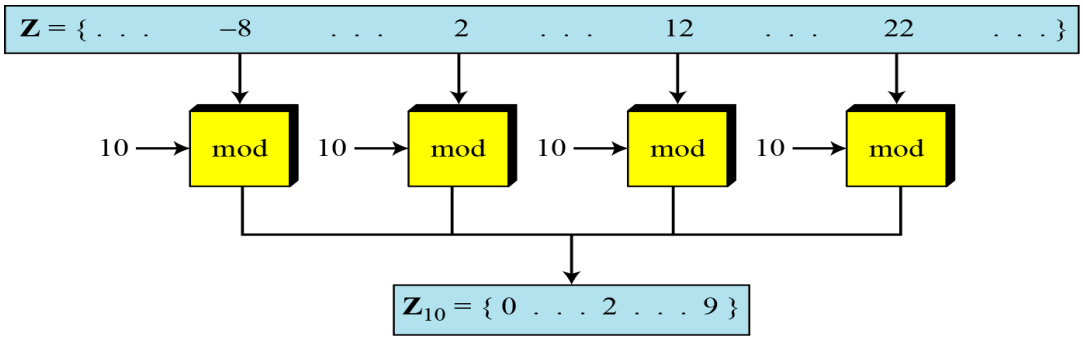
2,12,22,32,42……are called congruent mod 10.

To show that two integers are congruent, we use the congruent operator (≡). We add (mod n) to the right side

Ex: 2≡12(mod 10), 2≡82 (mod 10), 3≡23 (mod 5), -8≡2(mod 5)

Difference between equal to and congruence

1. Equality operator maps a member of Z to itself, the congruence operator maps a member of Z to a member Z_n
2. Equality operator is one to one mapping where as congruence operator is many to one.



$$-8 \equiv 2 \equiv 12 \equiv 22 \pmod{10}$$

Congruence Relationship

2.2.4 Residue Classes

Residue class [a] or [a]n is the set of integers congruent modulo n. it is the set of all integers such that x=a(mod n).

Ex: for n=5, we have five sets

[0]= {.....,-15,-10,-5,0,5,10,15.....}

[1]= {.....,-14,-9,-4,1,6,11,16.....}

[2]= {.....,-13,-8,-3,2,7,12,17.....}

[3]= {.....,-12,-7,-2,3,8,13,18.....}

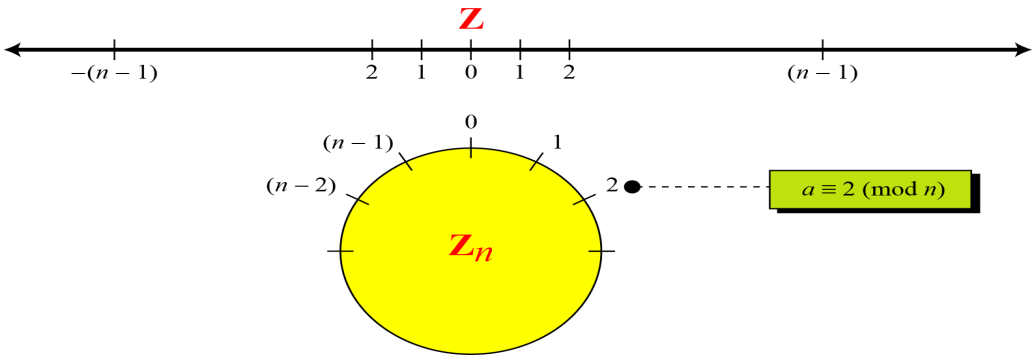
[4]= {.....,-11,-6,-1,4,9,14,19.....}

The integers in the set[0] are all reduced to 0 when we apply modulo 5 operation, the integers in the set [4] are all reduced to 4 when we apply modulo 5 operation.

The set of all these residues can be written as Z_5={0,1,2,3,4}

The set Z_n is the set of all least residue modulo n.

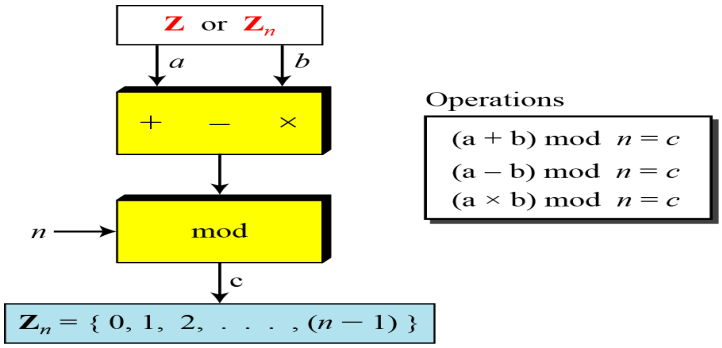
Comparison between Z and Zn using graphs.



The line representation is used to represent Z and circle representation is used for congruence. All congruent integers modulo n occupy the same point on the circle. Positive and negative integers from Z are mapped to the circle.

Operations on Zn

The three binary operations that we discussed for the set Z can also be defined for the set Zn. The result may need to be mapped to Zn using the mod operator.



Perform the following operations (the inputs come from Zn):

a. Add 7 to 14 in Z15.

b. Subtract 11 from 7 in Z13.

c. Multiply 11 by 7 in Z20.

Solution

a. (14+7) mod 15= 21 mod 15=6
b. (7-11) mod 13=-4 mod 13=9
c. (7x11) mod 20=77 mod 20=7

Perform the following operations (the inputs come from either Z or Z_n):

a. Add 17 to 27 in Z_{14}.

b. Subtract 43 from 12 in Z_{13}.

c. Multiply 123 by −10 in Z_{19}.

Solution

a. (17 + 27) mod 14=44 mod 14=2
b. (12-43) mod 13=-31 mod 13=8
c. (123x-10) mod 19=-1230 mod 19=5

2.2.5 Properties of modulus operator

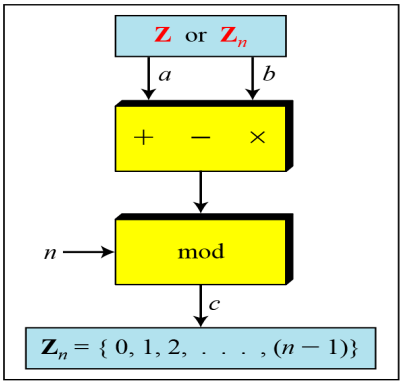
The three properties that are used in cryptography are

(a+b) mod n=[(a mod n) + (b mod n)]mod n

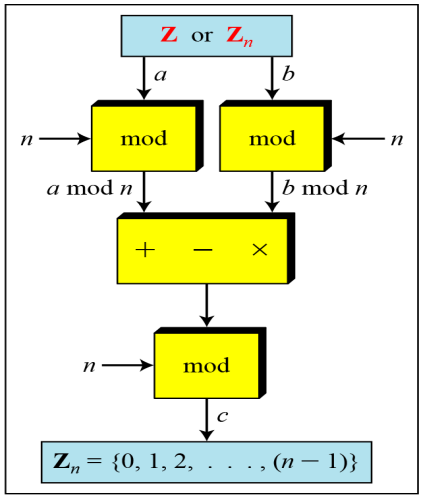
(a-b) mod n=[(a mod n) - (b mod n)]mod n

(a*b) mod n=[(a mod n) * (b mod n)]mod n

We use these properties in the cryptography because cryptography handles large numbers. If multiplying large number with another large number, it becomes problem to store the resultant large number. To overcome this problem the above properties are used to decrease the number.



a. Original process



b. Applying properties

Example1: The following shows the application of the above properties:

1. (1,723,345 + 2,124,945) mod 11 =(1,723,345 mod 11 + 2,124,945 mod 11) mod 11= (8 + 9) mod 11 = 6
2. (1,723,345 − 2,124,945) mod 16 =(1,723,345 mod 11 - 2,124,945 mod 11) mod 11= (8 − 9) mod 11 = 10
3. (1,723,345 × 2,124,945) mod 16 =(1,723,345 mod 11 x 2,124,945 mod 11) mod 11= (8 × 9) mod 11 = 6

Example 2:

10 mod 3=1 10^n mod 3=(10 mod 3)^n= 1 Applying property 3 n times

10 mod 9=1 10^n mod 9=(10 mod 9)^n= 1

10 mod 7=3 10^n mod 7=(10 mod 7)^n=3^n mod 7

Example3: We have been told in arithmetic that the remainder of an integer divided by 3 is the same as the remainder of the sum of its decimal digits. We write an integer as the sum of its digits multiplied by the powers of 10.

$$a = a_n \times 10^n + \cdots + a_1 \times 10^1 + a_0 \times 10^0$$

$$\text{For example: } 6371 = 6 \times 10^3 + 3 \times 10^2 + 7 \times 10^1 + 1 \times 10^0$$

$$\begin{aligned} a \bmod 3 &= (a_n \times 10^n + \cdots + a_1 \times 10^1 + a_0 \times 10^0) \bmod 3 \\ &= (a_n \times 10^n) \bmod 3 + \cdots + (a_1 \times 10^1) \bmod 3 + (a_0 \times 10^0) \bmod 3 \\ &= (a_n \bmod 3) \times (10^n \bmod 3) + \cdots + (a_1 \bmod 3) \times (10^1 \bmod 3) + \\ &\quad (a_0 \bmod 3) \times (10^0 \bmod 3) \\ &= a_n \bmod 3 + \cdots + a_1 \bmod 3 + a_0 \bmod 3 \\ &= (a_n + \cdots + a_1 + a_0) \bmod 3 \end{aligned}$$

There fore 6371 mod 3=(6+3+7+1) mod 3=17 mod 3=2

2.2.6 Inverses

Additive Inverse

In Z_n, two numbers *a* and *b* are additive inverses of each other if

a+b≡0(mod n)

In modular arithmetic, each integer has an additive inverse. The sum of an integer and its additive inverse is congruent to 0 modulo n.

find all additive inverse pairs in Z10.

The six pairs of additive inverses are (0, 0), (1, 9), (2, 8), (3, 7), (4, 6), and (5, 5).

Multiplicative Inverse

In Z_n, two numbers *a* and *b* are Multiplicative inverses of each other if

a x b≡1(mod n)

In modular arithmetic, an integer may or may not have a multiplicative inverse.When it does, the product of the integer and its multiplicative inverse is congruent to 1 modulo n.

In Z_n , a number 'a' has a multiplicative inverse if and only if gcd(n,a)=1

1. Find the multiplicative inverse of 8 in Z_{10}.

   There is no multiplicative inverse because gcd (10, 8) = 2 ≠ 1. In other words, we cannot find any number between 0 and 9 such that when multiplied by 8, the result is congruent to 1.

2. Find all multiplicative inverses in Z_{10}.

   There are only three pairs: (1, 1), (3, 7) and (9, 9). The numbers 0, 2, 4, 5, 6, and 8 do not have a multiplicative inverse.

3. Find all multiplicative inverse pairs in Z_{11}.

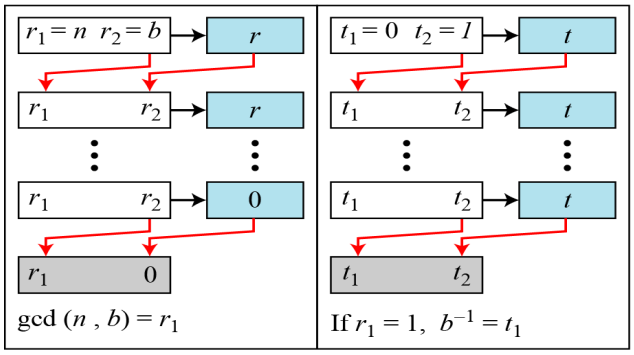
   We have seven pairs: (1, 1), (2, 6), (3, 4), (5, 9), (7, 8), (9, 5), and (10, 10).

Finding multiplicative inverse using Extended Euclidian Algorithm

The extended Euclidean algorithm finds the multiplicative inverses of b in Z_n when n and b are given and gcd (n, b) = 1.

The multiplicative inverse of b is the value of t after being mapped to Z_n.

The process and the algorithm for finding the multiplicative inverse using extended eucludian algorithm are



a. Process

$r_1 \leftarrow$ n; $r_2 \leftarrow b$;
$t_1 \leftarrow 0$; $t_2 \leftarrow 1$;
while $(r_2 > 0)$
{
$q \leftarrow r_1 / r_2$;
$r \leftarrow r_1 - q \times r_2$;
$r_1 \leftarrow r_2$; $r_2 \leftarrow r$;
$t \leftarrow t_1 - q \times t_2$;
$t_1 \leftarrow t_2$; $t_2 \leftarrow t$;
}
if $(r_1 = 1)$ then $b^{-1} \leftarrow t_1$

b. Algorithm

Find the multiplicative inverse of 11 in Z_{26}.

Solution:

Find the gcd(26,11)

q	r_1	r_2	r	t_1	t_2	t
2	26	11	4	0	1	−2
2	11	4	3	1	−2	5
1	4	3	1	−2	5	−7
3	3	1	0	5	−7	26
	1	0		−7	26	

gcd(26,11)=1 therefore 11 has multiplicative inverse in Z26

Multiplicative inverse of 11= t1 = -7

We cannot represent negative numbers as multiplicative inverse, there fore add 26 to -7 to get multiplicative inverse

Multiplicative inverse of 11=-7+29=19

Find the multiplicative inverse of 23 in Z_{100}.

Solution:

q	r_1	r_2	r	t_1	t_2	t
4	100	23	8	0	1	−4
2	23	8	7	1	−4	19
1	8	7	1	−4	9	−13
7	7	1	0	9	−13	100
	1	0		−13	100	

The gcd (100, 23) is 1; the inverse of 23 is -13 or 87.

Find the inverse of 12 in Z_{26}.

q	r_1	r_2	r	t_1	t_2	t
2	26	12	2	0	1	−2
6	12	2	0	1	−2	13
	2	0		−2	13	

The gcd (26, 12) is 2; the inverse does not exist.

2.2.7 Addition and Multiplication Tables

	0	1	2	3	4	5	6	7	8	9
0	**0**	1	2	3	4	5	6	7	8	9
1	1	2	3	4	5	6	7	8	9	**0**
2	2	3	4	5	6	7	8	9	**0**	1
3	3	4	5	6	7	8	9	**0**	1	2
4	4	5	6	7	8	9	**0**	1	2	3
5	5	6	7	8	9	**0**	1	2	3	4
6	6	7	8	9	**0**	1	2	3	4	5
7	7	8	9	**0**	1	2	3	4	5	6
8	8	9	**0**	1	2	3	4	5	6	7
9	9	**0**	1	2	3	4	5	6	7	8

Addition Table in $\mathbf{Z}_{10}$

	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	0	0	0	0	0
1	0	**1**	2	3	4	5	6	7	8	9
2	0	2	4	6	8	0	2	4	6	8
3	0	3	6	9	2	5	8	**1**	4	7
4	0	4	8	2	6	0	4	8	2	6
5	0	5	0	5	0	5	0	5	0	5
6	0	6	2	8	4	0	6	2	8	4
7	0	7	4	**1**	8	0	2	9	6	3
8	0	8	6	4	2	0	8	6	4	2
9	0	9	8	7	6	5	4	3	2	**1**

Multiplication Table in $\mathbf{Z}_{10}$

The set of multiplicative inverses Zn is called Zn*

Z_{10}*=(1,3,7,9}

Construct the multiplication table for Z_7

	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6
2	0	2	4	6	1	3	5
3	0	3	6	2	5	1	4
4	0	4	1	5	2	6	3
5	0	5	3	1	6	4	2
6	0	6	5	4	3	2	1

Z_7*={1,2,3,4,5,6}

2.3Matrices

2.3.1 Definitions

A Matrix is a rectangular array of I x m where l is the number of rows and m is the number of columns

The element aij is the element present at ith row and jth column of a matrix A

$$\text{Matrix } \mathbf{A}: \quad \overset{l \text{ rows}}{\left[\begin{array}{cccc} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \vdots & \vdots & & \vdots \\ a_{l1} & a_{l2} & \cdots & a_{lm} \end{array}\right]} \quad \overset{m \text{ columns}}{}$$

A matrix A having only one row(l=1) is called Row matrix

A matrix A having only one column(m=1) is called Column matrix

A matrix A having the same dimensions(i.e., l=m) then it is called Square matrix

A matrix A having all the elements as zero is called Zero matrix

A Square matrix A is said to be an Identity matrix if all the elements of principal diagonal are 1 and the remaing elements are 0, which is denoted by I

$$\underset{\text{Row matrix}}{\begin{bmatrix} 2 & 1 & 5 & 11 \end{bmatrix}} \qquad \underset{\text{Column matrix}}{\begin{bmatrix} 2 \\ 4 \\ 12 \end{bmatrix}} \qquad \underset{\text{Square matrix}}{\begin{bmatrix} 23 & 14 & 56 \\ 12 & 21 & 18 \\ 10 & 8 & 31 \end{bmatrix}} \qquad \underset{\mathbf{0}}{\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}} \qquad \underset{\mathbf{I}}{\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}}$$

2.3.2 Operations and relations on matrix

1. **Equality**

   If all the corresponding elements of the two matrices A and B are same then the two matrices are said

to be equal. i.e., aij=bij for all i and j

2. **Addition and subtraction**

The two matrices can be added or subtracted if they have the same dimensions

cij=aij+bij

dij=aij-bij

$$\begin{bmatrix} 12 & 4 & 4 \\ 11 & 12 & 30 \end{bmatrix} = \begin{bmatrix} 5 & 2 & 1 \\ 3 & 2 & 10 \end{bmatrix} + \begin{bmatrix} 7 & 2 & 3 \\ 8 & 10 & 20 \end{bmatrix}$$

$$\mathbf{C = A + B}$$

$$\begin{bmatrix} -2 & 0 & -2 \\ -5 & -8 & 10 \end{bmatrix} = \begin{bmatrix} 5 & 2 & 1 \\ 3 & 2 & 10 \end{bmatrix} - \begin{bmatrix} 7 & 2 & 3 \\ 8 & 10 & 20 \end{bmatrix}$$

$$\mathbf{D = A - B}$$

3. **Multiplication**

$$\underset{\mathbf{C}}{\begin{bmatrix} 53 \end{bmatrix}} = \underset{\mathbf{A}}{\begin{bmatrix} 5 & 2 & 1 \end{bmatrix}} \times \underset{\mathbf{B}}{\begin{bmatrix} 7 \\ 8 \\ 2 \end{bmatrix}}$$

In which: $53 = 5 \times 7 + 2 \times 8 + 1 \times 2$

$$\underset{\mathbf{C}}{\begin{bmatrix} 52 & 18 & 14 & 9 \\ 41 & 21 & 22 & 7 \end{bmatrix}} = \underset{\mathbf{A}}{\begin{bmatrix} 5 & 2 & 1 \\ 3 & 2 & 4 \end{bmatrix}} \times \underset{\mathbf{B}}{\begin{bmatrix} 7 & 3 & 2 & 1 \\ 8 & 0 & 0 & 2 \\ 1 & 3 & 4 & 0 \end{bmatrix}}$$

4. **Scalar Multiplication**

A matrix can be multiplied with a scalar value. i.e., we multiply every element of the matrix with a scalar value

$$\underset{\mathbf{B}}{\begin{bmatrix} 15 & 6 & 3 \\ 9 & 6 & 12 \end{bmatrix}} = 3 \times \underset{\mathbf{A}}{\begin{bmatrix} 5 & 2 & 1 \\ 3 & 2 & 4 \end{bmatrix}}$$

2.3.3. Determinant

The determinant of a square matrix A of size m × m denoted as det (A) is a scalar calculated recursively as shown below:

1. If $m = 1$, $\det(\mathbf{A}) = a_{11}$
2. If $m > 1$, $\det(\mathbf{A}) = \sum_{i=1\ldots m} (-1)^{i+j} \times a_{ij} \times \det(\mathbf{A}_{ij})$

Where Aij is a matrix obtained from A by deleting the ith row and jth column

we can calculate the determinant of a 2 × 2 matrix based on the determinant of a 1 × 1 matrix.

$$\det\begin{bmatrix}5 & 2\\ 3 & 4\end{bmatrix} = (-1)^{1+1}\times 5\times \det[\,4\,] + (-1)^{1+2}\times 2\times \det[\,3\,] \longrightarrow 5\times 4 - 2\times 3 = 14$$

or

$$\det\begin{bmatrix}a_{11} & a_{12}\\ a_{21} & a_{22}\end{bmatrix} = a_{11}\times a_{22} - a_{12}\times a_{21}$$

calculation of the determinant of a 3 × 3 matrix.

$$\det\begin{bmatrix}5 & 2 & 1\\ 3 & 0 & -4\\ 2 & 1 & 6\end{bmatrix} = (-1)^{1+1}\times 5\times \det\begin{bmatrix}0 & -4\\ 1 & 6\end{bmatrix} + (-1)^{1+2}\times 2\times \det\begin{bmatrix}3 & -4\\ 2 & 6\end{bmatrix} + (-1)^{1+3}\times 1\times \det\begin{bmatrix}3 & 0\\ 2 & 1\end{bmatrix}$$

$$= (+1)\times 5\times (+4) \quad + \quad (-1)\times 2\times (24) \quad + \quad (+1)\times 1\times (3) = -25$$

2.3.4 Residue Matrices

Cryptography uses residue matrices: matrices where all elements are in Z_n. A residue matrix has a multiplicative inverse if gcd (det(A), n) = 1.

Residue matrix and its multiplicative inverse

$$\mathbf{A} = \begin{bmatrix}3 & 5 & 7 & 2\\ 1 & 4 & 7 & 2\\ 6 & 3 & 9 & 17\\ 13 & 5 & 4 & 16\end{bmatrix} \qquad \det(\mathbf{A}) = 21$$

$$\mathbf{A}^{-1} = \begin{bmatrix}15 & 21 & 0 & 15\\ 23 & 9 & 0 & 22\\ 15 & 16 & 18 & 3\\ 24 & 7 & 15 & 3\end{bmatrix} \qquad \det(\mathbf{A}^{-1}) = 5$$

2.4 Linear Congruence

Cryptography often involves solving an equation or a set of equations of one or more variables with coefficient in Z_n. This section shows how to solve equations when the power of each variable is 1 (linear equation).

2.4.1Single Variable Linear Equation

Equations of the form ax ≡ b (mod n) might have no solution or a limited number of solutions.

Find d=gcd(a,n)

If $d \nmid b$, there is no solution.

If $d \mid b$, there are d solutions.

1. **Solve the equation 10 x ≡ 2(mod 15).**
   Find the gcd(15,10)=5
   Since 5 does not divide 2 ,there is no solution for this linear equation
2. **Solve the equation 14 x ≡ 12 (mod 18).**
   gcd(18,14)=2

since 2 divides 12, there are two solutions for x

14 x ≡ 12 (mod 18)

Divide the equation by gcd

14/2 x ≡ 12/2 (mod 18/2)

7 x ≡ 6 (mod 9)

General solution

$X_0=(6 \times 7^{-1})(\text{mod } 9)$

$X_0=(6x\ 4) \text{ mod } 9$

$X_0=24 \text{ mod } 9$

$X_0=6$

Particular solution

X=X0+k x (n/d)

X=X0+k* (18/2)

X=X0+9 k

X=6+1 x 9 substituting k=1

X=15

The two solutions are 6 and 9

3. **Solve the equation 3x + 4 ≡ 6 (mod 13)**

3x + 4 ≡ 6 (mod 13)

3x ≡ (6 - 4) (mod 13)

3x ≡ 2 (mod 13)

Gcd(13,3)=1 and 1 divides 2 there is a single solution for the equation

$X_0=(2 \times 3^{-1})(\text{mod } 13)$

$X_0=(2 \times 9)(\text{mod } 13)$

$X_0=(18)(\text{mod } 13)$

$X_0=5$

The solution for equation is 5

2.4.2 Set of Linear Equations of multiple variables

We can also solve a set of linear equations with the same modulus if the matrix formed from the coefficients of the variables is invertible.

$$\begin{array}{rcrcccrcl} a_{11}x_1 & + & a_{12}x_2 & + & \cdots & + & a_{1n}x_n & \equiv & b_1 \\ a_{21}x_1 & + & a_{22}x_2 & + & \cdots & + & a_{2n}x_n & \equiv & b_2 \\ \vdots & & \vdots & & & & \vdots & & \vdots \\ a_{n1}x_1 & + & a_{n2}x_2 & + & \cdots & + & a_{nn}x_n & \equiv & b_n \end{array}$$

a. Equations

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \equiv \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

b. Interpretation

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \equiv \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}^{-1} \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

c. Solution

Questions from old papers related to this chapter

1. Find the particular and the general solutions to the following linear Diophantine equation 19x +13y = 20.
2. State and prove the properties of modular arithmetic binary operations. [6]
3. Let us assign numeric values to the uppercase alphabet (A=0, B=1, . . ,Z=25).

   We can do modular arithmetic on the system using modulo 26

   (i) What is (A+N) mod 26 in this system?

   (ii) What is (C-10) mod 26 in this system? [3]
4. Explain the extended Euclidean algorithm. Find gcd(a, b) and the values of s

and t for given a=161 and b=28 [8]

5. Find the particular and the general solutions to the following linear Diophantine equation 25x +10y = 15. [3]
6. Explain the Euclidian algorithm. Find the greatest common divisor of 25 and 60 using this.[6]
7. Let us assign numeric values to the uppercase alphabet (A=0, B=1, . . , Z=25).

   We can do modular arithmetic on the system using modulo 26

   (i) What is (A+6) mod 26 in this system?

   (ii) What is (C-10) mod 26 in this system? [3]

8. What is a multiplicative inverse? Find all multiplicative inverse pairs in Z11. [4]

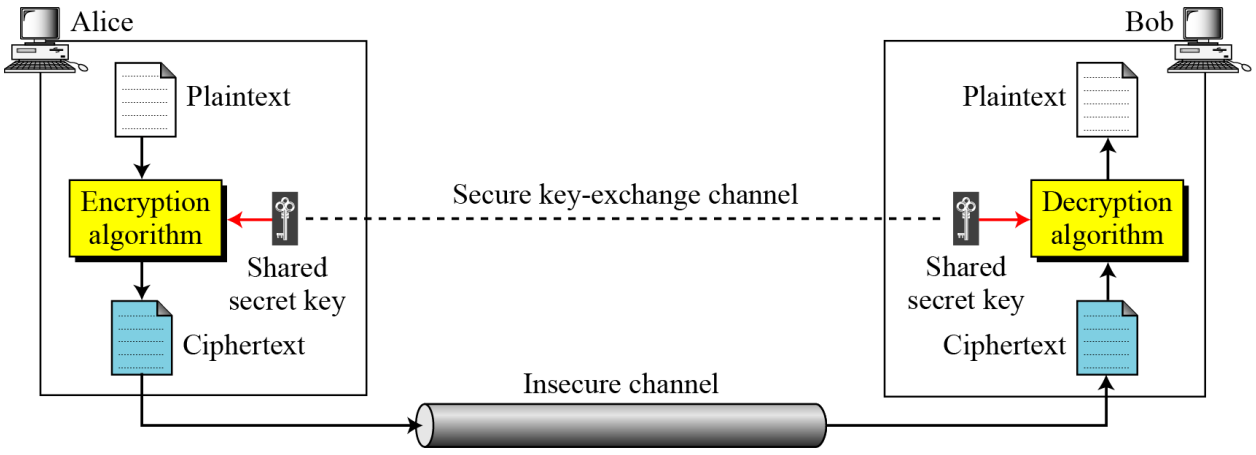
Unit-2
Syllabus

Symmetric Encryption
Mathematics of Symmetric Key Cryptography, Introduction to Modern Symmetric Key Ciphers, Data Encryption Standard, Advanced Encryption Standard.

Chapter-3: Introduction to Modern Symmetric Key Ciphers

3.1 Symmetric Key Cryptography

The original message from Alice to Bob is called plaintext. the message that is sent through the channel is called the ciphertext. To create the ciphertext from the plaintext, Alice uses an encryption algorithm and a shared secret key. To create the plaintext from ciphertext, Bob uses a decryption algorithm and the same secret key. A key is a set of values that the cipher and algorithm operates on. The symmetric key algorithm uses a single key for both encryption and decryption. The encryption and decryption algorithms are inverses of each other.



If P is the plaintext, C is the ciphertext, and K is the key,

$$\text{Encryption: } C = E_k(P) \qquad \text{Decryption: } P = D_k(C)$$

$$\text{In which, } D_k(E_k(x)) = E_k(D_k(x)) = x$$

We assume that Bob creates P_1; we prove that $P_1 = P$:

$$\textbf{Alice: } C = E_k(P)$$

$$\textbf{Bob: } P_1 = D_k(C) = D_k(E_k(P)) = P$$

Using symmetric key mechanism Alice and Bob share the same secret key for both encryption and decryption. This is why this method is called symmetric

Another element in symmetric key encryption is the number of keys. Alice needs a key for communication with each other participant. If m people are in the group there is a need of m(m-1)/2 keys for communication.

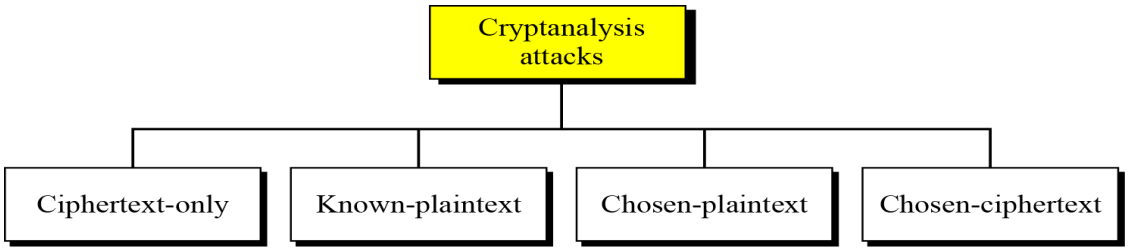
3.2 Kerckhoff's principle

Kerckhoffs' Principle states that the security of a cryptosystem must lie in the choice of its keys only; everything else (including the algorithm itself) should be considered public knowledge.

This principle states that Eve knows the encryption/decryption algorithm. The resistance of the cipher to attack must be based on only the secrecy of the key. The key domain for each algorithm is to be so large that it makes the cryptanalyst difficult to find the key.

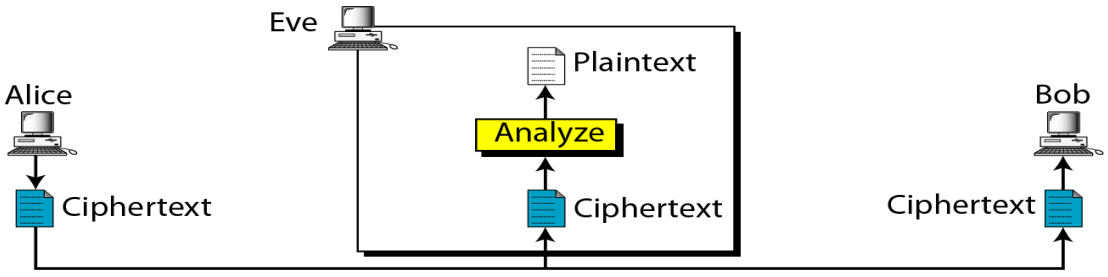
3.3 Cryptanalysis

Cryptography is the science and art of creating secret codes, Cryptanalysis is the science and art of breaking these codes.There are four common types of cryptanalysis attacks



1.Ciphertect-only Attack:

In this attack, Eve has the access to only some ciphertext. She tries to find the corresponding key and the plain text. The assumption is that Eve knows the algorithm and can interrupt the cipher text. This attack is the most probable one because every attacks has the cipher text only.



Various methods can be used in cipher text attacks they are

Brute-Force Attack:

It is also called exhaustive-key method, Eve tries to use all possible keys because she knows the encryption algorithm and the key domain. Using the intercepted cipher Eve decrypts the ciphertext with every possible key until the plain text is decrypted. To prevent this type of attack, the possible keys must be large.

Statistical Attack:

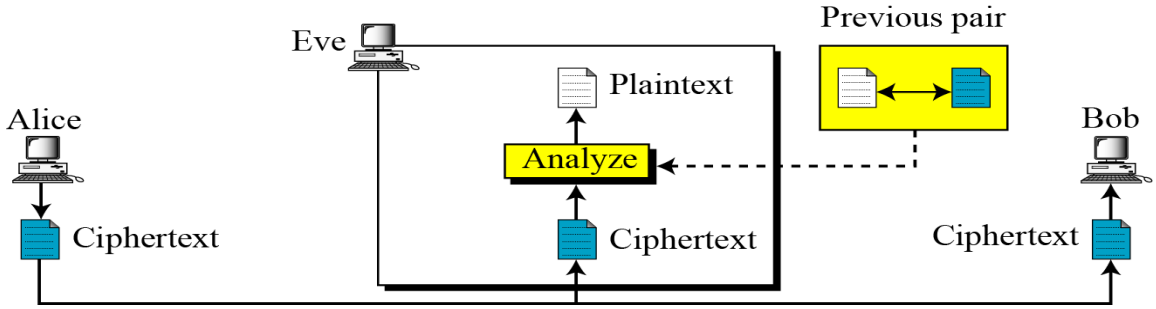
The cryptanalyst can benefit from some inherent characteristics of the plain text language to launch a statistical attack. For Ex. We know letter E is the most frequently used letter in English text. The cryptanalyst finds the mostly used character in cipher text and assumes that the corresponding plain text character is E. To prevent this type of attack, the cipher should hide the characteristics of the language

Pattern Attack:

Some ciphers may hide the characteristics of the language, but may create some patterns in the ciphertext. A cryptanalyst uses these patterns to decrypt the cipher text. There fore the cipher text should look as random as possible

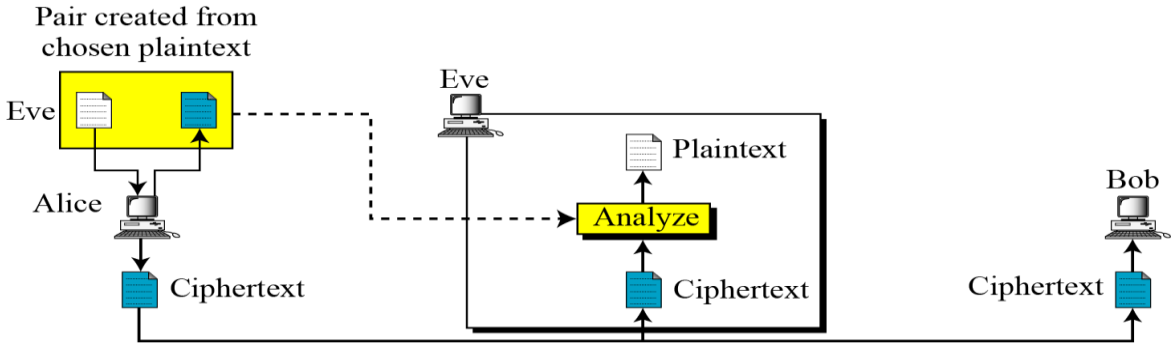
2.Known-Plain text Attack:

In this attack, Eve has access to some plaintext/ciphertext pairs in addition to the intercepted ciphertext that she wants to break. The plaintext/ciphertext pairs have been collected earlier when alice has sent a secret message to bob which has later on made public. Alice has kept these pairts to break the next secret message from Alice to Bob, assuming that the secret key has not changed. This attack is easier to implement because eve has information to use for analysis. However it is less likely to happen because Alice may have changed the key or may have not disclosed the contents of previous messages.



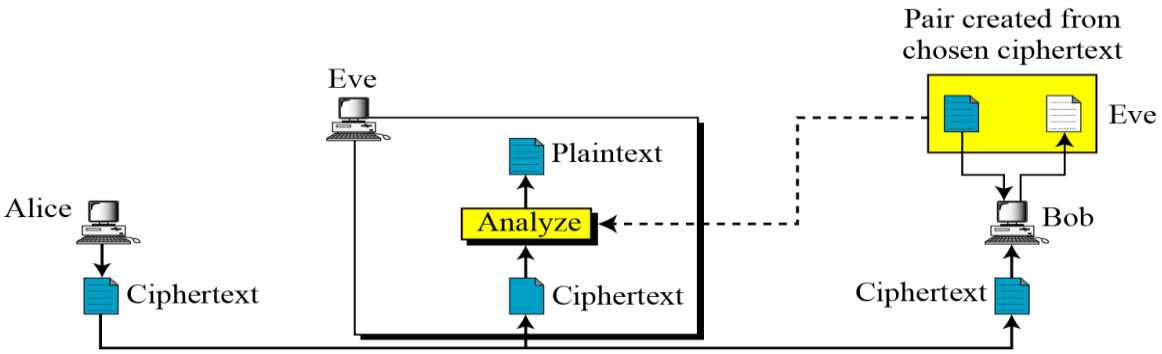
3.Chosen-Plaintext Attack:

This attack is similar to the known plain text attack, but the plaintext/ciphertext pairs have been chosen by the attacker herself. For Example if Evve has access to Alice's computer. She can choose some plaintext and intercept the created ciphertext. This type of attack is much easier to implement, but it is less likely to happen.



4.Chosen-Ciphertext Attack:

This attack is similar to the chosen-plain text attack , except that Eve chooses some ciphertext and decrypts it to form a ciphertext/plaintext pair. This can happen if Eve has access to bob's computer



3.4 Substitution Ciphers

A substitution cipher replaces one symbol with another. Substitution ciphers can be categorized as either monoalphabetic ciphers or polyalphabetic ciphers.

3.4.1 Monoalphabetical Ciphers

In monoalphabetic substitution, A character in the plaintext is always changed with the same character in the cipher text regardless of its position in the text. the relationship between a symbol in the plaintext to a symbol in the ciphertext is always one-to-one.

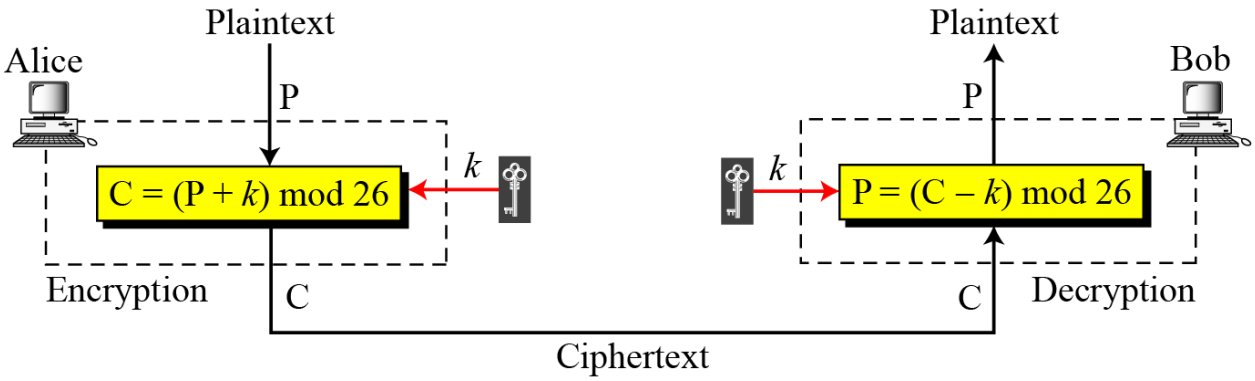
1. **Additive Ciphers**

The simplest monoalphabetic cipher is the additive cipher. This cipher is sometimes called a shift cipher and sometimes a Caesar cipher, but the term additive cipher better reveals its mathematical nature.

Plain text and cipher text mapping in Z26

Plaintext →	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
Ciphertext →	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
Value →	00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

When the cipher is additive, the plaintext, ciphertext, and key are integers in Z_{26}.



Example:

Use the additive cipher with key = 15 to encrypt the message “hello”.

We apply the encryption algorithm to the plaintext, character by character:

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

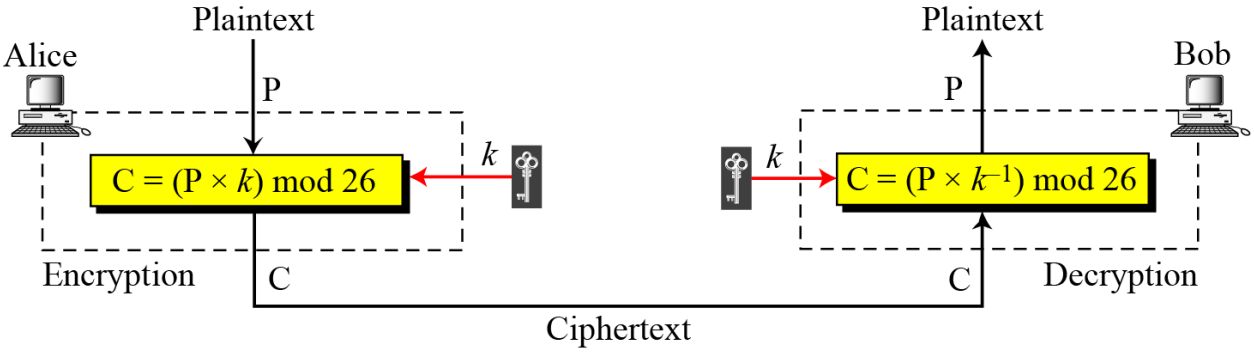
Use the additive cipher with key = 15 to decrypt the message "WTAAD".
We apply the decryption algorithm to the plaintext character by character:

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

Historically, additive ciphers are called shift ciphers. Julius Caesar used an additive cipher to communicate with his officers. For this reason, additive ciphers are sometimes referred to as the Caesar cipher. Caesar used a key of 3 for his communications.

2. Multiplicative Ciphers

In a multiplicative cipher, encryption algorithm specifies multiplication of the plain text by the key and the decryption algorithm specifies division or multiplication with multiplicative inverse of the key with the cipher text. the plaintext and ciphertext are integers in Z_{26}, the key is an integer in Z_{26}*.



The key needs to be in Z_{26}*. This set has only 12 members: 1, 3, 5, 7, 9, 11, 15, 17, 19, 21, 23, 25.

The inverse pairs are

Inverses mod 26												
x	1	3	5	7	9	11	15	17	19	21	23	25
x^{-1}	1	9	21	15	3	19	7	23	11	5	17	25

Example

Use a multiplicative cipher to encrypt the message "hello" with a key of 7. The ciphertext is "XCZZU".

Plaintext: h → 07	Encryption: $(07 \times 07) \bmod 26$	ciphertext: 23 → X
Plaintext: e → 04	Encryption: $(04 \times 07) \bmod 26$	ciphertext: 02 → C
Plaintext: l → 11	Encryption: $(11 \times 07) \bmod 26$	ciphertext: 25 → Z
Plaintext: l → 11	Encryption: $(11 \times 07) \bmod 26$	ciphertext: 25 → Z
Plaintext: o → 14	Encryption: $(14 \times 07) \bmod 26$	ciphertext: 20 → U

3.Affine Cipher

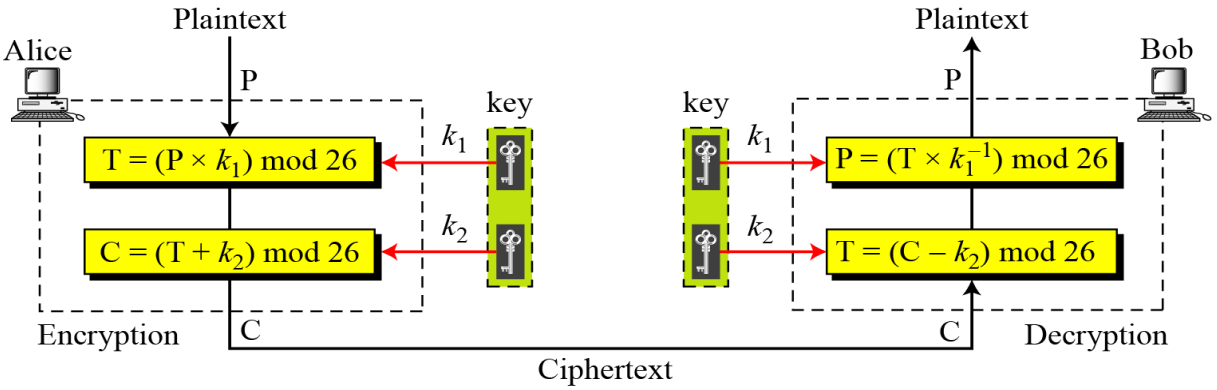
We combine Additive cipher and Multiplicative cipher to get Affine cipher. It uses a combination of both ciphers and a pair of keys. The first key is used with the multiplicative cipher, the second key is used with the additive cipher.

The encryption and decryption is done using the following formulas

$$C = (P \times k_1 + k_2) \bmod 26 \qquad P = ((C - k_2) \times k_1^{-1}) \bmod 26$$

where k_1^{-1} is the multiplicative inverse of k_1 and $-k_2$ is the additive inverse of k_2

The Encryption and Decryption process in affine cipher is



The affine cipher uses a pair of keys in which the first key is from Z_{26}* and the second is from Z_{26}. The size of the key domain is 26 × 12 = 312. The additive cipher is a special case of an affine cipher in which $k_1 = 1$. The multiplicative cipher is a special case of affine cipher in which $k_2 = 0$.

Example

Use an affine cipher to encrypt the message “hello” with the key pair(7,2).

Encryption.

We use 7 for the multiplicative key and 2 for the additive key.

P: h → 07	Encryption: $(07 \times 7 + 2) \bmod 26$	C: 25 → Z
P: e → 04	Encryption: $(04 \times 7 + 2) \bmod 26$	C: 04 → E
P: l → 11	Encryption: $(11 \times 7 + 2) \bmod 26$	C: 01 → B
P: l → 11	Encryption: $(11 \times 7 + 2) \bmod 26$	C: 01 → B
P: o → 14	Encryption: $(14 \times 7 + 2) \bmod 26$	C: 22 → W

Decryption:

The inverse of key 7 is 15 in Z26

C: Z → 25	Decryption: $((25-2)\times 7^{-1}) \bmod 26$	P:07 → h
C: E → 04	Decryption: $((04-2)\times 7^{-1}) \bmod 26$	P:04 → e
C: B → 01	Decryption: $((01-2)\times 7^{-1}) \bmod 26$	P:11 → l
C: B → 01	Decryption: $((01-2)\times 7^{-1}) \bmod 26$	P:11 → l
C: W → 22	Decryption: $((22-2)\times 7^{-1}) \bmod 26$	P:14 → o

4.Monoalphabetic substitution Cipher

Because additive, multiplicative and affine ciphers have small key domains, they are very vulnerable to brute force attack. After alice and bob agreed to a single key, that key is used to encrypt each letter in the plaintext or decrypt each letter in the ciphertext. A better solution is to create a mapping between each plaintext/ciphertext and the corresponding ciphertext character.

The key size of the monoalphabetic cipher is 26! Which is difficult for brute force attack.

Ex:

Plaintext →	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
Ciphertext →	N	O	A	T	R	B	E	C	F	U	X	D	Q	G	Y	L	K	H	V	I	J	M	P	Z	S	W

From this table we can encrypt the following message to ciphertext.

this message is easy to encrypt but hard to find the key

ICFVQRVVNEFVRNVSIYRGAHSLIOJICNHTIYBFGTICRXRS

3.4.2 Polyalphabetic Ciphers

In this type of ciphers, each occurrence of a character may have a different substitute. The relationship between a character in the plaintext to a character in the ciphertext is one to many. Ex: 'a' character could be enciphered as 'X' in the beginning, 'M' in the middle and 'T' at the end. These ciphers have the advantage of hiding the frequency of the underlying language.

1.Autokey Cipher

In this cipher the key is a stream of subkeys, in which each subkey is used to encrypt the corresponding character in the plaintext. The first subkey is a predetermined value secretely agreed between alice and bob. The second subkey is the value of the first plain text character. The third subkey is the value of the second plaintext and so on.

$$P = P_1P_2P_3 \ldots \qquad C = C_1C_2C_3\ldots \qquad k = (k_1, P_1, P_2, \ldots)$$

$$\text{Encryption: } C_i = (P_i + k_i) \bmod 26 \qquad \text{Decryption: } P_i = (C_i - k_i) \bmod 26$$

Assume that Alice and Bob agreed to use an autokey cipher with initial key value k_1 = 12. Now Alice wants to send Bob the message "Attack is today". Enciphering is done character by character.

Plaintext:	a	t	t	a	c	k	i	s	t	o	d	a	y
P's Values:	00	19	19	00	02	10	08	18	19	14	03	00	24
Key stream:	*12*	*00*	*19*	*19*	*00*	*02*	*10*	*08*	*18*	*19*	*14*	*03*	*00*
C's Values:	12	19	12	19	02	12	18	00	11	7	17	03	24
Ciphertext:	**M**	**T**	**M**	**T**	**C**	**M**	**S**	**A**	**L**	**H**	**R**	**D**	**Y**

2.Playfair cipher

It is the cipher used by the british army during the world war I. the secret key inn this cipher is made of 25 alphabet letters arranged in a 5 x 5 matrix(I and J are considered same). Different arrangements of the letters in the matrix can create many different secret keys

Example:

Let us take the key as "MONARCHY". The key is arranged in the following manner.

M	O	N	A	R
C	H	Y	B	D
E	F	G	I	K
L	P	Q	S	T
U	V	W	X	Z

Algorithm to encrypt the plain text: The plaintext is split into pairs of two letters (digraphs).

If there is an odd number of letters, a Z is added to the last letter.

For example:PlainText: "instruments"

After Split: 'in' 'st' 'ru' 'me' 'nt' 'sz'

Rules for Encryption:

1. If both the letters are in the same column: Take the letter below each one (going back to the top if at the bottom).

M	O	N	A	R
C	H	Y	B	D
E	F	G	I	K
L	P	Q	S	T
U	V	W	X	Z

For example:

Diagraph: "me"

Encrypted Text: cl

Encryption: m -> c e -> l

2.If both the letters are in the same row: Take the letter to the right of each one (going back to the leftmost if at the rightmost position).

For example:

Diagraph: "st" Encrypted Text: tl

Encryption: s -> t t -> l

3.If neither of the above rules is true: Form a rectangle with the two letters and take the letters on the horizontal opposite corner of the rectangle.

For example:

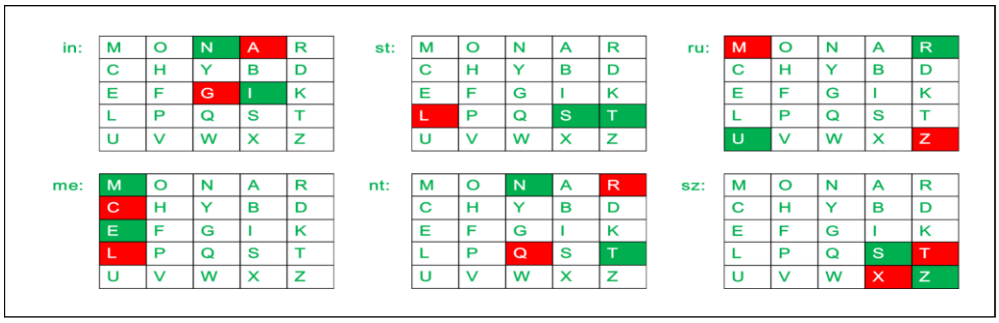
Diagraph: "nt" Encrypted Text: rq

Encryption: n -> r t -> q

"instruments" is encrypted as

i -> g n -> a s -> t t -> l r -> m u -> z m -> c e -> l n -> r t -> q s -> t z -> x

Encrypted Text: gatlmzclrqtx



3.Vigenere Cipher

A sixteenth century mathematician Blaise de Vigener designed Vigenere cipher. It uses a different strategy to create the key stream. The key stream is a repetition of an initial secretkey stream of length m, where 1<=m<=26. The cipher can be described as follows where (k1,k2,k3….km) is the initial secret key agreed to be Alice and Bod.

$$P = P_1P_2P_3 \ldots \qquad C = C_1C_2C_3 \ldots \qquad K = [(k_1, k_2, \ldots, k_m), (k_1, k_2, \ldots, k_m), \ldots]$$

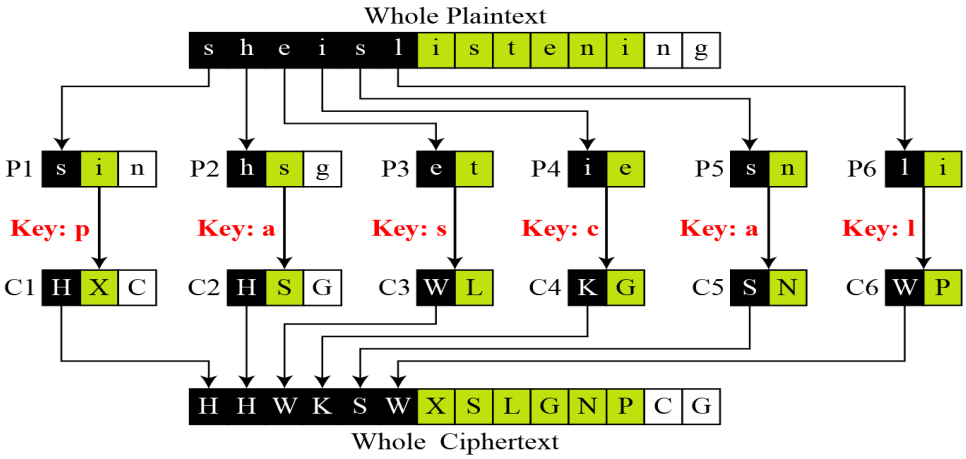
$$\text{Encryption: } C_i = P_i + k_i \qquad \text{Decryption: } P_i = C_i - k_i$$

Example:

We can encrypt the message “She is listening” using the 6-character keyword “PASCAL”.

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**

Vigenere cipher can be seen as combinations of m additive ciphers.



we can say that the additive cipher is a special case of Vigenere cipher in which *m* = 1.

	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

The first row shows the plain text character to be encrypted. The first column contains the characters to be used by the key. The rest of the tableau shows the ciphertext characters.

Cryptanalysis of vigenere ciphers

1. Kasiski test, is used to find the length of the key, the cryptanalyst searches for repeated text segments, of at least three characters in ciphertext.
2. She divides the ciphertext into m different pieces and applies the method used to cryptanalyse the additive cipher, including frequency attack. Each ciphertext piece can be decrypted and put together to create the whole plaintext.

LIOMWGFEGGDVWGHHCQUCRHRWAGWIOWQLKGZETKKMEVLWPCZVGTH-
VTSGXQOVGCSVETQLTJSUMVWVEUVLXEWSLGFZMVVWLGYHCUSWXQH-
KVGSHEEVFLCFDGVSUMPHKIRZDMPHHBVWVWJWIXGFWLTSHGJOUEEHH-
VUCFVGOWICQLTJSUXGLW

The Kasiski test for repetition of three-character segments yields the results shown in Table 3.

String	*First Index*	*Second Index*	*Difference*
JSU	68	168	100
SUM	69	117	48
VWV	72	132	60
MPH	119	127	8

3. Find the gcd of differences is 4, which means that the key length is multiple of 4.

C1: **LWGWCRAOKTEPGTQCTJVUEGVGUQGECVPRPVJGTJEUGCJG**
P1: *jueuapymircneroarhtsthihytrahcieixsthcarrehe*
C2: **IGGGQHGWGKVCTSOSQSWVWFVYSHSVFSHZHWWFSOHCOQSL**
P2: *usssctsiswhofeaeceihcetesoecatnpntherhctecex*
C3: **OFDHURWQZKLZHGVVLUVLSZWHWKHFDUKDHVIWHUHFWLUW**
P3: *lcaerotnwhiwedssirsiirhketehretltiideatrairt*
C4: **MEVHCWILEMWVVXGETMEXLMLCXVELGMIMBWXLGEVVITX**
P4: *iardysehaisrrtcapiafpwtethecarhaesfterectpt*

4.Hill Cipher

Hill cipher is a polyalphabetic cipher invented by Lester S Hill. The plain text is divided into equal size blocks. The blocks are encrypted one at a time in such a way that each character in the block . the hill ciphers belongs to block ciphers.

In a Hill cipher, the key is a square matrix of size m x m in which m is the size of the block.

The m characters in the plaintext block P1,P2,P3….Pm , the corresponding characters in the ciphertext block are C1,C2,C3….Cm.

$$\mathrm{K} = \begin{bmatrix} k_{11} & k_{12} & \cdots & k_{1m} \\ k_{21} & k_{22} & \cdots & k_{2m} \\ \vdots & \vdots & & \vdots \\ k_{m1} & k_{m2} & \cdots & k_{mm} \end{bmatrix}$$

$$\begin{aligned} \mathrm{C}_1 &= \mathrm{P}_1 k_{11} + \mathrm{P}_2 k_{21} + \cdots + \mathrm{P}_m k_{m1} \\ \mathrm{C}_2 &= \mathrm{P}_1 k_{12} + \mathrm{P}_2 k_{22} + \cdots + \mathrm{P}_m k_{m2} \\ &\cdots \\ \mathrm{C}_m &= \mathrm{P}_1 k_{1m} + \mathrm{P}_2 k_{2m} + \cdots + \mathrm{P}_m k_{mm} \end{aligned}$$

The equations show that each ciphertext character such as C1 depends on all plaintext characters in the block (P1,P2,P3,….Pm). The Key matrix K should have a Multiplicative Inverse.

Example 1: The plaintext “code is ready” can make a 3 × 4 matrix when adding extra bogus character “z” to the last block and removing the spaces. The ciphertext is “OHKNIHGKLISS”.

$$\underset{\mathrm{C}}{\begin{bmatrix} 14 & 07 & 10 & 13 \\ 08 & 07 & 06 & 11 \\ 11 & 08 & 18 & 18 \end{bmatrix}} = \underset{\mathrm{P}}{\begin{bmatrix} 02 & 14 & 03 & 04 \\ 08 & 18 & 17 & 04 \\ 00 & 03 & 24 & 25 \end{bmatrix}} \underset{\mathrm{K}}{\begin{bmatrix} 09 & 07 & 11 & 13 \\ 04 & 07 & 05 & 06 \\ 02 & 21 & 14 & 09 \\ 03 & 23 & 21 & 08 \end{bmatrix}}$$

a. Encryption

$$\underset{\mathrm{P}}{\begin{bmatrix} 02 & 14 & 03 & 04 \\ 08 & 18 & 17 & 04 \\ 00 & 03 & 24 & 25 \end{bmatrix}} = \underset{\mathrm{C}}{\begin{bmatrix} 14 & 07 & 10 & 13 \\ 08 & 07 & 06 & 11 \\ 11 & 08 & 18 & 18 \end{bmatrix}} \underset{\mathrm{K}^{-1}}{\begin{bmatrix} 02 & 15 & 22 & 03 \\ 15 & 00 & 19 & 03 \\ 09 & 09 & 03 & 11 \\ 17 & 00 & 04 & 07 \end{bmatrix}}$$

b. Decryption

Example2:Encrypt the message ‘act’ (n=3). The key is ‘GYBNQKURP’ which can be written as the nxn matrix:

$$\begin{bmatrix} 6 & 24 & 1 \\ 13 & 16 & 10 \\ 20 & 17 & 15 \end{bmatrix}$$

The message ‘act’ is written as vector:

$$\begin{bmatrix} 0 \\ 2 \\ 19 \end{bmatrix}$$

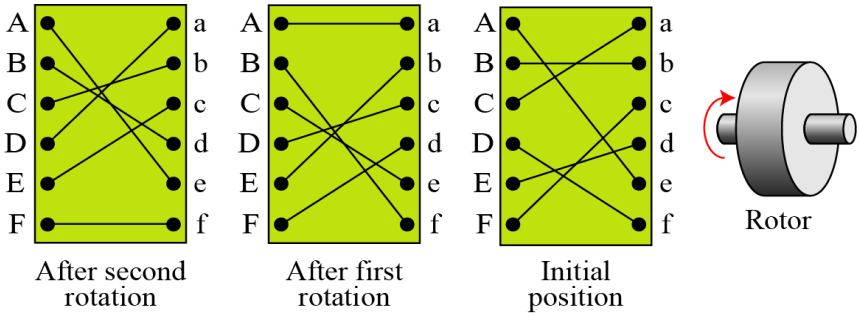
The process of Encryption is

$$\begin{bmatrix} 6 & 24 & 1 \\ 13 & 16 & 10 \\ 20 & 17 & 15 \end{bmatrix} \begin{bmatrix} 0 \\ 2 \\ 19 \end{bmatrix} = \begin{bmatrix} 67 \\ 222 \\ 319 \end{bmatrix} \equiv \begin{bmatrix} 15 \\ 14 \\ 7 \end{bmatrix} \pmod{26}$$

which corresponds to ciphertext of 'POH'

5.Rotor Cipher

Rotor ciphers uses the idea of monoalphabetic substitution cipher but changes the mapping between the plain text and the cipher text characters for each plain text character,

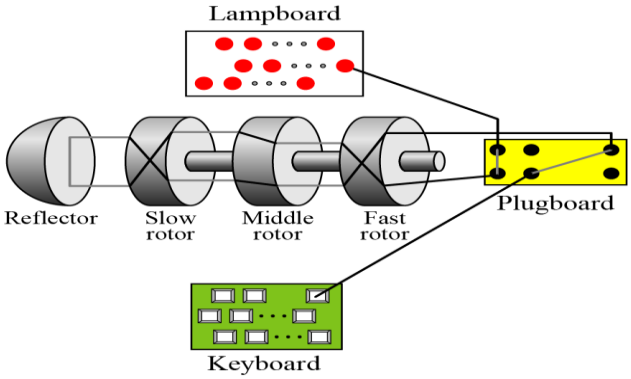


The rotor shown in the diagram uses only 6 characters but in actual rotor use 26 characters. The initial setting of the rotor is the secret key between Alice and Bob. The first plian text character is encrypted using the initial setting. The second character is encrypted using the first rotation and the third character is encrypted using second rotation.

For example a three character word "bee" is encrypted as "BCA". Where 'b' is mapped to 'B" in initial position, 'e' is mapped to 'C' in first rotation, 'e' is mapped to 'A' in second rotation. This shows that the rotor cipher is a polyalphabetic cipher because two occurances of the same character are encrypted as different characters. Rotor machines are resistant to brute force attack and statistical attack.

6.Enigma Machine

It was originally invented by sherbius, but was modified by the german army and used during world word II. This machine is based on the rotor machines.



The main concepts of the machine are

1. A keyboard with 26 keys used for entering the plaintext when encrypting and for entering the ciphertext when decrypting
2. A lampboard with 26 lamps that shows the ciphertext characters in encrypting and the plain text characters in decrypting

3. A plugboard with 26 plugs manually connected by 13 wires
4. Three wired rotors which are selected from 5 available rotors.
5. A reflector, which is stationary and prewired

Procedure for encrypting a message

1. Set the starting position of the rotors to the code of the day Ex: HUS
2. Choose a random three letter code Ex: ACF. Encrypt the text "ACFACF" using the initial settingof the rotors. Assume it is OPNABT.
3. Set the starting position of the rotors to OPN
4. Append the encrypted six letters obtained from step2 to the beginning of the message
5. Encrypt the message including the 6 letter code. Send the encrypted message

Procedure for decrypting a message

1. Receive the message and separate the first 6 letters
2. Set the starting position of the rotors to the code of the day
3. Decrypt the first six letters using the initial setting in step2.
4. Set the positions of the rotors to tht first half of the decrypted code
5. Decrypt the message.

3.5 Transposition Cipher

A transposition cipher does not substitute one symbol for another, instead it changes the location of the symbols. A transposition cipher reorders symbols.

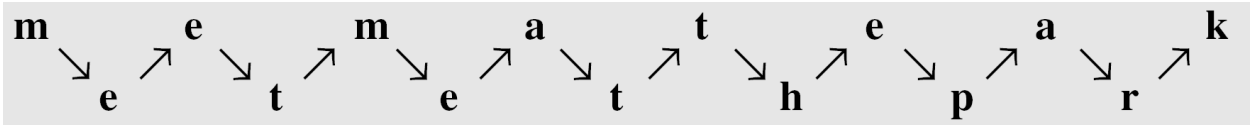
There are two types of Transposition ciphers

3.5.1 Keyless Transposition ciphers:

Simple transposition ciphers, which were used in the past, are keyless. There are two methods for permutation of characters. In the first method text is written into a table column by column and then transmitted row by row. In the second method , the text is written into the table row by row and then transmitted column by column

Example 1: Rail Fence cipher

A good example of a keyless cipher using the first method is the rail fence cipher. The ciphertext is created reading the pattern row by row. For example, to send the message "Meet me at the park" to Bob, Alice writes



She then creates the ciphertext "MEMATEAKETETHPR".

Example 2: Alice and Bob can agree on the number of columns and use the second method. Alice writes the same plaintext, row by row, in a table of four columns.

m	e	e	t
m	e	a	t
t	h	e	p
a	r	k	

She then creates the ciphertext “MMTAEEHREAEKTTP”.

Example3: The cipher in Example 3.23 is actually a transposition cipher. The following shows the permutation of each character in the plaintext into the ciphertext based on the positions.

01	02	03	04	05	06	07	08	09	10	11	12	13	14	15
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
01	05	09	13	02	06	10	13	03	07	11	15	04	08	12

The second character in the plaintext has moved to the fifth position in the ciphertext, the third character has moved to the ninth position; and so on. Although the characters are permuted,
there is a pattern in the permutation: (01, 05, 09, 13), (02, 06, 10, 13), (03, 07, 11, 15), and (08, 12). In each section, the difference between the two adjacent numbers is 4.

3.5.2 Keyed Transposition Ciphers

The keyless ciphers permute the characters by using writing plaintext in one way and reading it in another way The permutation is done on the whole plaintext to create the whole ciphertext. Another method is to divide the plaintext into groups of predetermined size, called blocks, and then use a key to permute the characters in each block separately.

Alice needs to send the message “Enemy attacks tonight” to Bob..

e	n	e	m	y		a	t	t	a	c		k	s	t	o	n		i	g	h	t	z

The key used for encryption and decryption is a permutation key, which shows how the character are permuted.

Encryption ↓

3	1	4	5	2
1	2	3	4	5

↑ Decryption

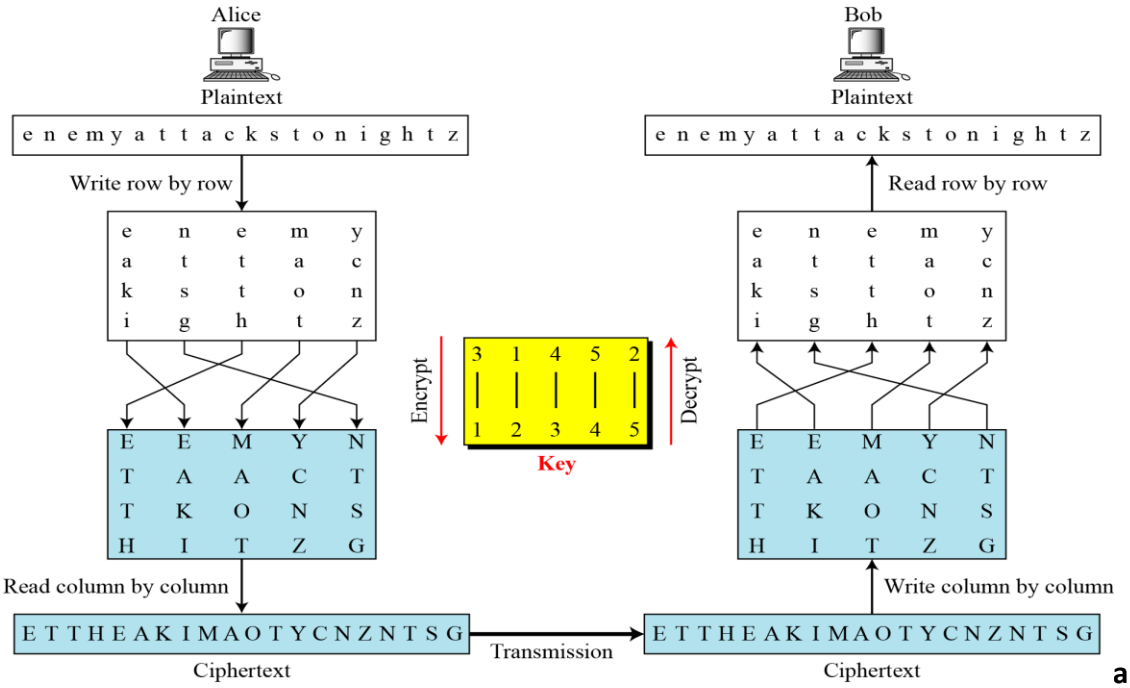
The permutation yields

E	**E**	**M**	**Y**	**N**		**T**	**A**	**A**	**C**	**T**		**T**	**K**	**O**	**N**	**S**		**H**	**I**	**T**	**Z**	**G**

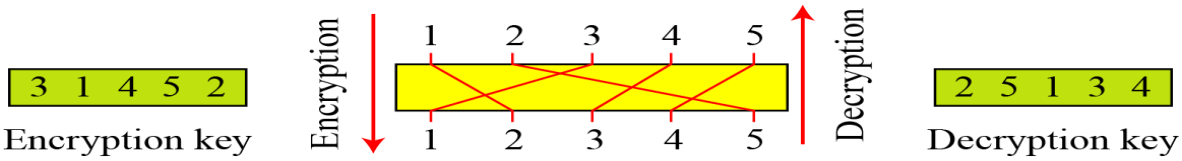
3.5.3Combining Two Approaches

Most recent transposition ciphers combine the two approaches to achieve better scrambling. Enryption or decryption is done in three steps.

1. The text is written into a table row by row
2. The permutation is done by reordering the columns
3. The new table is read column by column.



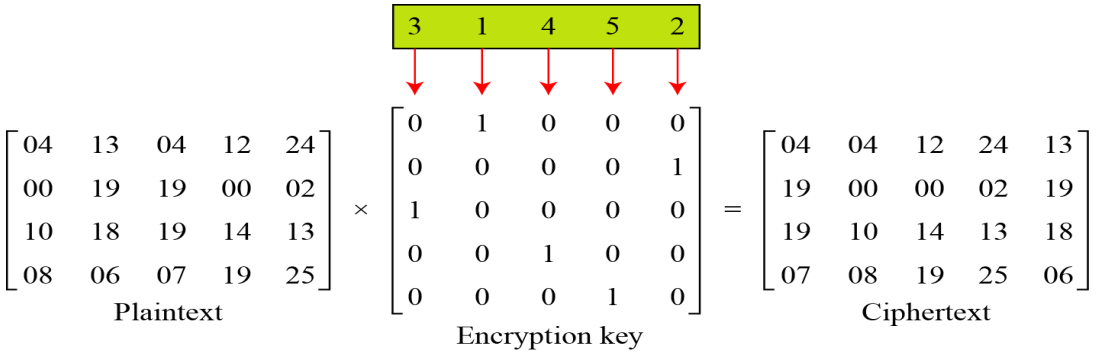
a single key was used in two directions for the column exchange: downward for encryption, upward for decryption. It is customary to create two keys.



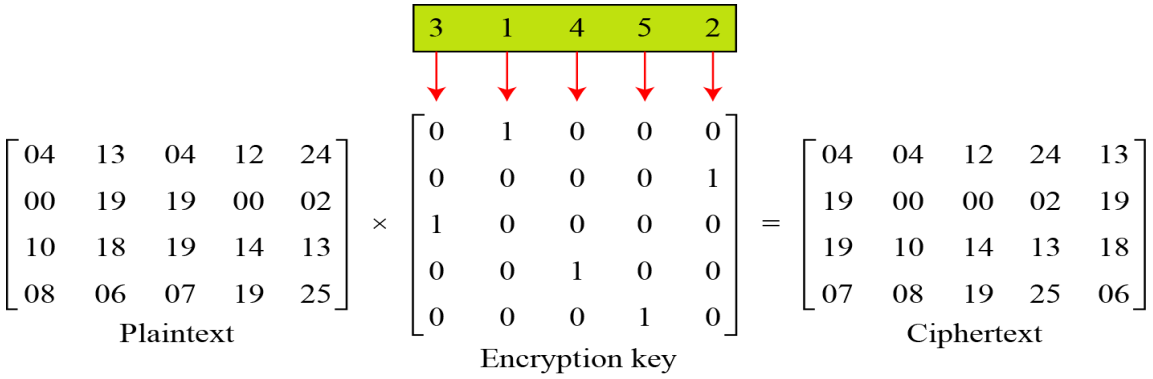
3.5.4 Implementing Transposition ciphers using Matrices

The plain text and cipher text are l x m matrices representing the numerical values of the characters. The keys are square matrices of size m x m. In a permutation matrix, every row or column has exactly one 1 and the rest of the values are 0. Encryption is performed by multiplying the plaintext matrix by the key matrix to get the ciphertext matrix. Decryption is performed by multiplying the ciphertext bythe inverse of the key matrix to get the plaintext.

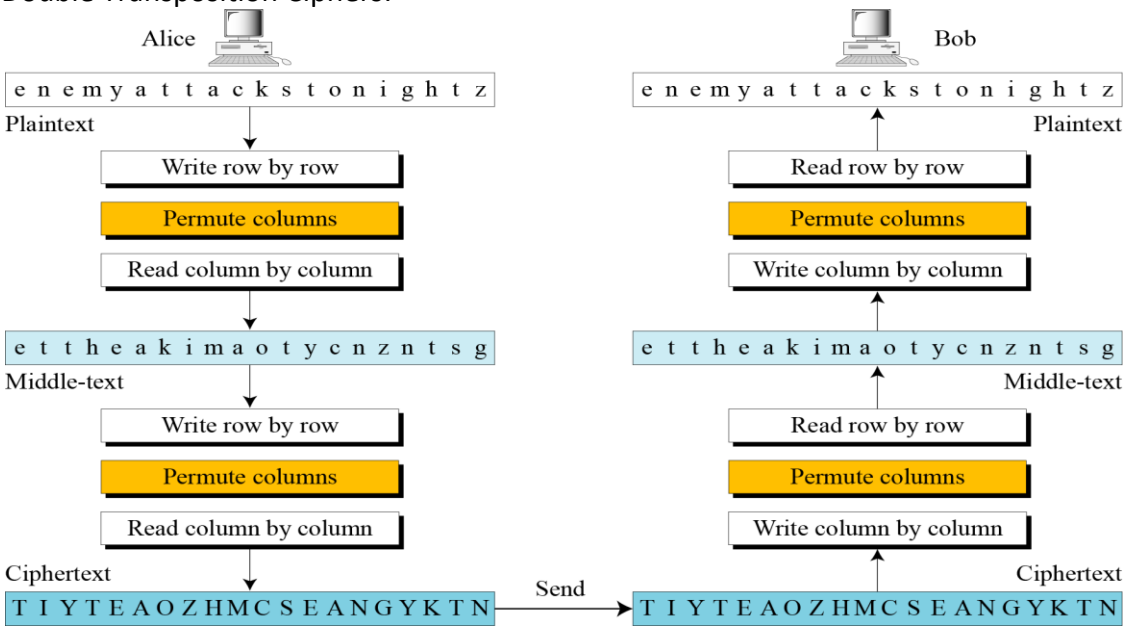
Encryption:



Decryption:



3.5.5 Double Transposition Ciphers:

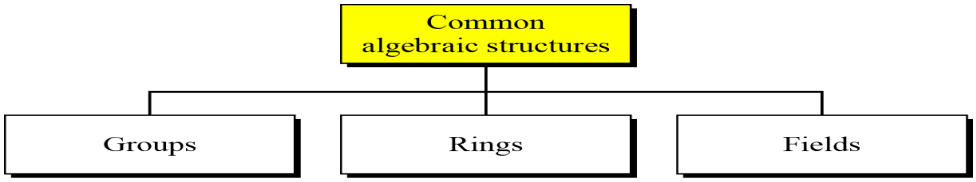


Chapter-4

Mathematics of Cryptography-Part-II Algebraic Structures

4.1 Algebraic Structures

Cryptography requires sets of integers and specific operations that are defined for those sets. The combination of the set and the operations that are applied to the elements of the set is called an algebraic structure. There are three types of Algebraic Structures



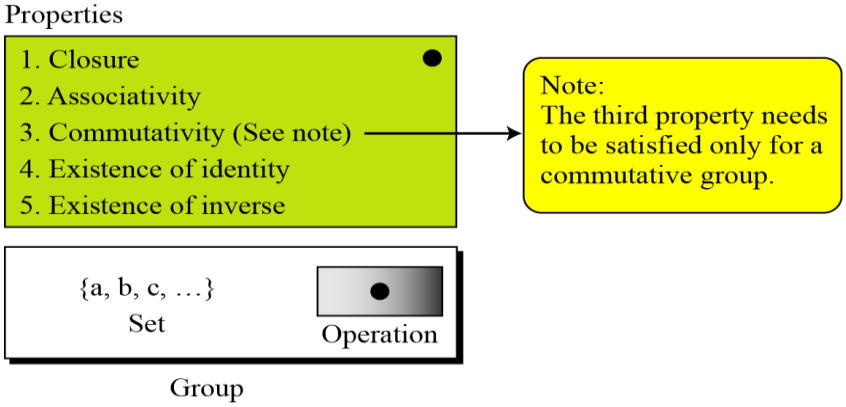
Group

A group (G) is a set of elements with a binary operation (•) that satisfies four properties (or axioms).

1. Closure: if a and b are elements of G, then c=a.b is also an element of G.
2. Associativity: if a,b and c are elements of G, then (a.b).c=a.(b.c)
3. Existence of Identity: for all a in G, there exists an element e, called the identity element, such that e.a=a.e=a
4. Existence of Inverse: for each a in G, there exists an element b, called the inverse of a, such that a.b=b.a=e

A commutative group or abelian group satisfies an extra property,

Commutativity: for all a,b in G, we have a.b=b.a.



Although a group involves a single operation, the properties imposed on the operation allow the use of a pair of operations as long as they are inverses of each other.

Example1: The set of residue integers with the addition operator,

$$G = < Z_n , + >,$$

is a commutative group. We can perform addition and subtraction on the elements of this set without moving out of the set.

1. Closure is satisfied. The result of adding two integers is always an integer.
2. Associativity is satisfied. The result of 4+(3+2) is same as (4+3)+2
3. Commutatitivity is satisfied. We have 3+4=4+3.
4. The identity element is 0
5. Every element has an additive inverse. The inverse of 3 is n-3.

Example2: The set Z_n^* with the multiplication operator, $G = <Z_n^*, \times>$, is also an abelian group.

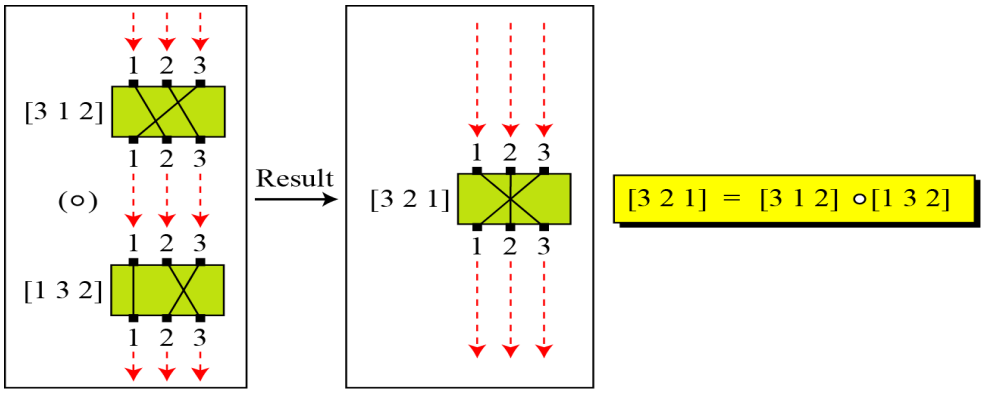
Example 3:Let us define a set G = < {a, b, c, d}, •> and the operation as shown in Table 4.1.

•	*a*	*b*	*c*	*d*
a	*a*	*b*	*c*	*d*
b	*b*	*c*	*d*	*a*
c	*c*	*d*	*a*	*b*
d	*d*	*a*	*b*	*c*

1. Closure is satisfied. Applying the operation on pair of elements result in another element in the set
2. Associativity is also satisfied
3. The operation is commutative
4. The group has an identity element a
5. Each element has an inverse. The pairs are (a,a),(b,d),(c,c)

Permutation Group

A very interesting group is the permutation group. The set is the set of all permutations, and the operation is composition: applying one permutation after another.



The operation table of permutation is

∘	[1 2 3]	[1 3 2]	[2 1 3]	[2 3 1]	[3 1 2]	[3 2 1]
[1 2 3]	[1 2 3]	[1 3 2]	[2 1 3]	[2 3 1]	[3 1 2]	[3 2 1]
[1 3 2]	[1 3 2]	[1 2 3]	[2 3 1]	[2 1 3]	[3 2 1]	[3 1 2]
[2 1 3]	[2 1 3]	[3 1 2]	[1 2 3]	[3 2 1]	[1 3 2]	[2 3 1]
[2 3 1]	[2 3 1]	[3 2 1]	[1 3 2]	[3 1 2]	[1 2 3]	[2 1 3]
[3 1 2]	[3 1 2]	[2 1 3]	[3 2 1]	[1 2 3]	[2 3 1]	[1 3 2]
[3 2 1]	[3 2 1]	[2 3 1]	[3 1 2]	[1 3 2]	[2 1 3]	[1 2 3]

In the previous example, we showed that a set of permutations with the composition operation is a group. This implies that using two permutations one after another cannot strengthen the security of a cipher, because we can always find a permutation that can do the same job because of the closure property.

Finite Group: A group is called a finite group if the set has a finite number of elements, otherwise it is an infinite group

Order of a group: the order of the group |G| is the number of elements in the group. If the group is not finite, its order is infinite. If the group is finite, the order is finite.

Subgroup: A subset H of a group G is a subgroup of G if H itself is a group with respect to the operation on G.

1. If a and b are members of both groups, then c=a.b is also the member of both groups
2. The group share the same identity.
3. If a is a member of both groups, the inverse of a is also a member of the both groups.
4. The group made of the identity element of G, H=<{e},.> is a subgroup of G.
5. Each group is a subgroup of itself

Cyclic Subgroups

If a subgroup of a group can be generated using the power of an element, the subgroup is called the cyclic subgroup. The term power represents repeatedly applying the group operation to the element

a^n=a.a.a.a……n times

Example

Four cyclic subgroups can be made from the group G = <Z_6, +>. They are H_1 = <{0}, +>, H_2 = <{0, 2, 4}, +>, H_3 = <{0, 3}, +>, and H_4 = G.

$$0^0 \bmod 6 = 0$$

$$1^0 \bmod 6 = 0$$
$$1^1 \bmod 6 = 1$$
$$1^2 \bmod 6 = (1 + 1) \bmod 6 = 2$$
$$1^3 \bmod 6 = (1 + 1 + 1) \bmod 6 = 3$$
$$1^4 \bmod 6 = (1 + 1 + 1 + 1) \bmod 6 = 4$$
$$1^5 \bmod 6 = (1 + 1 + 1 + 1 + 1) \bmod 6 = 5$$

$$2^0 \bmod 6 = 0$$
$$2^1 \bmod 6 = 2$$
$$2^2 \bmod 6 = (2 + 2) \bmod 6 = 4$$

$$3^0 \bmod 6 = 0$$
$$3^1 \bmod 6 = 3$$

$$4^0 \bmod 6 = 0$$
$$4^1 \bmod 6 = 4$$
$$4^2 \bmod 6 = (4 + 4) \bmod 6 = 2$$

$$5^0 \bmod 6 = 0$$
$$5^1 \bmod 6 = 5$$
$$5^2 \bmod 6 = 4$$
$$5^3 \bmod 6 = 3$$
$$5^4 \bmod 6 = 2$$
$$5^5 \bmod 6 = 1$$

Example:

Three cyclic subgroups can be made from the group G = <Z_{10}*, ×>. G has only four elements: 1, 3, 7, and 9. The cyclic subgroups are H_1 = <{1}, ×>, H_2 = <{1, 9}, ×>, and H_3 = G.

$$1^0 \bmod 10 = 1$$

$$3^0 \bmod 10 = 1$$
$$3^1 \bmod 10 = 3$$
$$3^2 \bmod 10 = 9$$
$$3^3 \bmod 10 = 7$$

$$7^0 \bmod 10 = 1$$
$$7^1 \bmod 10 = 7$$
$$7^2 \bmod 10 = 9$$
$$7^3 \bmod 10 = 3$$

$$9^0 \bmod 10 = 1$$
$$9^1 \bmod 10 = 9$$

Cyclic Group

A cyclic group is a group that is its own cyclic subgroup. The element with which cyclic group is generated is called generator. If g is a generator, the elements in a finite group can be written as.

$$\{e, g, g^2, \ldots, g^{n-1}\}, \text{ where } g^n = e$$

Lagrange's theorem

Assume that G is a group, and H is a subgroup of G. If the order of G and H are |G| and |H|, respectively, then, based on this theorem, |H| divides |G|.

Order of the element:

The order of an element is the order of the cyclic group it generates.

Example:

a. In the group G = $<Z_6, +>$, the orders of the elements are: ord(0) = 1, ord(1) = 6, ord(2) = 3, ord(3) = 2, ord(4) = 3, ord(5) = 6.

b. In the group G = $<Z_{10}^*, \times>$, the orders of the elements are: ord(1) = 1, ord(3) = 4, ord(7) = 4, ord(9) = 2.

Ring:

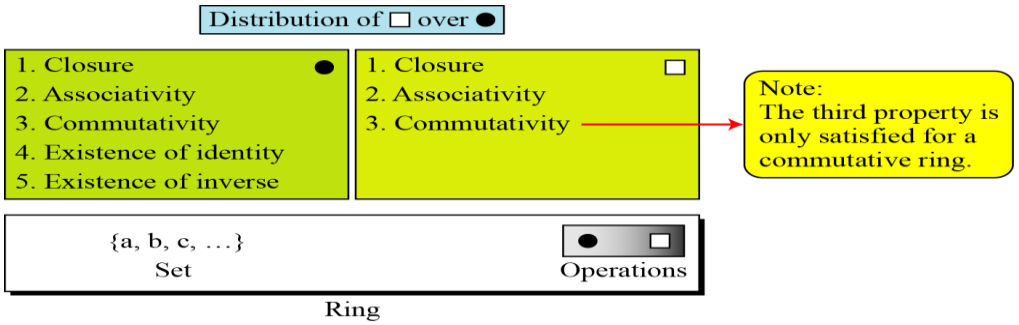
A ring, R = <{…}, •,□ >, is an algebraic structure with two operations. The first operation must satisfy all five properties required for an abelian group. The second operation must satisfy the first two. In addition second operation must be distributed over the first.

a□(b•c)=(a□b)• (a□c)

A commutative ring is a ring in which the commutative property is also satisfied for the second operation.

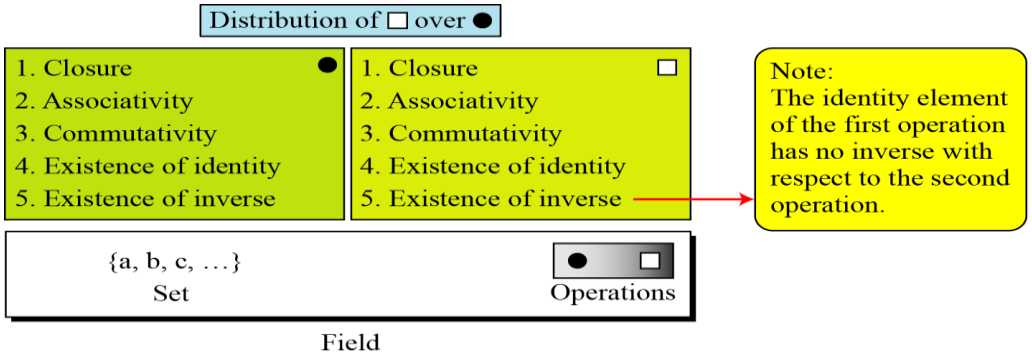
Example:

The set Z with two operations, addition and multiplication, is a commutative ring. We show it by R = <Z, +, ×>. Addition satisfies all of the five properties; multiplication satisfies only three properties.



Field:

A field, denoted by F = <{…}, •,□ > is a commutative ring in which the second operation satisfies all five properties defined for the first operation except that the identity of the first operation has no inverse.



Finite Fields

Galois showed that for a field to be finite, the number of elements should be p^n, where p is a prime and n is a positive integer.

A Galois field, GF(p^n), is a finite field with p^n elements.

When n = 1, we have GF(p) field. This field can be the set Z_p, {0, 1, …, p − 1}, with two arithmetic operations.

Example:GF(2)

A very common field in this category is GF(2) with the set {0, 1} and two operations, addition and multiplication

GF(2)

{0, 1} + ×

+	0	1
0	0	1
1	1	0

Addition

×	0	1
0	0	0
1	0	1

Multiplication

a	0	1
−a	1	0

a	0	1
a^{-1}	—	1

Inverses

Example: GF(5)

We can define GF(5) on the set Z_5 (5 is a prime) with addition and multiplication operators .

GF(5)

{0, 1, 2, 3, 4} + ×

+	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

Addition

×	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	3	1	4	2
4	0	4	3	2	1

Multiplication

Additive inverse

a	0	1	2	3	4
−a	0	4	3	2	1

a	0	1	2	3	4
a^{-1}	—	1	3	2	4

Multiplicative inverse

4-2 GF(2^n) FIELDS

In cryptography, we often need to use four operations (addition, subtraction, multiplication, and division). In other words, we need to use fields. We can work in GF(2^n) and uses a set of 2^n elements. The elements in this set are n-bit words.

Example: GF(2^2)

Let us define a GF(2^2) field in which the set has four 2-bit words: {00, 01, 10, 11}. We can redefine addition and multiplication for this field in such a way that all properties of these operations are satisfied

Addition

⊕	00	01	10	11
00	00	01	10	11
01	01	00	11	10
10	10	11	00	01
11	11	10	01	00

Identity: 00

Multiplication

⊗	00	01	10	11
00	00	00	00	00
01	00	01	10	11
10	00	10	11	01
11	00	11	01	10

Identity: 01

Polynomials

A polynomial of degree n − 1 is an expression of the form

$$f(x) = a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \cdots + a_1x^1 + a_0x^0$$

where x^i is called the ith term and a_i is called coefficient of the ith term.

Representation of 8 bit polynomial

we can represent the 8-bit word (10011001) using a polynomials.

n-bit word	1	0	0	1	1	0	0	1
Polynomial	$1x^7 + 0x^6 + 0x^5 + 1x^4 + 1x^3 + 0x^2 + 0x^1 + 1x^0$							

First simplification: $1x^7 + 1x^4 + 1x^3 + 1x^0$

Second simplification: $x^7 + x^4 + x^3 + 1$

Example:

To find the 8-bit word related to the polynomial $x^5 + x^2 + x$, we first supply the omitted terms. Since n = 8, it means the polynomial is of degree 7. The expanded polynomial is

$$0x^7 + 0x^6 + 1x^5 + 0x^4 + 0x^3 + 1x^2 + 1x^1 + 0x^0$$

This is related to the 8-bit word 00100110.

Polynomials representing n-bit words use two fields: GF(2) and GF(2^n).

Operations on polynomials

1. Modulus

For the sets of polynomials in GF(2^n), a group of polynomials of degree n is defined as the modulus. Such polynomials are referred to as irreducible polynomials. The list of irreducible polynomials are

Degree	*Irreducible Polynomials*
1	$(x + 1)$, (x)
2	$(x^2 + x + 1)$
3	$(x^3 + x^2 + 1)$, $(x^3 + x + 1)$
4	$(x^4 + x^3 + x^2 + x + 1)$, $(x^4 + x^3 + 1)$, $(x^4 + x + 1)$
5	$(x^5 + x^2 + 1)$, $(x^5 + x^3 + x^2 + x + 1)$, $(x^5 + x^4 + x^3 + x + 1)$, $(x^5 + x^4 + x^3 + x^2 + 1)$, $(x^5 + x^4 + x^2 + x + 1)$

2. Addition or Subtraction of two polynomials

Addition and subtraction operations on polynomials are the same operation.

Let us do ($x^5 + x^2 + x$) XOR ($x^3 + x^2 + 1$) in GF(2^8). We use the symbol Å to show that we mean polynomial addition. The following shows the procedure:

$$\begin{array}{lll} 0x^7 + 0x^6 + 1x^5 + 0x^4 + 0x^3 + 1x^2 + 1x^1 + 0x^0 & \oplus & \\ 0x^7 + 0x^6 + 0x^5 + 0x^4 + 1x^3 + 1x^2 + 0x^1 + 1x^0 & & \\ \hline 0x^7 + 0x^6 + 1x^5 + 0x^4 + 1x^3 + 0x^2 + 1x^1 + 1x^0 & \rightarrow & x^5 + x^3 + x + 1 \end{array}$$

There is also another short cut. Because the addition in GF(2) means the exclusive-or (XOR) operation. So we can exclusive-or the two words, bits by bits, to get the result. In the previous example, $x^5 + x^2 + x$ is 00100110 and $x^3 + x^2 + 1$ is 00001101. The result is 00101011 or in polynomial notation $x^5 + x^3 + x + 1$.

3. Multliplication of two polynomials

1. The coefficient multiplication is done in GF(2).

2. The multiplying x^i by x^j results in x^{i+j}.

3. The multiplication may create terms with degree more than n − 1, which means the result needs to be reduced using a modulus polynomial.

Example: Find the result of ($x^5 + x^2 + x$) ⊗ ($x^7 + x^4 + x^3 + x^2 + x$) in GF($2^8$) with irreducible polynomial ($x^8 + x^4 + x^3 + x + 1$). Note that we use the symbol ⊗ to show the multiplication of two polynomials.

$$P_1 \otimes P_2 = x^5(x^7 + x^4 + x^3 + x^2 + x) + x^2(x^7 + x^4 + x^3 + x^2 + x) + x(x^7 + x^4 + x^3 + x^2 + x)$$
$$P_1 \otimes P_2 = x^{12} + x^9 + x^8 + x^7 + x^6 + x^9 + x^6 + x^5 + x^4 + x^3 + x^8 + x^5 + x^4 + x^3 + x^2$$
$$P_1 \otimes P_2 = (x^{12} + x^7 + x^2) \bmod (x^8 + x^4 + x^3 + x + 1) = x^5 + x^3 + x^2 + x + 1$$

To find the final result, divide the polynomial of degree 12 by the polynomial of degree 8 (the modulus) and keep only the remainder.

$$
\begin{array}{r|l}
 & x^4 + 1 \\
\hline
x^8 + x^4 + x^3 + x + 1 & x^{12} + x^7 + x^2 \\
 & x^{12} + x^8 + x^7 + x^5 + x^4 \\
\hline
 & x^8 + x^5 + x^4 + x^2 \\
 & x^8 + x^4 + x^3 + x + 1 \\
\hline
\text{Remainder} & x^5 + x^3 + x^2 + x + 1
\end{array}
$$

4.Multiplicative Inverse of a polynomial

The extended Euclidian algorithm must be applied to the modulus and the polynomial. The process is exactly the same as the integers

Example1:

In GF (2^4), find the inverse of ($x^2 + 1$) modulo ($x^4 + x + 1$).

q	r_1	r_2	r	t_1	t_2	t
$(x^2 + 1)$	$(x^4 + x + 1)$	$(x^2 + 1)$	(x)	(0)	(1)	$(x^2 + 1)$
(x)	$(x^2 + 1)$	(x)	(1)	(1)	$(x^2 + 1)$	$(x^3 + x + 1)$
(x)	(x)	(1)	(0)	$(x^2 + 1)$	$(x^3 + x + 1)$	(0)
	(1)	(0)		$(x^3 + x + 1)$	(0)	

Example2:

In GF(2^8), find the inverse of (x^5) modulo ($x^8 + x^4 + x^3 + x + 1$).

q	r_1	r_2	r	t_1	t_2	t
(x^3)	$(x^8 + x^4 + x^3 + x + 1)$	(x^5)	$(x^4 + x^3 + x + 1)$	(0)	(1)	(x^3)
$(x + 1)$	(x^5)	$(x^4 + x^3 + x + 1)$	$(x^3 + x^2 + 1)$	(1)	(x^3)	$(x^4 + x^3 + 1)$
(x)	$(x^4 + x^3 + x + 1)$	$(x^3 + x^2 + 1)$	(1)	(x^3)	$(x^4 + x^3 + 1)$	$(x^5 + x^4 + x^3 + x)$
$(x^3 + x^2 + 1)$	$(x^3 + x^2 + 1)$	(1)	(0)	$(x^4 + x^3 + 1)$	$(x^5 + x^4 + x^3 + x)$	(0)
	(1)	(0)		$(x^5 + x^4 + x^3 + x)$	(0)	

Multiplication using computers:

The computer implementation uses a better algorithm, repeatedly multiplying a reduced polynomial by x.

Example1:

Find the result of multiplying $P_1 = (x^5 + x^2 + x)$ by $P_2 = (x^7 + x^4 + x^3 + x^2 + x)$ in GF(2^8) with irreducible polynomial ($x^8 + x^4 + x^3 + x + 1$) .

Solution:

We first find the partial result of multiplying x^0, x^1, x^2, x^3, x^4, and x^5 by P_2. Note that although only three terms are needed, the product of $x^m \otimes P_2$ for m from 0 to 5 because each calculation depends on the previous result.

Powers	*Operation*	*New Result*	*Reduction*
$x^0 \otimes P_2$		$x^7 + x^4 + x^3 + x^2 + x$	No
$x^1 \otimes P_2$	$x \otimes (x^7 + x^4 + x^3 + x^2 + x)$	$x^5 + x^2 + x + 1$	**Yes**
$x^2 \otimes P_2$	$x \otimes (x^5 + x^2 + x + 1)$	$x^6 + x^3 + x^2 + x$	No
$x^3 \otimes P_2$	$x \otimes (x^6 + x^3 + x^2 + x)$	$x^7 + x^4 + x^3 + x^2$	No
$x^4 \otimes P_2$	$x \otimes (x^7 + x^4 + x^3 + x^2)$	$x^5 + x + 1$	**Yes**
$x^5 \otimes P_2$	$x \otimes (x^5 + x + 1)$	$x^6 + x^2 + x$	No
$\mathbf{P_1 \times P_2} = (x^6 + x^2 + x) + (x^6 + x^3 + x^2 + x) + (x^5 + x^2 + x + 1) = \mathbf{x^5 + x^3 + x^2 + x + 1}$			

$P_1 = (x^5 + x^2 + x)$ P1 = 000100110

$P_2 = (x^7 + x^4 + x^3 + x^2 + x)$ P2 = 10011110

Modulus= $(x^8 + x^4 + x^3 + x + 1)$ modulus = 10011110

Powers	*Shift-Left Operation*	*Exclusive-Or*
$x^0 \otimes P_2$		10011110
$x^1 \otimes P_2$	00111100	(00111100) $\oplus$ (00011010) = **<u>00100111</u>**
$x^2 \otimes P_2$	01001110	**<u>01001110</u>**
$x^3 \otimes P_2$	10011100	10011100
$x^4 \otimes P_2$	00111000	(00111000) $\oplus$ (00011010) = 00100011
$x^5 \otimes P_2$	01000110	**<u>01000110</u>**
$\mathbf{P_1 \otimes P_2}$ = **(00100111)** $\oplus$ **(01001110)** $\oplus$ **(01000110)** = **00101111**		

Addition and Multiplication table for GF(2^3)

The GF(2^3) field has 8 elements. We use the irreducible polynomial ($x^3 + x^2 + 1$) and show the addition and multiplication tables for this field. We show both 3-bit words and the polynomials. Note that there are two irreducible polynomials for degree 3. The other one, ($x^3 + x + 1$), yields a totally different table for multiplication.

Addition table for GF(2^3)

$\oplus$	000 **(0)**	001 **(1)**	010 **(x)**	011 **($x+1$)**	100 **(x^2)**	101 **x^2+1**	110 **(x^2+x)**	111 **(x^2+x+1)**
000 **(0)**	000 **(0)**	001 **(1)**	010 **(x)**	011 **($x+1$)**	100 **(x^2)**	101 **(x^2+1)**	110 **(x^2+x)**	111 **(x^2+x+1)**
001 **(1)**	001 **(1)**	000 **(0)**	011 **($x+1$)**	010 **(x^2)**	101 **(x^2+1)**	100 **(x^2+x)**	111 **(x^2+x+1)**	110 **(x^2+x)**
010 **(x)**	010 **(x)**	011 **($x+1$)**	000 **(0)**	001 **(1)**	110 **(x^2+x)**	111 **(x^2+x+1)**	100 **(x^2+x)**	101 **(x^2+1)**
011 **($x+1$)**	011 **($x+1$)**	010 **(x)**	001 **(1)**	000 **(0)**	111 **(x^2+x+1)**	110 **(x^2+x)**	101 **(x^2+1)**	100 **(x^2)**
100 **(x^2)**	100 **(x^2)**	101 **(x^2+1)**	110 **(x^2+x)**	111 **(x^2+x+1)**	000 **(0)**	001 **(1)**	010 **(x)**	011 **($x+1$)**
101 **(x^2+1)**	101 **(x^2+1)**	100 **(x^2)**	111 **(x^2+x+1)**	110 **(x^2+x)**	001 **(1)**	000 **(0)**	011 **($x+1$)**	010 **(x)**
110 **(x^2+x)**	110 **(x^2+x)**	111 **(x^2+x+1)**	100 **(x^2)**	101 **(x^2+1)**	010 **(x)**	011 **($x+1$)**	000 **(0)**	001 **(1)**
111 **(x^2+x+1)**	111 **(x^2+x+1)**	110 **(x^2+x)**	101 **(x^2+1)**	100 **(x^2)**	011 **($x+1$)**	010 **(x)**	001 **(1)**	000 **(0)**

Multiplication Table for GF(2^3)

⊗	000 (0)	001 (1)	010 (x)	011 ($x+1$)	100 (x^2)	101 (x^2+1)	110 (x^2+x)	111 (x^2+x+1)
000 (0)	000 (0)	000 (0)	000 (0)	000 (0)	000 (0)	000 (0)	000 (0)	000 (0)
001 (1)	000 (0)	001 (1)	010 (x)	011 ($x+1$)	100 (x^2)	101 (x^2+1)	110 (x^2+x)	111 (x^2+x+1)
010 (x)	000 (0)	010 (x)	100 (x)	110 (x^2+x)	101 (x^2+1)	111 (x^2+x+1)	001 (1)	011 ($x+1$)
011 ($x+1$)	000 (0)	011 ($x+1$)	110 (x^2+x)	101 (x^2+1)	001 (1)	010 (x)	111 (x^2+x+1)	100 (x)
100 (x^2)	000 (0)	100 (x^2)	101 (x^2+1)	001 (1)	111 (x^2+x+1)	011 ($x+1$)	010 (x)	110 (x^2+x)
101 (x^2+1)	000 (0)	101 (x^2+1)	111 (x^2+x+1)	010 (x)	011 ($x+1$)	110 (x^2+x)	100 (x^2)	001 (1)
110 (x^2+x)	000 (0)	110 (x^2+x)	001 (1)	111 (x^2+x+1)	010 (x)	100 (x^2)	011 ($x+1$)	101 (x^2+1)
111 (x^2+x+1)	000 (0)	111 (x^2+x+1)	011 ($x+1$)	100 (x^2)	110 (x^2+x)	001 (1)	101 (x^2+1)	010 (x)

4.2.2 Using a Generator

Sometimes it is easier to define the elements of the GF(2^n) field using a generator.

$$\{0, g, g, g^2, \ldots, g^N\}, \text{ where } N = 2^n - 2$$

The elements 0, g^0, g^1, $g2$, and g^3 can be easily generated, because they are the 4-bit representations of 0, 1, x^2, and x^3. Elements g^4 through g^{14}, which represent x^4 though x^{14} need to be divided by the irreducible polynomial. To avoid the polynomial division, the relation $f(g) = g^4 + g + 1 = 0$ can be used .

0	=	0	=	0	=	0	⟶	0	= (0000)
g^0	=	g^0	=	g^0	=	g^0	⟶	g^0	= (0001)
g^1	=	g^1	=	g^1	=	g^1	⟶	g^1	= (0010)
g^2	=	g^2	=	g^2	=	g^2	⟶	g^2	= (0100)
g^3	=	g^3	=	g^3	=	g^3	⟶	g^3	= (1000)
g^4	=	g^4	=	g^4	=	$g+1$	⟶	g^4	= (0011)
g^5	=	$g\,(g^4)$	=	$g\,(g+1)$	=	g^2+g	⟶	g^5	= (0110)
g^6	=	$g\,(g^5)$	=	$g\,(g^2+g)$	=	g^3+g^2	⟶	g^6	= (1100)
g^7	=	$g\,(g^6)$	=	$g\,(g^3+g)$	=	g^3+g+1	⟶	g^7	= (1011)
g^8	=	$g\,(g^7)$	=	$g\,(g^3+g+1)$	=	g^2+1	⟶	g^8	= (0101)
g^9	=	$g\,(g^8)$	=	$g\,(g^2+1)$	=	g^3+g	⟶	g^9	= (1010)
g^{10}	=	$g\,(g^9)$	=	$g\,(g^3+g)$	=	g^2+g+1	⟶	g^{10}	= (0111)
g^{11}	=	$g\,(g^{10})$	=	$g\,(g^2+g+1)$	=	g^3+g^2+g	⟶	g^{11}	= (1110)
g^{12}	=	$g\,(g^{11})$	=	$g\,(g^3+g^2+g)$	=	g^3+g^2+g+1	⟶	g^{12}	= (1111)
g^{13}	=	$g\,(g^{12})$	=	$g\,(g^3+g^2+g+1)$	=	g^3+g^2+1	⟶	g^{13}	= (1101)
g^{14}	=	$g\,(g^{13})$	=	$g\,(g^3+g^2+1)$	=	g^3+1	⟶	g^{14}	= (1001)

The following show the results of addition and subtraction operations:

a. $g^3 + g^{12} + g^7 = g^3 + (g^3 + g^2 + g + 1) + (g^3 + g + 1) = g^3 + g^2 \rightarrow (1100)$

b. $g^3 - g^6 = g^3 + g^6 = g^3 + (g^3 + g^2) = g^2 \rightarrow (0100)$

The following show the result of multiplication and division operations:.

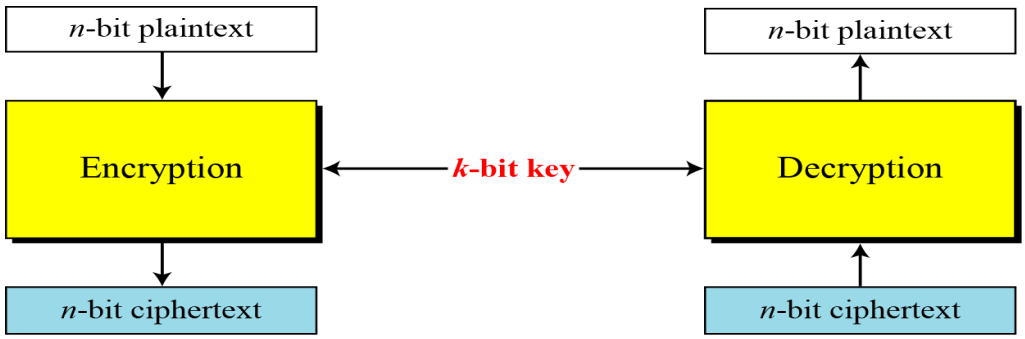
a. $g^9 \times g^{11} = g^{20} = g^{20 \bmod 15} = g^5 = g^2 + g \rightarrow (0110)$

b. $g^3 / g^8 = g^3 \times g^7 = g^{10} = g^2 + g + 1 \rightarrow (0111)$

Chapter 5
Introduction to Modern Symmetric-key Ciphers

5.1Modern Block Ciphers

A symmetric-key modern block cipher encrypts an n-bit block of plaintext or decrypts an n-bit block of ciphertext. The encryption or decryption algorithm uses a k-bit key. The decryption algorithm must be the inverse of encryption algorithm, and both operations use the same secret key



If the message has a fewer than n bits, padding must beaded to make it an n bit block. If the message has more than n bits, it should be divided into n bit blocks and the appropriate padding must be added to the last block if necessary. The common values of n are 64, 128, 256 or 512 bits.

Example:

How many padding bits must be added to a message of 100 characters if 8-bit ASCII is used for encoding and the block cipher accepts blocks of 64 bits?

Encoding 100 characters using 8-bit ASCII results in an 800-bit message. The plaintext must be divisible by 64. If | M | and |Pad| are the length of the message and the length of the padding,

$$|M| + |Pad| = 0 \bmod 64 \quad \rightarrow \quad |Pad| = -800 \bmod 64 \quad \rightarrow \quad 32 \bmod 64$$

Substitution or Transposition

A modern block cipher can be designed to act as a substitution cipher or a transposition cipher.

If the cipher is designed as a substitution cipher, a 1-bit or a 0-bit in the plain text can be replaced by either 0 or 1. The plain text and the cipher text will have a different number of 1's. a 64 bit plaintext block of 12 0's and 52 1s can be encrypted to a cipher text of 4 0's and 30 1's. if the cipher is designed as a transposition cipher, the bits are only reordered, there is a same number of 1's and 0's in the cipher text. To be resistant to exhaustive-search attack, a modern block cipher needs to be designed as a substitution cipher.

Example: Suppose that we have a block cipher where n = 64. If there are 10 1's in the ciphertext, how many trial-and-error tests does Eve need to do to recover the plaintext from the intercepted ciphertext in each of the following cases?

a. The cipher is designed as a substitution cipher.
b. The cipher is designed as a transposition cipher.

Solution

a. In the first case, Eve has no idea how many 1's are in the plaintext. Eve needs to try all possible 2^{64} 64-bit blocks to find one that makes sense.
b. In the second case, Eve knows that there are exactly 10 1's in the plaintext. Eve can launch an exhaustive-search attack using only those 64-bit blocks that have exactly 10 1's.

Block Ciphers as Permutation Groups

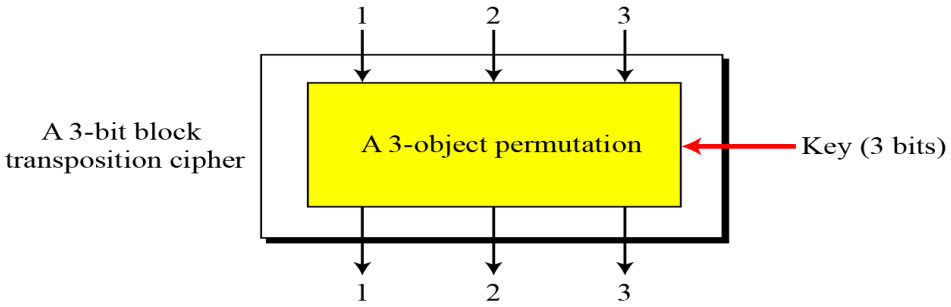
Full Size Key ciphers

Full size key Transposition Ciphers

A full size key transposition ciphers only transpose bits with out changing their values, so it can be modeled as a n object perution with a set of n! permutation tables in which the key defines which table is used by Alice and Bob.

The model and the set of permutation tables for a 3-bit block transposition cipher where the block size is 3 bits.

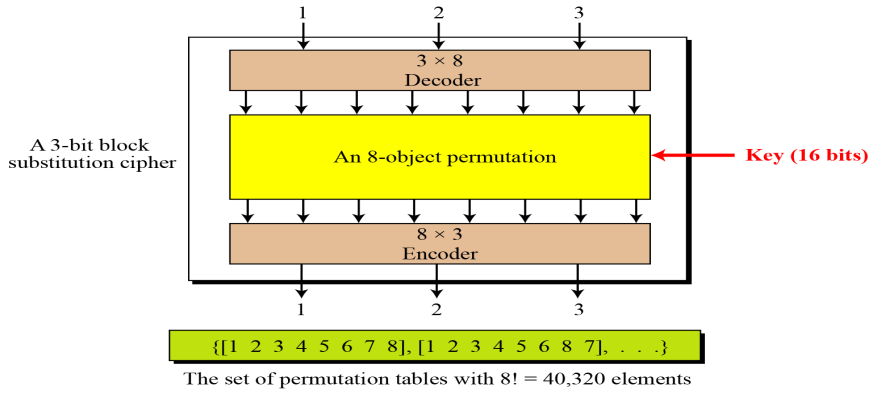
The set of permutation tables has 3! = 6 elements.



The set of permutation tables with 3! = 6 elements

Full size key substitution Block ciphers

A full size key substitution cipher does not transpose bits. We can model the substitution cipher as a permutation if we can decode the input and encode the output. Decoding means transforming an n-bit integer into 2n bit string with only a single 1 and 2^n-1 0's. the position of single 1 is the integer, in which the position ranges from 0 to 2^n-1. Encoding is the reverse process.



The set of permutation tables with 8! = 40,320 elements

Key less ciphers

Although a key less cipher is practically useless by itself, keyless ciphers are used as components of keyed ciphers.

Keyless transposition ciphers: A keyless transposition cipher is a prewired transposition cipher when implemented in hardware. The fixed key can be represented as a table when the unit is implemented in software

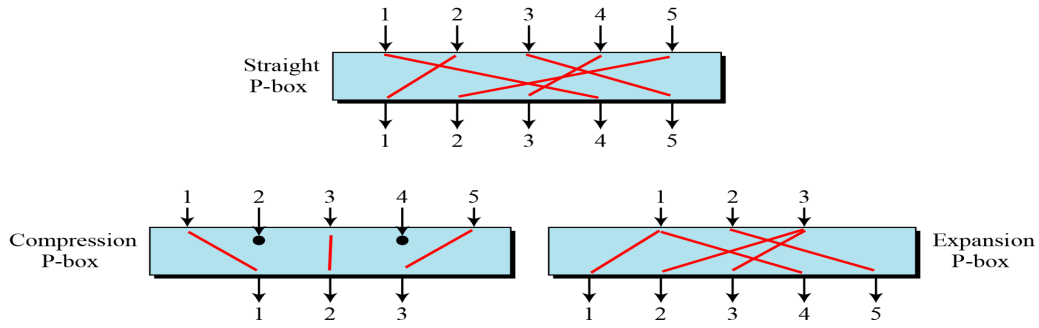
Keyless substitution cipher: A key less substitution cipher can be thought of as a predefined mapping from the input to the output. The mapping can be defined as a table, mathematical function and so on.

5.2 Components of a Modern Block Cipher

Modern block ciphers normally are keyed substitution ciphers in which the key allows only partial mappings from the possible inputs to the possible outputs. Modern block ciphers are not designed as a single units. To provide the required properties of a modern block cipher, such as diffusion and confusion a modern block cipher is made o f a combination of S-Box and P-Boxes.

1.P-box

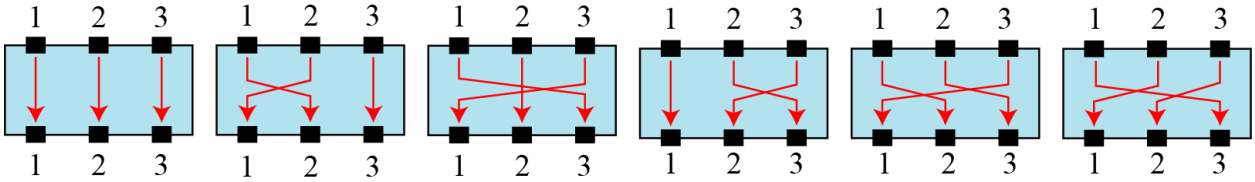
A P-box (permutation box) parallels the traditional transposition cipher for characters. It transposes bits. There are three types of P-boxes



Straight P-box

A straight P-box with n inputs and n outputs is a permutation. There are n! possible mappings.P boxes are normally keyless, which means that the mapping is predetermined. If the P box is implemented in hardware, it is prewired. If it is implemented in software a permutation table shows the mapping.

Ex: possible mappings of 3x3 P-box



Example of permutation table for straight box containing 8 bits

4 3 6 5 1 2 8 7

Compression P-box

It is a P-box with n inputs and m outputs where m<n. some of the inputs are blocked and do not reach the output. They are used when we need to permute the bits and the same time decrease the number of bits for the next stage.

Example of 32x24 permutation table.

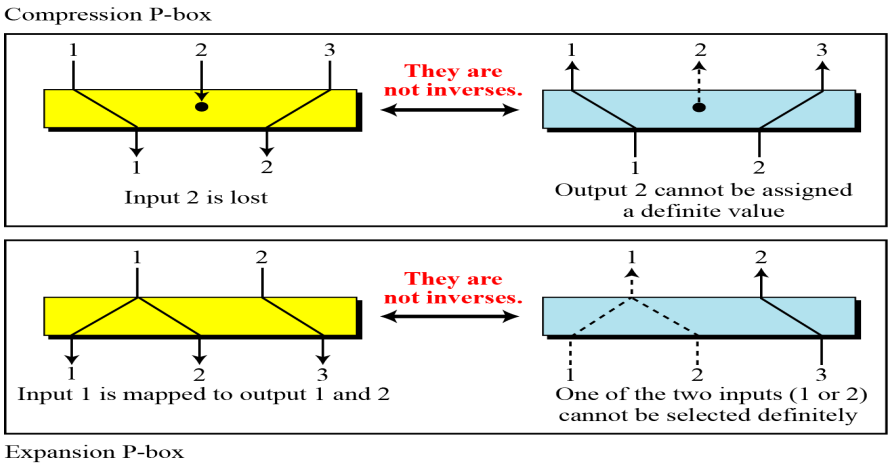
01	02	03	21	22	26	27	28	29	13	14	17
18	19	20	04	05	06	10	11	12	30	31	32

Expansion P-box

It is a P-box with n inputs and m outputs where m>n. some of the inputs are connected to more than one outputs.some of the entries in the P-boxes are repeated.

01	02	03	21	22	26	27	28	29	13	14	17
18	19	20	04	05	06	10	11	12	30	31	32

Invertability of P-boxes: Straight P-box is invertible where as expansion and compression P-boxes are not invertible. In compression P-Box an input can be dropped during encryption, the decryption does not replave the dropped bit. In an expansion p-box, an input may be mapped to more than one output during encryption, the decryption algorithm does not have the clue which of the several inputs are mapped to an output



2.S-Box

It can be thought of as a substitution cipher. An S-box can have a different number of inputs and outputs. The input to an S-box could be an n- bit word, but the output can be an m-bit word, where m and n not necessarily the same.

Linear and Non Linear S-Box:

In an S-box with n inputs and m outputs, we call the inputs x0,x1,….xn and the outputs y1,y1,….yn . the relationship between inputs and the outputs can be expressed as a set of equation

Y1=f1(x1,x2,…xn)

Y2=f2(x1,x2,…xn)

Ym=fm(x1,x2,..xn)

Example: In an S-box with three inputs and two outputs, we have

$$y_1 = x_1 \oplus x_2 \oplus x_3 \qquad y_2 = x_1$$

The S-box is linear because $a_{1,1} = a_{1,2} = a_{1,3} = a_{2,1} = 1$ and $a_{2,2} = a_{2,3} = 0$. The relationship can be represented by matrices, as shown below:

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

In Non Linear S-box the relationship is non linear

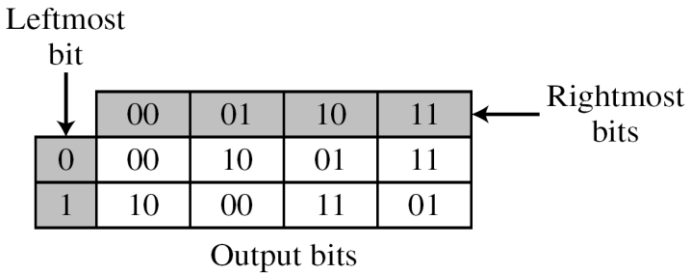
Example

In an S-box with three inputs and two outputs, we have

$$y_1 = (x_1)^3 + x_2 \qquad y_2 = (x_1)^2 + x_1x_2 + x_3$$

where multiplication and addition is in GF(2). The S-box is nonlinear because there is no linear relationship between the inputs and the outputs.

Example: The following table defines the input/output relationship for an S-box of size 3 × 2. The leftmost bit of the input defines the row; the two rightmost bits of the input define the column. The two output bits are values on the cross section of the selected row and column.

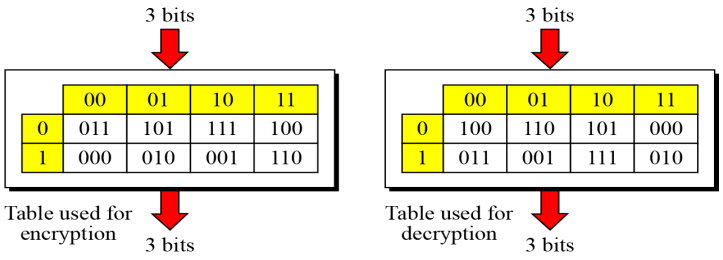


	00	01	10	11
0	00	10	01	11
1	10	00	11	01

Based on the table, an input of 010 yields the output 01. An input of 101 yields the output of 00.

Properties of S-Box

Invertability: An S-box may or may not be invertible. In an invertible S-box, the number of input bits should be the same as the number of output bits. For example, if the input to the left box is 001, the output is 101. The input 101 in the right table creates the output 001, which shows that the two tables are inverses of each other.

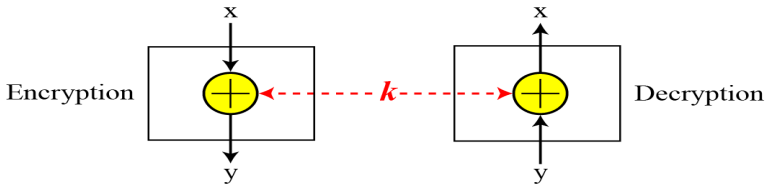


	00	01	10	11
0	011	101	111	100
1	000	010	001	110

	00	01	10	11
0	100	110	101	000
1	011	001	111	010

3.Exclusive-OR:

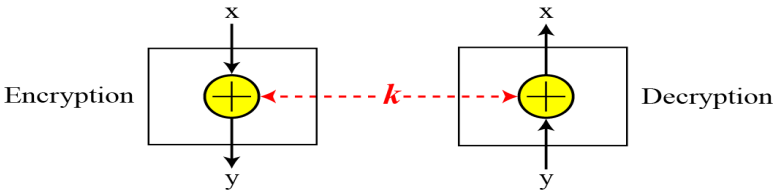
An important component in most block ciphers is the exclusive-or operation.



An important component in most block ciphers is the exclusive-or operation. addition and subtraction operations in the GF(2^n) field are performed by a single operation called the exclusive-or (XOR).

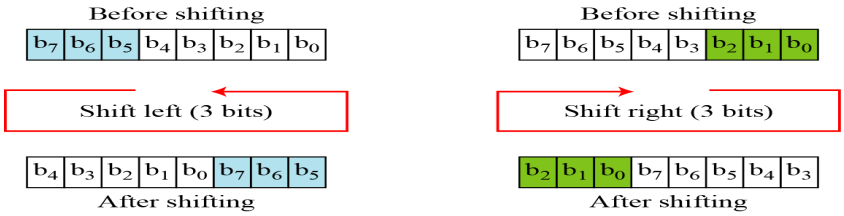
The five properties of the exclusive-or operation in the GF(2n) field makes this operation a very interesting component for use in a block cipher: closure, associativity, commutativity, existence of identity, and existence of inverse.

The inverse of a component in a cipher makes sense if the component represents a unary operation (one input and one output). For example, a keyless P-box or a keyless S-box can be made invertible because they have one input and one output. An exclusive operation is a binary operation. The inverse of an exclusive-or operation can make sense only if one of the inputs is fixed (is the same in encryption and decryption). For example, if one of the inputs is the key, which normally is the same in encryption and decryption, then an exclusive-or operation is self-invertible



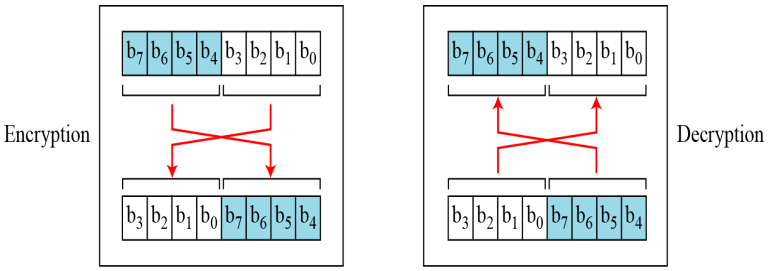
4.Circular Shift:

Another component found in some modern block ciphers is the circular shift operation. Shifting can be to the left or to the tight. The circular left shift operation shifts each bit in an n bit word k positions to the left, the leftmost k bits are removed from the left and become the rightmost bits. The circular shift operation mixes the bits in a word and helps hide the patterns in the original word.



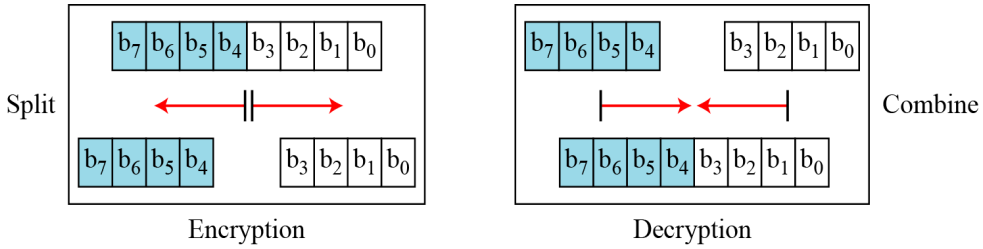
5.Swap

The swap operation is a special case of the circular shift operation where k = n/2. This operation is valid only if the n value is even. A swap operation in the encryption cipher can be totally canceled by a swap operation in the decryption operation



6.Split and Combine

Two other operations found in some block ciphers are split and combine. Split operation splits an n bit word in the middle, crating two equal length words. The combine operation normally concatenates two equal length words to create nbit word. These two operations are inverses of each other. If one is used in encryption, the other is used in decryption.



5.3Product Ciphers

Shannon introduced the concept of a product cipher. A product cipher is a complex cipher combining substitution, permutation, and other components. Shannon's idea in introducing the product cipher was to have two properties diffusion and confustion

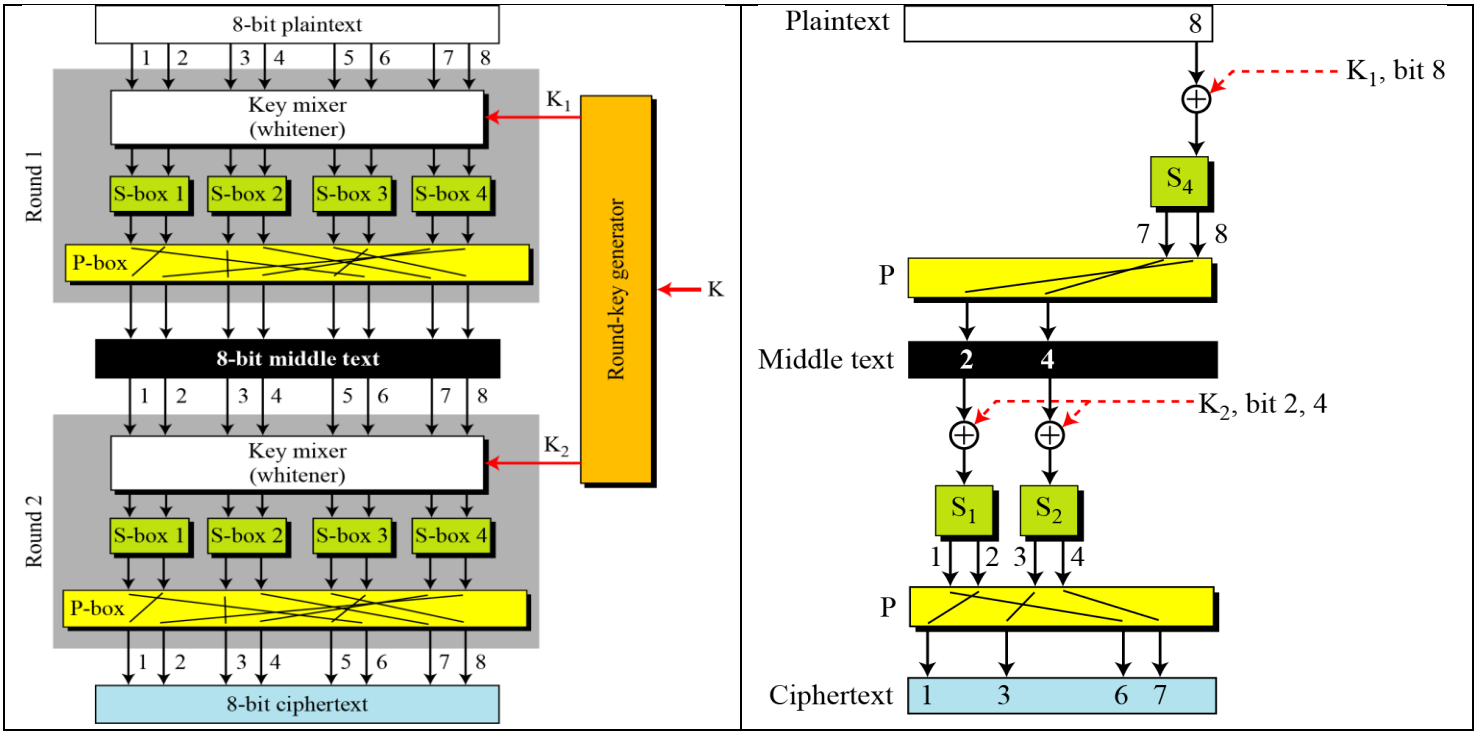
Diffusion: hide the relationship between the ciphertext and the plaintext.

Confusion:hide the relationship between the ciphertext and the key.

Rounds

Diffusion and confusion can be achieved using iterated product ciphers where each iteration is a combination of S-boxes, P-boxes, and other components. Each iteration is referred as a round. The block cipher uses s key schedule or key generator that creates different keys for each round from the cipher key. In an N-round cipher, the plaintext is encypted N times to create the ciphertext, the ciphertext is decrypted N times to create the plaintext.

A product cipher made up of two rounds



Diffusion and confusion made in each round

In the first round, bit , after being XORed with correspondingbit of K1, affects two bits 7 and 8 through S-box. Bit 7 is permuted and becomes bit 2, bit 8 is permuted and becomes bit 4. After the first round, bit 8 affects bits 2 and 4. Similarly after second round bit 8 affects 1,3,6 and 7 bits.

Types of Product Ciphers

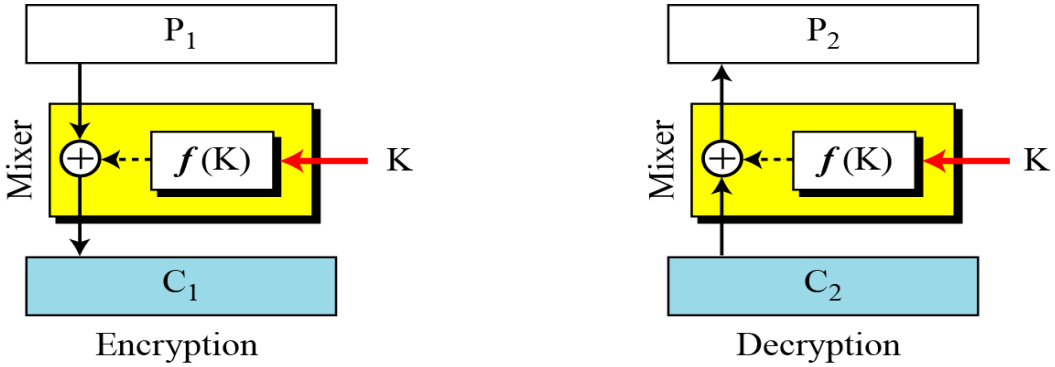
Modern block ciphers are all product ciphers, but they are divided into two classes.

1. Feistel ciphers
2. Non-Feistel ciphers

Feistel Ciphers

Feistel designed a very intelligent and interesting cipher that has been used for decades. A Feistel cipher can have three types of components: self-invertible, invertible, and noninvertible. A fiestal cipher combines all invertible elements in a unit and uses the same unit in the encryption and decryption. Fiestal showed that the encryption and decryption are inverse of each other.

First design of Fiestal Cipher:



Example:

This is a trivial example. The plaintext and ciphertext are each 4 bits long and the key is 3 bits long. Assume that the function takes the first and third bits of the key, interprets these two bits as a decimal number, squares the number, and interprets the result as a 4-bit binary pattern. Show the results of encryption and decryption if the original plaintext is 0111 and the key is 101.

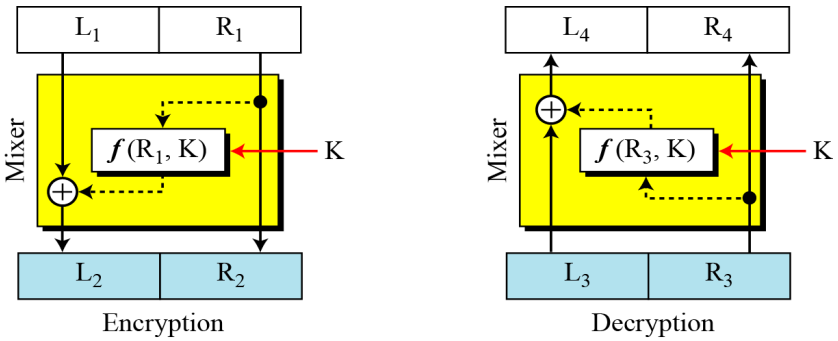
Solution:

The function extracts the first and second bits to get 11 in binary or 3 in decimal. The result of squaring is 9, which is 1001 in binary.

$$\textbf{Encryption: } C = P \oplus f(K) = 0111 \oplus 1001 = 1110$$

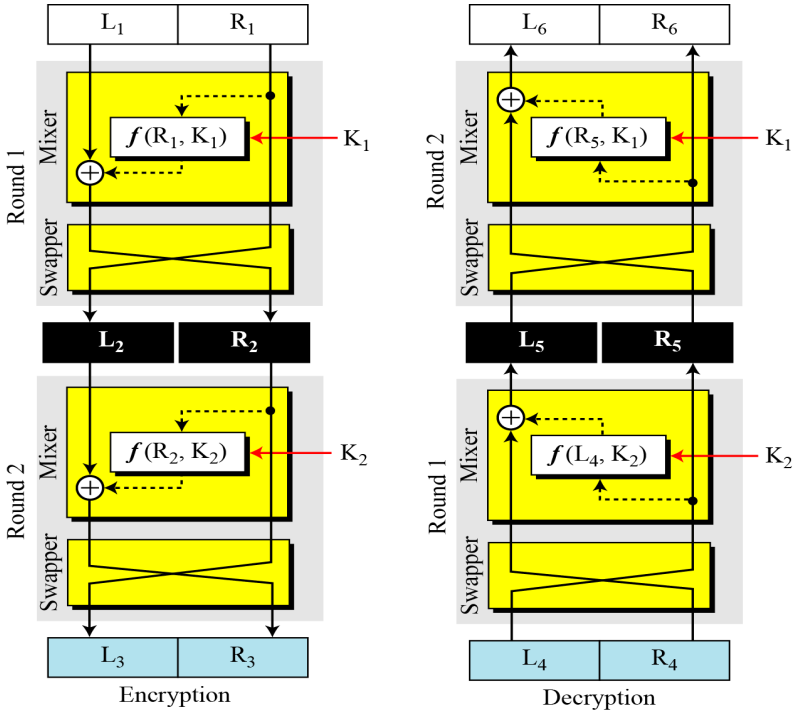
$$\textbf{Decryption: } P = C \oplus f(K) = 1110 \oplus 1001 = 0111$$

Improvement of first design



Final Design of Fiestal Cipher: the preceding improvement has one flaw. The right half of the plain text never changes.eve can immediately find the right half of the plaintext by intercepting the ciphertext and extracting

half of it. The design has been improved by increasing the number of rounds and adding a new element to each round called swapper. It allows us to swap the left and right halves in each round.



There are two different keys k1,k2 for two rounds, the keys are used in reverse order in encryption and decryption. Because two mixers are inverses of each other, and the sappers are inverses of each other, it should be clear that the encryption and decryption ciphers are inverses of each other

$$L_5 = R_4 \oplus f(L_4, K_2) = R_3 \oplus f(R_2, K_2) = L_2 \oplus f(R_2, K_2) \oplus f(R_2, K_2) = L_2$$
$$R_5 = L_4 = L_3 = R_2$$

Then it is easy to prove that the equality holds for two plaintext blocks.

$$L_6 = R_5 \oplus f(L_5, K_1) = R_2 \oplus f(L_2, K_1) = L_1 \oplus f(R_1, K_1) \oplus f(R_1, K_1) = L_1$$
$$R_6 = L_5 = L_2 = R_1$$

Non-Feistel Ciphers

A non-Feistel cipher uses only invertible components. A component in the encryption cipher has the corresponding component in the decryption cipher. S boxes need to have an equal number of inputs and outputs to be compatable. No compression or expansion P boxes are allowed, because they are not invertible. In a non fiestal cipher there is no need to divide the plaintext into two halves . the only components in each round are the XOR operations.

5.4 *Attacks on Block Ciphers*

There are two types of attacks on modern block ciphers

1.Differential cryptanalysis:

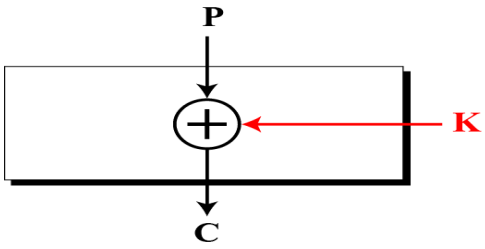
This attack is the known plaintext attack. Eve can some how access alice's computer, submitting chosen plain text and obtaining the corresponding ciphertext. The goal is to find Alice's cipher key.

Before eve uses the chosen plaintext attack, she needs to analyze the encryption algorithm in order to collect some information about plaintext- ciphertext relationships. Eve does not know the cipher key. However some ciphers have weakness in their structures that can allow eve to find a relationship between the plaintext

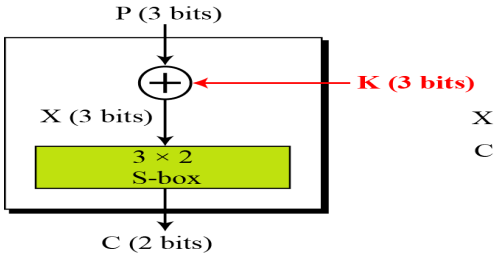
differences without knowing the key.

Example1: assume that the cipher is made only of one XOR operation. With out knowing the key, eve can easily find the relation between plaintext differences and ciphertext differences.

$$C_1 = P_1 \oplus K \qquad C_2 = P_2 \oplus K \qquad \rightarrow \qquad C_1 \oplus C_2 = P_1 \oplus K \oplus P_2 \oplus K = P_1 \oplus P_2$$



We can add an SBox



X	000	001	010	011	100	101	110	111
C	11	00	10	10	01	00	11	00

S-box table

Differential input/output for the cipher

$C_1 \oplus C_2$

$P_1 \oplus P_2$	00	01	10	11
000	8			
001	2	2		4
010	2	2	4	
011		4	2	2
100	2	2	4	
101		4	2	2
110	4		2	2
111			2	6

$C_1 \oplus C_2$

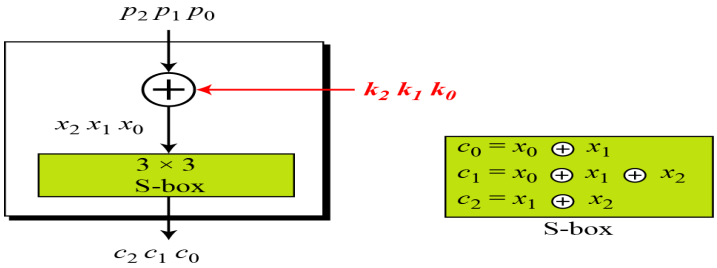
$P_1 \oplus P_2$	00	01	10	11
000	1	0	0	0
001	0.25	0.25	0	0.50
010	0.25	0.25	0.50	0
011	0	0.50	0.25	0.25
100	0.25	0.25	0.50	0
101	0	0.50	0.25	0.25
110	0.50	0	0.25	0.25
111	0	0	0.25	0.75

General procedure for differential cryptanalysis

1. Because each round is the same, even can create a differential distribution table for each S-box and combine them to create the distribution for each round.
2. Assuming that each round is independent. Eve can create distribution table for the whole cipher by multiplying the corresponding probabilities.
3. Eve now can make a list of plaintexts for attacks based on the distribution table in step2
4. Eve chooses a ciphertext and finds the corresponding plaintext
5. Eve repeats step4 to find more bits in the key
6. After finding the enough bits in the key, eve can use brute-force attack to fing the whole key.

Linear Cryptanalysis

The analysis is known as known plaintext attacks. Assume that the cipher ismade of a single round. Where c0,c1 and c2 represents the input. With linear component, we can create three linear equations between plaintext and ciphertext bits



$$c_0 = p_0 \oplus k_0 \oplus p_1 \oplus k_1$$
$$c_1 = p_0 \oplus k_0 \oplus p_1 \oplus k_1 \oplus p_2 \oplus k_2$$
$$c_2 = p_1 \oplus k_1 \oplus p_2 \oplus k_2$$

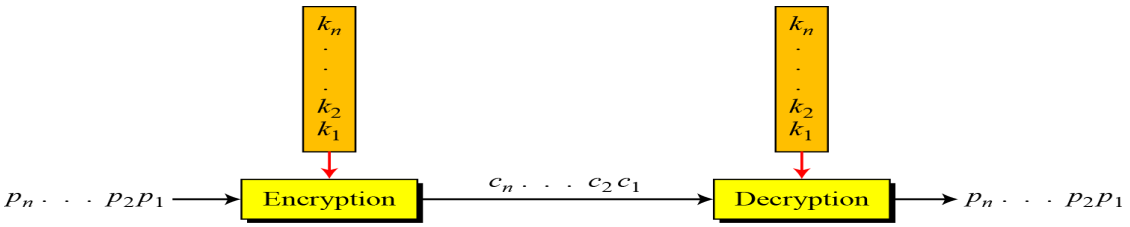
Solving for three unknowns, we get

$$k_1 = (p_1) \oplus (c_0 \oplus c_1 \oplus c_2)$$
$$k_2 = (p_2) \oplus (c_0 \oplus c_1)$$
$$k_0 = (p_0) \oplus (c_1 \oplus c_2)$$

This means that three known-plaintext attacks can find the values of k_0, k_1, and k_2 .

5.5Modern Stream Ciphers

In a modern stream cipher, encryption and decryption are done r bits at a time. We have a plaintext bit stream P = p_n…p_2 p_1, a ciphertext bit stream C = c_n…c_2 c_1, and a key bit stream K = k_n…k_2 k_1, in which p_i , c_i , and k_i are r-bit words.



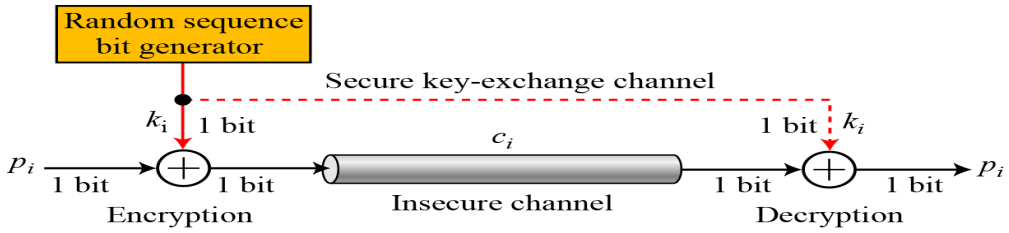
Stream ciphers are faster than block ciphers. The hardware implementation of a stream cipher is also easier. When we need to encrypt binary stream and transmit them at a constant rate, a strea cipher is the better choice to use. Stream ciphers are also more immune to the corruption of bits during transmission. Ther ar two types of stream ciphers.

1. Synchronous Stream ciphers
2. NonSynchronous Stream ciphers

Synchronous stream ciphers

In a synchronous stream cipher the key is independent of the plaintext or ciphertext.

One time pad: the simplest synchronous stream cipher is one time pad

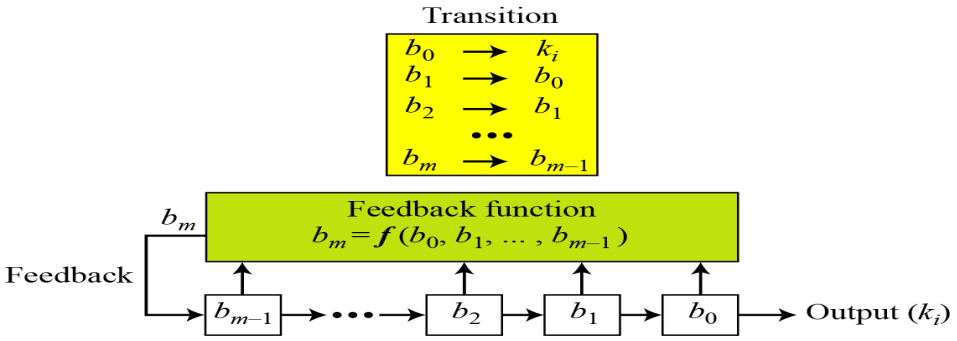


The one-time pad is an ideal cipher. It is perfect. There is no way that an adversary can guess the key or the plaintext and ciphertext statistics. There is no relationship between the plaintext and the ciphertext, either.

Eve cannot break the cipher unless she tries all possible random key steams.

Feedback Shift Register

One compromise to the one-time pad is the feedback shift register(FSR).FSR can be implemented in either software or hardware, but the hardware implementation is easier.



The shift register is a sequence of bits b0 to bm-1, where each cell holds a single bit. The cells are initialized to an m-bit word, called the initial value or the seed. Whenever output bit is needed, every bit is shifted one cell to the right, which means that each cell gives its value to the cell to its right and receives the value of the cell to its left.

Linear Feedback shift register(LFSR):

It is a linear function of b0,b1,b2…..bm-1

$b_m=c_{m-1}b_{m-1}+\ldots\ldots\ldots\ldots c_2b_2+c_1b_1+c_0b_0$

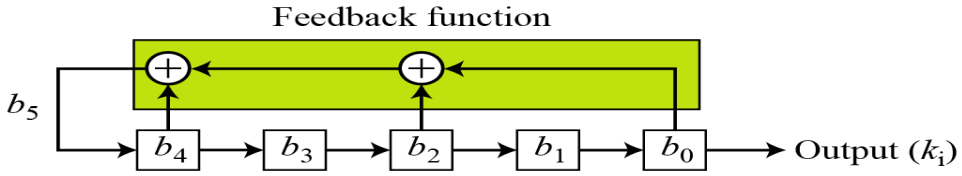
we are dealing with binary digits because the multiplication and addition are in GF(2) field, so the values of ci is either 0 or 1, but c0 value should be 1 to get feedback from the output

Example:

Create a linear feedback shift register with 5 cells in which

$$b_5 = b_4 \oplus b_2 \oplus b_0 .$$

Solution:If c_i = 0, b_i has no role in calculation of b_m. This means that b_i is not connected to the feedback function. If c_i = 1, b_i is involved in calculation of bm. In this example, c1 and c3 are 0's, which means that we have only three connections.



Non Linear Feedback Shift Register(NLFSR)

The linear feedback shift register is vulnerable to attacks mainly because of its linearity. A better stream cipher can be achieved using a nonlinear feedback shift register. An NLFSR has the same structure as an LSFR except that the bm is the nonlinear function of b0,b1,….bm. NLFSR uses AND means bit wise and operation, OR means bit wise or operation, the bar means the complement. NLFSR's re not common because there is no mathematical foundation for how to make an NLFSR with the maximum period.

b4=(b3 AND b2) OR (b1 AND b0)

Non Synchronous Stream Ciphers

In a nonsynchronous stream cipher, each key in the key stream depends on previous plaintext or ciphertext.

Chapter 6
Data Encryption Standard

6.1Introduction

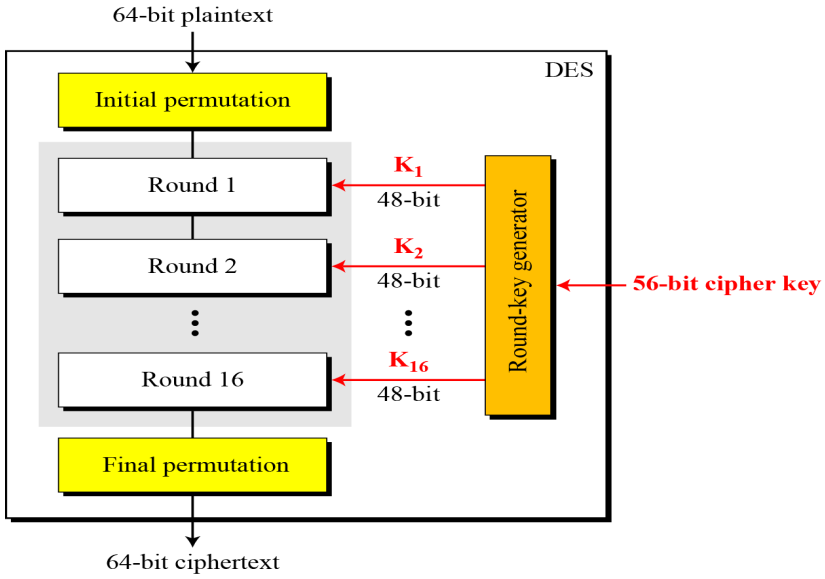
The Data Encryption Standard (DES) is a symmetric-key block cipher published by the National Institute of Standards and Technology (NIST). In 1973, NIST published a request for proposals for a national symmetric-key cryptosystem. A proposal from IBM, a modification of a project called Lucifer, was accepted as DES. DES was published in the Federal Register in March 1975 as a draft of the Federal Information Processing Standard (FIPS).

At the encryption site, DES takes a 64 bit plaintext and creates a 64 bit ciphertext. at decryption it takes 64 bit ciphertext and creates 64 bit plaintext. The same 56 bit key is used for both encryption and decryption.

6.2DES Structure

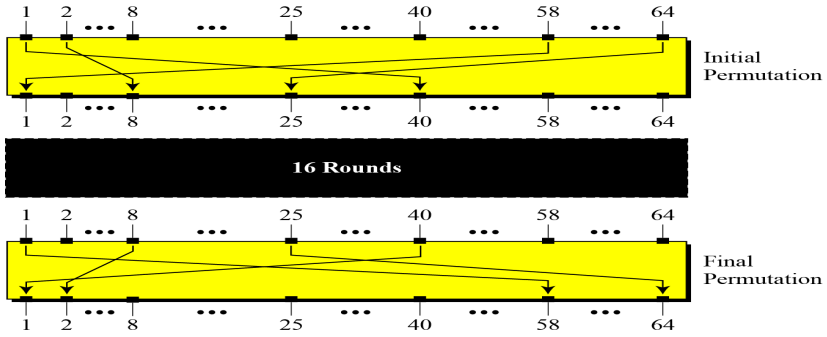
The encryption process is made of two permutations (P-boxes), which we call initial and final permutations, and sixteen Feistel rounds. Each round uses a different 48 bit round key generated from the cipher key according to the pre defined algorithm.

General Structure of DES



6.2.1 Initial and Final Permutations

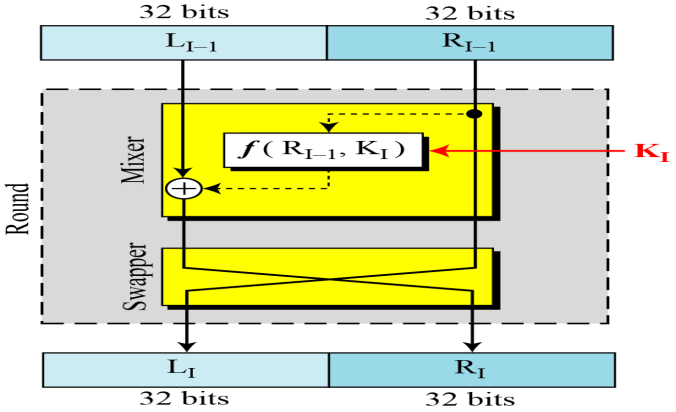
Each of these permutations takes 64 bit input and permutates according to a predefined rule. These permutations are keyless straight permutations that are the inverse of each other. For example in the initial permutation, the 58th bit in the input becomes the first bit in the output. Similarly in the final permutation the first bit in the input becomes the 58th bit in the output.



These two permutations have no cryptography significance in DES. Both permutations are keyless and predetermined. The reason they are included in DES is not clear and has not been revealed by the DES designers.

Rounds

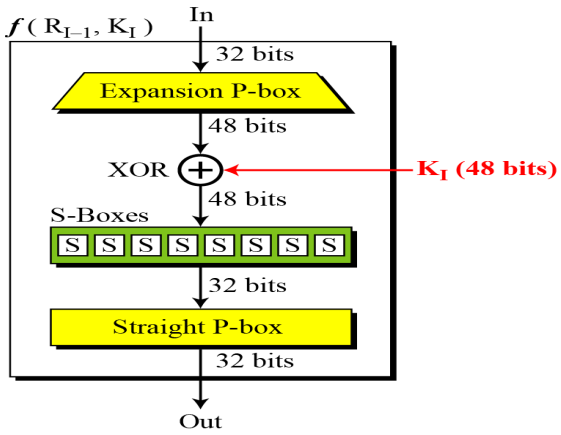
DES uses 16 rounds, each round of DES is a feistal cipher.



The round takes Li-1 and R i-1 from previous round and creates Li and Ri, which go to the next round. Each round has two cipher elements mixer and swapper. Each of these elements are invertible.

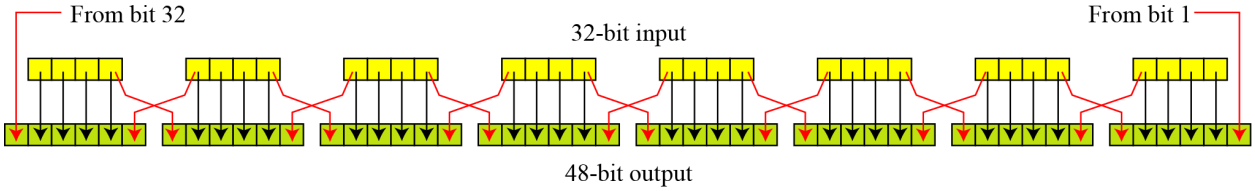
6.2.2DES Function

The heart of the DES is the DES function. The DES function applies a 48 bit key to the right most 32 bits(Ri-1) to produce a 32 bit output(Ri). This function is made up of four sections: an expansion P-box, a whitener that adds key, a group of S-boxes and a straight P-box



Expansion P-Box: since Ri-1 is a 32 bit input and ki is a 48 bit key, we first need to expand Ri-1 to 48 bits. Ri-1 is divided into 8 4-bit sections. Each 4 bit section is then expanded to 6 bits. For each section input bits 1,2,3 and

4 are copied to output bits 2,3,4 and 5. Output bit 1 comes from bit 4 of the previous section, output bit 6 comes from bit 1 of the next section. sections 1 and 8 can be considered as adjacent sections. The same rules applies to 1 and 32



DES used the table

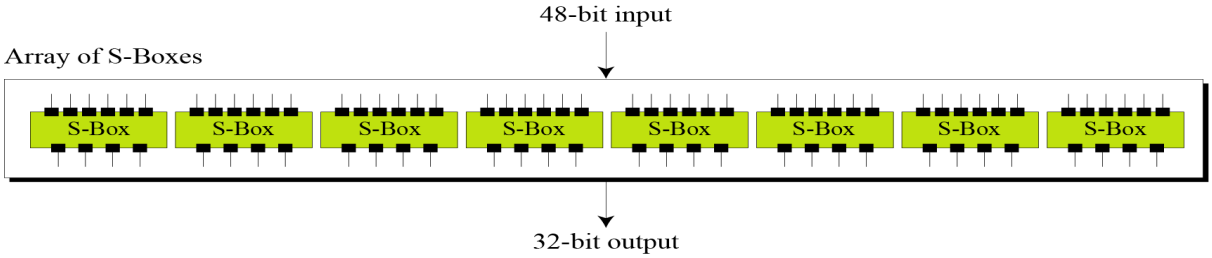
32	01	02	03	04	05
04	05	06	07	08	09
08	09	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	31	31	32	01

Whitener(XOR)

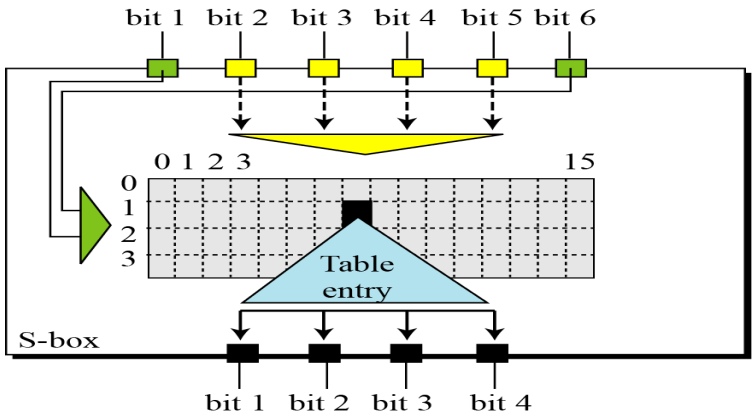
After the expansion permutation, DES uses the XOR operation on the expanded right section and the round key. Note that both the right section and the key are 48-bits in length. Also note that the round key is used only in this operation.

S-Boxes

The S-boxes do the real mixing (confusion). DES uses 8 S-boxes, each with a 6-bit input and a 4-bit output.



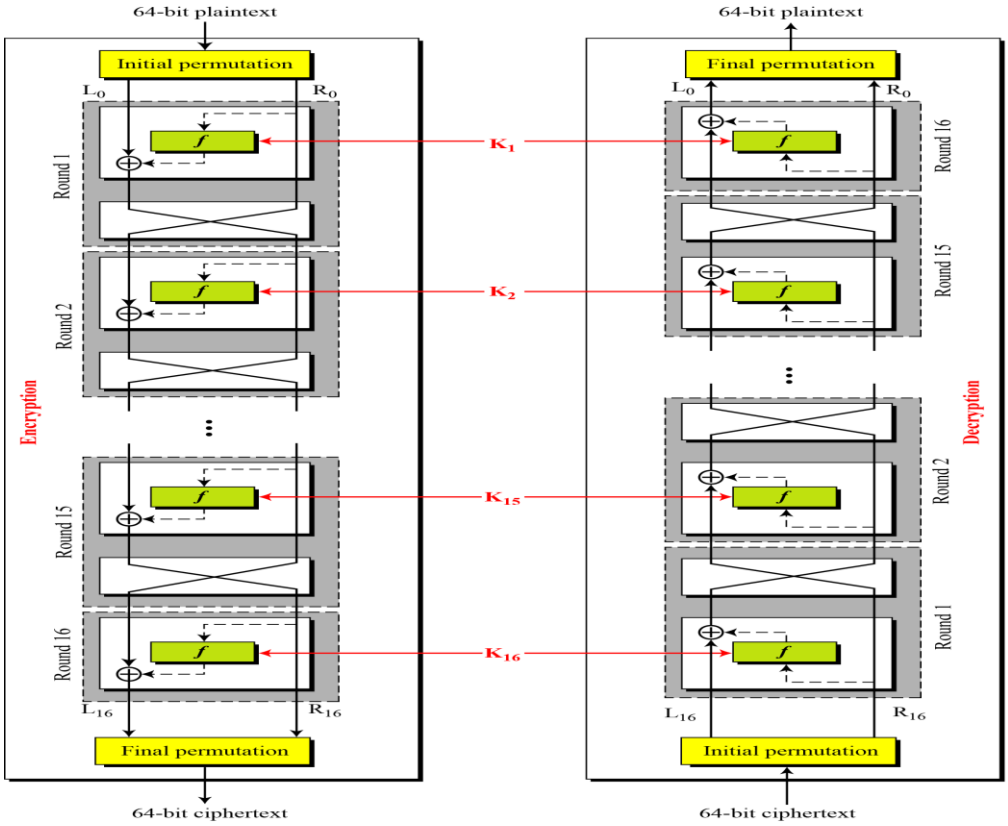
The 48 bitdata from the second operation is divided into eight 6 bit chunks, and each chunk is fed into a box. The result of each box is a 4 bit chunk. When these are combined the result is a 32 bit text. The substitution in each box follows a predetermined rule based on a 4 row by 16 column table. The combination of bits 1 and 6 of the input defines one of 4 rows. The combination of bits 2,3,4,5 defines one of the 16 columns.



6.2.3 Cipher and Reverse Cipher

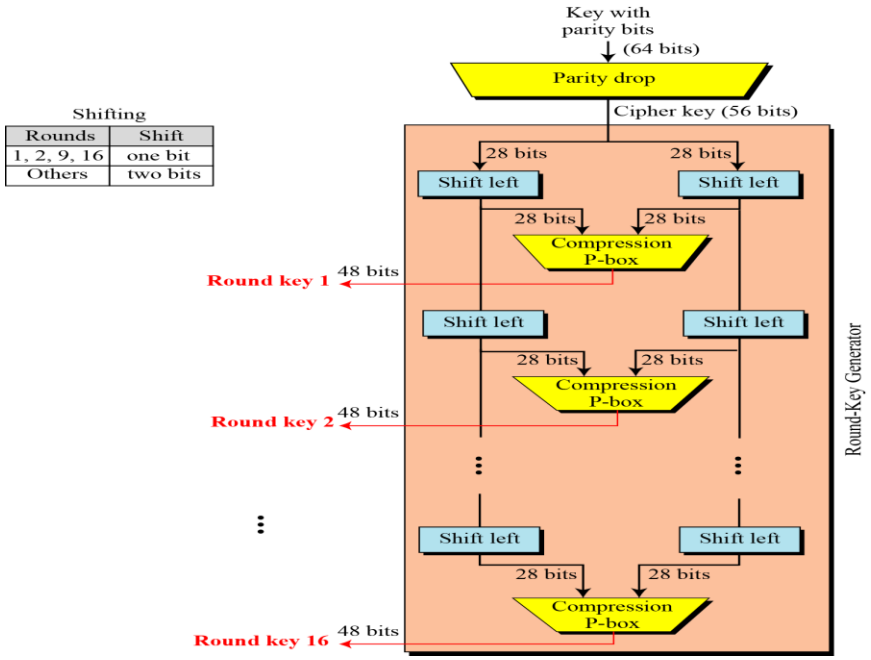
Using mixers and swappers, we can create the cipher and reverse cipher, each having 16 rounds.

To achieve this goal, one approach is to make the last round (round 16) different from the others; it has only a mixer and no swapper.



6.2.4Key Generation

The round-key generator creates sixteen 48-bit keys out of a 56-bit cipher key.



Shifting

Rounds	Shift
1, 2, 9, 16	one bit
Others	two bits

Parity Drop: The preprocess before key expansion is a compression permutation called parity bit drop. It drops the parity bits at 8,16,32…64 from the 64 bit key and permutes the rest of the bits. The remaining 56 bitvalue is the actual cipher key which is used to generate round keys.

Shift left: after the straight permutation, the key is divided into two 28 bit parts. Each part is circular shift left one or two bits. In rounds 1,2,9,16 shifting is done be one bit and in all other rounds, shifting is done for two bits. The two parts combined to form a 56 bit part.

Compression permutation: the compression permutation(P-box) changes the 58 bits to 48 bits, which are used as a key for a round.

6.3 DES Analysis

Two desired properties of a block cipher are

Avalanche Effect: It specifies that a small change in the plaintext(or key) should create a significant change in the cipher text. DES has proved to be strong with this property.

Completeness Effect: It means that each bit of the ciphertext needs to depend on many bits on the plaintext. The diffusion and confusion produced by P-boxes and S-boxes, show a very strong completeness effect.

Design Criteria

S-Box

1. The entries of each row are permutations of 0 to 15
2. S-Boxes are non linear. The output is not an affine transformation of input
3. If we change a single bit in the input, two or more bits will be changed in the output
4. If two inputs of s-box differ only in two middle bits, the output must differ in atleast two bits.
5. If two inputs to an S-box differ in the first two bits(1 and 2) and are the same in the last tow bits(5 and 6) , the two outputs must be different
6. There are only 32 6 bit input word pairs in which x1 XOR x2 not equal to 000000.. there 32 inputs pairs create 32 4 bit output word pairs
7. A criteria similar to 6 is applied to three S-boxes
8. In any S-box, if a single input bit is held constant(0 Or 1) and the other bits are changed randlomly, the difference between number of 0's and 1's is minimized.

P-box

1. Each S-box input comes from the output of different S-box
2. No input to a given S-box comes from the output from the same box
3. The four outputs from each S-box goes to six different S-boxes in the next round
4. No two output bits from an s-box go to the same s-box in the next round
5. If we number the eight S-boxes as s1,s2,….s8
    a. An output of Sj-2 goes to one of the first two bits of Sj in the next round
    b. An output bit from Sj-1 goes to one of the last two bits of Sj in the next round
    c. An output of Sj+1 goes to one of the two middle bits of the Sj in the next round

Number of Rounds

DES uses 16 rounds of Fiestal ciphers. It has been proved that after eight rounds, each ciphertext is a function of every plaintext and ciphertext. However experiment have found that DES with less than 16 rounds are even more vulnerable for known plaintext and brute force attacks.

DES Weaknesses

Weaknesses in Cipher Design

S-boxes:

1. In S-box 4 the last three output bits can be derived in the same way as the first output bit by complementing some of the input bits
2. Two specifically chosen inputs to an S-Box array can create the same output.
3. It is possible to obtain the same output in single round by changing bits in only three neighboring S-boxes

P-boxes:

1. It is not clear why the designers of DES used the initial and final permutations
2. In the expansion permutation inside the function the first and fourth bits of every four bit series are repeated.

Weakness in the cipher key:

Several weaknesses have been found in the cipher key.

Key Size

1. Critics believes that the most serious weakness of DES is the key size of 56 bits
2. To do a brute force attack on a given cipher text block the attacker needs on 2^{56} keys.
3. With the available technology it is possible to check one million keys per second, the designers told that it takes two thousand years to break DES, but using parallel processing technique it took only 112 hours to break this algorithm in 1998.

Weak Keys: four keys are called weak keys because after parity drop the key consists of all 0's,all 1's, half 0's and half 1's. if we encrypt original block with these key and

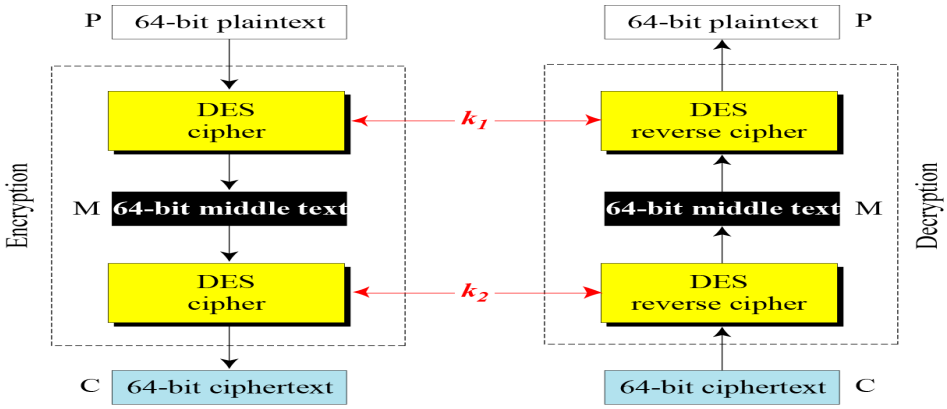
6.4 Multiple DES

6.4.1 Double DES

The first approach is to use double DES (2DES).

Meet-in-the-Middle Attack

However, using a known-plaintext attack called meet-in-the-middle attack proves that double DES improves this vulnerability slightly (to 2^{57} tests), but not tremendously (to 2^{112}). Alice uses two keys k1, k2 for encrypt the text, Bob uses ciphertext C and k2,k1 to recover P

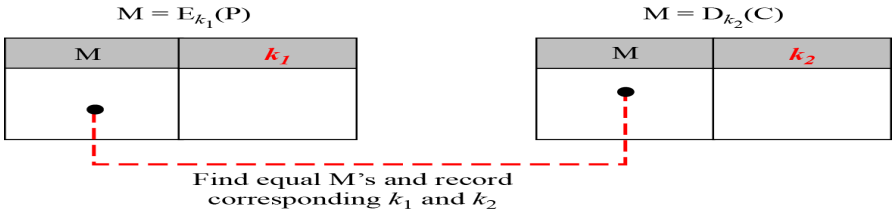


The text created by first encryption or first decryption, M should be same for encryption and decryption.

Assume that Eve has intercepted a previous pair P and C(Known plain text attack). Eve encrypts all possible

values 256 of K1 and records all values obtained for M. Eve decrypts C using all possible values 256 of K2 and records all the values of M. Eve creates two tables sorted by M values. She then compares thevalues of M until she finds those pair of k1 and K2 for which the value M is the same in both tables.

Tables for meet-in-the-middle attack

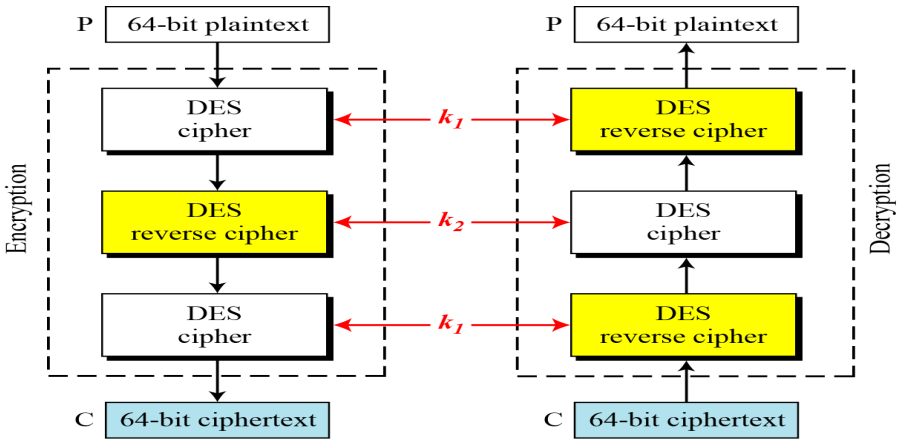


6.4.2 Triple DES

To improve the security of DES, triple DES(3DES) was proposed. This uses three stages of DES for encryption and decryption. Two versions of triple DES are in use: triple DES with two keys and triple DES with three keys.

Triple DES with two keys

In triple DES with two keys, there are only two keys k1 and k2. The first and the third stages use k1, the second stage uses k2. To make triple DES, the middle stage uses decryption in the encryption site and encryption in decryption site. It is much stronger than double DES



Triple DES with three keys.

In Triple DES with three keys at the encryption site uses EDE and decryption site uses DED. The three stages uses three different keys . Triple DES with three keys is used by many applications like PGP.

Chapter 7
Advanced Encryption Standard
(AES)

7-1 INTRODUCTION

7.1.1 History.

In February 2001, NIST announced that a draft of the Federal Information Processing Standard (FIPS) was available for public review and comment. Finally, AES was published as FIPS 197 in the Federal Register in December 2001.

7.1.2 Criteria

The criteria defined by NIST for selecting AES fall into three areas:

1. Security : the main emphasis was on security. Because NIST explicitly demanded a 128-bit key, this criterion focused on resistance to cryptanalysis attacks other than brute force attack.

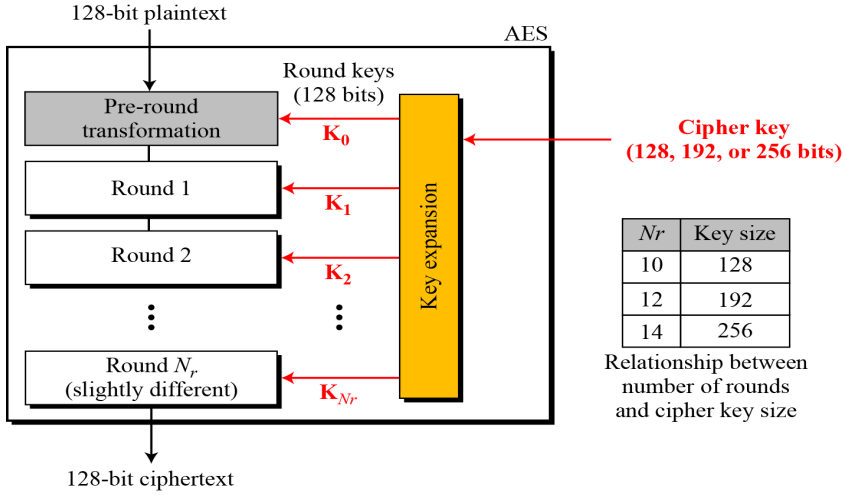
2. Cost: the second criteria was cost, which covers the computational efficiency and storage requirement for different implementations such as hardware, software or smart cards.

3. Implementation: The criteria included the requirement that the algorithm must have flexibility of implementing on any platform and simplicity.

7.1.3 Rounds

AES is a non-Feistel cipher that encrypts and decrypts a data block of 128 bits. It uses 10, 12, or 14 rounds. The key size, which can be 128, 192, or 256 bits, depends on the number of rounds. Each version uses a different cipher key size (128, 192, or 256), but the round keys are always 128 bits.

General design of AES encryption cipher



Nr	Key size
10	128
12	192
14	256

Relationship between number of rounds and cipher key size

7.1.4 *Data Units. :*

AES uses five units of measurement to refer the data

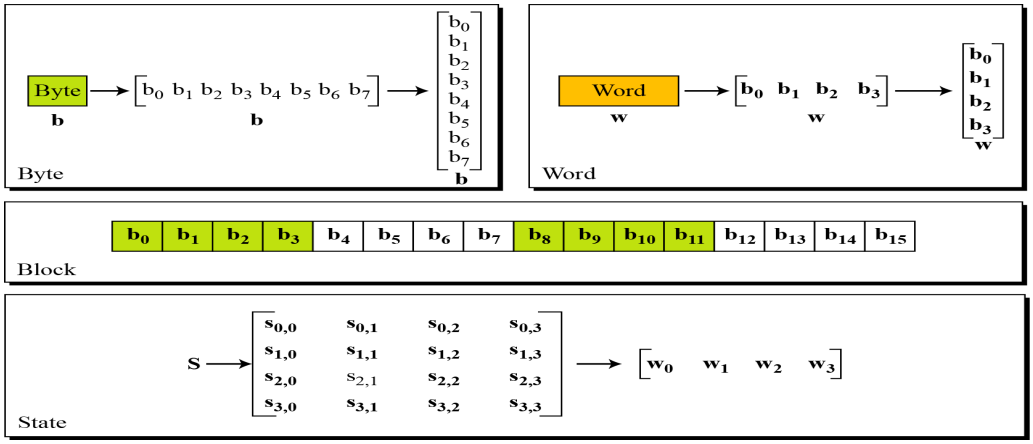
Bit: In AES, a bit is a binary digit with a value of 0or 1.

Byte: A byte is a group of eight bits can be treated as a single entity, a row matrixof eight bits or a column matrix of eight bits. When treated as row matrix, the bits are inserted from left to right, when treated as a columnmatrix the bits are inserted from top to bottom

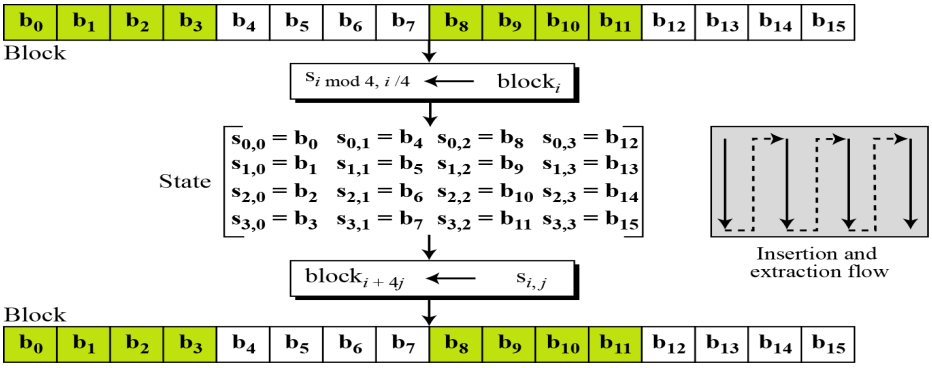
Word: A word is a group of 32 bits that can be treated as a row matrix of four bytes or a column matrix of four bytes.

Block: AES encrypts and decrypts data blocks. A block in AES is a group of 128 bits, a block can be represented as a row matrix of 16 bytes.

State: AES uses several rounds in which each round is made of several stages. Data block is transformed from one stage to another. Before and after each stage, the data block is referred as state. State is made up of 16 bytes, treated as 4x4 byte matrix. Bytes in data block are inserted into a state column, and in each column, from top to bottom.



Block-to-state and state-to-block transformation



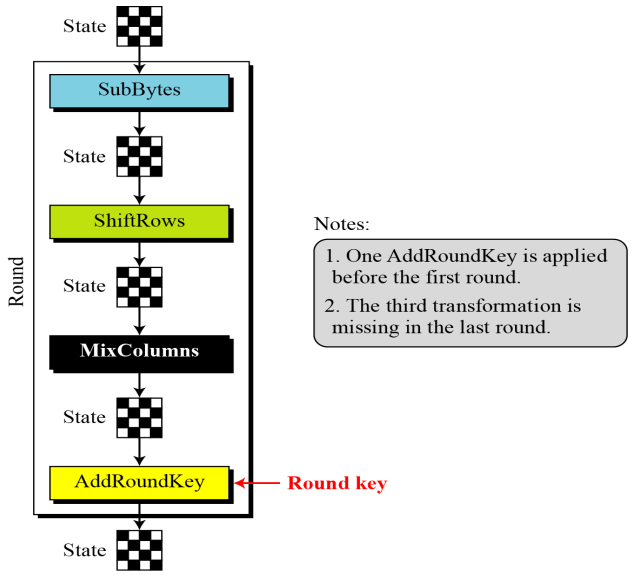
Example:

Text	A	E	S	U	S	E	S	A	M	A	T	R	I	X	Z	Z
Hexadecimal	00	04	12	14	12	04	12	00	0C	00	13	11	08	23	19	19

$$\begin{bmatrix} 00 & 12 & 0C & 08 \\ 04 & 04 & 00 & 23 \\ 12 & 12 & 13 & 19 \\ 14 & 00 & 11 & 19 \end{bmatrix}$$ State

7.1.5 Structure of Each Round

Each round except the last, uses four transformations that are invertible. The last round has only three transformations. Each transformation takes a state and creates another state to be used for the next transformation or the next round. The pre-round section uses only one transformation(AddRoundKey), the last round uses only three transformations(MixColumns transformation is missing).



7-2 TRANSFORMATIONS

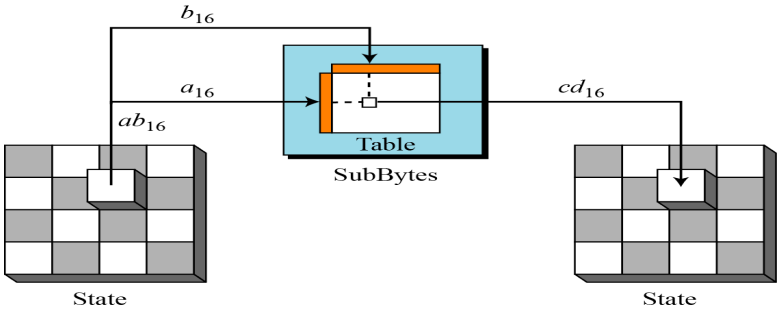
To provide security, AES uses four types of transformations: substitution, permutation, mixing, and key-adding.

7.2.1 Substitution

AES, like DES, uses substitution. AES uses two invertible transformations.

SubBytes

The first transformation, SubBytes, is used at the encryption site. To substitute a byte, we interpret the byte as two hexadecimal digits. The left digit defines the row and the right digit defines the column of the substitution table. The two hexadecimal digits at the junction of the row and column are the new byte. The SubBytes operation involves 16 independent byte-to-byte transformations.



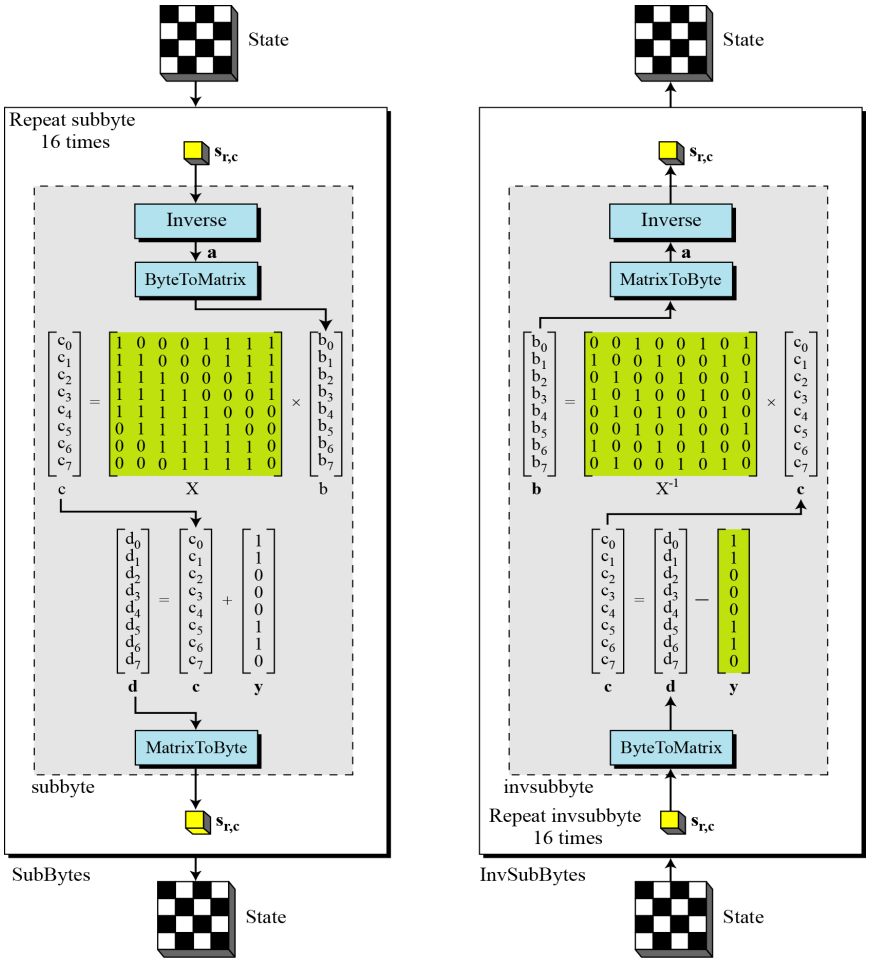
Transformation Using the GF(2^8) Field

AES also defines the transformation algebraically using the GF(28) field with the irreducible polynomials ($x^8 + x^4 + x^3 + x + 1$), as shown

$$\text{subbyte:} \rightarrow \mathbf{d} = \mathbf{X}\,(s_{r,c})^{-1} \oplus \mathbf{y}$$
$$\text{invsubbyte:} \rightarrow [\mathbf{X}^{-1}(\mathbf{d} \oplus \mathbf{y})]^{-1} = [\mathbf{X}^{-1}(\mathbf{X}\,(s_{r,c})^{-1} \oplus \mathbf{y} \oplus \mathbf{y})]^{-1} = [(s_{r,c})^{-1}]^{-1} = s_{r,c}$$

The SubBytes and InvSubBytes transformations are inverses of each other.

SubBytes and InvSubBytes processes

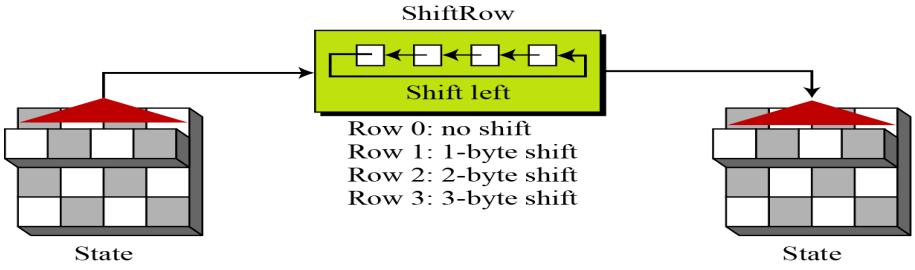


7.2.2 Permutation

Another transformation found in a round is shifting, which permutes the bytes. Unlike the DES, in which permutation is done at bit level, shifting transformation in AES is done at the byte level, the order of the bits in the byte are not changed

ShiftRows

In the encryption, the transformation is called ShiftRows and the shifting is to the left. The number of shifts depends on the row number of the matrix.



InvShiftRows

In the decryption, the transformation is called InvShiftRows and the shifting is to the right.

7.2.3 Mixing

We need an interbyte transformation that changes the bits inside a byte, based on the bits inside the neighboring bytes. We need to mix bytes to provide diffusion at the bit level.

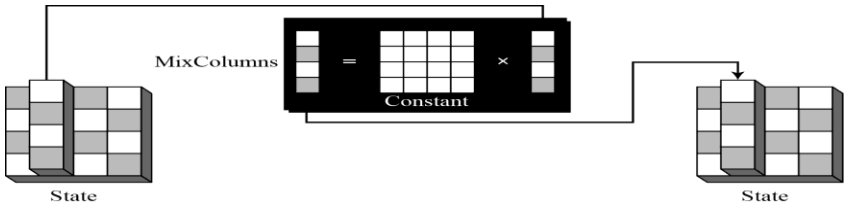
Mixing bytes using matrix multiplication

$$\begin{bmatrix} ax + by + cz + dt \\ ex + fy + gz + ht \\ ix + jy + kz + lt \\ mx + ny + oz + pt \end{bmatrix} = \begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{bmatrix} \times \begin{bmatrix} x \\ y \\ z \\ t \end{bmatrix}$$

New matrix Constant matrix Old matrix

MixColumns

The substitution provided by the SubBytes transformation changes the value of the byte based only on the original value and an entry in the table. It does not include neighboring bytes. SubBytes is intrabyte transformation. The MixColumns transformation operates at the column level; it transforms each column of the state to a new column. It is an interbyte transformation that changes the bits inside the byte, based on the bits inside the neighboring bytes. It provides diffusion at the bit level.



InvMixColumns

The InvMixColumns transformation is basically the same as the MixColumns transformation.

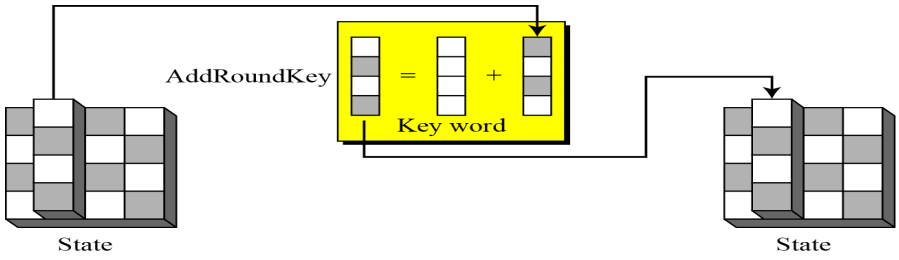
7.2.4 Key Adding

The most important transformation is the one that includes the cipher key. AES uses a process called key expansion that creates Nr+1 round keys from the cipher key. Each round key is 128 bit long. It is treated as four 32 bit words.

AddRoundKey

AddRoundKey proceeds one column at a time. AddRoundKey adds a round key word with each state column matrix; the operation in AddRoundKey is matrix addition.

The AddRoundKey transformation is the inverse of itself.



7-3 KEY EXPANSION

To create round keys for each round, AES uses a key-expansion process. If the number of rounds is N_r , the key-expansion routine creates $N_r + 1$ 128-bit round keys from one single 128-bit cipher key. The first round key is used for pre-round transformation, the remaining round keys are used for the last transformation at the end of each round.

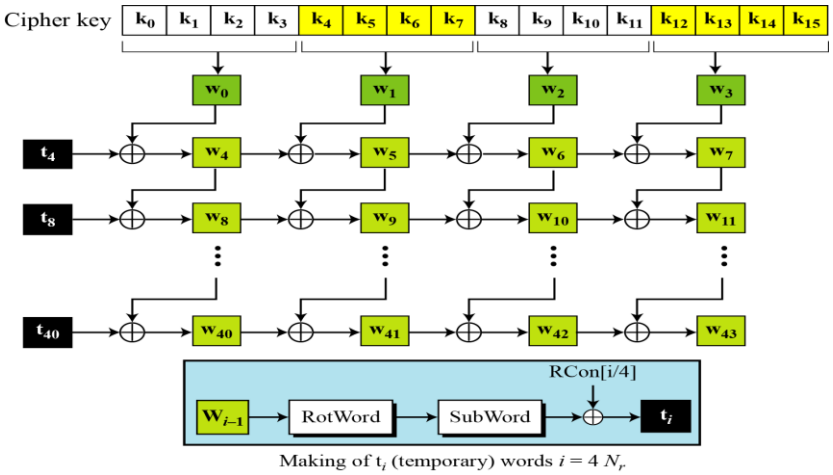
7.3.1 Key Expansion in AES-128

The first four words(w0,w1,w2,w3) are made from the cipher key(k0 to k15). The first four bytes(k0 to k3) become w0, k4 to k7 become w1 and so on.

The rest of the words(wi for i=4 to 43) are made as follows.

a. If (i mod 4)!=0 w_i=w_{i-1}XOR w_{i-4}

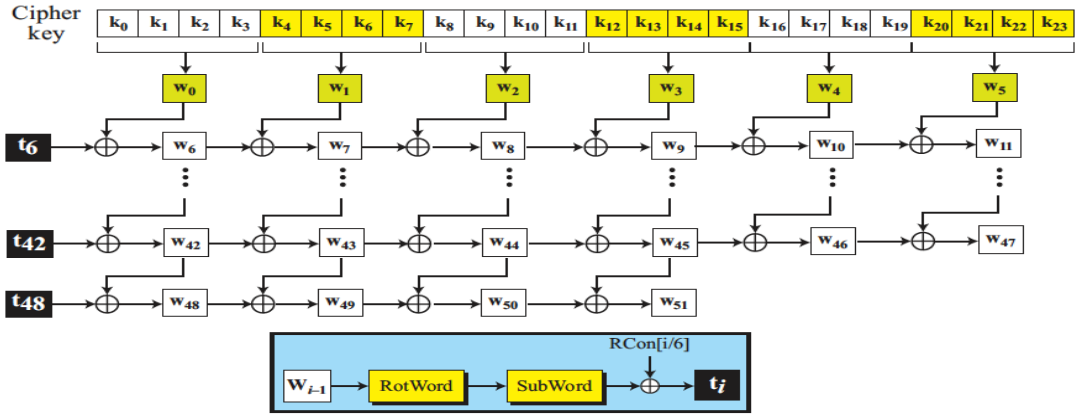
b. If (i mod 4)=0 w_i=t XOR w_{i-4}



Making of t_i (temporary) words $i = 4\ N_r$

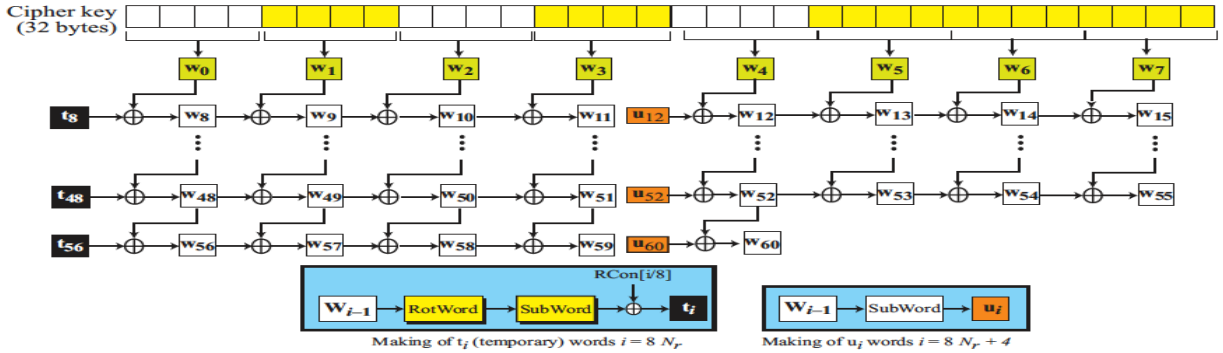
The key-expansion routine can either use the above table when calculating the words or use the GF(2^8) field to calculate the leftmost byte dynamically, as shown below (prime is the irreducible polynomial):

7.3.2 Key Expansion in AES-192



Making of t_i (temporary) words $i = 6\ N_r$

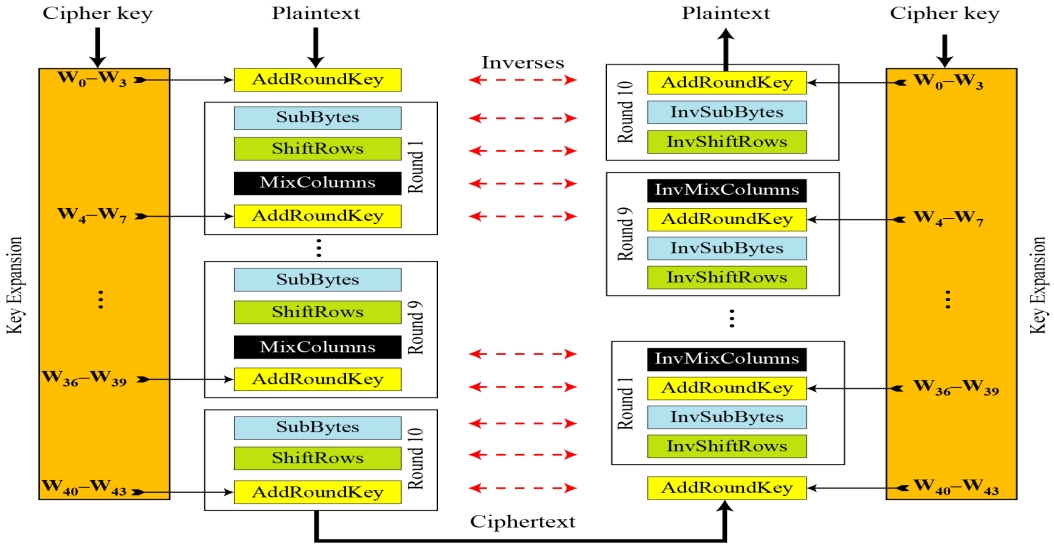
7.3.2 Key Expansion in AES-256



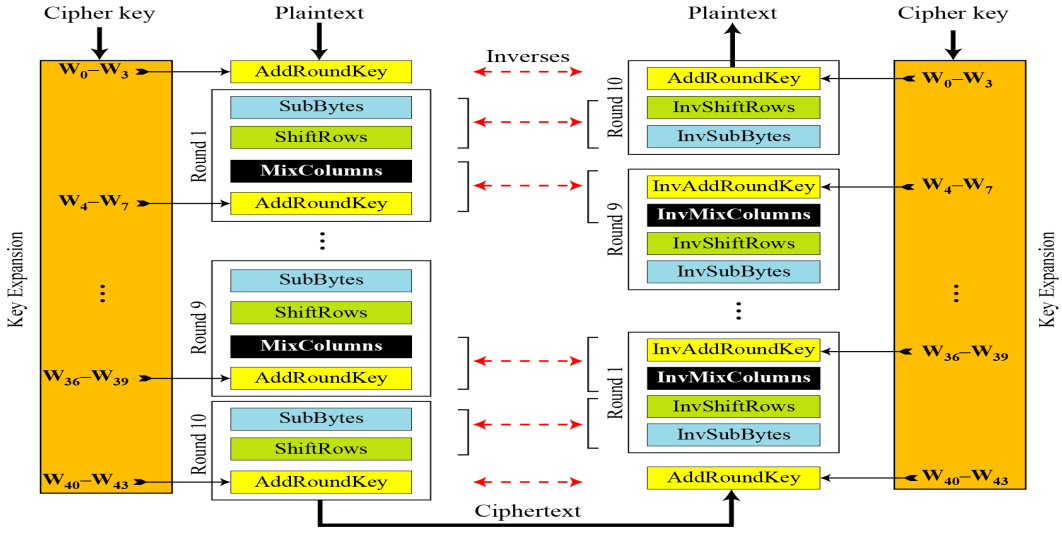
7-4 CIPHERS

AES uses four types of transformations for encryption and decryption. In the standard, the encryption algorithm is referred to as the cipher and the decryption algorithm as the inverse cipher.

7.4.1 Original Design



7.4.2 Cipher and reverse cipher in alternate design



7-6 ANALYSIS OF AES

AES was designed after DES. Most of the known attacks on DES were already tested on AES.

Brute-Force Attack

AES is definitely more secure than DES due to the larger-size key(128,192,256 bits). For DES we need 2^{56}tests to find the key, but for AES we need minimum of 2^{128} tests to find the key. If we break DES in t seconds, we need 2^{72}xt seconds to break AES which is impossible. There are no weak keys in AES

Statistical Attacks

The strong diffusion and confusion provided by the combination of the SubBytes, ShiftRows and MixColumns removes the frequency pattern in the plaintext. Numerous tests have failed to do statistical analysis of the ciphertext.

Differential and Linear Attacks

There are no differential and linear cryptanalysis attacks on AES as yet.

7.6.2 Implementation

AES can be implemented in software, hardware, and firmware. The implementation can use table lookup process or routines that use a well-defined algebraic structure. The transformation can be either byte oriented or word oriented. In the byte oriented version, the whole algorithm can use 8 bit processor. In word oriented version, it can use a 32-bit processor.

7.6.3 Simplicity and Cost

The algorithms used in AES are so simple that they can be easily implemented using cheap processors and a minimum amount of memory.

UNIT III

Asymmetric Key Cryptograhy

Mathematics of Asymmetric Key Cryptography, Asymmetric Key Cryptography

Chapter 9

Mathematics of Cryptography *Part III:*

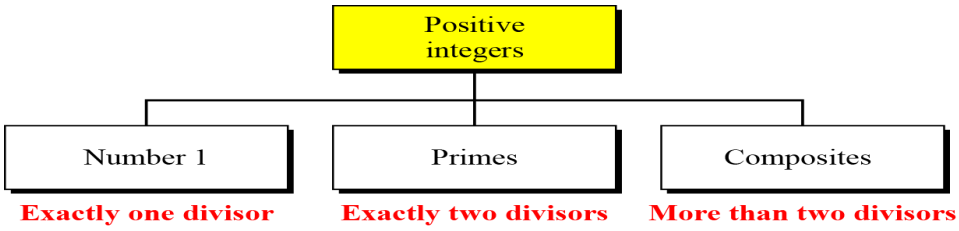
Primes and Related Equations

9.1 PRIMES

9.1.1 Definition

A prime is divisible only by itself and 1.

The positive integers are divided into three groups.



What is the smallest prime?

The smallest prime is 2, which is divisible by 2 (itself) and 1.

List the primes smaller than 10.

There are four primes less than 10: 2, 3, 5, and 7. It is interesting to note that the percentage of primes in the range 1 to 10 is 40%. The percentage decreases as the range increases.

Coprimes: the positive integers a and b are relatively prime or coprime, if gcd(a,b)=1. The number is relatively prime with any integer. if p is prime, then all the integers from 1 to p-1 are relatively prime to p.

9.1.2 Cardinality of Primes

Given a number n, how many primes are smaller than or equal to n is the cardinality of primes.

The number of primes is infinite.

Suppose that the set of primes is finite, with p as largest prime.

Multiply the set of primes and call the result as P=2x3x5….xp

The integer (P+1) cannot have a factor q<=p. we know that q divides P

If q also divides (P+1) then q divides (P+1)-P=1

The only number that divides 1 is 1, which is not a prime.

Therefore q is larger than p

Example

As a trivial example, assume that the only primes are in the set {2, 3, 5, 7, 11, 13, 17}.

Here P = 510510 and P + 1 = 510511.
However, 510511 = 19 × 97 × 277;
none of these primes were in the original list. Therefore, there are three primes greater than 17.

Number of Primes : A function called π(n) is defined that finds the number of primes smaller than or equal to n. the following shows the values of this function

$$[n / (\ln n)] \quad < \quad \pi(n) \quad < \quad [n/(\ln n - 1.08366)]$$

Gauss defend theupper limit and Lagrange discovered the lower limit

Ex: Find the number of primes less than 1,000,000.
Find the number of primes less than 1,000,000.

9.1.3 Checking for Primeness

Given a number n, how can we determine if n is a prime? The answer is that we need to see if the number is divisible by all primes less than $\sqrt{n}$.We know that this method is inefficient, but it is a good start.

Theorem

If n is composite, then n has a prime divisor less than or equal to $\sqrt{n}$.

Proof.

- Let $n = ab$, $1 < a < n$, $1 < b < n$.
- We can't have both $a > \sqrt{n}$ and $b > \sqrt{n}$ since this would lead to $ab > n$.
- Therefore, n must have a prime divisor less than or equal to $\sqrt{n}$.

□

Example:

1. Is 97 a prime?

The floor of √97 = 9. The primes less than 9 are 2, 3, 5, and 7. We need to see if 97 is divisible by any of these numbers. It is not, so 97 is a prime.

2. **Is 301 a prime?**

The floor of √301 = 17. We need to check 2, 3, 5, 7, 11, 13, and 17. The numbers 2, 3, and 5 do not divide 301, but 7 does. Therefore 301 is not a prime.

Sieve of Eratosthenes

The greek mathematician Eratosthenes devised a method to find all primes less than n. The method is called sieve of Ertosthenes.

If we want to find all the primes less than 100. We write all the numbers from 2 to 100. We find the square root of 100. It is 10. we need to see if any number is divisible by the prime numbers less than 10 i.e., 2,3,5,7.

We cross out all the numbers divisible by 2,3,5,7 except those numbers

Table 9.1 *Sieve of Eratosthenes*

	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	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

9.1.4 Euler's Phi-Function

Euler's phi-function, f (n), which is sometimes called the Euler's totient function plays a very important role in cryptography.

1. $\phi(1) = 0$.
2. $\phi(p) = p - 1$ if p is a prime.
3. $\phi(m \times n) = \phi(m) \times \phi(n)$ if m and n are relatively prime.
4. $\phi(p^e) = p^e - p^{e-1}$ if p is a prime.

We can combine the above four rules to find the value of f(n). For example, if n can be factored as

$$n = p_1^{e1} \times p_2^{e2} \times \ldots \times p_k^{ek}$$

then we combine the third and the fourth rule to find

$$\phi(n) = (p_1^{e_1} - p_1^{e_1-1}) \times (p_2^{e_2} - p_2^{e_2-1}) \times \cdots \times (p_k^{e_k} - p_k^{e_k-1})$$

The difficulty of finding f(n) depends on the difficulty of finding the factorization of n.

Examples:

1.What is the value of Ø(13)?

We can use the third rule: Ø(10) = Ø(2) × Ø (5) = 1 × 4 = 4, because 2 and 5 are primes.

2. **What is the value of Ø (240)?**

Solution

We can write $240 = 2^4 \times 3^1 \times 5^1$. Then

$$Ø\ (240) = (2^4 - 2^3) \times (3^1 - 3^0) \times (5^1 - 5^0) = 64$$

3. **Can we say that Ø (49) = Ø (7) × Ø (7) = 6 × 6 = 36?**

Solution

No. The third rule applies when m and n are relatively prime. Here $49 = 7^2$. We need to use the fourth rule: Ø (49) = $7^2 - 7^1 = 42$.

4. What is the number of elements in Z_{14}*?

Solution

The answer is Ø (14) = Ø (7) × Ø (2) = 6 × 1 = 6. The members are 1, 3, 5, 9, 11, and 13.

9.1.5 Fermat's Little Theorem

Fermats Little theorem provides a very important role in number theory and cryptography.

First Version :

Thr first version says that if p is a prime and a is an integer such that p does not divide a then

$a^{p-1} \equiv 1 \bmod p$

Second Version

The second version removes the condition on a. it says that if p is a prime and a is an integer, then

$a^p \equiv a \bmod p$

Examples:

1.Find the result of 6^{10} mod 11

Using the rule $a^{p-1} \equiv \bmod p$

$6^{11-1} \equiv 1 \bmod 11$

i.e., $6^{10} \bmod 11 \equiv 1$

2. Find the result of 3^{12} mod 11

$=3^{12} \bmod 11 = (3^{11} x 3) \bmod 11$

$= (311 \bmod 11) x (3 \bmod 11)$

$= (3 \times 3) \bmod 11$

$= 9$

Multiplicative Inverses

If p is a prime and a is an integer such that p does not divide a then

$a^{-1} \bmod p = a^{p-2} \bmod p$

Proof:

$axa^{-1} \bmod p = axa^{p-2} \bmod p$

$1 \bmod p = a^{p-2} \bmod p$

$a^{p-2} \bmod p = 1 \bmod p$

Examples:

1. find the result of 8^{-1} mod 17

According to Fermats Little theorem

$8^{-1} \bmod 17 = 8^{17-2} \bmod 17$

$=8^{15} \bmod 17$

Using fast modular exponentiation

Me mod n

e=15, m=5, n=17

convert 15 to binary

	1	1	1	1
d^2	1	13	4	4
d^2m	8	2	15	**15**

8-1 mod 17=15 mod 17

2.fi**nd the result of 5^{-1} mod 23**

According to Fermats Little theorem

$5^{-1} \bmod 23 = 5^{23-2} \bmod 23$

$=5^{21} \bmod 23$

Using fast modular exponentiation

$m^e \bmod n$

e=21, m=5, n=23

convert 21 to binary

	1	0	1	0	1
d^2	1	2	4	9	12
d^2m	5	x	20	x	14

$5^{-1} \bmod 23 = 14 \bmod 23$

3.find the result of $60^{-1} \bmod 101$

According to Fermats Little theorem

$60^{-1} \bmod 101 = 60^{101-2} \bmod 101$

$=60^{99} \bmod 101$

Using fast modular exponentiation

$m^e \bmod n$

e=99, m=60, n=101

convert 99 to binary

	1	1	0	0	0	1	1
d^2	1	65	6	36	84	87	14
d^2m	60	62	x	x	x	69	32

$60^{-1} \bmod 101 = 32 \bmod 101$

Find the results of the following, using Fermat's little theorem:

a. $5^{15} \bmod 13$

b. $15^{18} \bmod 17$

c. $456^{17} \bmod 17$

d. $145^{102} \bmod 101$

Find the results of the following, using Fermat's little theorem:

a. $5^{-1} \bmod 13$

b. $15^{-1} \bmod 17$

c. $27^{-1} \bmod 41$

d. $70^{-1} \bmod 101$

Note that all moduli are primes.

9.1.6 Euler's Theorem

Eulers theorem is a generalization of fermats little theorem. The modulus in the fermats theorem is prime, the modulus in eulers theorem is an integer

First Version:

If a and n are coprime, then

$a^{f(n)} \equiv 1 \pmod{n}$

Second Version:

It removes a condition that a and n are coprimes. If n=p x q , a<n, k is an integer then

$a^{k \times f(n) + 1} \equiv a \pmod{n}$

The second version of Euler's theorem is used in the RSA cryptosystem.

Proof for second version:

1. if a is neither a multiple of p nor a multiple of q then a and n are coprimes.
   $a^{kx\emptyset(n)+1} \bmod n = ((a^{\emptyset(n)})^k \times a) \bmod n$
   $= (a^{\emptyset(n)} \bmod n)^k \times a \bmod n$
   $= 1^k \times a \bmod n$
   $= a \bmod n$
2. if a is multiple of p(a=i x p) but not a multiple of q
   i. $a^{\emptyset(n)} \bmod q = a^{\emptyset(p \times q)} \bmod q = (a^{\emptyset(q)})^{\emptyset(p)} \bmod q = (a^{\emptyset(q)} \bmod q)^{\emptyset(p)} \bmod q = (1)^{\emptyset(p)} \bmod q = 1$
   ii. $a^{k \times \emptyset(n)} \bmod q = (a^{\emptyset(q)})^k \bmod q = (a^{\emptyset(q)} \bmod q)^k \bmod q = 1^k \bmod q = 1$
   iii. $a^{k \times \emptyset(n)} \bmod q = 1 + j \times q$
   iv. $a^{k \times \emptyset(n)+1} = a \times a^{k \times \emptyset(n)} = a \times (1 + j \times q) = a + j \times q \times a = a + j \times q \times i \times p = a + (i \times j) \times p \times q = a + (i \times j) \times n$
   v. $a^{k \times \emptyset(n)+1} = a + (i \times j) \times n$ implies $a^{k \times \emptyset(n)+1} = a \bmod n$
   vi.
3. If a is a multiple of q(a=i x q) bt not multiple of p. we can prove this in the same way as case 2 by interchanging p and q.

Examples

1. **Find $6^{24} \bmod 35$**
   $6^{24} \bmod 35 = 6^{4 \times 6} \bmod 35 = 6^{\emptyset(5) \times \emptyset(7)} \bmod 35$
   $= 6^{\emptyset(35)} \bmod 35 = 1$
2. **Find the result of $20^{62} \bmod 77$**
   $20^{62} \bmod 77 = (20 \bmod 77) \times (20^{61} \bmod 77)$
   $= (20 \bmod 77) \times (20^{60+1} \bmod 77)$
   $= (20 \bmod 77) \times (20^{6 \times 10+1} \bmod 77)$
   $= (20 \bmod 77) \times (20^{\emptyset(7) \times \emptyset(11)+1} \bmod 77)$
   $= (20 \bmod 77) \times (20^{\emptyset(77)+1} \bmod 77)$
   $= (20 \bmod 77) \times (20 \bmod 77)$
   $= (20 \times 20) \bmod 77$
   $= 400 \bmod 77 = 15$

Multiplicative Inverses

Euler's theorem can be used to find multiplicative inverses modulo a composite. If n and a are coprime, then $a^{-1} \bmod n = a^{\emptyset(n)-1} \bmod n$

Examples

1.Find 8^{-1} mod 77

$8^{-1} \bmod 77 = 8^{\emptyset(77)-1} \bmod 77$

$= 8^{\emptyset(7)x\ \emptyset(11)-1} \bmod 77$

$= 8^{6x\ 10-1} \bmod 77$

$= 8^{59} \bmod 77$

Using fast modular exponentiation

Write 59 in binary

	1	1	1	0	1	1
d^2	1	64	36	15	71	71
d^2m	8	50	57	x	29	29

$8^{-1} \bmod 77 = 29 \bmod 77$

2. Find 7^{-1} mod 15

$7^{-1} \bmod 15 = 7^{\emptyset(15)-1} \bmod 15$

$= 7^{\emptyset(3)x\ \emptyset(5)-1} \bmod 15$

$= 7^{2x\ 4-1} \bmod 15$

$= 7^{7} \bmod 15$

Using fast modular exponentiation

Write 7 in binary

	1	1	1
d^2	1	4	4
d^2m	7	13	13

$7^{-1} \bmod 15 = 13 \bmod 15$

3. Find 60^{-1} mod 187

$60^{-1} \bmod 187 = 60^{\emptyset(187)-1} \bmod 187$

$= 60^{\emptyset(17)x\ \emptyset(11)-1} \bmod 187$

$= 60^{16x\ 10-1} \bmod 187$

$= 60^{159} \bmod 187$

Using fast modular exponentiation

Write 159 in binary

	1	0	0	1	1	1	1	1
d^2	1	47	152	103	81	4	4	4
d^2m	60	x	x	9	185	53	53	53

$60^{-1} \bmod 187 = 53 \bmod 187$

3. Find 60^{-1} mod 187

60^{-1} mod 187= $60^{\emptyset(187)-1}$ mod 187

=$60^{\emptyset(17)x\ \emptyset(11)-1}$ mod 187

=$60^{16x\ 10-1}$ mod 187

=60^{159} mod 187

Using fast modular exponentiation

Write 159 in binary

	1	0	0	1	1	1	1	1
d^2	1	47	152	103	81	4	4	4
d^2m	60	x	x	9	185	53	53	53

60^{-1} mod 187=53 mod 187

4. Find 71^{-1} mod 100

60^{-1} mod 100= $71^{\emptyset(100)-1}$ mod 100

$\emptyset(100)= \emptyset(2^2 \times 5^2)=(2^2-2^1) \times(5^2-5^1)=2*20=40$

=71^{40-1} mod 100

=71^{39} mod 100

Using fast modular exponentiation

Write 39 in binary

	1	0	0	1	1	1
d^2	1	41	81	61	61	61
d^2m	71	x	x	31	31	31

71^{-1} mod 100=31 mod 100

Find the results of the following, using Euler's theorem:

a. 12^{-1} mod 77

b. 16^{-1} mod 323

c. 20^{-1} mod 403

d. 44^{-1} mod 667

Note that $77 = 7 \times 11$, $323 = 17 \times 19$, $403 = 31 \times 13$, and $667 = 23 \times 29$.

9.1.7 Generating Primes

Two Mathematicians Mersenne and Fermant , attempted to develop a formula that could generate primes.

1. **Mersenne Primes**

A number in the form $M_p = 2^p - 1$ is called a Mersenne number and may or may not be a prime.

Where p is prime, them Mp is prime.

$M_2 = 2^2 - 1 = 3$
$M_3 = 2^3 - 1 = 7$
$M_5 = 2^5 - 1 = 31$
$M_7 = 2^7 - 1 = 127$
$M_{11} = 2^{11} - 1 = 2047$ **Not a prime ($2047 = 23 \times 89$)**
$M_{13} = 2^{13} - 1 = 8191$
$M_{17} = 2^{17} - 1 = 131071$

M11 is not a prime.
The latest one is M124036583 which is 7253733 digits.

2.Fermat Primes

$$F_n = 2^{2^n} + 1$$

$F_0 = 3$ $F_1 = 5$
$F_2 = 17$ $F_3 = 257$
$F_4 = 65537$
$F_5 = 4294967297 = 641 \times 6700417$ Not a prime

9-2 PRIMALITY TESTING

Finding an algorithm to correctly and efficiently test a very large integer and output a prime or a composite has always been a challenge in number theory, and consequently in cryptography. However, recent developments look very promising.
Algorithms that handle this issue can be divided into two categories: probabilistic and Deterministic algorithms. Deterministic algorithms give correct answer but probabilistic algorithms give almost correct answer.

9.2.1 Deterministic Algorithms

A deterministic primality testing algorithm accepts an integer and always output a prime or composite.
Divisibility Algorithm:
The most elementary deterministic test for primality is the divisibility test. We use as divisors all the numbers smaller than square root of n. if any of these numbers divide n, the n is composite.
The algorithm can be improved by testing only odd numbers. It can be further improved by testing only primes between 2 and square root on n.

Algorithm 9.1 *Pseudocode for the divisibility test*

```
Divisibility_Test (n)                    // n is the number to test for primality
{
  r ← 2
  while (r < √n )
  {
   if (r | n) return "a composite"
   r ← r + 1
  }
  return "a prime"
}
```

Assume n has 200 bits. What is the number of bit operations needed to run the divisibility-test algorithm?
Solution
The bit-operation complexity of this algorithm is $2^{nb/2}$. This means that the algorithm needs 2^{100} bit operations. On a computer capable of doing 2^{30} bit operations per second, the algorithm needs 2^{70} seconds to do the testing (forever).

AKS Algorithm

In 2002, Agarwal, Kayal and Saxena found an algorithm for primality testing with a polynomial bit operation of $O((\log n_b)^{12})$.the algorithm uses the fact that $(x-a)^p=(x^p-a) \bmod p$.

Assume n has 200 bits. What is the number of bit operations needed to run the AKS algorithm?

Solution

This algorithm needs only $(\log_2 200)^{12}$ = 39,547,615,483 bit operations. On a computer capable of doing 1 billion bit operations per second, the algorithm needs only 40 seconds.

9.2.2 Probabilistic Algorithms

Before the AKS algorithm, all efficient methods for primality testing have been probabilistic. These methods areused until the AKS is formally accepted as a standard. A probabilistic algorithm does not guarantee the correctness of the result.we can make the probability of error so small that it is almost the correct answer. A probabilistic algorithm returns either a prime or composite based on the following rules

a. If the integer to be tested is actually a prime, the algorithm definitely returns a prime.
b. If the integer to be tested is actually a composite, it returns a composite with probability 1-ε, but it returns a prime with the probability of ε.

The probability of the mistake can be improved if we run the algorithm more than once with different parameters or using different methods. If we run the algorithm m times, the probability of error may reduce to ε^m

Fermat Test

the fermats theorem specifies that

$$\textbf{If } n \textbf{ is a prime, then } a^{n-1} \equiv 1 \bmod n.$$

If n is a prime, the congruence holds. It does not mean that if congruence holds, n is a prime. The integer can be prime or composite.

If n is prime, an-1 ≡ 1 mod n

If n is composite, it is possible that a n-1≡ 1 mod n

A prime passes the test, a composite may pass the fermat test with a probability of ε.

Example

Does the number 561 pass the Fermat test?

Solution

Use base 2

$$2^{561-1} = 1 \bmod 561$$

The number passes the Fermat test, but it is not a prime, because 561 = 33 × 17.

Square Root Test

In modular arithmetic if n is prime, the square root of 1 is either +1 or -1. If n is composite the square root if +1 or _1, but there may be other roots.this is knownas the square root primality test. In modular arithmetic, -1 means n-1

If n is a prime, $\sqrt{1} \bmod n = \pm 1$.
If n is a composite, $\sqrt{1} \bmod n = \pm 1$ and possibly other values.

What are the square roots of 1 mod n if n is 7 (a prime)?

Solution

The only square roots are 1 and −1. We can see that

$$1^2 = 1 \bmod 7 \qquad (-1)^2 = 1 \bmod 7$$
$$2^2 = 4 \bmod 7 \qquad (-2)^2 = 4 \bmod 7$$
$$3^2 = 2 \bmod 7 \qquad (-3)^2 = 2 \bmod 7$$

What are the square roots of 1 mod n if n is 8 (a composite)?

Solution

There are four solutions: 1, 3, 5, and 7 (which is −1). We can see that

$$1^2 = 1 \bmod 8 \qquad (-1)^2 = 1 \bmod 8$$
$$3^2 = 1 \bmod 8 \qquad 5^2 = 1 \bmod 8$$

What are the square roots of 1 mod n if n is 17 (a prime)?

$$1^2 = 1 \bmod 17 \qquad (-1)^2 = 1 \bmod 17$$
$$2^2 = 4 \bmod 17 \qquad (-2)^2 = 4 \bmod 17$$
$$3^2 = 9 \bmod 17 \qquad (-3)^2 = 9 \bmod 17$$
$$4^2 = 16 \bmod 17 \qquad (-4)^2 = 16 \bmod 17$$
$$5^2 = 8 \bmod 17 \qquad (-5)^2 = 8 \bmod 17$$
$$6^2 = 2 \bmod 17 \qquad (-6)^2 = 2 \bmod 17$$
$$(7)^2 = 15 \bmod 17 \qquad (-7)^2 = 15 \bmod 17$$
$$(8)^2 = 13 \bmod 17 \qquad (-8)^2 = 13 \bmod 17$$

Solution

There are only two solutions: 1 and −1

What are the square roots of 1 mod n if n is 22 (a composite)?

Solution

Surprisingly, there are only two solutions, +1 and −1, although 22 is a composite.

$$1^2 = 1 \bmod 22$$
$$(-1)^2 = 1 \bmod 22$$

Miller-Rabin Test

Miller Rabin primality test combines the fermat test and the square root test to find a strong pseudo prime that is a prime with a very high probability. In this test we write n-1 as the product of an odd number m and a power of 2

$$n-1 = m \times 2^k$$

Idea behind Fermat primality test

$$a^{n-1} = a^{m \times 2^k} = [a^m]^{2^k} = [a^m]^{2^{2^{\cdot^{\cdot^{\cdot^{2}}}}}} \quad \text{(k times)}$$

Instead of calculating an-1 (mod n) in one step, we can do it in k+1 steps. The benefit is that, in each step the square root test can be performed. If it fails we can declare the number as composite.

Algorithm 9.2 *Pseudocode for Miller-Rabin test*

```
Miller_Rabin_Test (n, a)                    // n is the number; a is the base.
{
    Find m and k such that n − 1 = m × 2^k
    T ← a^m mod n
    if (T = ± 1)  return "a prime"
    for (i ← 1 to k − 1)                    // k − 1 is the maximum number of steps.
    {
        T ← T^2 mod n
        if (T = +1) return "a composite"
        if (T = −1) return "a prime"
    }
    return "a composite"
}
```

Does the number 561 pass the Miller-Rabin test?

Solution

Using base 2, let $561 - 1 = 35 \times 2^4$, which means m = 35, k = 4, and a = 2.

Initialization:	$T = 2^{35} \bmod 561 = 263 \bmod 561$	
$k = 1$:	$T = 263^2 \bmod 561 = 166 \bmod 561$	
$k = 2$:	$T = 166^2 \bmod 561 = 67 \bmod 561$	
$k = 3$:	$T = 67^2 \bmod 561 = +1 \bmod 561$	$\rightarrow$ **a composite**

We already know that 27 is not a prime. Let us apply the Miller-Rabin test.

Solution

With base 2, let $27 - 1 = 13 \times 2^1$, which means that m = 13, k = 1, and a = 2. In this case, because k − 1 = 0, we should do only the initialization step: $T = 2^{13} \bmod 27 = 11 \bmod 27$. However, because the algorithm never enters the loop, it returns a composite.

We know that 61 is a prime, let us see if it passes the Miller-Rabin test.

Solution

We use base 2.

$61 - 1 = 15 \times 2^2$	$\rightarrow \quad m = 15 \quad k = 2 \quad a = 2$	
Initialization:	$T = 2^{15} \bmod 61 = 11 \bmod 61$	
$k = 1$	$T = 11^2 \bmod 61 = -1 \bmod 61$	$\rightarrow$ **a prime**

9.2.3 Recommended Primality Test

Today, one of the most popular primality test is a combination of the divisibility test and the Miller-Rabin test. The number 4033 is a composite (37 × 109). Does it pass the recommended primality test?

Solution

1. Perform the divisibility tests first. The numbers 2, 3, 5, 7, 11, 17, and 23 are not divisors of 4033.
2. Perform the Miller-Rabin test with a base of 2, 4033 − 1 = 63 × 26, which means m is 63 and k is 6.

Initialization: $T \equiv 2^{63} \pmod{4033} \equiv 3521 \pmod{4033}$

$k = 1$ $\quad T \equiv T^2 \equiv 3521^2 \pmod{4033} \equiv -1 \pmod{4033}$ $\rightarrow$ **Passes**

3. But we are not satisfied. We continue with another base, 3.

Initialization: $T \equiv 3^{63} \pmod{4033} \equiv 3551 \pmod{4033}$

$k = 1$ $\quad T \equiv T^2 \equiv 3551^2 \pmod{4033} \equiv 2443 \pmod{4033}$

$k = 2$ $\quad T \equiv T^2 \equiv 2443^2 \pmod{4033} \equiv 3442 \pmod{4033}$

$k = 3$ $\quad T \equiv T^2 \equiv 3442^2 \pmod{4033} \equiv 2443 \pmod{4033}$

$k = 4$ $\quad T \equiv T^2 \equiv 2443^2 \pmod{4033} \equiv 3442 \pmod{4033}$

$k = 5$ $\quad T \equiv T^2 \equiv 3442^2 \pmod{4033} \equiv 2443 \pmod{4033}$ $\rightarrow$**Failed (composite)**

9-3 FACTORIZATION

Factorization has been the subject of continuous research in the past; such research is likely to continue in the future. Factorization plays a very important role in the security of several public-key cryptosystems

9.3.1 Fundamental Theorem of Arithmetic

Greatest Common Divisor

$$a = p_1^{a_1} \times p_2^{a_2} \times \cdots \times p_k^{a_k} \qquad b = p_1^{b_1} \times p_2^{b_2} \times \cdots \times p_k^{b_k}$$

$$\gcd(a, b) = p_1^{\min(a_1, b_1)} \times p_2^{\min(a_2, b_2)} \times \cdots \times p_k^{\min(a_k, b_k)}$$

Least Common Multiplier

$$a = p_1^{a1} \times p_2^{a2} \times \cdots \times p_k^{ak} \qquad b = p_1^{b1} \times p_2^{b2} \times \cdots \times p_k^{bk}$$

$$\text{lcm}(a, b) = p_1^{\max(a_1, b_1)} \times p_2^{\max(a_2, b_2)} \times \cdots \times p_k^{\max(a_k, b_k)}$$

$$\text{lcm}(a, b) \times \gcd(a, b) = a \times b$$

9.3.2 Factorization Methods

There has been a long search for efficient algorithm to factor large composite numbers. Unfortunately ther is no perfect algorithm capable of factoring a very large number in reasonable time.

Trial Division Method

The simplest and the least efficient algorithm is the trial division factorization method. It tries all the integers starting with 2 which divides n. we know that if n is composite, then it will have a prime p<= square root of n.

Algorithm 9.3 *Pseudocode for trial-division factorization*

```
Trial_Division_Factorization (n)          // n is the number to be factored
{
    a ← 2
    while (a ≤ √n )
    {
        while (n mod a = 0)
        {
            output a                      // Factors are output one by one
            n = n / a
        }
        a ← a + 1
    }
    if (n > 1) output n                   // n has no more factors
}
```

9.3.3 Fermat Method

This method divides a number n into two positive integers a and b so that n= a x b.

The fermats method is based on the fact that we can find x and y such that $n=x^2-y^2$ the we have

$$n = x^2 - y^2 = a \times b \qquad \text{with a} = (\text{x} + \text{y}) \text{ and b} = (\text{x} - \text{y})$$

The method tries to find two integers a and b close to each other. It starts from the smallest integer greater than x=√n and tries to find the integer y such that the relation $y^2=x^2-n$ holds. In each iteration we see the result x2-n is a perfect square. If we find such value for y, we calculate a and b and break the loop.

Algorithm 9.4 *Pseudocode for Fermat factorization*

```
Feramat_Factorization (n)                 // n is the number to be factored
{
    x ← √n                                // smallest integer greater than √n
    while (x < n)
    {
    w ← x² − n
    if (w is perfect square)  y ← √w;  a ← x+y;  b ← x−y;   return a and b
    x ← x + 1
    }

}
```

9.3.4 Pollard p – 1 Method

In 1974 John P Pollard developed a method that finds a prime factor p of a number based on the condition that p-1 has no factor larger than the predefined value B, called the bound. Pollard showed that

$$p = \gcd\,(2^{\mathbf{B!}} - 1, n)$$

Algorithm 9.5 *Pseudocode for Pollard p −1 factorization*

```
Pollard_ (p − 1) _Factorization (n, B)          // n is the number to be factored
{
   a ← 2
   e ← 2
   while (e ≤ B)
   {
      a ← a^e mod n
      e ← e + 1
   }
   p ← gcd (a −1, n)
   if 1 < p < n    return p
   return failure
}
```

9.3.5 Pollard rho Method

In 1975 John P Pollard developed second method for factorization. The Pollard rho factorization is based on the following points

a. Assume that there are two integers, x1 and x2, such that p divides x1-x2, but n does not.
b. It can be proven that p=gcd(x1-x2,n). because p divides x1-x2. It can be written as x1-x2=q x p. but because n does not divide x1-x2, it is obvious that q does not divide n. this means that gcd(x1-x2,n) is either 1 or factor of n.

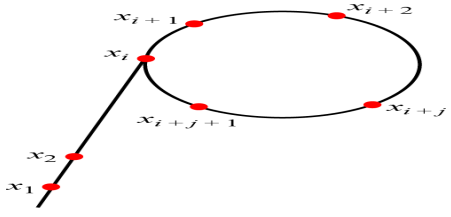
Algorithm Steps

1. Choose x1, a small integer called seed
2. Use the function $x2=f(x)=x1^2+a$(a is normally 1) such that n does not divide x1-x2.
3. Calculate the gcd(x1-x2,n) if it is 1, the result is a factor of n. if it is not 1 repeat the process by making x2 as x1.

Algorithm 9.6 *Pseudocode for Pollard rho method*

```
Pollard_ rho _Factorization (n, B)          // n is the number to be factored
{
   x ← 2
   y ← 2
   p ← 1
   while (p = 1)
   {
      x ← f(x) mod n
      y ← f (f (y) mod n) mod n
      p ← gcd (x − y, n)
   }
   return p                                 // if p = n, the program has failed
}
```

Pollard Rho successive numbers



9-4 CHINESE REMAINDER THEOREM

The Chinese remainder theorem (CRT) is used to solve a set of congruent equations with one variable but different moduli, which are relatively prime, as shown

$$\begin{aligned} x &\equiv a_1 \pmod{m_1} \\ x &\equiv a_2 \pmod{m_2} \\ &\cdots \\ x &\equiv a_k \pmod{m_k} \end{aligned}$$

Solution To Chinese Remainder Theorem

1. Find $M = m_1 \times m_2 \times \ldots \times m_k$. This is the common modulus.
2. Find $M_1 = M/m_1$, $M_2 = M/m_2$, …, $M_k = M/m_k$.
3. Find the multiplicative inverse of M_1, M_2, …, M_k using the corresponding moduli (m_1, m_2, …, m_k). Call the inverses M_1^{-1}, M_2^{-1}, …, M_k^{-1}.
4. The solution to the simultaneous equations is

$$x = (a_1 \times M_1 \times M_1^{-1} + a_2 \times M_2 \times M_2^{-1} + \cdots + a_k \times M_k \times M_k^{-1}) \bmod M$$

Examples:

1. **Find the solution to the simultaneous equations:**

x ≡ 2 mod 3

x ≡ 3 mod 5

x ≡ 1 mod 7

Solution

We follow the four steps.

1. $M = 3 \times 5 \times 7 = 105$
2. $M_1 = 105 / 3 = 35$, $M_2 = 105 / 5 = 21$, $M_3 = 105 / 7 = 15$
3. The inverses are $M_1^{-1} = 2$, $M_2^{-1} = 1$, $M_3^{-1} = 1$

   35 x X= 1 mod 3 X=2

   21 x X= 1 mod 5 X=1

   15 x X= 1 mod 7 X=1
4. $x = (2 \times 35 \times 2 + 3 \times 21 \times 1 + 2 \times 15 \times 1) \bmod 105 = 23 \bmod 105$

2. **Find an integer that has a remainder of 3 when divided by 7 and 13, but is divisible by 12.**

Solution

This is a CRT problem. We can form three equations and solve them to find the value of x.

x ≡ 3 mod 7

x ≡ 3 mod 13

x ≡ 0 mod 12

If we follow the four steps,

1. M=7x13x12=1092
2. M1=1092/7=156, M2=1092/13=84, M3=1092/12=91
3. The inverses are M

156 x M_1^{-1}= 1 mod 7 M_1^{-1}=4

84 x M_2^{-1}=1 mod 13 M_2^{-1}=11

91 x M_3^{-1}=1 mod 12 M_3^{-1}=7

4. X=(3 x 156 x 4+3 x 84 x 11+ 0x 91 x 7) mod 1092= 4474 mod 1092= 276

we find X = 276.

We can check that

276 = 3 mod 7, 276 = 3 mod 13 and 276 is divisible by 12 (the quotient is 23 and the remainder is zero).

3. Solve the linear system of congruence

x≡ 3 mod 5

x≡ 2 mod 6

x≡ 4 mod 7

M= 5 x 6 x 7=210

ni	a_i	M_i	Mi^{-1}	
5	3	210/5=42	42 x u1=1 m0d 5	u1=3
6	2	210/6=35	35 xu2= 1mod 6	u2=5
7	4	210/7=30	30 x u3=1 mod 6	u3=4

X=3 x 42 x 3+2 x 35 x 5+4 x 30 x 4= 1208 ≡158 (mod 210)

4. Solve the linear system of congruence

x≡ 1 mod 2

x≡ 2 mod 3

x≡ 3 mod 5

x≡ 4 mod 7

M= 2 x 3 x5 x 7=210

Ni	a_i	M_i	Mi^{-1}	
2	1	210/2=105	105 x u1=1 mod 2	u1=1
3	2	210/3=70	70 x u2= 1mod 3	u2=1
5	3	210/5=42	42 x u3=1 mod 5	u3=3
7	4	210/7=30	30 x u4=1 m0d 7	U4=4

X=1 x 105 x 1+2 x 70 x 1+3 x 42 x 3+4 x 30 x 4= 1103 ≡53 (mod 210)

9-5 QUADRATIC CONGRUENCE

In cryptography, we also need to discuss quadratic congruence¾that is, equations of the form $a_2x^2 + a_1x + a_0 \equiv 0 \pmod n$. We limit our discussion to quadratic equations in which $a_2 = 1$ and $a_1 = 0$, that is equations of the form

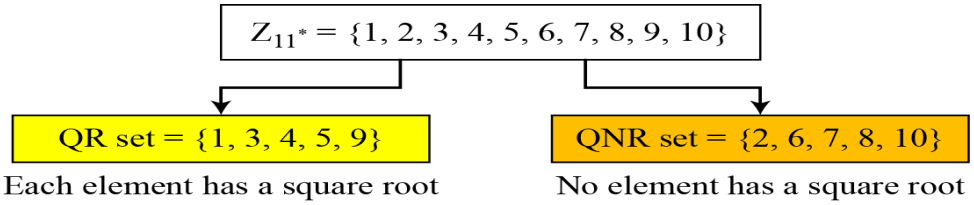
$x^2 \equiv a \pmod n$.

9.5.1 Quadratic Congruence Modulo a Prime

Quadratic Residues and Nonresidue

In the equation $x^2 \equiv a \pmod{p}$, a is called a quadratic residue (QR) if the equation has two solutions; a is called quadratic nonresidue (QNR) if the equation has no solutions.

There are 10 elements in Z_{11}^*. Exactly five of them are quadratic residues and five of them are nonresidues. In other words, Z_{11}^* is divided into two separate sets, QR and QNR, as shown in Figure



Euler's Criterion

a. If $a^{(p-1)/2} \equiv 1 \pmod{p}$, a is a quadratic residue modulo p.

b. If $a^{(p-1)/2} \equiv -1 \pmod{p}$, a is a quadratic nonresidue modulo p.

Find if 14 or 16 is a QR in Z_{23}^*, we calculate:

$14^{(23-1)/2} \bmod 23 \rightarrow 22 \bmod 23 \rightarrow -1 \bmod 23$ nonresidue

$16^{(23-1)/2} \bmod 23 \rightarrow 16^{11} \bmod 23 \rightarrow 1 \bmod 23$ residue

Solving Quadratic Equation Modulo a Prime

Eulers criteria tells us if an integer is QR or QNR in Zp*, it cannot find a solution for $x^2 \equiv a \pmod{p}$. to find solution to this quadratic equation, we notice that a prime ca be either p=4k+1 or p=4k+3, in which k is a positive integer. It is easier to find the solution with second case.

Special Case: p = 4k + 3

$$x \equiv a^{(p+1)/4} \pmod{p} \qquad \text{and} \qquad x \equiv -a^{(p+1)/4} \pmod{p}$$

Solve the following quadratic equations:

a. $x^2 \equiv 3 \pmod{23}$
b. $x^2 \equiv 2 \pmod{11}$
c. $x^2 \equiv 7 \pmod{19}$

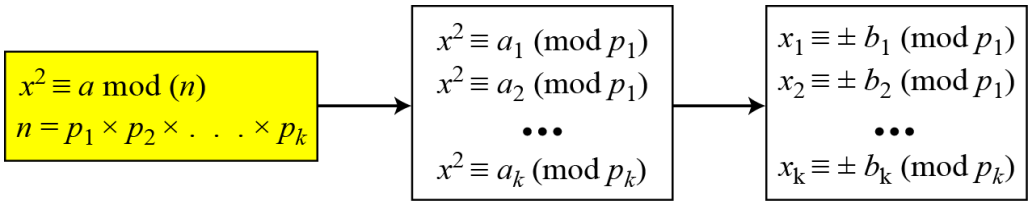
Solution

a. In the first equation 3 is in QR in Z23. The solution is $x \equiv \pm 16 \pmod{23}$ $\sqrt{3} \equiv \pm 16 \pmod{23}$. In other words $\sqrt{3} = \pm 16 \pmod{23}$.
   $3^{(23+1)/4} \bmod 23 = 16$
b. in the second equation, 2 is QNR in Z11. There is no solution for $\sqrt{2}$ in Z11.
c. In the third equation 7 is a QR in Z19. The solution is $x \equiv \pm 11 \pmod{19}$. $\sqrt{7} \equiv \pm 11 \pmod{19}$.
   $7^{(19+1)}/4 = 11$

9.5.2 Quadratic Congruence Modulo a Composite

Quadratic congruence modulo a composite can be done by solving a set of congruence modulo a prime. We decompose x2≡a mod n if we factorize n. we can solve each decomposed equation and find k pairs of answers

Decomposition of congruence modulo a composite



How hard is it to solve a quadratic congruence modulo a composite? The main task is the factorization of the modulus. In other words, the complexity of solving a quadratic congruence modulo a composite is the same as factorizing a composite integer. As we have seen, if *n* is very large, factorization is infeasible.

Example:

Assume that $x^2 \equiv 36 \pmod{77}$. We know that 77 = 7 × 11. We can write

$$x^2 \equiv 36 \pmod 7 \equiv 1 \pmod 7 \qquad \text{and} \qquad x^2 \equiv 36 \pmod{11} \equiv 3 \pmod{11}$$

The answers are x ≡ +1 (mod 7), x ≡ − 1 (mod 7), x ≡ + 5 (mod 11), and x ≡ − 5 (mod 11).

Now we can make four sets of equations out of these:

Set1: x≡+1 mod 7 x≡+5 mod 11

Set2: x≡+1 mod 7 x≡-5 mod 11

Set3: x≡-1 mod 7 x≡+5 mod 11

Set4: x≡-1 mod 7 x≡-5 mod 11

The answers are x = ± 6 and ± 27.

9-6 EXPONENTIATION AND LOGARITHM

Exponentiation and logarithm are inverses of each other. a is the base of the exponentiation or logarithm

$$\textbf{Exponentiation: } y = a^x \quad \rightarrow \quad \textbf{Logarithm: } x = \log_a y$$

9.6.1 Exponentiation

*I*n cryptography a common modular operation is exponentiation

$y=a^x \bmod n$

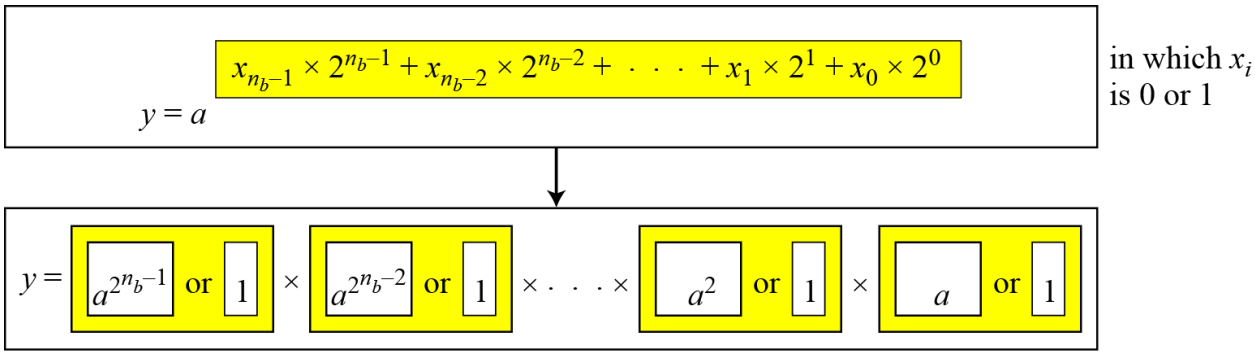
The RSA cryptosystem uses exponentiation for both encryption and decryption with large exponents.

Fast Exponentiation

Fast exponentiation is possible using the square and multipley method. In traditional algorithms multiplication is used to simulate exponentiation, but the fast exponentiation uses both squaring and multiplication.

The main idea behing this method is to treat the exponent as a binary number of n_b bits

The idea behind the square-and-multiply method



Example:

$$y = a^9 = a^{1001_2} = a^8 \times 1 \times 1 \times a$$

Algorithm 9.7 *Pseudocode for square-and-multiply algorithm*

```
Square_and_Multiply (a, x, n)
{
    y ← 1
    for (i ← 0 to n_b − 1)                    // n_b is the number of bits in x
    {
        if (x_i = 1)   y ← a × y mod n        // multiply only if the bit is 1

        a ← a^2 mod n                         // squaring is not needed in the last iteration
    }
    return y
}
```

calculating $y = a^x$ using the Algorithm for $x = 22 = (10110)2$ in binary.

The exponent has five bits.

Demonstration of calculation of a^{22} using square-and-multiply method

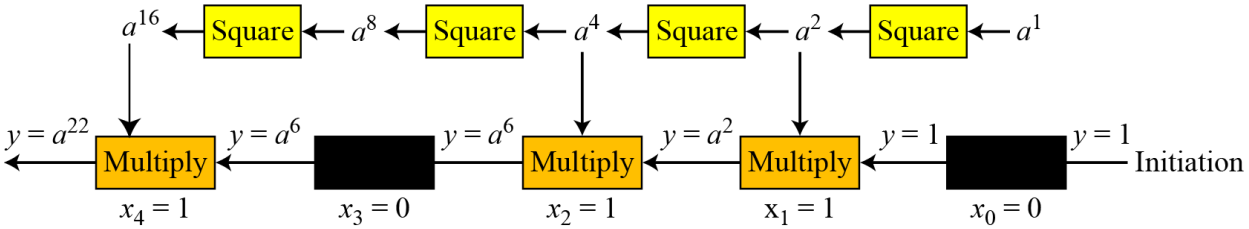


Table 9.3 *Calculation of 17^{22} mod 21*

i	x_i	*Multiplication (Initialization: $y = 1$)*	*Squaring (Initialization: $a = 17$)*
0	0	→	$a = 17^2 \bmod 21 = 16$
1	1	$y = 1 \times 16 \bmod 21 = 16$ →	$a = 16^2 \bmod 21 = 4$
2	1	$y = 16 \times 4 \bmod 21 = 1$ →	$a = 4^2 \bmod 21 = 16$
3	0	→	$a = 16^2 \bmod 21 = 4$
4	1	$y = 1 \times 4 \bmod 21 = 4$ →	

9.6.2 Logarithm

In cryptography, we also need to discuss modular logarithm. If we use exponentiation to encrypt or decrypt, the attacker can use logarithm to attack.

Exhaustive Search

The first solution is solve $x=\log_a y \pmod n$

Algorithm 9.8 *Exhaustive search for modular logarithm*

```
Modular_Logarithm (a, y, n)
{
    for (x = 1 to n −1)                          // k is the number of bits in x
    {
        if (y ≡ a^x mod n) return x
    }
    return failure
}
```

Primitive roots of 19

Table 8.3 Powers of Integers, Modulo 19

a	a^2	a^3	a^4	a^5	a^6	a^7	a^8	a^9	a^{10}	a^{11}	a^{12}	a^{13}	a^{14}	a^{15}	a^{16}	a^{17}	a^{18}
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	4	8	16	13	7	14	9	18	17	15	11	3	6	12	5	10	1
3	9	8	5	15	7	2	6	18	16	10	11	14	4	12	17	13	1
4	16	7	9	17	11	6	5	1	4	16	7	9	17	11	6	5	1
5	6	11	17	9	7	16	4	1	5	6	11	17	9	7	16	4	1
6	17	7	4	5	11	9	16	1	6	17	7	4	5	11	9	16	1
7	11	1	7	11	1	7	11	1	7	11	1	7	11	1	7	11	1
8	7	18	11	12	1	8	7	18	11	12	1	8	7	18	11	12	1
9	5	7	6	16	11	4	17	1	9	5	7	6	16	11	4	17	1
10	5	12	6	3	11	15	17	18	9	14	7	13	16	8	4	2	1
11	7	1	11	7	1	11	7	1	11	7	1	11	7	1	11	7	1
12	11	18	7	8	1	12	11	18	7	8	1	12	11	18	7	8	1
13	17	12	4	14	11	10	16	18	6	2	7	15	5	8	9	3	1
14	6	8	17	10	7	3	4	18	5	13	11	2	9	12	16	15	1
15	16	12	9	2	11	13	5	18	4	3	7	10	17	8	6	14	1
16	9	11	5	4	7	17	6	1	16	9	11	5	4	7	17	6	1
17	4	11	16	6	7	5	9	1	17	4	11	16	6	7	5	9	1
18	1	18	1	18	1	18	1	18	1	18	1	18	1	18	1	18	1

The primitive roots of 19 are 2,3,10,13,14,15

Discrete Logarithm

The second approach is to use the concept of discrete logarithms. To understand this we should know the groups

Order of the Group.

What is the order of group G = $<Z_{21}^*, \times>$? $|G| = \phi(21) = \phi(3) \times \phi(7) = 2 \times 6 = 12$. There are 12 elements in this group: 1, 2, 4, 5, 8, 10, 11, 13, 16, 17, 19, and 20. All are relatively prime with 21.

Order of an Element

Solution

Find the order of all elements in G = $<Z_{10}^*, \times>$.

This group has only f(10) = 4 elements: 1, 3, 7, 9. We can find the order of each element by trial and error.

a. $1^1 \equiv 1 \bmod (10) \rightarrow \text{ord}(1) = 1$.

b. $3^4 \equiv 1 \bmod (10) \rightarrow \text{ord}(3) = 4$.

c. $7^4 \equiv 1 \bmod (10) \rightarrow \text{ord}(7) = 4$.
d. $9^2 \equiv 1 \bmod (10) \rightarrow \text{ord}(9) = 2$.

Euler's Theorem
If a is a member of G=<Z_n*, x> then $a^{\emptyset(n)}=1 \bmod n$
This theorem is very helpful because it shows the relationship $a^i=1 \pmod n$ holds when i= Ø(n)

Example: **the result of $a^i \equiv x \pmod 8$ for the group G = <Z_7*, ×>. In this group, Ø(8) = 4.**

	$i=1$	$i=2$	$i=3$	$i=4$	$i=5$	$i=6$	$i=7$
$a=1$	**x: 1**	x: 1	x: 1	x: 1	x: 1	x: 1	x: 1
$a=3$	x: 3	**x: 1**	x: 3	x: 1	x: 3	x: 1	x: 3
$a=5$	x: 5	**x: 1**	x: 5	x: 1	x: 5	x: 1	x: 5
$a=7$	x: 7	**x: 1**	x: 7	x: 1	x: 7	x: 1	x: 7

Primitive Roots In the group G = <Z_n*, ×>, when the order of an element is the same as Ø(n), that element is called the primitive root of the group.
shows the result of $a^i \equiv x \pmod 7$ for the group G = <Z_7*, ×>. In this group, Ø(7) = 6.

	$i=1$	$i=2$	$i=3$	$i=4$	$i=5$	$i=6$
$a=1$	x: 1	x: 1	x: 1	x: 1	x: 1	x: 1
$a=2$	x: 2	x: 4	x: 1	x: 2	x: 4	x: 1
Primitive root → $a=3$	x: 3	x: 2	x: 6	x: 4	x: 5	x: 1
$a=4$	x: 4	x: 2	x: 1	x: 4	x: 2	x: 1
Primitive root → $a=5$	x: 5	x: 4	x: 6	x: 2	x: 3	x: 1
$a=6$	x: 6	x: 1	x: 6	x: 1	x: 6	x: 1

The group G = <Z_n*, ×> has primitive roots only if *n* is 2, 4, p^t, or $2p^t$.

For which value of *n*, does the group G = <Z_n*, ×> have primitive roots: 17, 20, 38, and 50?
Solution

a. G = <Z_{17}*, ×> has primitive roots, 17 is a prime.
b. G = <Z_{20}*, ×> has no primitive roots.
c. G = <Z_{38}*, ×> has primitive roots, $38 = 2 \times 19$ prime.
d. G = <Z_{50}*, ×> has primitive roots, $50 = 2 \times 5^2$ and 5 is a prime.

If the group G = <Z_n*, ×> has any primitive root, the number of primitive roots is $\emptyset(\emptyset(n))$.

Cyclic Group If g is a primitive root in the group, we can generate the set Z_n* as $Z_n* = \{g^1, g^2, g^3, \ldots, g^{f(n)}\}$

The group G = <Z_{10}*, ×> has two primitive roots because Ø(10) = 4 and Ø(Ø(10)) = 2. It can be found that the primitive roots are 3 and 7. The following shows how we can create the whole set Z_{10}* using each primitive root.

$g = 3 \rightarrow$	$g^1 \bmod 10 = 3$	$g^2 \bmod 10 = 9$	$g^3 \bmod 10 = 7$	$g^4 \bmod 10 = 1$
$g = 7 \rightarrow$	$g^1 \bmod 10 = 7$	$g^2 \bmod 10 = 9$	$g^3 \bmod 10 = 3$	$g^4 \bmod 10 = 1$

The group $G = <Z_n^*, \times>$ is a cyclic group if it has primitive roots.
The group $G = <Z_p^*, \times>$ is always cyclic.

The idea of Discrete Logarithm
Properties of $G = <Z_p^*, \times>$:
1. Its elements include all integers from 1 to p − 1.
2. It always has primitive roots.
3. It is cyclic. The elements can be created using g^x where x is an integer from 1 to Ø(n) = p − 1.
4. The primitive roots can be thought as the base of logarithm.

Solution to Modular Logarithm Using Discrete Logs

Tabulation of Discrete Logarithms

Table 9.6 *Discrete logarithm for* $\mathbf{G} = <\mathbf{Z}_7^*, \times>$

y	1	2	3	4	5	6
$x = \mathrm{L}_3\, y$	6	2	1	4	5	3
$x = \mathrm{L}_5\, y$	6	4	5	2	1	3

Find x in each of the following cases:
a. $4 \equiv 3^x \pmod 7$.
b. $6 \equiv 5^x \pmod 7$.
Solution
We can easily use the tabulation of the discrete logarithm in Table 9.6.
a. $4 \equiv 3^x \bmod 7 \rightarrow x = \mathrm{L}_3 4 \bmod 7 = 4 \bmod 7$
b. $6 \equiv 5^x \bmod 7 \rightarrow x = \mathrm{L}_5 6 \bmod 7 = 3 \bmod 7$

Using Properties of Discrete Logarithms

Table 9.7 *Comparison of traditional and discrete logarithms*

Traditional Logarithm	*Discrete Logarithms*
$\log_a 1 = 0$	$\mathrm{L}_g 1 \equiv 0 \pmod{\phi(n)}$
$\log_a (x \times y) = \log_a x + \log_a y$	$\mathrm{L}_g(x \times y) \equiv (\mathrm{L}_g x + \mathrm{L}_g y) \pmod{\phi(n)}$
$\log_a x^k = k \times \log_a x$	$\mathrm{L}_g x^k \equiv k \times \mathrm{L}_g x \pmod{\phi(n)}$