Cryptography 1

Introduction To Cryptography

Chittaranjan Pradhan
School of Computer Engineering,
KIIT University

Security Goals

Security Goals

- Confidentiality: refers to secrecy of information
- Integrity: changes need to be done only by authorized entities and through authorized mechanisms
- Availability: information needs to be available to authorized entities

Security Attacks

Threat to Confidentiality

- Snooping: refers to unauthorized access to or interception of data
- Traffic Analysis: refers to obtaining some other type of information by monitoring online traffic

Security Attacks...

Threat to Integrity

- Modification: attacker intercepts the message and modifies it
- Masquerading/Spoofing: attacker impersonates somebody else
- Replaying: attacker obtains a copy of a message sent by a user and later tries to replay it
- Repudiation: the sender of the message might later deny that he/she has sent the message; the receiver of the message might later deny that he/she has received the message

Security Attacks...

Threat to Availability

- Denial of Service: may slow down or totally interrupt the service of a system

Passive vs. Active Attacks

- Passive Attacks: goal is to obtain information. It is very difficult to detect. Ex: Snooping, Traffic analysis
- Active Attacks: may change the data or harm the system. Ex: Modification, Masquerading, Replaying, Repudiation, Denial of Service

Security Services

Security Services

- Data Confidentiality: protects data from disclosure attack
- Data Integrity: protects data from modification, insertion, deletion and replaying by an adversary
- Authentication: authentication of the party at the other end of the line
- Nonrepudiation: protects against repudiation by either the sender or the receiver of the data
- Access Control: protects against unauthorized access to data

Security Mechanisms

Security Mechanisms

- Encipherment: hiding or covering data can provide confidentiality
- Data Integrity: appends a short checkvalue to the data that has been created by a specific process from the data itself. Receiver creates a new checkvalue from the received data and compares the checkvalues
- Digital Signature: sender can electronically sign the data and receiver can electronically verify the signature
- Authentication Exchange: two entities exchange some messages to provide their identity to each other
- Notarization: selecting a third trusted party to control the communication between two entities
- Access Control: proves that a user has access right to the data or resources owned by a system

Cryptography 2

Substitution & Transposition Techniques

Chittaranjan Pradhan
School of Computer Engineering,
KIIT University

Concepts of Cryptography

Cryptography

Systematic and well-structured process

Readable Message → Cryptography → Un-readable Message

Cryptanalysis

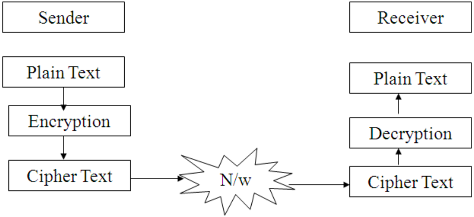
Trial and error process

Un-readable Message → Cryptanalysis → Readable Message

Plain Text and Cipher Text

Plain Text and Cipher Text

- **Plain Text**: Language that can be easily understood
- **Cipher Text**: Language that cannot be understood



Techniques for Plain Text to Cipher Text Conversion

PT–>CT

- Substitution technique/ Cipher
  - Each character in the PT is substituted for another character in the CT
- Transposition technique/ Cipher
  - Encrypt PT by moving small pieces of the message around
- Product Cipher
  - When the 2 approaches are used together

Substitution Cipher

Substitution Cipher

Here, each character in the plain text substituted for another character in the cipher text. Substitution ciphers can be categorized as:

- **Monoalphabetic Ciphers**: relationship between a symbol in the PT to a symbol in the CT is always one-to-one
- **Polyalphabetic Ciphers**: each occurence of a symbol may have a different substitute. The relationship between a symbol in the PT to a symbol in the CT is one-to-many

Caesar Cipher

Caesar Cipher

- Proposed by Julius Caesar
- Mechanism to make a message non-understandable
- Replaces each alphabet with the one three places down
- CT_i= **E(**PT_i**)=** PT_i **+ 3**

- PT: KIIT
- CT: NLLW

Shift Cipher / Modified Caesar Cipher

Shift Cipher / Modified Caesar Cipher

- The CT alphabets corresponding to the original PT alphabets may not necessarily be 3 places down the line, it can be any places down the line
- CT_i**= E(**PT_i**)=** PT_i **+ n; 1** $\leq$ **n** $\leq$ **25**
- Once the replacement scheme is decided, it could be constant and will be used for all other alphabets in that message
- For each alphabet, we have 25 possibilities of replacement

- PT: KIIT
- CT: PNNY

Brute- Force Attack

Brute- Force Attack

Process where the attacker attempts to use all possible permutations and combinations to get PT from CT

- CT: PNNY
  - Key=1, PT: OMMX
  - Key=2, PT: NLLW
  - Key=3, PT: MKKV
  - Key=4, PT: LJJU
  - Key=5, PT: KIIT

Affine Cipher

Affine Cipher

- It uses two keys (one for multiplicative cipher & other for additive cipher)
- **C = (P x k_1 + k_2) mod 26**
- **P = (C - k_2) x (k_1^{-1})**, where k_1^{-1} is the multiplicative inverse of k_1 and -k_2 is the additive inverse of k_2
- Encrypt message "hello"with the key pair (7,2)

Polyalphabetic Substitution Cipher/ Vigenere Cipher

Polyalphabetic Substitution Cipher/ Vigenere Cipher

- Proposed by Leon Battish in 1568
- The cipher uses multiple one-character keys
- Each key encrypts one PT character
- After all the keys are used, they are recycled
- Period of Cipher
- It uses a key as well as a Vigenere table for the encryption of PT

Vigenere Table

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

Polyalphabetic Substitution Cipher/ Vigenere Cipher

Polyalphabetic Substitution Cipher/ Vigenere Cipher

- If the key length is lesser than the PT length; then make the length same by repeating the key
- For key letter **p** and PT letter **q**, the corresponding CT letter is at the intersection of row titled **p** and column titled **q**. Therefore, the CT could be F

- Key: orissa
- PT: bhubaneswar
- Key: orissaoriss
- CT:PYCTSNSJESJ

Playfair Cipher

Playfair Cipher

- Proposed by Charles Wheatstone in 1854
- Named by the name of Wheatstone's friend Lord Playfair
- Used by British army in World War-I, and by Australian in World War-II
- Quite fast
- Used to protect important, but not very critical information

Creation and Population of Matrix

Creation and Population of Matrix

- Uses matrix of 5 X 5
- Used to store the keyword
  - Enter the keyword in the matrix row-wise
  - Drop duplicate letters
  - Fill the remaining spaces with the rest of the English alphabets
  - Both I and J has same precedence

Keyword: NETWORK SECURITY

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

Encryption Process

Encryption Process

- The PT message is broken down into groups of 2 alphabets
- If both alphabets are same or only one is left, add X after the alphabet
- If one character is ’, then that character will be replaced by the respective character in the previous pair
- If both the alphabets in the pair is in the same row of the matrix, replace them with alphabets to their immediate right respectively. If the original pair is on the right side of the row , then wrapping around to the left side of the same row happens
- If both the alphabets in the pair appears in the same column of the matrix, replace them with alphabets immediately below them respectively . If the original pair is on the bottom side of the column, then wrapping round to the top side of the same column happens

Encryption Process

Encryption Process

- If the alphabets are not in the same row or column, replace them with the alphabets in the same row respectively, but at the other pair of corners of the rectangle defined by the original pair. The 1st encrypted alphabet of the pair is the one that is present on the same row as the 1st PT alphabet

Keyword: NETWORK SECURITY
PT: HAPPY NEW YEAR
HA PP YN EW YE AR $\rightarrow$ HA PX YN EW YE AR

HA	PX	YN	EW	YE	AR
VH	QZ	IE	TO	GK	IS

CT:VHQZIETOGKIS

To decrypt the message, simply reverse the entire process. Break the CT into pairs of letters

Hill Cipher

Hill Cipher

- It is a type of polygraphic substitution cipher
- Invented by Lester Hill in 1929
- Works on inverse matrix theory
- It uses a key for the generation of key matrix. If the key is not given, then it chooses a random key
- It is a block cipher

Hill Cipher...

Encryption Process

- Treat every letter in PT as a number in base 26
- Extra bogus character 'z' may be added to the last block for the construction of PT matrix of size l x m, where l is the number of blocks
- The key is a square matrix of size m x m, where m is the size of the block
- The key matrix should be chosen in such a way that it should have multiplicative inverse in Z_{26}
- Multiply PT matrix with the key matrix to generate CT **CT=(PT x Key) mod 26**
- Compute mod26 value of the matrix
- Translate the numbers into alphabets, which is the CT

Hill Cipher...

PT: CODE IS READY

$$\begin{pmatrix} 02 & 14 & 03 & 04 \\ 08 & 18 & 17 & 04 \\ 00 & 03 & 24 & 25 \end{pmatrix} \times \begin{pmatrix} 09 & 07 & 11 & 13 \\ 04 & 07 & 05 & 06 \\ 02 & 21 & 14 & 09 \\ 03 & 23 & 21 & 08 \end{pmatrix} = \begin{pmatrix} 92 & 267 & 218 & 169 \\ 190 & 631 & 500 & 397 \\ 135 & 1100 & 876 & 434 \end{pmatrix}$$

$$\begin{pmatrix} 92 & 267 & 218 & 169 \\ 190 & 631 & 500 & 397 \\ 135 & 1100 & 876 & 434 \end{pmatrix} \text{ MOD 26 } = \begin{pmatrix} 14 & 07 & 10 & 13 \\ 08 & 07 & 06 & 07 \\ 05 & 08 & 18 & 18 \end{pmatrix}$$

CT: OHKNIHGHFISS

Decryption Process

- **PT= (CT X key^{-1}) mod 26**

Transposition Cipher

Transposition Cipher

It encrypts PT by moving small pieces of the message around. The common types of transposition techniques are discussed

Rail Fence Technique

Rail Fence Technique

- Write down the PT message as a sequence of diagonals
- Read the PT as a sequence of rows
- This technique is quite simple for a cryptanalyst to break into

PT: come home tomorrow

CT: cmhmtmrooeoeoorw

This technique can be applicable for more number of lines in the similar manner

Single Columnar Transposition Technique

Single Columnar Transposition Technique

- One keyword is used whose letters are numbered according to their presence in the alphabet
- If the same letter has occurred more than one time, it should be numbered 1, 2 ... from left to right
- PT is written in rows under the numbered keyword, one letter under each letter of the keyword
- CT can be generated by reading the PT letters column wise in the order stated by the enumeration of the keyword

Keyword: heaven

PT: WE ARE THE BEST

CT: ABEEESWHTTRE

If keyword is not given, then the number of columns will be given; which are numbered in increasing order

Double- Columnar Transposition

Double- Columnar Transposition

- Similar to single- columnar transposition
- The process is repeated twice
- Two keywords are used or one keyword may be repeated

Keyword1: heaven

Keyword2: another

PT: WE ARE THE BEST

CT: AHSEEBTETWER

Vernam Cipher / One- Time Pad

Vernam Cipher / One- Time Pad

- Uses a random set of non-repeating characters as the input CT
- Input CT will be used only once
- Length of input CT = length of PT
- Treat each PT alphabet as a number as in dictionary
- Do the same for each alphabet of the input CT
- Add each number corresponding to the PT alphabet to the corresponding input CT alphabet number
- If sum > 25, then subtract 26 from it
- Translate each number of the sum back to the alphabet

Vernam Cipher / One- Time Pad

Vernam Cipher / One- Time Pad

- Highly secured
- Suitable for small PT message
- Impractical for large message

PT: ANNUAL FUNCTION

Onetime pad: SAFGHI WEYUOPLI

CT: SNSAHTBYLWHXZV

Cryptographic Mechanisms

Cryptographic Mechanisms

- **Key**
  - Similar to the one-time pad used in vernam cipher
  - Algorithm is known to everybody. The key is the thing which makes the cryptographic system secure
- **Symmetric-key Cryptography**
  - Symmetric algorithms or secret key algorithms
  - Same key is used both for encryption and decryption processes
  - **CT=E_k(PT), PT=D_k(CT)**
  - For n persons, the number of keys required is n*(n-1)/2
- **Asymmetric-key Cryptography**
  - Asymmetric algorithms or public key algorithms
  - Different keys are used for encryption and decryption processes
  - **CT=E_{k1}(PT), PT=D_{k2}(CT); k1 $\neq$ k2**
  - For n persons, the number of keys required is 2n (n private keys and n public keys)

Key Range & Key Size

Key Range & Key Size

- **Key Range**
  - Key range specifies the number of possible keys
- **Key Size**
  - key size is represented in bits
  - If the key size is 2, the key range is 4. The possible key values are 00,01,10,11
  - **If the key size is *n*, the key range is** 2^n
  - Larger the key size means greater security

Cryptanalysis and Attack Models

Cryptanalysis and Attack Models

Cryptanalysis is the science and art of breaking the secret codes

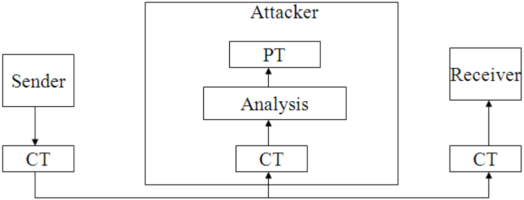
An attempted cryptanalysis is called an attack

Different attack models are discussed here

Cipher Text Only Attack

Cipher Text Only Attack

- Attacker has access to only some CT
- Attacker tries to find the corresponding key and PT



Cipher Text Only Attack

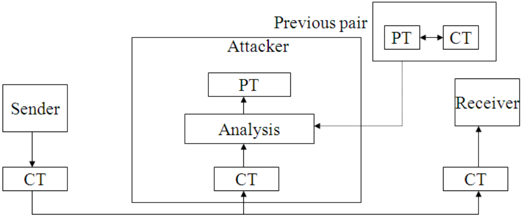
Common Cipher Text Only Attacks

- **Brute- Force / Exhaustive- Key- Search Attack**: Attacker tries to use all possible keys
- **Statistical Attack**: Attacker tries to use the inherent characteristics of the PT language
- **Pattern Attack**: Attacker tries to exploit the hidden characteristics of the language

Known Plain Text Attack

Known Plain Text Attack

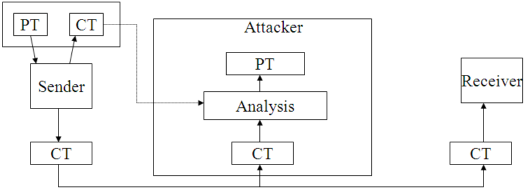
- Attacker knows some pairs of PT & corresponding CT
- Using the above information, attacker tries to find other pairs
- Ex: Company Banner, file headers



Chosen-Plain Text Attack

Chosen-Plain Text Attack

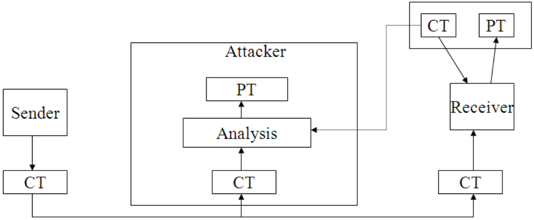
- The attacker selects a PT block and tries to look for the encryption of the same in the CT
- Here, the attacker chooses some PT and pays the company to encrypt it
- Attacker has access to the sender's computer



Chosen Cipher Text Attack

Chosen Cipher Text Attack

- Similar to Chosen Plain Text Attack
- Here, the attacker knows the CT to be decrypted, the encryption algorithm that was used to produce this CT and the corresponding PT block
- Attacker has access to the receiver's computer



Cryptography 3
GCD, Modularity Arithmetic

Chittaranjan Pradhan
School of Computer Engineering,
KIIT University

GCD (Greatest Common Divisor)

GCD (Greatest Common Divisor)

GCD of two positive integers is the largest integer that can divide both integers

Euclidean Algorithm

Euclidean Algorithm

- *Fact 1: gcd (a, 0) = a*
- *Fact 2: gcd (a, b) = gcd (b, r), where r is the remainder of dividing a by b*
- When gcd (a, b) = 1, we say that a and b are relatively prime or coprime

```
r1←a; r2← b;        //Initialization
while(r2>0)
        {
        q←r1/r2;
        r←r1 – q x r2;
        r1←r2;
        r2←r;
        }
gcd (a, b)←r1
```

Euclidean Algorithm...

- Gcd of 25 & 60

q	r1	r2	r
0	25	60	25
2	60	25	10
2	25	10	5
2	10	5	0
	5	0	

Gcd of 60 & 25

q	r1	r2	r
2	60	25	10
2	25	10	5
2	10	5	0
	5	0	

Extended Euclidean Algorithm

Extended Euclidean Algorithm

- Given two integers a and b, we often need to find other two integers s and t such that
  s * a + t * b = gcd (a,b)

```
r1←a; r2← b;     s1←1; s2←0;     t1←0, t2←1;   //Initialization
while(r2>0)
        {
        q←r1/r2;
        r←r1 – q x r2;
        r1←r2; r2←r;
        s←s1 – q x s2;
        s1←s2; s2←s;
        t←t1 – q x t2;
        t1←t2; t2←t;
        }
gcd (a, b)←r1, s←s1, t←t1
```

Extended Euclidean Algorithm...

q	r1	r2	r	s1	s2	s	t1	t2	t
2	60	25	10	1	0	1	0	1	-2
2	25	10	5	0	1	-2	1	-2	5
2	10	5	0	1	-2	5	-2	5	-12
	5	0		-2	5		5	-12	

Extended Euclidean Algorithm...

Linear Diophantine Equation

A linear Diophantine equation of two variables is ax + by = c

- d= gcd(a,b)
  if d|c: infinite solution
  else no solution
- **Particular Solution** Since d divides a, b and c, reduce the equation to a1x+b1y=c1. Then solve a1s+b1t =0
  - x_0 = (c/d)s and y_0 = (c/d)t
- **General Solutions**
  - x = x_0+k(b/d) and y = y_0-k(a/d), where k is an integer
- Find the particular and general solutions to the equation 21x + 14y = 35
- If we want Rs.100 note to be changed with Rs.20 and Rs.5 notes, then what are the possible cases

Modular Arithmetic

Modular Arithmetic

- 27 mod 10
- -7 mod 10
- a $\equiv$ b(mod n)
- Ex: 2 $\equiv$ 12 (mod 10)

Z_n (Set of Residues)

Z_n (Set of Residues)

- The result of a mod n is always a non negative integer less than n i.e. 0 to n-1
- Z_{10}, Z_7
- Property1: (a+b) mod n =[(a mod n) + (b mod n)] mod n
- Property2: (a-b) mod n =[(a mod n) - (b mod n)] mod n
- Property3: (axb) mod n =[(a mod n) x (b mod n)] mod n
- 10^n mod x = $(10 mod x)^n$

Inverse

Inverse

Additive Inverse

- Let n be a positive integer
  If a, b $\in Z_n$, then

$$\begin{aligned} (a+b) mod n &= a+b, if (a+b) < n \\ &= a+b-n, if (a+b) >= n \end{aligned} \tag{1}$$

- In Z_n, two numbers a & b are additive inverses of each other if

  a +b $\equiv$ 0 (mod n)

- *In modular arithmetic, each number has an additive inverse and the inverse is unique; and each number has one and only one additive inverse*
- Find additive inverse of 6 in Z_{10}
- Find all additive inverse pairs in Z_{10}

Inverse...

Inverse...

Multiplicative Inverse

- The integer a in Z_n has a multiplicative inverse iff **gcd(n,a) $\equiv$ 1 (mod n)**
- In Z_n, two numbers a and b are the multiplicative inverse of each other if
  a x b $\equiv$ 1 (mod n)
- Find all multiplicative inverse pairs in Z_{10}
- The Extended Euclidean algorithm can find the multiplicative inverse of b in Z_n where n & b are given and the inverse exist, i.e. gcd (n, b)=1
  s x n + b x t= gcd (n,b)
- If the multiplicative inverse of b exists, gcd (n,b) =1
  s x n + b x t= 1
- *The multiplicative inverse of b is the value of t after being mapped to Z_n*

Inverse...

Multiplicative Inverse...

```
r1←n; r2← b;    t1←0, t2←1;   //Initialization
while(r2>0)
        {
        q←r1/r2;
        r←r1 – q x r2;
        r1←r2; r2←r;
        t←t1 – q x t2;
        t1←t2; t2←t;
        }
If (r1=1), then b^-1←t1
```

Find the multiplicative inverse of 11 in Z_{26}

Find the multiplicative inverse of 12 in Z_{26}

Additive & Multiplicative Tables

Additive & Multiplicative 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 $\mathbf{Z}_{10}$

Cryptography 4

Group, Ring, Field

Chittaranjan Pradhan
School of Computer Engineering,
KIIT University

Group

Group

A group **<G, . >** is a set of elements with a binary operation **.** that associates to each pair (a, b) of elements in G an element (a.b) in G such that the following properties are satisfied:

- **Closure**: If a and b belong to G, then *a.b* is also in G
- **Associative**: If a, b and c are elements of G, then *a.(b.c) = (a.b).c*
- **Identity Element**: For all a in G, there exists an element e, called as identity element such that *e.a = a.e = a*
- **Inverse Element**: For each a in G, there exists an element a', called as inverse of a such that *a.a' = a'.a = e*
- Ex: <Z, + >
- <{ 0,1,2,3,4 }, + >

Group...

Group...

- **Finite Group**: If the group has a finite number of elements, it is called as finite group
- Ex: <{ 0,1 }, + >, <{ 0,1 }, * >
- <{ -1,1 }, * >
- **Order of a Group**: It is the number of elements in the group
- **Abelian Group**: It is the group with additional condition **Commutative**: For all a and b in G, *a.b = b.a*
- Ex: $<Z_n, + >$
- **Cyclic Group**: On a group, an element a is called generator if a $\in$ G, and $\forall$ a $\in$ G can be represented using power of a, a^k. *A group is said to be cyclic if it contains at least one generator element*
- Ex: $<\{0, 1, 2, 3\}, +_4 >$
- $<Z_6, + >$

Group...

- **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*
  - If a and b are members of both groups, then a.b is also a member of both groups
  - The group share the same identity element
  - If a is a member of both groups, then the inverse of a is also a member of both groups
  - The group made of the identity element of G, H = $<$ {e}, .$>$, is a subgroup of G
  - Each group is a subgroup of itself
- Ex: <{0, 2, 4}, $+_6$ >subgroup of <{0, 1, 2, 3, 4, 5}, $+_6$ >
- <Z, + >subgroup of <Q, + >
- **Semigroup**: It is an Algebraic structure satisfying associative property

Ring

Ring

<R, +, * >is said to be ring iff

- <R,+>is abelian group
- <R,*>is semigroup
- * operator is distributed over + i.e. a*(b+c)=a*b+a*c or (b+c)*a=b*a+c*a

A **commutative ring** is a ring in which the commutative property is also satisfied for the second operation
Ex: <Z, +, * >
$< Z_6, +, * >$

A **Ring with identity** is a ring if unity of its multiplicative identity exists,
e*a = a = a*e, $\forall$ a $\in$ R

Field

Field

<F, +, * >is said to be field iff

- <F,+ >is abelian group
- The nonzero elements of F form an abelian group w.r.t. *
- The distributive law holds

- Ex: <R,+,* >, <Z,+,* >
- <{0, 1, 2},$+_3$,$*_3$ >

Galois Field

Galois Field

GF(p^n) is a finite field with p^n elements, where p is prime and n is positive integer

- When n=1, we have GF(p) field
- Z_p, {0, 1, 2, ... ,p-1} with + and *
- Ex: GF(2), GF(5)

GF(2^n) Fields

GF(2^n) Fields

To use fields in computers, there are two options:

- GF(p) is used with set Z_p, where p is the largest prime number less than 2^n. This scheme is inefficient because the integers from p to 2^n-1 are not used
- GF(2^n) can be used with 2^n elements
- Ex: GF(2^2)

Polynomials

A polynomial of degree n-1 is an expression in the form:
$f(x) = a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + ... + a_1x^1 + a_0x^0$
where, power of x defines the position of the bit, and coefficients of the terms define the value of bits

Ex: Represent 10011001 using polynomials

Polynomials representing n-bit words use two fields: GF(2) for coefficients and GF(2^n) for operations on polynomials

GF(2^n) Fields...

Polynomials...

Addition

- Addition and subtraction operations on polynomials are same operation
- Ex: Addition of $x^5 + x^2 + x$ and $x^3 + x^2 + 1$ in GF(2^8)

Multiplication

- It is the sum of the multiplication of each term of the first polynomial with each term of the second polynomial such that:
  The coefficient multiplication is done in GF(2), and multiplication is done using modulus polynomial
- Ex: Multiplication of $x^5 + x^2 + x$ and $x^7 + x^4 + x^3 + x^2 + x$ in GF(2^8) with irreducible polynomial $x^8 + x^4 + x^3 + x + 1$

Cryptography 5

Modern Symmetric-key Ciphers & Algorithm Modes

Chittaranjan Pradhan
School of Computer Engineering,
KIIT University

Stream Cipher vs. Block Cipher

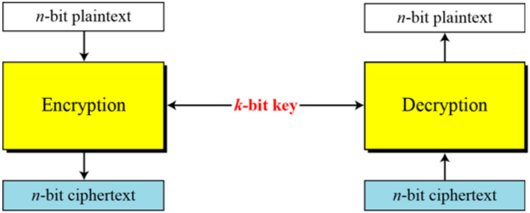
Stream Cipher vs. Block Cipher

- **Stream Cipher**
  - Encryption and Decryption are done one symbol (such as a bit or a byte) at a time
- **Block Cipher**
  - A group of PT symbols of size m (m>1) are encrypted together creating a group of CT of the same size
  - A single key is used to encrypt the whole block even if the key is made of multiple values
- *Stream cipher is very time consuming*

Modern Block Ciphers

Modern Block Ciphers

- A symmetric-key modern block cipher encrypts an n-bit block of plaintext or decrypts an n-bit block of cipher text. The encryption or decryption algorithm uses a k-bit key
- A modern block cipher can be designed to act as a substitution cipher or a transposition cipher



Component of a Modern Block Cipher

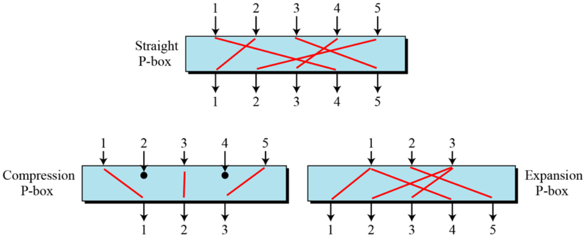
Component of a Modern Block Cipher

- Modern block ciphers normally are keyed substitution ciphers in which the key allows only partial mappings from the possible inputs to the possible outputs

- **P-Box**
    - A P-box (permutation box) parallels the traditional transposition cipher for characters
    - It transposes bits
    - P-boxes are normally keyless
    - Sometimes called as D-box (Diffusion box)

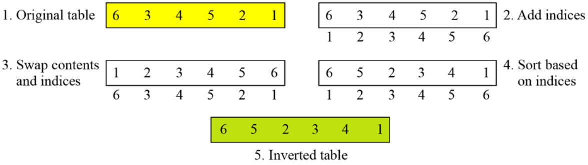
- *Straight P-box*: n inputs and n outputs. There are n! possible mappings
- *Compressed P-box*: n inputs and m outputs, m < n
- *Expansion P-box*: n inputs and m outputs, m > n

Component of a Modern Block Cipher...



Invertibility Feature of P-Box

A straight P-box is invertible. Compression and expansion P-boxes have no inverses



Component of a Modern Block Cipher...

Component of a Modern Block Cipher...

- **S-Box**
  - An S-box (substitution box) can be thought of as a miniature substitution cipher
  - S-box can have a different number of inputs and outputs
  - Modern block ciphers normally use keyless S-Boxes

  - *Linear S-box*: The relationship between inputs and outputs can be represented as a set of equations
  - *Nonlinear S-box*: For every outputs, there may not be relationships like linear type
  - A S-box may be invertible. In an invertible S-box, the number of input bits should be same as the number of output bits
- **XOR**
  - XOR is reversible:- when used twice, it produces the original value

Component of a Modern Block Cipher...

Component of a Modern Block Cipher...

- **Circular Shift**
  - Shifting can be to the left or to the right
  - Circular left shift operation shifts each bit in an n-bit word k positions to the left
  - Circular right shift operation shifts each bit in an n-bit word k positions to the right
  - A circular left-shift operation is the inverse of the circular right-shift operation
- **Swap**
  - Special case of circular shift operation where k=n/2
  - *Swap operation is valid only if n is an even number*
- **Split & Combine**
  - Split operation splits an n-bit word in the middle, creating two equal-length words
  - Combine operation concatenates two equal-length words to create an n-bit word

Product Cipher

Product Cipher

- Shannon introduced the concept of a product cipher. A product cipher is a complex cipher combining substitution, permutation, and other components discussed in previous sections

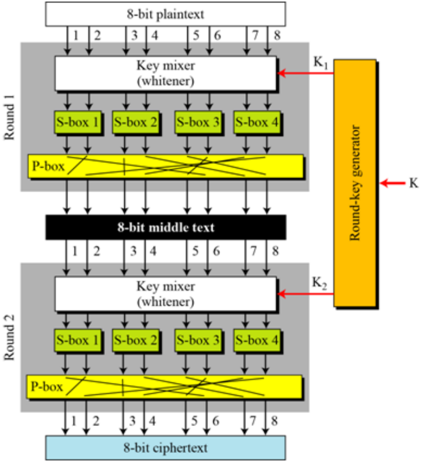
- **Diffusion**
  - The idea of diffusion is to hide the relationship between the ciphertext and the plaintext
  - If a single symbol in the PT is changed, several or all symbols in the CT will also be changed

- **Confusion**
  - The idea of confusion is to hide the relationship between the ciphertext and the key
  - If a single bit in the key is changed, most or all bits in CT will also be changed

Non-Feistel ciphers

Non-Feistel ciphers

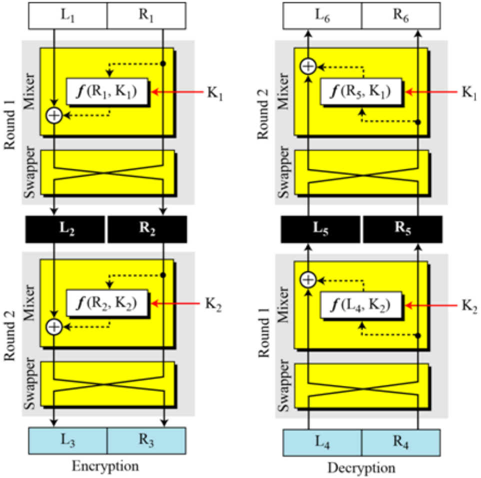
It uses only invertible components, like S-Box, P-Box, XOR operation. A component in the PT has the corresponding component in the cipher



Feistel ciphers

Feistel ciphers

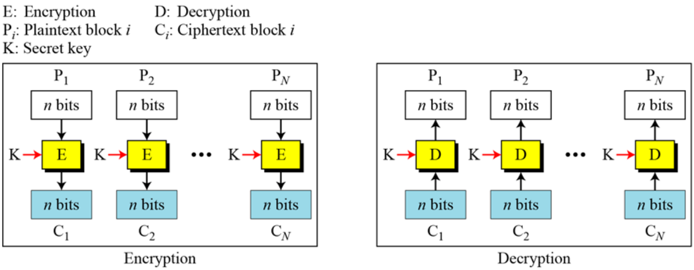
Uses Split & Combine, Swap, XOR, Circular Shift operation



Electronic Codebook (ECB) Mode

Electronic Codebook (ECB) Mode

- The simplest mode of operation is called the ECB mode
- Each block is encrypted independently
- Parallel processing can be used



Encryption: $C_i = E_K(P_i)$ Decryption: $P_i = D_K(C_i)$

Cipher Block Chaining (CBC) Mode

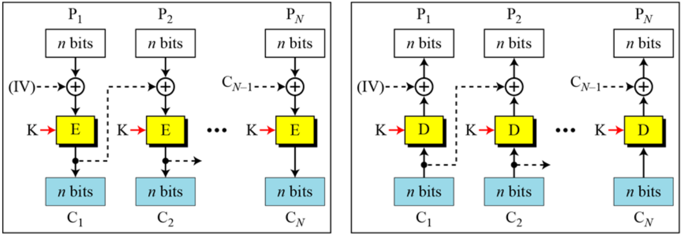
Cipher Block Chaining (CBC) Mode

- In ECB, a PT block always produces the same CT block, which provides some clue to a cryptanalyst
- In CBC mode, each plaintext block is XORed with the previous ciphertext block before being encrypted
- Feedback mechanism is used by chaining
- Initialization Vector(IV)
    - IV should be known by the sender & the receiver
    - It should be agreed upon by sender & receiver when the secret key is established
    - It can be part of the secret key

Cipher Block Chaining (CBC) Mode...

E: Encryption
P_i: Plaintext block i
K: Secret key

D : Decryption
C_i : Ciphertext block i
IV: Initial vector (C_0)



Encryption:

$C_0 = IV$

$C_i = E_K(P_i \oplus C_{i-1})$

Decryption:

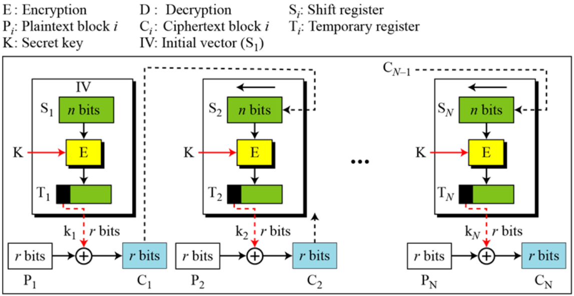
$C_0 = IV$

$P_i = D_K(C_i) \oplus C_{i-1}$

Cipher Feedback (CFB) Mode

Cipher Feedback (CFB) Mode

- ECB & CBC modes encrypt and decrypt blocks of the message. The block size, n, is predetermined by the underlying cipher. Ex: n=64
- When we have to use DES or AES as secure ciphers, the plaintext or ciphertext block sizes are to be smaller

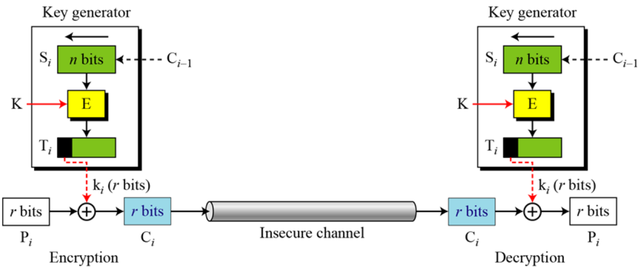


Encryption

Encryption: $C_i = P_i \oplus \text{SelectLeft}_r\{E_K[\text{ShiftLeft}_r(S_{i-1}) \mid C_{i-1})]\}$

Decryption: $P_i = C_i \oplus \text{SelectLeft}_r\{E_K[\text{ShiftLeft}_r(S_{i-1}) \mid C_{i-1})]\}$

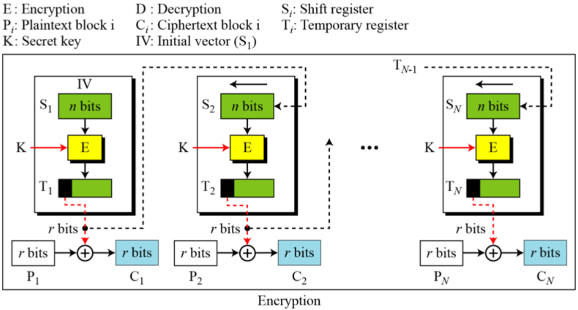
CFB as Stream Cipher



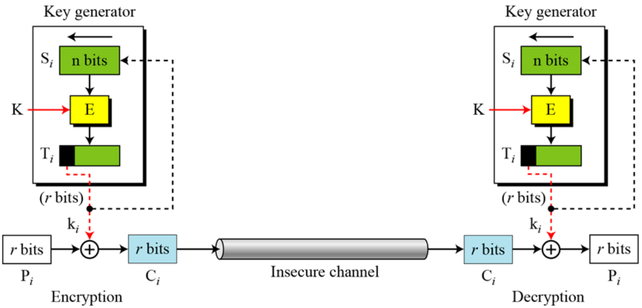
Output Feedback (OFB) Mode

Output Feedback (OFB) Mode

- In this mode each bit in the ciphertext is independent of the previous bit or bits. This avoids error propagation



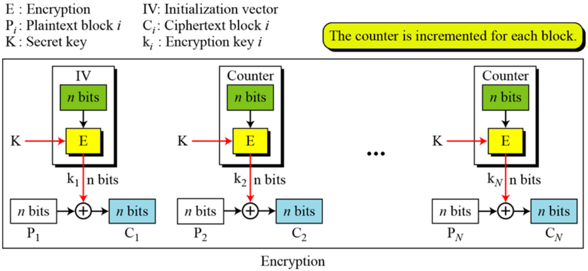
OFB as Stream Cipher



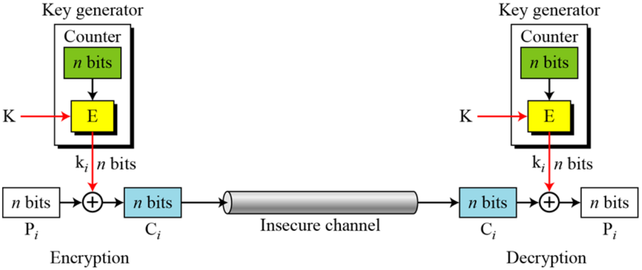
Counter (CTR) Mode

Counter (CTR) Mode

- In the counter mode, there is no feedback. The pseudo randomness in the key stream is achieved using a counter
- A n- bit counter is initialized to a predetermined value(IV) and increment based on a predefined rule
- The plaintext & ciphertext blocks have the same block size as the underlying cipher
- Counter is incremented for each block



CTR as Stream Cipher



Cryptography 6

DES, AES & Diffie-Hellman Key Distribution

Chittaranjan Pradhan
School of Computer Engineering,
KIIT University

DES

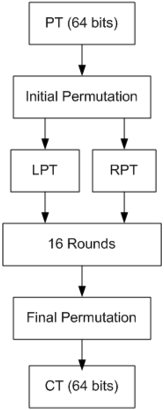
DES

- The Data Encryption Standard (DES) is a symmetric-key block cipher published by the National Institute of Standards and Technology (NIST) in 1975
- Modified Lucifer project of IBM was chosen as DES
- DES is generally used in ECB, CBC or CFB mode

DES Overview

DES Overview

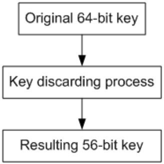
The encryption process is made of two permutations (P-boxes) called initial and final permutations, and sixteen Feistel rounds



DES Overview...

DES Overview...

- Original key consists of 64bits
- 56-bit key can be generated by discarding every 8^{th} bit of the key



Initial Permutation (IP) & Final Permutation (FP)

Initial Permutation (IP)

Initial & Final permutations are keyless straight P-boxes that are inverse of each other. Happens only once

58	50	42	34	26	18	10	2	60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6	64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1	59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5	63	55	47	39	31	23	15	7

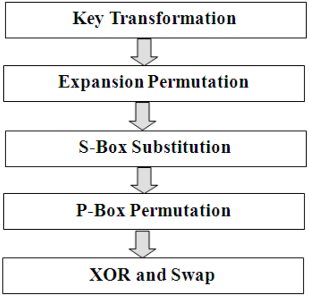
Final Permutation (FP)

40	8	48	16	56	24	64	32	39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30	37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28	35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26	33	1	41	9	49	17	57	25

Details of One Round in DES

Details of One Round in DES

DES uses 16 rounds. Each round of DES is a Feistel cipher



a. Key Transformation

a. Key Transformation

- From the 56- bit key, a 48- bit sub key is generated during each round
- 56- bit key is divided into 2 halves, each of 28- bits. These halves are circularly shifted left by 1 or 2 positions, depending on the round

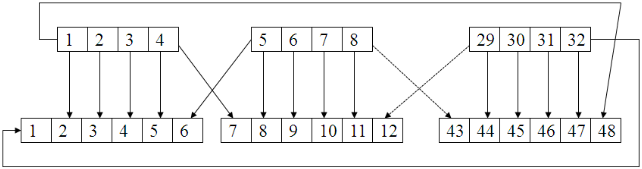
Round	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Bits shifted	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

It is also called as compression permutation
In each round, a different subset of key bits is used

14	17	11	24	1	5	3	28	15	6	21	10
23	19	12	4	26	8	16	7	27	20	13	2
41	52	31	37	47	55	30	40	51	45	33	48
44	49	39	56	34	53	46	42	50	36	29	32

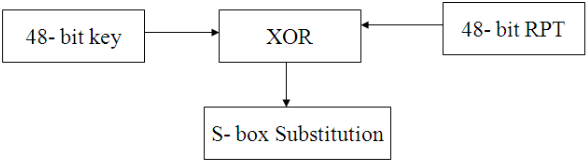
b. Expansion Permutation

- After Initial Permutation, we have 32- bit LPT & 32- bit RPT
- Now, RPT will be expanded to 48- bit
- After expansion permutation, DES uses XOR operation on expanded RPT and round key



b. Expansion Permutation...

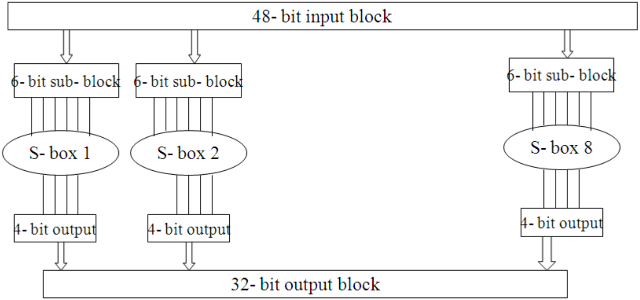
32	1	2	3	4	5	4	5	6	7	8	9
8	9	10	11	12	13	12	13	14	15	16	17
16	17	18	19	20	21	20	21	22	23	24	25
24	25	26	27	28	29	28	29	30	31	32	1



c. S- Box Substitution

c. S- Box Substitution

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



c. S- Box Substitution...

S- box 1

14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

S- box 2

15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

S- box 3

10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

S- box 4

7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

c. S- Box Substitution...

S- box 5

2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

S- box 6

12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

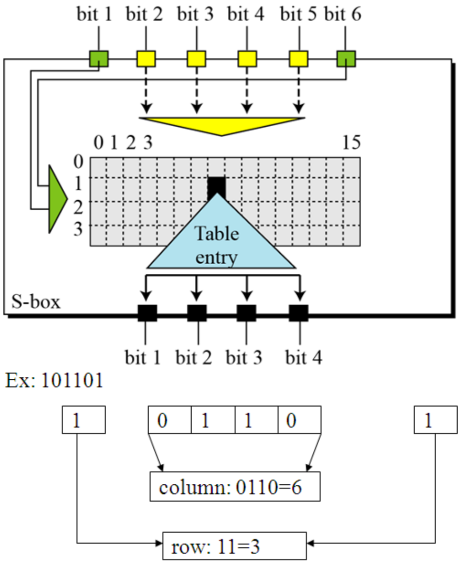
S- box 7

4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

S- box 8

13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

c. S- Box Substitution...



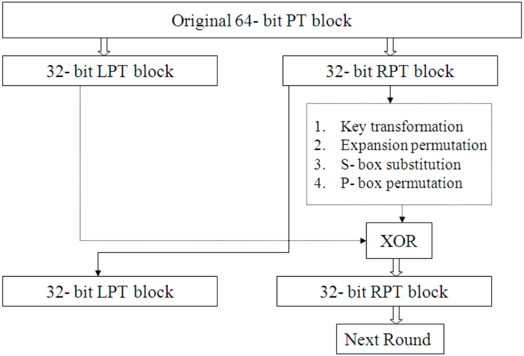
d. P- Box Permutation

d. P- Box Permutation

The last operation in DES round is a permutation with a 32-bit input and a 32-bit output

16	7	20	21	29	12	28	17	1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9	19	13	30	6	22	11	4	25

e. XOR & Swap



Original 64- bit PT block
32- bit LPT block
32- bit RPT block
1. Key transformation
2. Expansion permutation
3. S- box substitution
4. P- box permutation
XOR
32- bit LPT block
32- bit RPT block
Next Round

DES Analysis

DES Analysis

Avalance Effect: a small change in the PT (or key) should create a significant change in CT. DES has been proved to be strong w.r.t. this property

Completeness Effect: each bit of CT needs to depend on many bits on PT. The diffusion and confusion produced by P-boxes and S-boxes in DES, show a very strong completeness effect

Weakness of DES

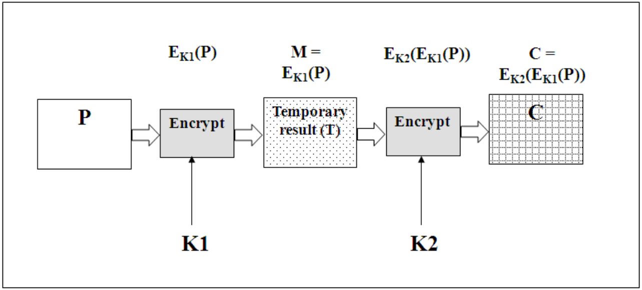
Weakness of DES

- Key size is 56 bit
- Brute force attack needs to check 2^{56} keys, i.e. a computer performing one DES encryption per microsecond would require more than 1000 years to break DES
- A computer with 1 million chips (parallel processing) can find the key in 20 hours
- In 1998, a special computer was built, which found the key in 112 hours

Double DES

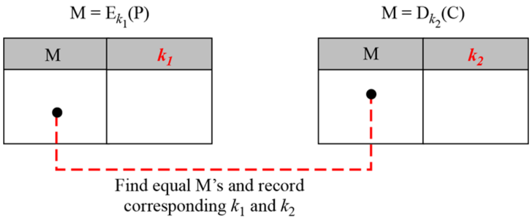
Double DES

- Does twice what DES normally does only once
- Uses 2 keys K1 & K2



Meet-in-the Middle Attack in 2DES

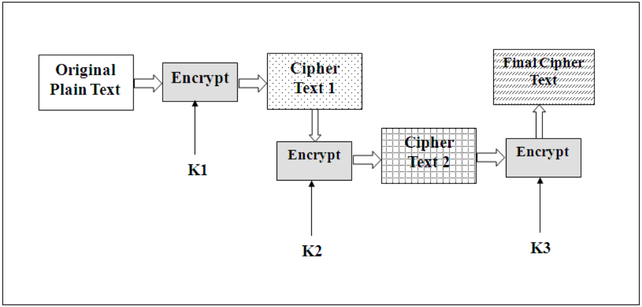
- Cryptanalyst needs 2^{112} keys. It is vulnerable to known-PT attack, called Meet- in- the middle attack
- **Step1**
  - Cryptanalyst uses a large memory
  - Cryptanalyst tried to find out M by using all possible values of K1 and store the values of M in a table in the memory
  - M=E_{k1}(P)
- **Step2**
  - Cryptanalyst decrypts CT with different keys
  - M=D_{k2}(C)



Triple DES with Three Keys

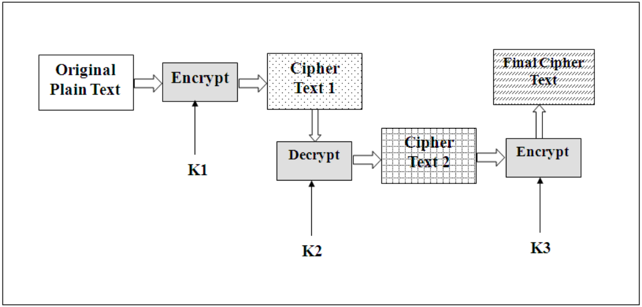
Triple DES with Three Keys

- Does thrice what DES normally does only once
- Uses 3 keys K1, K2 & K3



Triple DES with Three Keys...

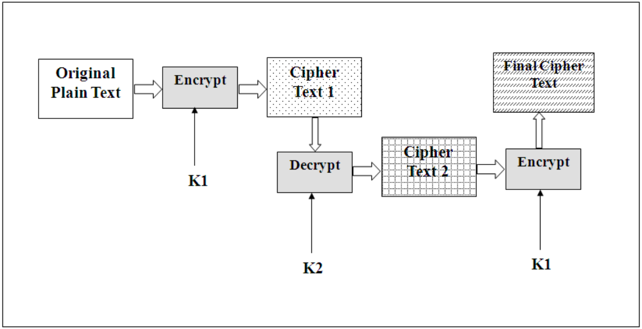
Backward compatibility



Triple DES with Two Keys

Triple DES with Two Keys

Uses 2 Keys K1 & K2



AES (Advanced Encryption Standard)

AES (Advanced Encryption Standard)

- Developed by Rijndael (Rijmen & Daemen) in Nov 2001
- Security
- Cost
- Implementation

- PT block size: 128 bits
- No of rounds: 10 or 12 or 14
- Key size: 128 or 192 or 256 bits

- AES-128, AES-192 & AES-256

One time Initialization

One time Initialization

- Generation of State
  - 16-byte PT block is copied into a 2-D 4X4 array called as state. The order is in the column order

B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	B13	B14	B15	B16

B1	B5	B9	B13
B2	B6	B10	B14
B3	B7	B11	B15
B4	B8	B12	B16

S_{00}	S_{01}	S_{02}	S_{03}
S_{10}	S_{11}	S_{12}	S_{13}
S_{20}	S_{21}	S_{22}	S_{23}
S_{30}	S_{31}	S_{32}	S_{33}

One time Initialization...

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
00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19

- AES USES A MATRIXZZ
- A E S U S E S A M A T R I X Z Z
- 00 04 12 14 12 04 12 00 0C 00 13 11 08 17 19 19

00	12	0C	08
04	04	00	17
12	12	13	19
14	00	11	19

Key Expansion

Key Expansion

- Expands the 4-words (16-byte) key into 11 array, each of size 4X4, i.e. original 16-byte key is expanded to 44-words (11X4X4=176 bytes)
- The first array (4-words) is initialized by the original key. The other 10 arrays (40-words) are used in the 10 rounds, one array per round

- The key is copied into the first four words of the expanded key. The reminder of the expanded key is filled in four words at a time
- Each added word w[i] depends on the immediately preceding word, w[i-1], and the word four positions back, w[i-4]
- In 3 out of 4 cases, a simple XOR is used. For a word whose position in the w array is a multiple of 4, a more complex function is used.

Key Expansion...

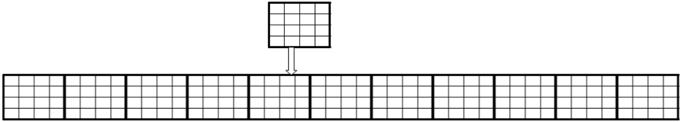
Key Expansion...

The function consists of:

- One-byte circular left shift happens on a word; i.e. an input word [B0, B1, B2, B3] is transformed into [B1, B2, B3, B0]
- Byte substitution on each byte of its input word using S-Box
- The result of the above 2 steps is XORed with a round constant Rcon[j]
- The round constant is a word in which the 3 rightmost bytes are always 0. Thus, the effect of an XOR of a word with Rcon is to only perform an XOR on the leftmost byte of the word. Rcon[j] is calculated as (RC[j],0,0,0)

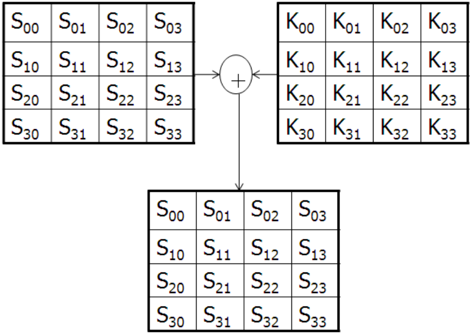
j	1	2	3	4	5	6	7	8	9	10
RC[j]	01	02	04	08	10	20	40	80	1B	36

Key Expansion...

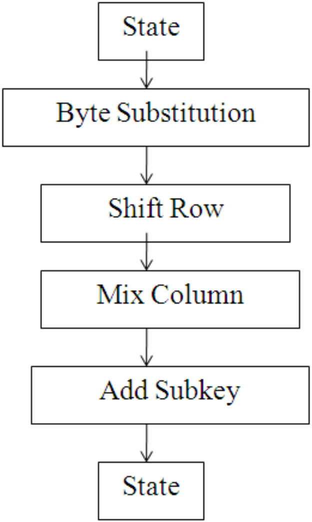


Key Expansion...

XOR the state with the key block



Round


State
Byte Substitution
Shift Row
Mix Column
Add Subkey
State

R1. Byte Substitution

R1. Byte Substitution

- Replace each byte in the state array with its corresponding value from the S-box. Only one S- box is used in AES

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

R2. Shift Row

R2. Shift Row

- Each row of the 4 rows of the state array are rotated to the left. Row 0 by 0B, row 1 by 1B, row 2 by 2B and row 3 by 3B

S_{00}	S_{01}	S_{02}	S_{03}
S_{10}	S_{11}	S_{12}	S_{13}
S_{20}	S_{21}	S_{22}	S_{23}
S_{30}	S_{31}	S_{32}	S_{33}

$\Rightarrow$

S_{00}	S_{01}	S_{02}	S_{03}
S_{11}	S_{12}	S_{13}	S_{10}
S_{22}	S_{23}	S_{20}	S_{21}
S_{33}	S_{30}	S_{31}	S_{32}

1	5	9	13
2	6	10	14
3	7	11	15
4	8	12	16

$\Rightarrow$

1	5	9	13
6	10	14	2
11	15	3	7
16	4	8	12

R3. Mix- Column

R3. Mix- Column

- Each column of the state is multiplied with a fixed Polynomial C(x)= $3x^3+x^2+x+2$

$$\begin{pmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{pmatrix}$$

- b_1=(b_1 X 2) XOR (b_2 X 3) XOR (b_3 X 1) XOR (b_4 X 1)
- b_2=(b_1 X 1) XOR (b_2 X 2) XOR (b_3 X 3) XOR (b_4 X 1)
- b_3=(b_1 X 1) XOR (b_2 X 1) XOR (b_3 X 2) XOR (b_4 X 3)
- b_4=(b_1 X 3) XOR (b_2 X 1) XOR (b_3 X 1) XOR (b_4 X 2)

R4. Add Sub key

R4. Add Sub key

- XOR each byte of the round key with its corresponding byte in the state array

Properties	AES	DES
Block Size (in bits)	128	64
Key size(in bits)	128/ 192/ 256	56
Speed	High	Low
Encryption primitives	Substitution, shift, bit mixing	Substitution, permutation
Rounds	10/ 12/ 14	16

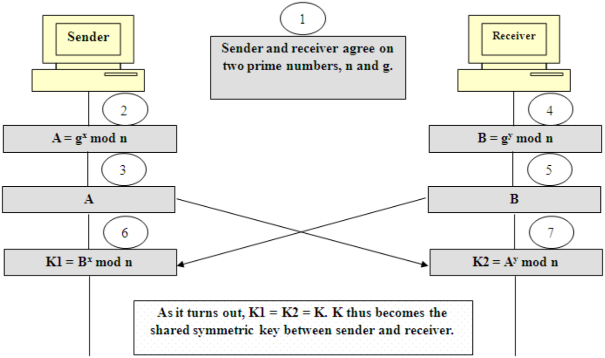
Properties	AES	3-DES
Key size(in bits)	128/ 192/ 256	112/ 168
Speed	High	Low
Time to crack (in a machine with 255 keys/second)	149 Trillion years	4.6 Billion years
Resource Consumption	Low	medium

Diffie- Hellman Key Agreement

Diffie- Hellman Key Agreement

- Devised by Whitefield Diffie and Martin Hellman in 1976 for the solution to the key exchange problem
- Two parties create a symmetric session key without the need of a KDC
- Two parties choose two large prime numbers n and g, which need not be kept secret
- Alice chooses a large random number x such that $0 \leq x \leq n-1$ and calculates $A = g^x \bmod n$
- Bob chooses another large random number y such that $0 \leq y \leq n-1$ and calculates $B = g^y \bmod n$
- Alice sends A to Bob. Similarly, Bob sends B to Alice
- Alice calculates key $K = B^x \bmod n$
- Bob calculates key $K = A^y \bmod n$

Diffie- Hellman Key Agreement...



$$K = B^x \bmod n = (g^y)^x \bmod n = A^y \bmod n = (g^x)^y \bmod n = g^{xy} \bmod n$$

Ex: n=23, g=7, x=3, y=6

Problems in Diffie- Hellman Algorithm/ Man-in-the-middle Attack

Problems in Diffie- Hellman Algorithm/ Man-in-the-middle Attack

- Eve can fool Alice and Bob by creating 2 keys: one between himself and Alice & another between himself and Bob
- n and g are public
- Alice chooses x, calculates $A = g^x \bmod n$ and sends A to Bob
- Eve intercepts A. He chooses z, calculates $C = g^z \bmod n$ and sends C to both Alice and Bob
- Bob chooses y, calculates $B = g^y \bmod n$ and sends B to Alice. But, B is intercepted by the Eve
- Alice and Eve calculates $K1=g^{xz} \bmod n$, which becomes a shared key between Alice and Eve
- Eve and Bob calculates $K2=g^{zy} \bmod n$, which becomes a shared key between Eve and Bob

Cryptography 7

Primes, Primality Test, Factorization & CRT

Chittaranjan Pradhan
School of Computer Engineering,
KIIT University

Primes

Primes

- A prime is divisible only by itself and 1
- Number 1 is relatively prime with any integer

- Number of Primes:
  [n/(ln n)] < π(n) < [n/(ln n - 1.08366)]
- Check whether the number n is divisible by the primes less than $\sqrt{n}$
- Ex: 97 is prime? 301 is prime?

1. Sieve of Eratosthenes

- Write all the numbers between 2 and n
- Check any number in the above range is divisible by the primes less than $\sqrt{n}$
- Cross out the numbers divisible by the above primes
- Remaining numbers are the primes
- Ex: Primes under 50

Primes...

Euler's Phi-Function

- **Φ(1) = 0**
- **Φ(p) = p-1, if p is prime**
- **Φ(m x n) = Φ(m) x Φ(n), if m & n are relatively prime**
- **$\Phi(p^e) = p^e - p^{e-1}$, if p is prime**

- We can combine the above four rules to find the value of Φ(n)
- Φ(n) finds the number of integers that are both smaller than n and relatively prime to n

- Φ(10), Φ(13)
- What is the number of elements in ${Z_{14}}^*$

Primes...

Fermat's Little Theorem

- If p is a prime number and a is an integer such that p doesn't divide a, then
  $a^{p-1} \equiv$ **1 mod p**
- If p is a prime number and a is an integer, then
  $a^{p} \equiv$ **a mod p**

- If the exponent and the modulus are not the same, with substitution this can be solved
- Ex: 6^{10} mod 11, 3^{12} mod 11

Application of Fermat's Little Theorem:

- Used to find multiplicative inverses quickly if the modulus is a prime
- If p is a prime and a is an integer such that p doesn't divide a, then
  a^{-1} **mod p =** a^{p-2} **mod p**
- Ex: 8^{-1} mod 17, 5^{-1} mod 23

Primes...

Euler's Theorem

- Generalization of Fermat's Little theorem
- Here, the modulus is an integer

- If a & n are coprimes, $a^{\Phi(n)} \equiv$ **1 (mod n)**
- If a & n are not coprimes and if n = p x q,
  $a^{k*\Phi(n)+1} \equiv$ **a (mod n)**

- Ex: 6^{24} mod 35, 20^{62} mod 77

Application of Euler's Theorem:

- Used to find multiplicative inverses modulo a composite
  a^{-1} **mod n =** $a^{\Phi(n)-1}$ **mod n**
- Ex: 8^{-1} mod 77, 7^{-1} mod 15, 71^{-1} mod 100

Generation of Primes

Generation of Primes

- Mersenne Primes

  $M_p = 2^p - 1$

  Ex: p= 2, 3, 5, 7, 11

- Fermat Primes

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

  Ex: n=1, 2, 3, 4, 5

Primality Testing

Primality Testing

- **Deterministic algorithms**: Gives correct answer
- **Probabilistic algorithms**: Gives an answer that is correct most of the times, but not always

Deterministic algorithms

Divisibility Algorithm

- All divisors smaller than $\sqrt{n}$ are used. If any of these numbers divides n, then n is composite

```
divisibility_test(n){
            r← 2
            while(r< √n)
            {
            if(r I n)
                return (composite)
            r←r+1
            }
            return (prime)
        }
```

- The algorithm can be improved by testing only odd numbers

Deterministic algorithms...

Divisibility Algorithm...

- It can be further improved by using a table of primes between 2 & $\sqrt{n}$
- If each arithmetic operation uses only one bit operation, then the bit-operation complexity is $\sqrt{2^{n_b}} = 2^{n_b/2}$, where n_b is the number of bits in n
- The complexity can be represented as $O(2^{n_b})$
- This algorithm is infeasible (intractable) if n_b is large
- Ex: $n_b = 200 bits$

Deterministic algorithms...

AKS (Agrawal - Kayal - Saxena) Algorithm

- Can be used to verify the primality of any general number given with time complexity $O((log_2{}^{n_b})^{12})$
  $(x-a)^p \equiv (x^p$**-a) (mod p)**

```
If (n=a^b with b>1)
    output composite
r = 2
While (r < n){
    if (gcd(r, n) ≠ 1)
        output composite;
    if (r is prime, r > 2)
        q = greatest prime divisor of (r - 1);
            if ( (q ≥ 4) and (n^((r - 1)/q) ≠ 1 (mod r)) )
                break;
    r = r + 1;
    }
For a = 1 to
    if (x - a)^n ≡ x^n - a (mod x^r − 1,n)
        output composite;
Output prime;
```

Probabilistic algorithms

1. Fermat Primality Test

- **If p is a prime, then $a^{p-1} \equiv$ 1 mod p**
- Bit-operation complexity $O(n_b)$
- Ex: 5, 561

2. Square Root Test

- **If n is a prime, $\sqrt{1}$ mod n = $\pm$1**
- **If n is a composite, $\sqrt{1}$ mod n = $\pm$1 and possibly other values**
- Ex: 7, 8, 17, 22

Probabilistic algorithms...

3. Miller - Rabin Test

- Combines the Fermat test and Square root test
- n-1 is written as the product of an odd number **m** & a power of **2**
  n-1=m x 2^k

$$a^{n-1} = a^{m \times 2^k} = [a^m]^{2^k} = [a^m]^{\underbrace{2^{2^{\cdot^{\cdot^{\cdot^{2}}}}}}_{k\text{ times}}}$$

- Run time complexity of $O((logn)^3)$

Probabilistic algorithms...

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"
}
```

- Ex: 561, 27, 61

Recommended Primality Test

Recommended Primality Test

Most popular primality test is a combination of the divisibility test and the Miller - Rabin test

- Choose an odd integer
- Do some trivial divisibility tests on some known primes such as 3, 5, 7, 11, 13 ...
  - If the number passes all of these tests, go to next step
  - else, go back to step 1 and choose another odd number
- Choose a set of bases for testing. A large set of bases is preferable
- Do Miller - Rabin tests on each of the bases
  - If any of them fails, go back to step 1 and choose another odd number
  - If the test passes for all bases, number is prime

Ex: 4033

Factorization

Factorization

- Any positive integer can be written as
  n = p_1^{e1} **x** p_2^{e2} **x ... x** p_k^{ek}

- **GCD (Greatest Common Divisor)**:
  $a = p1^{a1} x p2^{a2} x ... x pk^{ak}$
  $b = p1^{b1} x p2^{b2} x ... x pk^{bk}$

  $gcd(a, b) = p1^{min(a1,b1)} x p2^{min(a2,b2)} x ... x pk^{min(ak,bk)}$

- **LCM (Least Common Multiplier)**:
  $a = p1^{a1} x p2^{a2} x ... x pk^{ak}$
  $b = p1^{b1} x p2^{b2} x ... x pk^{bk}$

  $lcm(a, b) = p1^{max(a1,b1)} x p2^{max(a2,b2)} x ... x pk^{max(ak,bk)}$

Factorization...

1. Trial Division Method

- Trial division can be attempted by all primes up to $\sqrt{n}$
- This method is good if $n < 2^{10}$, but it is inefficient and infeasible for factoring large integers
- Ex: 1233

```
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
}
```

Factorization...

2. Fermat Method

- It divides a number n into two positive numbers a & b so that n = a x b
- n = x^2 - y^2 = a x b with a = (x + y) and b = (x - y)
- It tries to find two integers a and b close to each other

```
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
    }
}
```

Factorization...

3. Pollard's p-1 Method

- Pollard's p-1 factoring algorithm is a special-purpose factoring algorithm that can be used to efficiently find any prime factors p of a composite integer n for which p - 1 is smooth with respect to some relatively small bound B
- Let B be a positive integer. An integer n is said to be B-smooth, or smooth with respect to a bound B, if all its prime factors are $\leq$ B. Ex: 57247159 with B=8

```
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
}
```

Factorization...

4. Pollard rho Method

- Choose x1, a small random integer called, seed
- Use a function to calculate x2 such that n doesn't divide x1-x2
- Calculate gcd(x1-x2,n):
  - If it isn't 1, the result is a factor of n; stop
  - If it is 1, return to step 1 and repeat the process with x2

```
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
}
```

Chinese Remainder Theorem (CRT)

Chinese Remainder Theorem (CRT)

- The CRT is used to solve a set of congruent equations with one variable but different moduli, which are relatively prime

  $x \equiv a_1 \pmod{m_1}$

  $x \equiv a_2 \pmod{m_2}$

  ...

  $x \equiv a_k \pmod{m_k}$

- CRT states that these equations have a unique solution if the moduli are relatively prime:
  - Find M = m_1 x m_2 x ... x m_k
  - Find M_1 = M/m_1, M_2 = M/m_2, ... , M_k = M/m_k
  - Find the multiplicative inverse of M_1, M_2, ..., M_k using the corresponding moduli (m_1, m_2, ..., m_k). Let they are M_1^{-1}, M_2^{-1}, ..., M_k^{-1}
  - The solution to the simultaneous equations is:
    x = (a_1 x M_1 x M_1^{-1} + a_2 x M_2 x M_2^{-1} + ... + a_k x M_k x M_k^{-1}) mod M

CRT...

CRT...

- $x \equiv 2 \bmod 3$
  $x \equiv 3 \bmod 5$
  $x \equiv 2 \bmod 7$
- Find an integer that has a remainder of 3 when divided by 7 and 13, but is divisible by 12