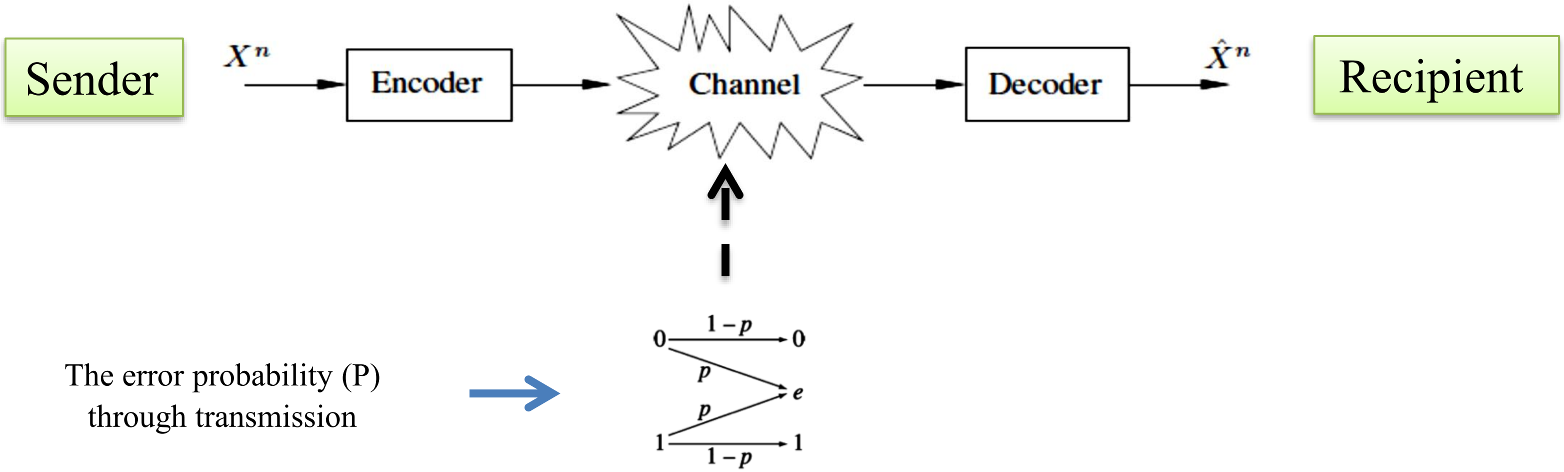
CODING THEORY

In daily life most people in the world uses applications resulting from what today is known as the areas of source coding and channel coding. These applications may for instance be compact discs (CDs), mobile phones, MP3 players, digital versatile discs (DVDs), digital television, voice over IP (VoIP), video streaming etc.



Sender

X^n

Encoder

Channel

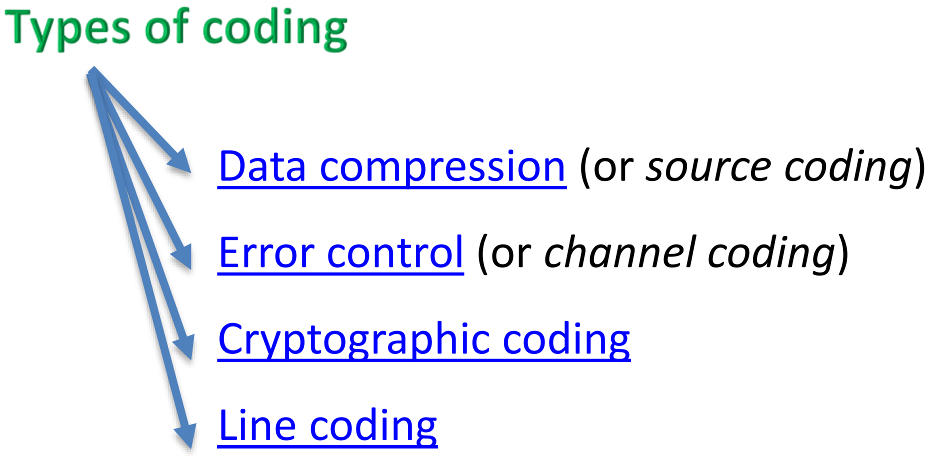
Decoder

$\hat{X}^n$

Recipient

The error probability (P) through transmission

Type of Coding (codes that used in the communication channels)



Types of coding

Data compression (or *source coding*)

Error control (or *channel coding*)

Cryptographic coding

Line coding

What challenges does channel coding address?

- Reduced error rates and retransmission

- Increased capacity

- Increased throughput سعة المعالجة

- Reduced power usage

Linear codes

Algebraic coding theory is basically divided into two major types of codes:

- Linear block codes رموز الكتلة الخطية
- Convolutional codes الرموز التلفيفية

It analyzes the following three properties of a code – mainly:

- Code word length

- Total number of valid code words

- The minimum distance between two valid code words, using mainly the Hamming distance, sometimes also other distances like the Lee distance

Linear block codes

- Linear block codes have the property of linearity , Linear block codes are summarized by their symbol alphabets (e.g., binary or ternary) and parameters **(*n*, *m*, d_{min})** where:
- **n** is the length of the codeword, in symbols,
- **m** is the number of source symbols that will be used for encoding at once,
- d_{min} is the minimum hamming distance for the code.
- There are many types of linear block codes, such as
- Cyclic codes (e.g., Hamming codes)
- Repetition codes
- Parity codes
- Polynomial codes (e.g., BCH codes)
- Reed–Solomon codes
- Algebraic geometric codes
- Reed–Muller codes
- Perfect codes

Convolutional codes

- The idea behind a convolutional code is to make every codeword symbol be the weighted sum of the various input message symbols.
- Fundamentally, convolutional codes do not offer more protection against noise than an equivalent block code. **In many cases**, they generally offer greater simplicity of implementation over a block code of equal power. The encoder is usually a simple circuit which has state memory and some feedback logic, normally XOR gates. The decoder can be implemented in software or firmware.
- *The Viterbi algorithm is the optimum algorithm used to decode convolutional codes.*(1) There are simplifications(تبسيطات) to reduce the computational load. (2) They rely on (تعول على) searching only the most likely paths. (3) Although not optimum(ليست الافضل), (4) they have generally been found to give good results in low noise environments.
- Convolutional codes are used in voice band modems (V.32, V.17, V.34) and in GSM mobile phones, as well as satellite and military communication devices.

Cryptographic coding

Cryptography or cryptographic coding is the practice and study of techniques for secure communication in the presence of third parties (called adversaries). More generally, it is about constructing and analyzing protocols that block adversaries; various aspects in information security such as data confidentiality, data integrity, authentication, and non-repudiation are central to modern cryptography. Modern cryptography exists at the intersection of the disciplines of mathematics, computer science, and electrical engineering. Applications of cryptography include ATM cards, computer passwords, and electronic commerce.

Cryptography prior to the modern age was effectively synonymous with *encryption*, the conversion of information from a readable state to apparent nonsense. The originator of an encrypted message shared the decoding technique needed to recover the original information only with intended recipients, thereby precluding unwanted persons from doing the same. Since World War I and the advent of the computer, the methods used to carry out cryptology have become increasingly complex and its application more widespread.

Modern cryptography is heavily based on mathematical theory and computer science practice; cryptographic algorithms are designed around computational hardness assumptions, making such algorithms hard to break in practice by any adversary. It is theoretically possible to break such a system, but it is infeasible to do so by any known practical means. These schemes are therefore termed computationally secure; theoretical advances, e.g., improvements in integer factorization algorithms, and faster computing technology require these solutions to be continually adapted. There exist information-theoretically secure schemes that provably cannot be broken even with unlimited computing power—an example is the one-time pad—but these schemes are more difficult to implement than the best theoretically breakable but computationally secure mechanisms.

Line coding

A line code (also called digital baseband modulation or digital baseband transmission method) is a code chosen for use within a communications system for baseband transmission purposes. Line coding is often used for digital data transport.

Line coding consists of representing the digital signal to be transported by an amplitude- and time-discrete signal that is optimally tuned for the specific properties of the physical channel (and of the receiving equipment). The waveform pattern of voltage or current used to represent the 1s and 0s of a digital data on a transmission link is called *line encoding*. The common types of line encoding are unipolar, polar, bipolar, and Manchester encoding.

Other applications of coding theory

Another concern of coding theory is designing codes that help synchronization. A code may be designed so that a phase shift can be easily detected and corrected and that multiple signals can be sent on the same channel.

Another application of codes, used in some mobile phone systems, is code-division multiple access (CDMA). Each phone is assigned a code sequence that is approximately uncorrelated with the codes of other phones. When transmitting, the code word is used to modulate the data bits representing the voice message. At the receiver, a demodulation process is performed to recover the data. The properties of this class of codes allow many users (with different codes) to use the same radio channel at the same time. To the receiver, the signals of other users will appear to the demodulator only as a low-level noise.

Another general class of codes are the automatic repeat-request (ARQ) codes. In these codes the sender adds redundancy to each message for error checking, usually by adding check bits. If the check bits are not consistent with the rest of the message when it arrives, the receiver will ask the sender to retransmit the message. All but the simplest wide area network protocols use ARQ. Common protocols include SDLC (IBM), TCP (Internet), X.25 (International) and many others. There is an extensive field of research on this topic because of the problem of matching a rejected packet against a new packet. Is it a new one or is it a retransmission? Typically numbering schemes are used, as in TCP.*"RFC793". RFCs. Internet Engineering Task Force (IETF). September 1981.*

Group testing

Group testing uses codes in a different way. Consider a large group of items in which a very few are different in a particular way (e.g., defective products or infected test subjects). The idea of group testing is to determine which items are "different" by using as few tests as possible. The origin of the problem has its roots in the Second World War when the United States Army Air Forces needed to test its soldiers for syphilis.

Analog coding

Information is encoded analogously in the neural networks of brains, in analog signal processing, and analog electronics. Aspects of analog coding include analog error correction, analog data compression and analog encryption.

Neural coding

Neural coding is a neuroscience-related field concerned with how sensory and other information is represented in the brain by networks of neurons. The main goal of studying neural coding is to characterize the relationship between the stimulus and the individual or ensemble neuronal responses and the relationship among electrical activity of the neurons in the ensemble. It is thought that neurons can encode both digital and analog information, and that neurons follow the principles of information theory and compress information, and detect and correct errors in the signals that are sent throughout the brain and wider nervous system.

End of introduction of the Coding theory

Subject : Data and computer Security

Shatt Al-arab University College

Department of Computer Sciences

Fourth Stage

Lecturer : Assist. Prof. Dr. Basim Sahar Yaseen

Building an infrastructure for the subject for students, and establishing an information basis for the various topics of information security and cybersecurity, so that the student can delve into this field and develop himself, whether this development is vertical through higher academic studies, or horizontally through building fieldwork experiences and scientific research.

Data (Data, Information and Knowledge) and computer(Computer and Networks) Security

Divided into:

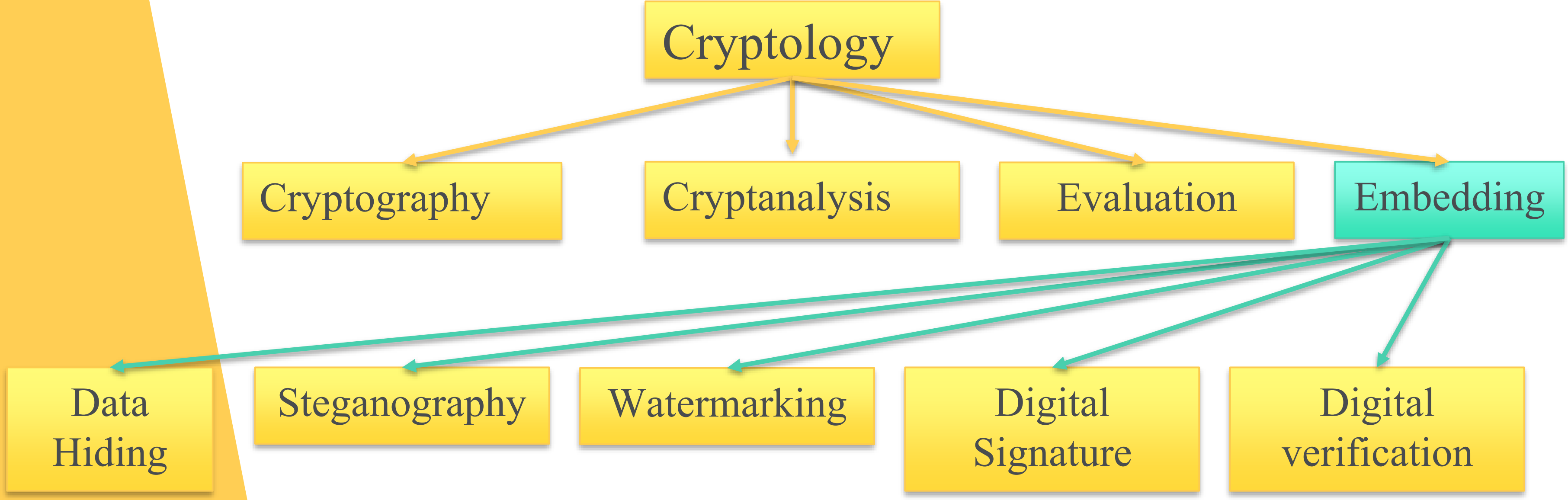
Cyber Security

Information Security

Covered by:

Information Technology(IT)

Security Tools



Cryptology

Cryptography

Cryptanalysis

Evaluation

Embedding

Data Hiding

Steganography

Watermarking

Digital Signature

Digital verification

Security Tools

Password’s System

System of Authorization of the Access

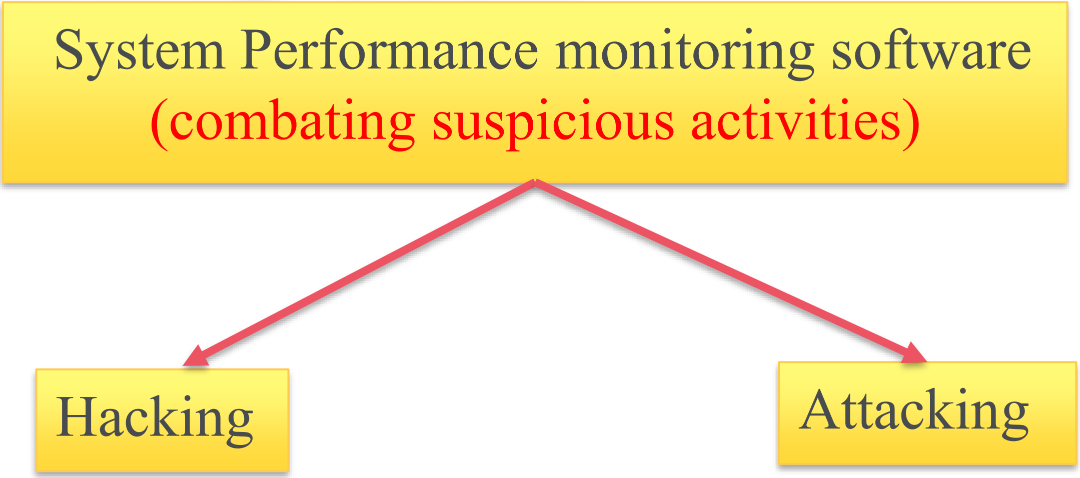
Security Tools

Antiviruses programs

Firewalls programs

Anti :Worms , Trojan Horse, Spyware…

Security Tools



System Performance monitoring software (combating suspicious activities)

Hacking

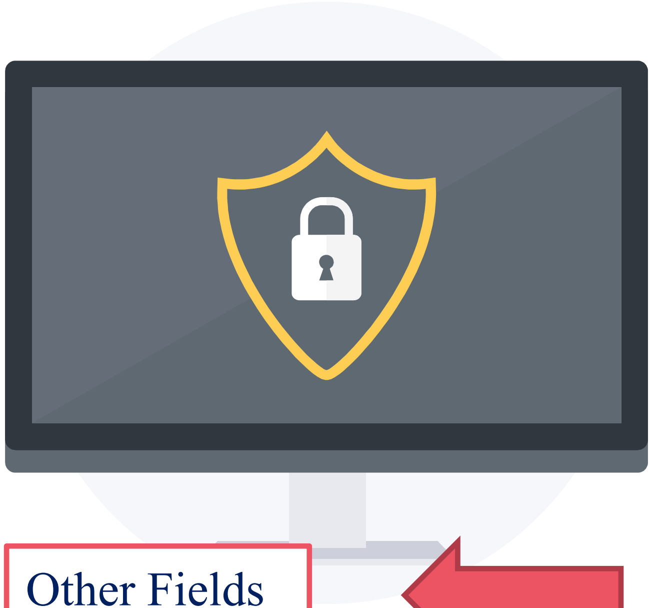
Attacking

Physical Methods and Techniques

Information and its devices

Computer networks , Communication systems , distributed data systems , Mobile’s Systems , Sensors Systems , Satellites Systems , Control Systems , Cameras and display devices , Navigation Systems ,….

Cognitive Requirements



Other Fields

Mathematics

Information Theory

Statistic

Mathematic Logic

Propositional logic

Logic Design

Numbers Theory

Communication Theory

Coding Theory

Image Processing

Computing Theory

Graph Theory

Formal Language Theory

Architecture design

Computer-based Hardware

Processing Theory

Networking

Security Goals

High protection

Breach Detection

Data Integrity

Availability

High Security

Confidentiality

Authentication verification

Data transmission efficiency

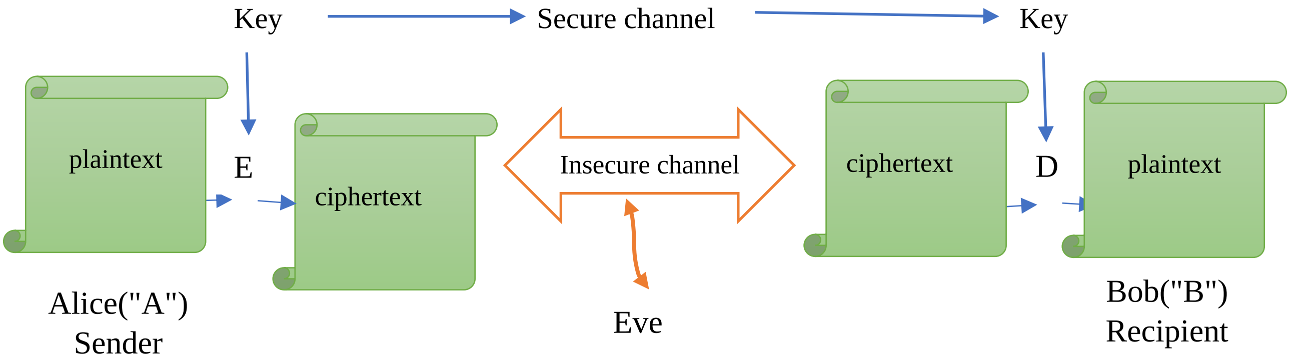
??

Head Lines

- Cryptography
- Computer and Information security Objectives
- Classical Methods
- Transposition Cipher
- Columnar cipher

Cryptography

In computer science, cryptography refers to secure information and communication techniques derived from mathematical concepts and a set of rule-based calculations called algorithms, to transform messages in ways that are hard to decipher. These deterministic algorithms are used for cryptographic key generation, digital signing, verification to protect data privacy, web browsing on the internet, and confidential communications such as credit card transactions and email. Cryptography prior to the modern age was effectively synonymous with *encryption*, converting information from a readable state to unintelligible nonsense. The sender of an encrypted message shares the decoding technique only with intended recipients to preclude access from adversaries. The cryptography literature often uses the names Alice ("A") for the sender, Bob ("B") for the intended recipient, and Eve ("eavesdropper") for the adversary. Cryptography methods have become increasingly complex and its applications more varied. Figure-1 below is an encrypted communication system.



Modern cryptography is heavily based on mathematical theory and computer science practice; cryptographic algorithms are designed around computational hardness assumptions, making such algorithms hard to break in actual practice by any adversary. While it is theoretically possible to break into a well-designed such system, it is infeasible in actual practice to do so. Such schemes, if well designed, are therefore termed "computationally secure"; theoretical advances, e.g., improvements in integer factorization algorithms, and faster computing technology require these designs to be continually reevaluated, and if necessary, adapted.

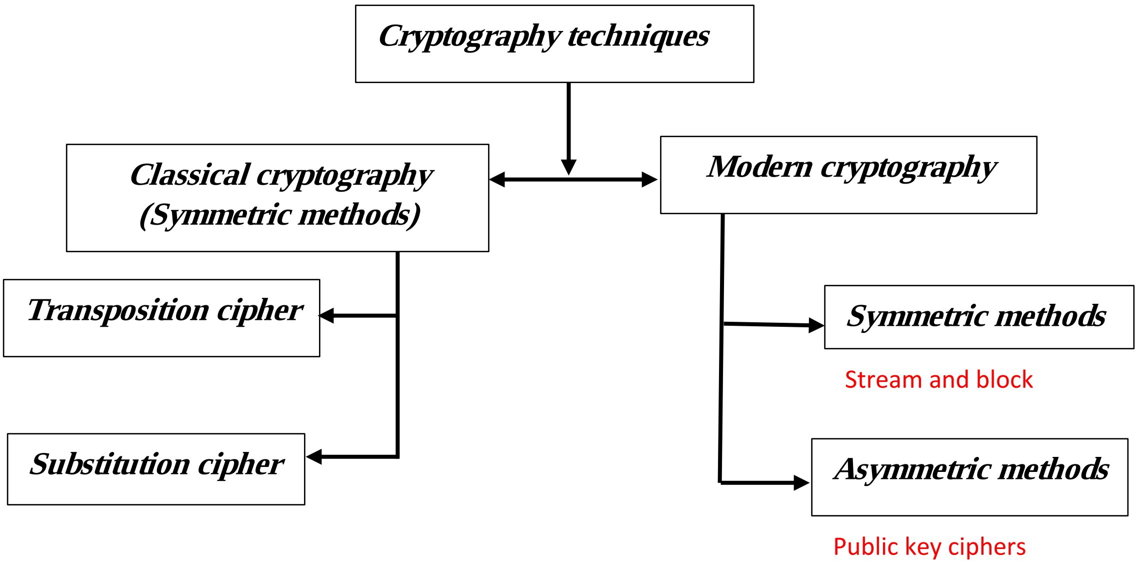


Figure-2: classification of the cryptography techniques

Note: We can classify the classical methods as a symmetric methods. Because they are using same key in the encryption and decryption processes.

Computer and Information security Objectives

All scientific fields that are concerned with computer and information security, like cryptography, Authentication techniques, Steganography, seek to achieve important goals illustrated in the figure 3.

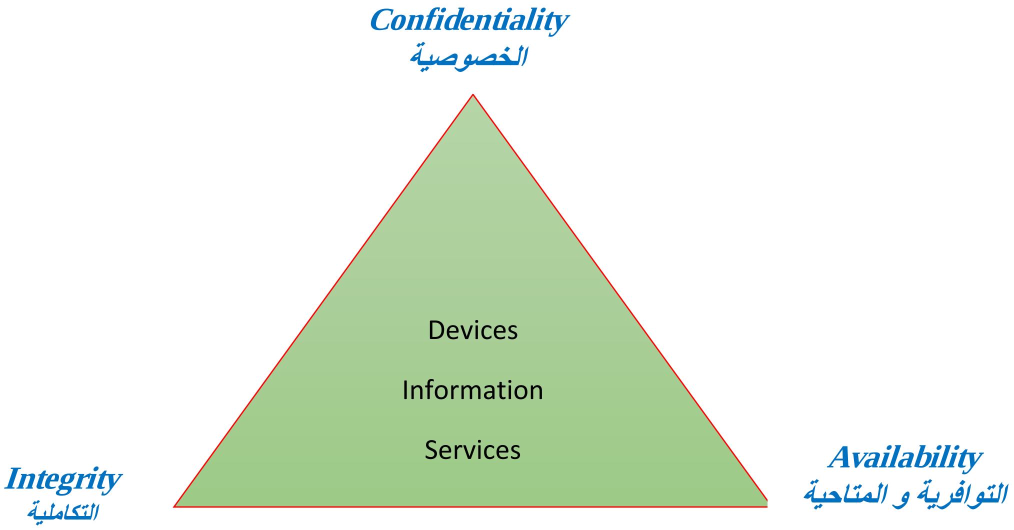


Figure 3: Objectives of computer and information security

Confidentiality

Data confidentiality: Assures that private or confidential information is not made available or disclosed to unauthorized individuals

Privacy: Assures that individuals control or influence what information related to them may be collected and stored and by whom and to whom that information may be disclosed

Integrity

Data integrity: Assures that information and programs are changed only in a specified and authorized manner.

System integrity: Assures that a system performs its intended function in an unimpaired manner, free from deliberate or inadvertent unauthorized manipulation of the system.

Availability

Assures that systems work promptly and service is not denied to authorized users.

Classical cryptography(methods)

By classical methods, we mean those methods that have been used in the ancient eras. For their simplicity and because most of them are done manually. Today, using the classical methods greatly reduced, they are including two classes: transposition methods that based on concept of transforming clear information into incomprehensible form by rearrangement it, and substitution methods that achieving same this goal by substituting original alphabet of the clear information with replacement alphabet(s).

Transposition Ciphers

Transposition ciphers jumble the letters of the message in a way that is designed to confuse the attacker, but can be unjumbled by the intended recipient. The concept of transposition is an important one and is widely used in the design of modern ciphers, as will be seen in subsequent chapters. Note that the key must provide sufficient information to unscramble the ciphertext.

Columnar Transposition

Suppose we have plaintext SEETHELIGHT and we want to encrypt this using a columnar transposition cipher. We first put the plaintext into the rows of an array of some given dimension. Then we read the ciphertext out of the columns. The key consists of the number of columns in the array. For example, suppose we choose the key to be four, which means that we write the plaintext in four columns as

$$\begin{bmatrix} S & E & E & T \\ H & E & L & I \\ G & H & T & X \end{bmatrix}$$

where the final X is used as to fill out the array. The ciphertext is then read from the columns, which in this case yields SHGEEHELTTIX. The intended recipient, who knows the number of columns, can put the ciphertext into an appropriate-sized array and read the plaintext out from the rows. Not surprisingly, a columnar transposition is not particularly strong. To perform a ciphertext only attack on this cipher, we simply need to test all possible decrypts using c columns, where c is a divisor of the number of characters in the ciphertext.

Keyword Columnar Transposition

The columnar transposition cipher can be strengthened by using a keyword, where the keyword determines the order in which the columns of ciphertext are transcribed. We refer to this as a keyword columnar transposition cipher. For example, consider encrypting the plaintext CRYPTOISFUN using a keyword columnar transposition cipher with keyword MATH, again using four columns. In this case, we get the array that we write the plaintext in four columns as:

$$\begin{array}{cccc} M & A & T & H \\ \hline C & R & Y & P \\ T & O & I & S \\ F & U & N & X \end{array}$$

The ciphertext is read from the columns in alphabetical order (as determined by the keyword), so that, in this example, the ciphertext is ROUPSXCTFYIN where key is AHMT.

Question: Encrypt the plain text "COMMUNICATIONSYSTEM", by using columnar transposition algorithm and key is "24135" ?

Alice(Sender)

$$\begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} C & O & M & M & U \\ N & I & C & A & T \\ I & O & N & S & Y \\ S & T & E & M & X \end{bmatrix} \end{array}$$

To encrypting →

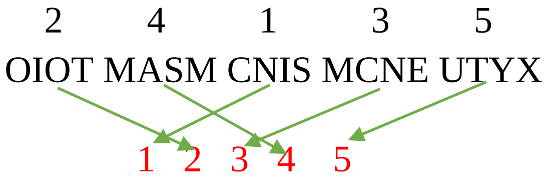
2 4 1 3 5

OIOT MASM CNIS MCNE UTYX

Question: Decrypt the cipher text "OIOTMASMCNISMCNEUTYX", by using columnar transposition algorithm and key is "24135" ?

Bob(Recipient)

Note that: the largest column number in the key is 5, then , the number of columns is 5 , and number of characters in each one is 20 div 5 = 4.



2 4 1 3 5

OIOT MASM CNIS MCNE UTYX

1 2 3 4 5

$$\begin{array}{ccccc} C & O & M & M & U \\ N & I & C & A & T \\ I & O & N & S & Y \\ S & T & E & M & X \end{array}$$

Head Lines

- Examples of the Columnar cipher
- Cryptanalysis of Columnar cipher
- Double columnar cipher

Examples of the Columnar cipher

Example1:

Plain Text: SHATTALARABCOLLAGE

Encryption algorithm: columnar transposition algorithm

Key encryption: 612453

Use the above information to produce the cipher text?

To Encryption:

1	2	3	4	5	6
S	H	A	T	T	A
L	A	R	A	B	C
O	L	L	A	G	E

Cipher Text

6	1	2	4	5	3
ACE	SLO	HAL	TAA	TBG	ARL

Example2:

Cipher Text: ACESLOHALTAATBGARL

Decryption algorithm: columnar transposition cipher

Decryption key: 612453

Use the above information to get the plain text?

To decryption:

6	1	2	4	5	3
ACE	SLO	HAL	TAA	TBG	ARL

1	2	3	4	5	6
S	H	A	T	T	A
L	A	R	A	B	C
O	L	L	A	G	E

Plain Text

SHATTALARABCOLLAGE

Question: Encrypt the plain text “WE ARE DISCOVERED FLEE AT ONCE” by using columnar transposition cipher and the encryption key is 614325?

Cryptanalysis of Columnar cipher

Cryptanalysis is the scientific field from cryptology, that analysis of cryptosystems for the discovery of weaknesses that these systems are suffering. Therefore, for columnar cipher we should testing all possible encryption key permutations and cipher text segmentations, this analysis strategy is called “brute force attack”

Goal of cryptanalysis: obtainment the plain text, encryption key or both.

Question: **cryptanalyze the following cipher text, and determine the encryption key and encryption matrix size? “RMRNXWAUCEAOEEXEPSCXECTIX” ?**

Double columnar cipher

A double transposition was often used to make the cryptography stronger. This is simply a columnar transposition applied twice. The same key can be used for both transpositions, or two different keys can be used.

As an example, we can take the result of the irregular columnar transposition in the previous section” ACESLOHALTAATBGARL”, and perform a second encryption with a different keyword "564231”

1	2	3	4	5	6
A	C	E	S	L	O
H	A	L	T	A	A
T	B	G	A	R	L

As before, this is read off column wise to give the cipher text:

LAR OAL STA CAB ELG AHT

If multiple messages of exactly the same length are encrypted using the same keys, they can be anagrammed simultaneously. This can lead to both recovery of the messages, and to recovery of the keys (so that every other message sent with those keys can be read).

Question: Encrypt the plain text “ WE ARE A COMPUTER SCIENCE” by using the double columnar transposition cipher, and encryption key1 is 651243, and encryption key2 is 15432?

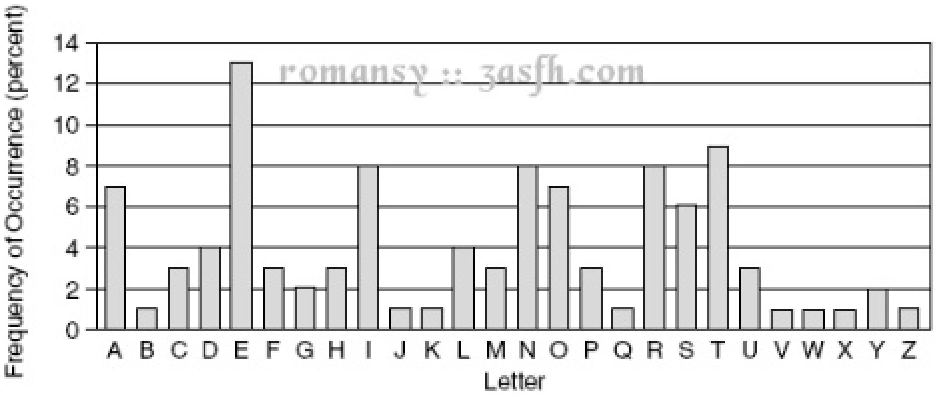
Important Note: In this lecture, we have three questions, so, each student must present answers for these questions in date of next lecture.

Head Lines

- Plain text statistics in the transposition cipher
- Substitution cipher
- Monoalphabetic substitution cipher
- Additive algorithm

Plain text statistics in the transposition cipher

For every natural language, there are specifications of frequencies of its letters. In accordance with this, there are high-frequency letters in every text of the language, and there are letters with low frequencies, and other frequencies between that. In English language, we can noting that letters like E, L, H with high frequencies, and z, q has a low frequencies. The graph below shows the distributing of frequencies of the English letters.



The transposition ciphers have a specialty of preserving the statistical distribution of the language's letters frequencies and not distorting them as substitution cipher does. Only it changes characters positions In the cipher text while preserving its original frequencies from the plain text.

For example, in the following texts of the columnar algorithm the frequency of "A" is 5 in both plain and cipher text, the frequency of "L" is 3,..and so on.

Plain Text: SHATTALARABCOLLAGE

Cipher Text: ACESLOHALTAATBGARL

Substitution cipher

A substitution cipher is a method of encrypting in which units of plaintext are replaced with ciphertext, according to a fixed system; the "units" may be single letters (the most common), pairs of letters, triplets of letters, mixtures of the above, and so forth. The receiver deciphers the text by performing the inverse substitution .

Substitution ciphers can be *compared with transposition ciphers*. In a transposition cipher, the units of the plaintext are rearranged in a different and usually quite complex order, but the units themselves are left unchanged. By contrast, in a substitution cipher, the units of the plaintext are retained in the same sequence in the ciphertext, but the units themselves are altered .

There are a number of different types of substitution cipher. If the cipher operates on single letters, it is termed a simple substitution cipher; a cipher that operates on larger groups of letters is termed polygraphic. A monoalphabetic cipher uses fixed substitution over the entire message, whereas a polyalphabetic cipher uses a number of substitutions at different positions in the message, where a unit from the plaintext is mapped to one of several possibilities in the ciphertext and vice versa.

The figure 1 describes the classification of the classical cryptography, and algorithms as examples of this ciphers.

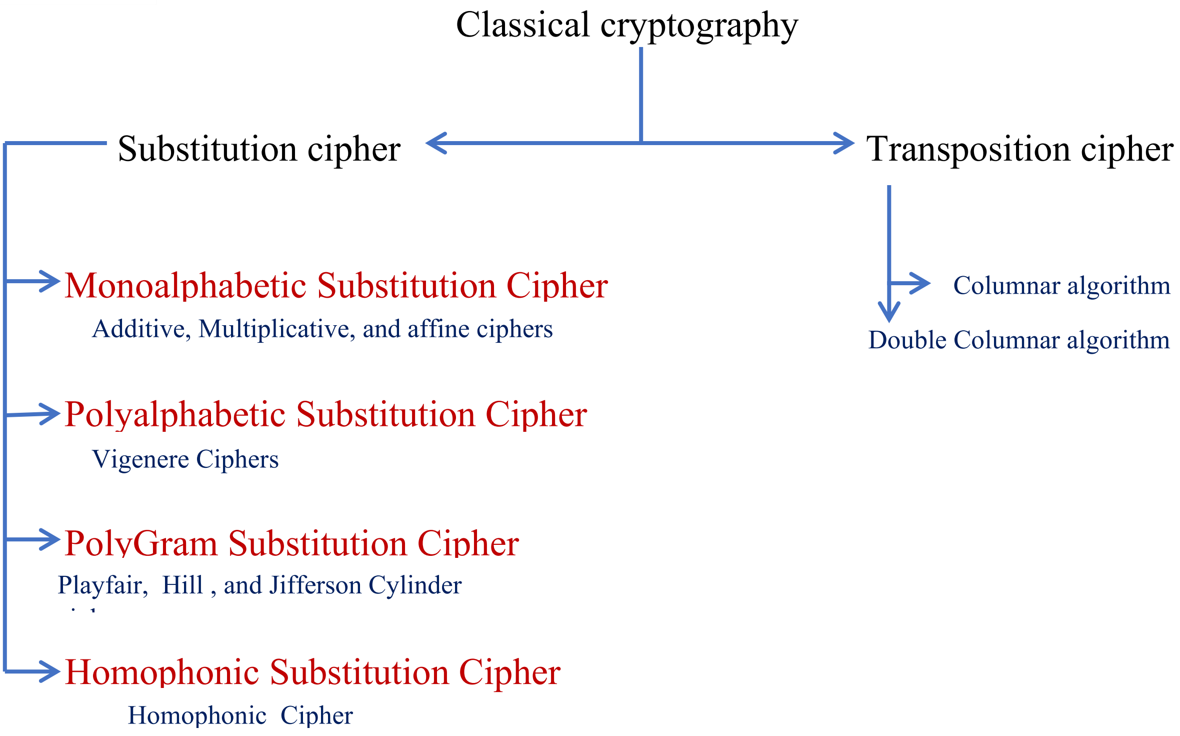

Classical cryptography
Substitution cipher
Transposition cipher
Monoalphabetic Substitution Cipher
Additive, Multiplicative, and affine ciphers
Polyalphabetic Substitution Cipher
Vigenere Ciphers
PolyGram Substitution Cipher
Playfair, Hill , and Jifferson Cylinder
Homophonic Substitution Cipher
Homophonic Cipher
Columnar algorithm
Double Columnar algorithm

Figure 1: Classification of the Classical cryptography

Monoalphabetic substitution cipher

Monoalphabetic S.C. is an encryption algorithms which substitutes plain text characters belonging to the standard alphabet with another belonging to an unique alternate alphabet, and this is formed by a fixed and systematic procedure.

Additive algorithm

The algorithm adds the key value for plain text character order, to produce cipher character.

Encryption: $C=E_{key}(P) = P + Key \mod 26$

Decryption: $P=D_{key}(C) = C - Key \mod 26$

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

Example: Encrypt the plain text “ORIGINALIDEA”, by using additive algorithm and key value 5 ?

Plain text character	**Its order**	**Evaluating cipher text Character**
O	14	14+5 mod 26=19=T
R	**17**	**17+5 mod 26=22=W**
I	**8**	**8+5 mod 26=13=N**
G	**6**	**6+5 mod 26=11=L**
I	**8**	**8+5 mod 26=13=N**
N	**13**	**13+5 mod 26=18=S**
A	**0**	**0+5 mod 26=5=F**
L	**11**	**11+5 mod 26=16=Q**
I	**8**	**8+5 mod 26=13=N**
D	**3**	**3+5 mod 26=8=I**
E	**4**	**4+5 mod 26=9=K**
A	**0**	**0+5 mod 26=5=F**

Then, the cipher text is “**TWNLNSFQNIKF**”.

Example: Decrypt the cipher text “ MFAUWDDMDIJ”, by using additive algorithm and key 18 ?

Cipher text character	**Its order**	**Evaluating plain text Character**
M	**12**	**12-18 mod 26=20=U**
F	5	5-18 mod 26=13=N
A	0	0-18 mod 26=8=I
U	20	20-18 mod 26=2=C
W	22	22-18 mod 26=4=E
D	3	3-18 mod 26=11=L
D	3	3-18 mod 26=11=L
M	12	12-18 mod 26=20=U
D	3	3-18 mod 26=11=L
I	18	18-18 mod 26=0=A
J	9	9-18 mod 26=17=R

Then the plain text is “**UNICELLULAR**”.

Head Lines

- Multiplicative algorithm
- Affine algorithm

Multiplicative algorithm

The algorithm multiplies the key value by plain text character value to produce cipher character.

Encryption: $C=E_{key}(P) = P * Key \mod 26$

Decryption: $P=D_{key}(C) = C * inverse(Key)$

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

Example: Encrypt the plain text "INTELLIGENT", by using multiplicative algorithm and key value 7?

Plain text character	**Its order**	**Evaluating cipher text Character**
I	8	8*7 mod 26=4=E
N	**13**	**13*7 mod 26=13=N**
T	**19**	**19*7 mod 26=3=D**
E	**4**	**4*7 mod 26=2=C**
L	**11**	**11*7 mod 26=25=Z**
L	**11**	**11*7 mod 26=25=Z**
I	**8**	**8*7 mod 26=4=E**
G	**6**	**6*7 mod 26=16=Q**
E	**4**	**4*7 mod 26=2=C**
N	**13**	**13*7 mod 26=13=N**
T	**19**	**19*7 mod 26=3=D**

Then, the cipher text is “ENDCZZEQCND”.

Note **/ The key value can be any ODD number from range [2-25],values 1 and 13 can’t be a key value, Why?**

Example: Decrypt the cipher text “ WYCCANH”, by using multiplicative algorithm and key 11 ?

Cipher character	**Its order**	**Evaluating plain character**
W	**22**	**(22+(0*26)))/11=2=C**
Y	24	(24+(5*26)) /11=14=O
C	2	(2+(5*26))/11=12=M
C	2	(2+(5*26))/11=12=M
A	0	(0+(0*26))/11=0=A
N	13	(13+(5*26))/11=13=N
H	7	(7+(1* 26))/11=3=D

Then the plain text is “**COMMAND**”.

- **Affine algorithm**

  Is a monoalphabetic class, and it combines the previous two algorithms. (i.e. it have two keys : key1 for additive ,and key2 for multiplicative)

Encryption: C=E(P) = (P*key2) + Key1 mod 26 ….. (1)

C=E(P) = (P+key1)key2 mod 26 ………(2)

H.W.

Encrypt the plain text "ORGENIZATION" by using affine equation 1, and equation 2, then evaluate and discus the result cipher text?

Head lines

- Vigenere examples
- Playfair examples
- Homophonic substitution ciphers

- Vigenere examples

EXAMPLE 1:

Encrypt the plain text 'CRYPTOGRAPHY FIELD' by using Vigenere substitution algorithm and key is 'TOURING'?

So, we encipher the plain text either by using Vigenere table or by using equations:

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
0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**

Plain text: **2 17 24 15 19 14 6 17 0 15 7 24 5 8 4 11 3**

Key : 19 14 20 17 8 13 6 19 14 20 17 8 13 6 19 14 20

Encryption's Equation: $C_i = P_i + Key_j$ mod 26

C1=2 + 19 mod 26 = 21 = V

C2=17 + 14 mod 26 = 5 = F

C3=24 + 20 mod 26 = 18 = S

C4=15 + 17 mod 26 = 6 = G

C5=19 + 8 mod 26 = 1 = B

C6=14 + 13 mod 26 = 1 = B

C7=6 + 6 mod 26 =12 = M

C8=17 + 19 mod 26 = 10 = K

C9=0 + 14 mod 26 = 14 = O

C10=15 + 20 mod 26 = 9 = J

C11=7 + 17 mod 26 = 24 = Y

C12=24 + 8 mod 26 = 6 = G

C13=5 + 13 mod 26 = 18 = S

C14=8 + 6 mod 26 = 14 = O

C15=4 + 19 mod 26 = 23 = X

C16=11 + 14 mod 26 = 25 = Z

C17=3 + 20 mod 26 = 23 = X

Then the cipher text is 'VFCGBBMKOJYGSOXZX'

EXAMPLE 2:

Decrypt the cipher text 'WFAUGHACZQ ' by using Vigenere algorithm and key is 'PROGRAM'?

Decryption equation: $P_i = C_i - Key_j$ mod 26

Cipher text → 22 5 0 20 6 7 0 2 25 16

Key → 15 17 14 6 17 0 12 15 17 14

P1=22-15 mod 26 = 7 = H

P2=5-17 mod 26 = 14 = O

P3=0-14 mod 26 = 12 = M

P4=20-6 mod 26 = 14 = O

P5=6-17 mod 26 = 15 = P

P6=7-0 mod 26 = 7 = H

P7=0-12 mod 26 = 14 = O

P8=2-15 mod 26 = 13 = N

P9=25-17 mod26 = 8 = I

P10=16-14 mod26 = 2 = C

Then, the plain text is 'HOMOPHONIC'

- Playfair examples

Example 1:

Encrypt the plain text 'SUBSTITUTION' by using playfair algorithm and keyword is 'SOCIAL MEDIA'?

First, we construct the encryption matrix:

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

SECOND, encrypt the plain text as units (2 characters per one unit). Note that we apply rows strategy:

Plain text → SU BS TI TU TI ON

Cipher text →PA AL KY UP KY GA

Question : Decrypt the cipher text 'PAALKYUPKYGA' by using playfair algorithm and the keyword is 'SOCIAL MEDIA'?

- Homophonic substitution ciphers

Homophonic substitution cipher is a much more complicated variant of substitution cipher where, instead of using one to one mapping of simple substitution, one to many mapping is used. In one to many mapping, each plaintext letter can be substituted with multiple ciphertext symbols.

The Homophonic Substitution cipher is a substitution cipher in which single plaintext letters can be replaced by any of several different ciphertext letters. They are generally much more difficult to break than standard substitution ciphers. The number of characters each letter is replaced by is part of the key, e.g. the letter 'E' might be replaced by any of 5 different symbols, while the letter 'Q' may only be substituted by 1 symbol.

The easiest way to break standard substitution ciphers is to look at the letter frequencies, the letter 'E' is usually the most common letter in English, so the most common ciphertext letter will probably be 'E' (or perhaps 'T'). If we allow the letter 'E' to be replaced by any of 3 different characters, then we can no longer just take the most common letter, since the letter count of 'E' is spread over several characters. As we allow more and more possible alternatives for each letter, the resulting cipher can become very secure.

An Example

Our cipher alphabet is as follows:

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

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

9 7 3 5 0 4 6

2

1

To encipher the message DEFEND THE EAST WALL OF THE CASTLE, we find 'D' in the top row, then replace it with the letter below it, 'F'. The second letter, 'E' provides us with several choices, we could use any of 'Z', '7', '2' or '1'. We choose one of these at random, say '7'. After continuing with this, we get the ciphertext:

plaintext: DEFEND THE EAST WALL OF THE CASTLE

ciphertext: F7EZ5F UC2 1DR6 M9PP 0E 6CZ SD4UP1

The number of ciphertext letters assigned to each plaintext letter was chosen to flatten the frequency distribution as much as possible. Since 'E' is normally the most common letter, it is allowed more possibilities so that the frequency peak from the letter 'E' will not be present in the ciphertext.

Asymmetric Encryption

RSA Steps and Example

Why Asymmetric Encryption?

Symmetric Encryption problems:

1- keys management problem 2- Authentication problem

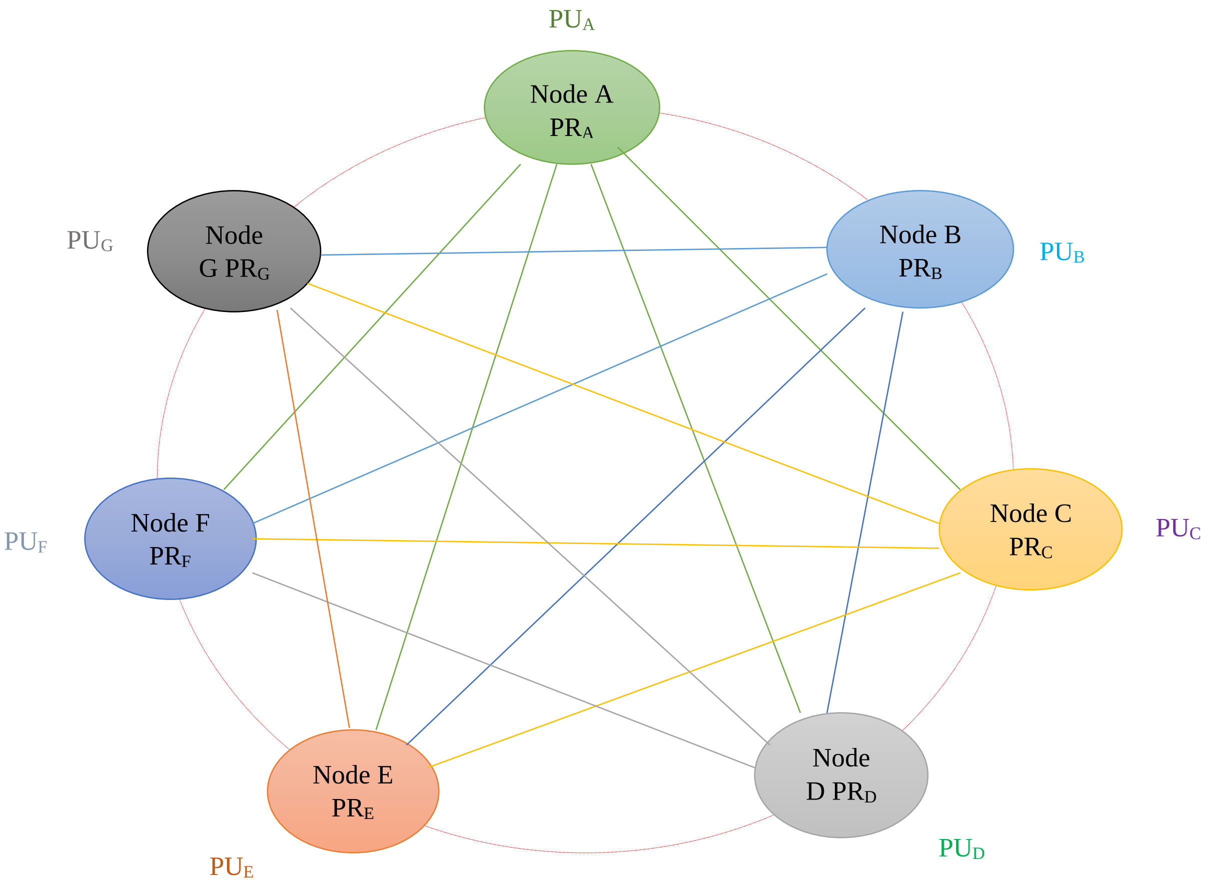


Figure: Cryptosystem of seven nodes Full connection network topology

Keys ➔ PU_x are public keys , PR_x are privet keys

The RSA algorithm is the most widely used Asymmetric Encryption algorithm deployed to date.

The acronym is derived from the last names of the three mathematicians who created it in 1977: Ron **R**ivest, Adi **S**hamir, Leonard **A**dleman.

In order to understand the algorithm, there are a few terms we have to define:

- Simplify the term e^X mod n into sub terms ➔ (e^{x1} mod n)* (e^{x2} mod n)*….* (e^{xm} mod n)
- **Prime** – A number is said to be Prime if it is only divisible by 1 and itself. Such as: 2, 3, 5, 7, 11, 13, etc.
- **Factor** – A factor is a number you can multiple to get another number. For example, the factors of 12 are 1, 2, 3, 4, 6, and 12.
- **Semi-Prime** – A number is Semi Prime if its only factors are prime (excluding 1 and itself). For example:
  12 is *not* semi-prime — one of its factors is 6, which is not prime.
  21 *is* semi-prime — the factors of 21 are 1, **3**, **7**, 21. If we exclude 1 and 21, we are left with 3 and 7, both of which are Prime.
  (Hint: Anytime you multiply two Prime numbers, the result is always Semi Prime)
- **Modulos** – This is a fancy way of simply asking for a remainder. If presented with the problem 12 MOD 5, we simply are asking for the remainder when dividing 12 by 5, which results in 2.

With that out of the way, we can get into the algorithm itself.

RSA Key Generation

The heart(principle) of Asymmetric Encryption lies(يكمن) in finding two mathematically linked values which can serve as our Public and Private keys. As such, the bulk of the work lies in the generation of such keys.

To acquire such keys, there are five steps:

1. Select two Prime Numbers: P and Q

This really is as easy as it sounds. Select two prime numbers to begin the key generation. For the purpose of our example, we will use the numbers **7** and **19**, and we will refer to them as **P** and **Q**.

2. Calculate the Product: (P*Q)

We then simply multiply our two prime numbers together to calculate the product:

7 * 19 = **133**

We will refer to this number as **N**. Bonus question: given the terminology we reviewed above, what kind of number is N?

3. Calculate the Totient of N: (P-1)*(Q-1) (Euler's function)

There is a lot of clever math that goes into both defining and acquiring a Totient. Most of which will be beyond the intended scope of this article. So, for now, we will simply accept that the formula to attain the Totient on a Semi Prime number is to calculate the product of one subtracted from each of its two prime factors. Or more simply stated, to calculate the Totient of a Semi-Prime number, calculate P-1 times Q-1.

Applied to our example, we would calculate:

(7-1)*(19-1) = 6 * 18 = **108**

We will refer to this as **T** moving forward.

4. Select a Public Key

The Public Key is a value which must match three requirements:

- It must be Prime
- It must be less than the Totient
- It must NOT be a factor of the Totient

Let us see if we can get by with the number 3: 3 is indeed Prime, 3 is indeed less than 108, but regrettably 3 is a factor of 108, so we cannot use it. Can you find another number that would work? Here is a hint, there are multiple values that would satisfy all three requirements.

For the sake of our example, we will select **29** as our **Public Key**, and we will refer to it as **E** going forward.

5. Select a Private Key

Finally, with what we have calculated so far, we can select our Private Key (which we will call **D**). The Private Key only has to match one requirement: The Product of the Public Key and the Private Key when divided by the Totient, must result in a remainder of 1. Or, to put it simply, the following formula must be true:

(D*E) MOD T = 1 ➔ $$d = \frac{(\text{Ø}(n)i + 1)}{e}$$

There are a few values that would work for the Private Key as well. But again, for the sake of our example, we will select **41**. To test it against our formula, we could calculate:

(41*29) MOD 108

We can use a calculator to validate the result is indeed 1. Which means **41** will work as our Private Key.

And there you have it, we walked through each of these five steps and ended up with the following values:

P
Q
N
T
E
D
7
19
133
108
29
41

Now we simply pick a value to be used as our Plaintext message, and we can see if Asymmetric encryption really works the way they say it does.

For our example, we will go ahead and use **99** as our Plaintext message.

(the math gets pretty large at this point, if you are attempting to follow along, I suggest to use the Linux Bash Calculator utility)

Message Encryption

Using the keys we generated in the example above, we run through the Encryption process. Recall, that with Asymmetric Encryption, we are encrypting with the Public Key, and decrypting with the Private Key.

The formula to Encrypt with RSA keys is: Cipher Text = **M^E MOD N**

If we plug that into a calculator, we get:

$99^{\wedge 29}$ MOD 133 = **92**

The result of **92** is our Cipher Text. This is the value that would get sent across the wire, which only the owner of the correlating Private Key would be able to decrypt and extract the original message. Our key pair was 29 (public) and 41 (private). So, let's see if we really can extract the original message, using our Private Key:

The formula to Decrypt with RSA keys is: **Original Message = M^D MOD N**

If we plug that into a calculator, we get:

$92^{\wedge 41}$ MOD 133 = **99**

As an experiment, go ahead and try plugging in the Public Key (29) into the Decryption formula and see if that gets you anything useful. You'll notice that, as was stated before, it is impossible to use the same key to both encrypt and decrypt.

Message Signing

But remember, that isn’t all we can do with a key pair. We can also use same key pair in the opposite order in order to verify a message’s signature.

To do this, we will use the same formulas as above, except this time we will revere the use of the Public and Private Key. We’re going to encrypt with the Private Key and see if we can decrypt with the Public Key.

We’ll use the same formula to encrypt, except this time we will use the Private Key:

Signature = **M^D MOD N**

If we plug that into a calculator, we get:

99^41 MOD 133 = **36**

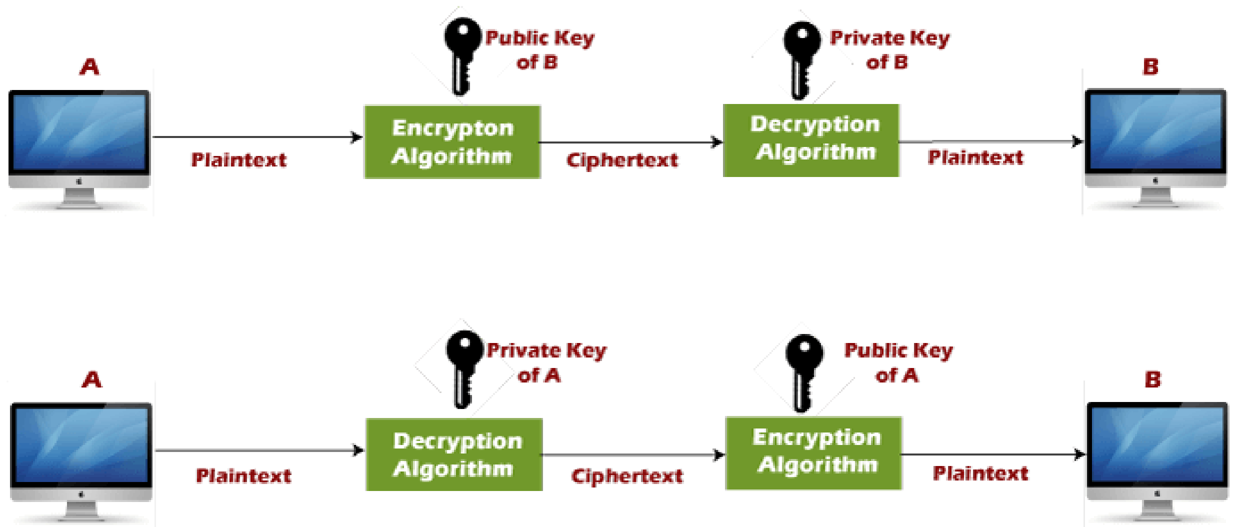
The result of **36** is the Signature of the message. If we can use the correlating public key to decrypt this and extract the original message, then we know that only whoever had the original Private Key could have generated a signature of 36.

Again, the same Decryption formula, except this time we will use the Public Key:

Original **M**essage = **M^E MOD N**

If we plug that into a calculator, we get:

36^29 MOD 133 = **99**



Encryption/decryption using public/private keys

And there you have it. A fully working example of RSA's Key generation, Encryption, and Signing capabilities.

Example1

- Choose p = 3 and q = 11
- Compute n = p * q = 3 * 11 = 33
- Compute φ(n) = (p - 1) * (q - 1) = 2 * 10 = 20
- Choose e such that 1 < e < φ(n) and e and φ (n) are coprime. Let e = 7
- Compute a value for d such that (d * e) Mod φ(n) = 1. One solution is d = 3 [(3 * 7) Mod 20 = 1]
- Public key is (e, n) => (7, 33)
- Private key is (d, n) => (3, 33)
- The encryption of $m = 2$ is $c = 2^7 \ Mod\ 33 = 29$
- The decryption of $c = 29$ is $m = 29^3 \ Mod\ 33 = 2$

Example2

Suppose P = 53 and Q = 59.

n = P*Q = 3127.

We also need a small exponent say e :

1 < e < Φ(n) [Φ(n) is discussed below],

Let us now consider it to be equal to 3.

Our Public Key is made of n and e

Generating Private Key:

Φ(n) = (P-1) *(Q-1)

so, Φ(n) = 3016

Now calculate Private Key, d:

d = (k*Φ(n) + 1) / e for some integer k

For k = 2, value of d is 2011.

Now we are ready with our – Public Key (n = 3127 and e = 3) and Private Key(d = 2011) Now we will encrypt "HI" :

Encryption

Convert letters to numbers: H = 8 and I = 9

Thus, Encrypted Data c = 89e mod n.

Thus, our Encrypted Data comes out to be 1394

Decryption

Now we will decrypt 1394:

Decrypted Data = cd mod n.

Thus, our Encrypted Data comes out to be 89

8 = H and I = 9 i.e. "HI".

An Example

Steps to generate **public key** (e, n) & **private key** (d, n)

1. First, select two prime numbers p=7 and q=11.
2. Now calculate n= p X q = 7 X 11

   n = 77
3. Calculate Ø(n)= Ø(pXq)

   = Ø(p) X Ø(q)

   = (p-1) X (q-1) ……. Ø (a) = (a-1) if **a** is a prime number.

   =(7-1) X (11-1)

   = 6 X 10

   Ø(n) = 60
4. Select e such that **1 ≤ e < Ø(n)** and also ‘e’ should be **coprime** to Ø(n).

   So, I select **e=7.**

   Our **Public Key** for this particular example is **(7,77)**.

5. Now we will determine the value of **d**. The value of d can be calculated from the formula given below:

$$ed = 1\ mod Ø(n)$$

In the expression above we know that and e and Ø(n) are the coprime numbers so in this case d is the multiplicative inverse of e. To calculate the value of d use the formula below:

$$d = \frac{(\text{Ø}(n)i + 1)}{e}$$

In this equation above we know the value of Ø(n), e, the value of i is unknown. First, we have to put the value of i=1.

$$d = \frac{(60 * i + 1)}{7}$$

If the result is in decimals then we have to compute the equation again but this time we have to increment the value of i by 1 so we will compute the equation with i=2. Keep on incrementing the value of i till the above equation results in a proper integer.

So, by trial and error method, for i=5 we get the result 43 i.e.

$$d = \frac{(60 * 5 + 1)}{7}$$

$$d = 43$$

Now we have generated both the private and public key.

Private Key (43, 77)

Public Key (7, 77)

RSA Encryption

Now, after generating the private and public key we will now encrypt the message. In RSA the plain text is always encrypted in **blocks.** The **binary value** of each plain text block should be **<n**.

Encryption is done with the intended receiver's **public key**. The expression to calculate cipher text is as follow:

$$\mathbf{C= M^{e} \bmod n}$$

In our example, the value of e=7 and n=77 i.e. public key (e, n) and we have to take the value of M such that **M<n**. We will take the value of M=15. So, the expression becomes

$$\mathbf{C= 15^{7} \bmod 77}$$

$$\mathbf{C= [\ (15^{4} \bmod 77)*(15^{2} \bmod 77)*}$$
$$\mathbf{(\ 15^{1} \bmod 77)\ ]mod77}$$

$$\mathbf{C= [(36)*(71)*(15)]\ mod77}$$

C=71 ……… Cipher Text

RSA Decryption

Done with the encryption now its time to decrypt the message. For decryption in RSA, we require a cipher text and the private key of the corresponding public key used in encryption.

In our example the cipher text we have M'=71 and the private key we have (43, 77). The expression to calculate plain text is as follow:

$$\mathbf{M= C^{d} \bmod n}$$

$$\mathbf{M= 71^{43} \bmod 77}$$

$$\mathbf{M= 15}$$

So, this is the method to encrypt and decrypt the message in RSA. It is very important to remember that in RSA we have to encrypt the message using the intended receiver's public key.

So, the message can only be decrypted by the intended receiver private key. This provides **confidentiality** to our message.

An Example

Perform encryption and decryption using RSA algorithm for p =11, q = 13, e = 7, m = 9.

$$Step\ 1: \quad p = 11, q = 13$$

$$Step\ 2: \quad n = p \times x = 11 \times 13 = 143$$

$$Step\ 3: \quad Calculate$$

$$\varphi(n) = (p-1)(q-1)$$

$$= (11-1)(13-1) = 10 \times 12 = 120$$

$$Step\ 4: \quad Determine\ d\ such\ that\ de \equiv 1(mod\ 160)$$

$$d = e^{-1} mod\ 160$$

Using extended Euclidean algorithm we calculate d

$$= -17\ mod\ 120$$

$$d = 103$$

$$Public\ key = \{7, 143\}$$

$$Private\ key = \{103, 143\}$$

$$Encryption\ (C) = M^e\ (mod\ n)$$

$$M = 9$$

$$C = 9^7 mod\ 143$$

$$= [(9^4\ mod\ 143\ ) \times (9^2\ mod\ 143)(9^1\ mod\ 143)]mod\ 143$$

$$= (126 \times 81 \times 9)mod\ 143$$

$$= 91854\ mod\ 143$$

$$= 48$$

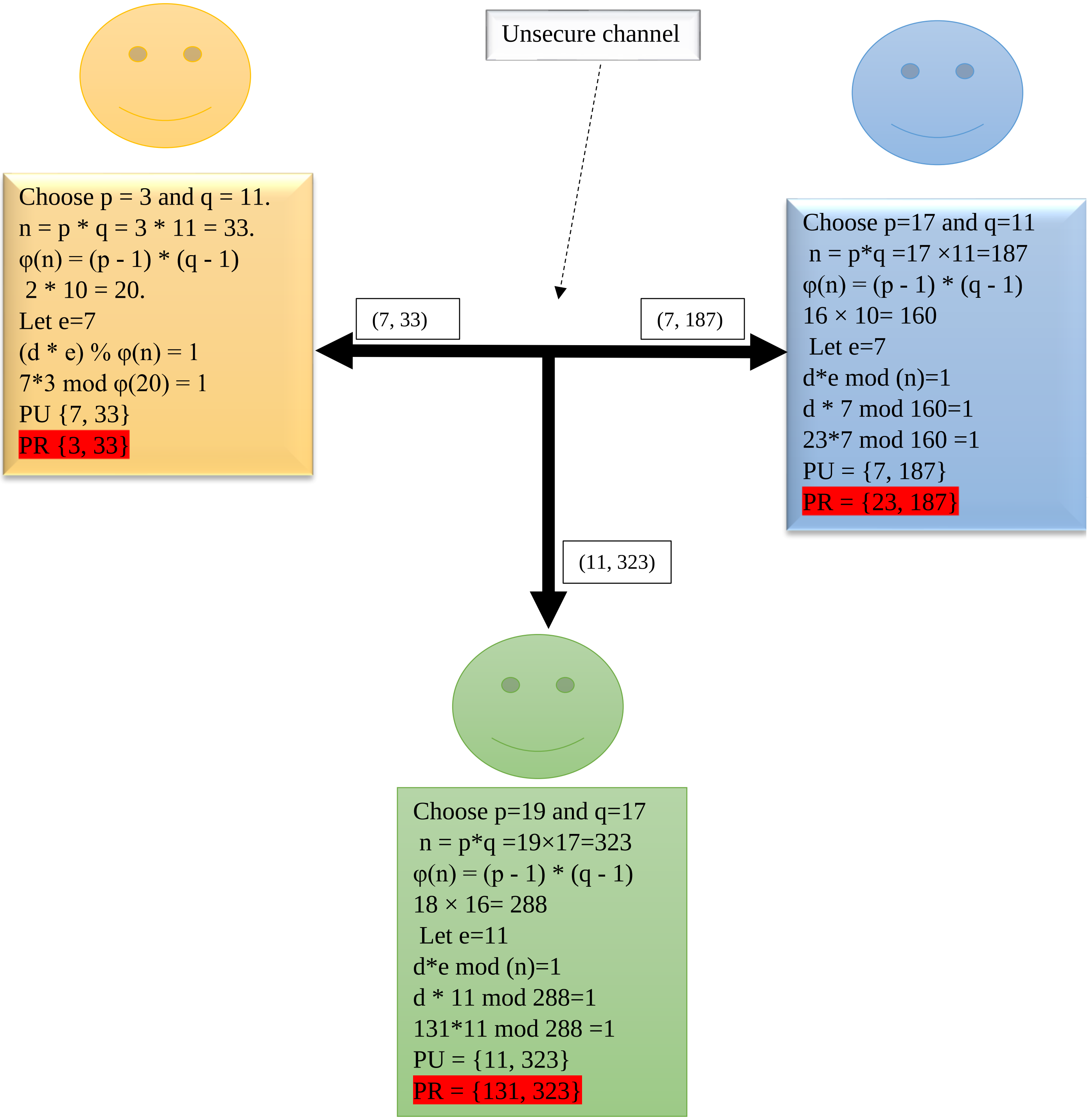
$$Decryption\ (M) = 13^{103} mod\ 143$$

An Example

- p = 3 and q = 11➔ n = p * q = 3 * 11 = 33
- φ(n) = (p - 1) * (q - 1) = 2 * 10 = 20
- Choose e such that 1 < e < φ(n) and e and φ (n) are coprime. Let e = 7
- (d * e) mod φ(n) = 1. One solution is d = 3 [(3 * 7) mod 20 = 1]
- Public key is (e, n) => (7, 33)
- Private key is (d, n) => (3, 33)

The encryption of $m = 2$ is $c = 2^7\ mod\ 33 = 29$
The decryption of $c = 29$ is $m = 29^3\ mod\ 33 = 2$

Protected Network Example



Choose p = 3 and q = 11.
n = p * q = 3 * 11 = 33.
φ(n) = (p - 1) * (q - 1)
2 * 10 = 20.
Let e=7
(d * e) % φ(n) = 1
7*3 mod φ(20) = 1
PU {7, 33}
PR {3, 33}

Unsecure channel

(7, 33)

(7, 187)

Choose p=17 and q=11
n = p*q =17 ×11=187
φ(n) = (p - 1) * (q - 1)
16 × 10= 160
Let e=7
d*e mod (n)=1
d * 7 mod 160=1
23*7 mod 160 =1
PU = {7, 187}
PR = {23, 187}

(11, 323)

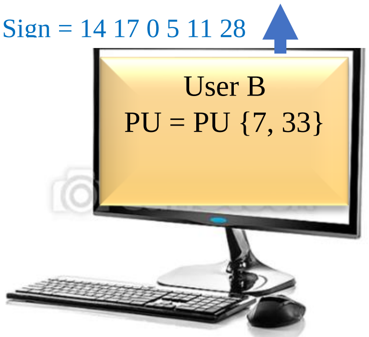
Choose p=19 and q=17
n = p*q =19×17=323
φ(n) = (p - 1) * (q - 1)
18 × 16= 288
Let e=11
d*e mod (n)=1
d * 11 mod 288=1
131*11 mod 288 =1
PU = {11, 323}
PR = {131, 323}

User B want to send a sign message to user A, the message is" THANKS".➔20 8 0 14 11 19

By using user B privet key:

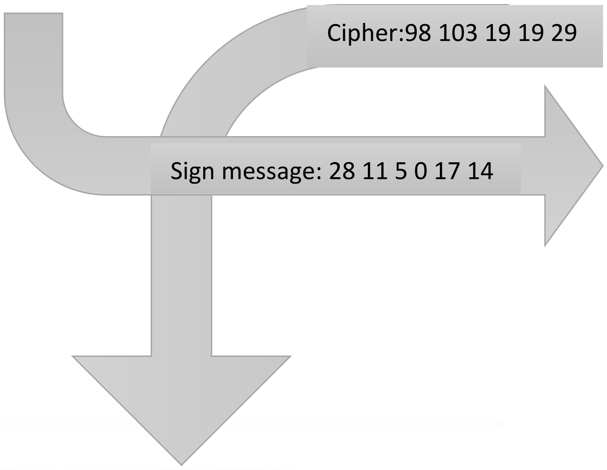
S1=20^3 mod 33=14, S2=8^3 mod 33=17, S3=0^3 mod 33=0, S4=14^3mod 33=5, S5=11^3 mod 33=11, S6=19^3 mod 33=28

Sign = 14 17 0 5 11 28



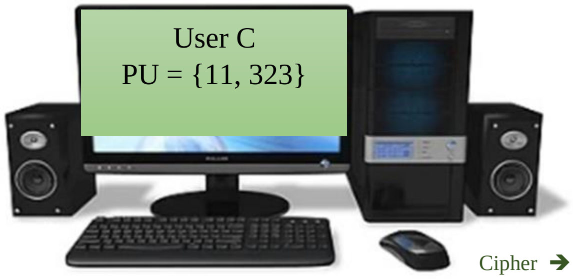
User B
PU = PU {7, 33}

PR {3, 33} secret



Cipher:98 103 19 19 29

Sign message: 28 11 5 0 17 14



User C
PU = {11, 323}

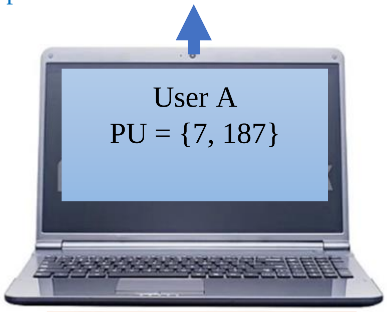
PR = {131, 323} secret

User A want to send a cipher text to user C, the message is" HELLO". HELLO ➔ 72 69 76 76 79, By using user C public key:

C1=72^{11} mod 323=98, C2=69^{11} mod 323=103, C3=76^{11} mod 323=19

C4=76^{11}mod 323=19, C5=79^{11} mod 323=29

Cipher = 98 103 19 19 29



User A
PU = {7, 187}

PR = {23, 187} secret

Sign ➔ 14 17 0 5 11 28, By using user B public key:

m1=14^7 mod 33=20, m2=17^7 mod 33=8, m3=0^7 mod 33=0, m4=5^7 mod 33=14, m5=11^7 mod 33=11, m5=28^7 mod 33=19

Sign message = 20 8 0 14 11 19 =THANKS

Cipher ➔ 98 103 19 19 29, By using user C privet key:

m1=98^{131} mod 323=72, m2=103^{131} mod 323=69, m3=19^{131} mod 323=76, m4=19^{131}mod 323=76, m5=29^{131} mod 323=79

Plain text = 72 69 76 76 79 Message =HELLO

Sign with Encryption

Send from A to B: processes➔ encryption (1) then signing (2)

$$S_C = \underbrace{(P^{PU(B)} \bmod n_{(B)})}_{\text{ENCRYPTION}} \underbrace{{}^{PR(A)} \bmod n_{(A)}}_{\text{Signature}}$$

processes ➔ signing (1) then encryption (2)

$$M = \underbrace{(S_C{}^{PU(A)} \bmod n_{(A)})}_{\text{Signature Verification}} \underbrace{{}^{PR(B)} \bmod n_{(B)}}_{\text{DECRYPTION}}$$

Note:

It should be noted here that what you see above is what is regarded as “vanilla” RSA. In production use of RSA encryption, the numbers used are *significantly* larger. In fact, modern RSA best practice is to use a key size of 2048 bits. This correlates to the ***N*** value in our calculation above. The two primes used in modern RSA must result in a product that is 2048 bits.

And just to give you an idea of how big 2048-bit number is. We saw earlier that a 128-bit number can be written in decimal with 39 digits. A 2048-bit key is exponentially longer – it would require approximately 617 digits to fully write out.

Note2: In our examples, the digital values of encrypted of signed blocks message should be less than n value.

RSA Advantages and Disadvantages

Advantages:

- **Convenience:** It solves the problem of distributing the key for encryption.
- **Provides message authentication:** Public key encryption allows the use of digital signatures which enables the recipient of a message to verify that the message is from a particular sender.
- **Detection of tampering:** The use of digital signatures in public key encryption allows the receiver to detect if the message was altered in transit. A digitally signed message cannot be modified without invalidating the signature.
- **Provides non-repudiation:** Digitally signing a message is related to physically signing a document. It is an acknowledgement of the message and thus, the sender cannot deny it.

Disadvantages:

- **Public keys should/must be authenticated:** No one can be absolutely sure that a public key belongs to the person it specifies and so everyone must verify that their public keys belong to them.
- **Slow:** Public key encryption is slow compared to symmetric encryption Not feasible for use in decrypting bulk messages.
- **Uses more computer resources:** It requires a lot more computer supplies compared to single-key encryption.
- **Widespread security compromise is possible:** If an attacker determines a person’s private key, his or her entire messages can be read.
- **Loss of private key may be irreparable:** The loss of a private key means that all received messages cannot be decrypted.

How to solve Two Symmetric cryptography problems?

The two problems of symmetric key cryptography i.e. confidentiality and authentication can be overcome by the double use of public key cryptography.

1. **First**, encrypt the message by the **sender's private key** which can be decrypted by the sender's public key(known to all). This provides a digital signature to the sender's message and thus **authentication** is achieved.

$$\mathbf{E(PR_s, M)}$$

2. In the **next step**, encrypt again with the **receiver's public key**. This will allow only the intended receiver to decrypt the message, this provides the **confidentiality** to the message.

$$\mathbf{C= E(PU_r, E(PR_s, M)}$$

The Decryption is shown by the following expression: $\mathbf{M= D(PU_s, E(PR_r,}$

$$\mathbf{C)}$$

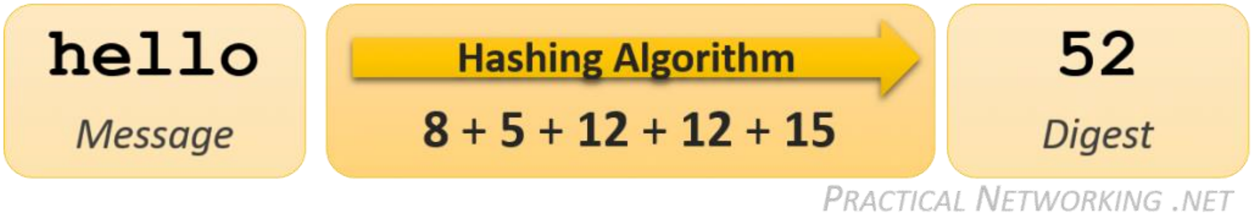
BLOCK Ciphers: Symmetric Encryption

Hashing Algorithm

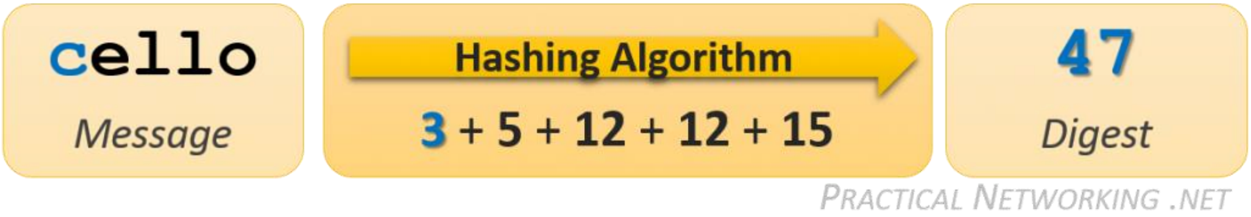
The first concept we need to discuss in our exploration of Cryptography is that of a **Hashing Algorithm**.

A Hashing Algorithm is a mathematical **formula that takes a Message of arbitrary length as input and produces as output a representational sample** of the original data.

For instance, a rudimentary example of a hashing algorithm is simply adding up all the letter values of a particular message. (A=1, B=2, C=3, etc…):



The result of a hashing algorithm is called a message **Digest** (or sometimes Checksum, or Fingerprint). The result of our example hashing on the original message of **hello** was **52**. If someone were to change our original message and process it through the same hashing algorithm, the result would be different:



By comparing the message digests of each calculation, it is easy to determine that our message has changed.

Obviously, the Hashing algorithm used in the example is full of flaws. There are many words that when processed through the example algorithm that might result in the same Digest. Had the original Message been changed to *cellt*, the resulting digest would still be 52, and we would be unaware that the original Message had been altered.

In reality, a legitimate hashing algorithm must maintain four qualities before it is approved for industry usage:

1. It is mathematically impossible to extract the original message from the digest.

It should be impossible to reverse the hashing algorithm and recover the original Message knowing just the resulting Digest. In fact, Hashing is sometimes referred to as *one-way encryption*: the message can be encrypted but is impossible to decrypt. This is accomplished using one-way functions within the hashing algorithm.

In a way, our example Hashing algorithm satisfied this condition. It is impossible to derive *hello* knowing only a resulting digest of *52*. Mostly because there could be thousands of messages that result in the identical digest.

2. A slight change to the original message causes a drastic change in the resulting digest.

Any minor modification – even as small as changing a single character – to the original Message should greatly alter the computed digest. This is sometimes referred to as the Avalanche effect.

It is possible because a hashing algorithm is not simply one calculation. It is a series of calculations, done iteratively over and over. As a result, a small change in the beginning, creates an exponentially bigger and bigger change in the resulting digest. Just like a snowball tumbling down a mountain forming an avalanche.

3. The result of the hashing algorithm is always the same length.

It is vital for the resulting Digest to not provide any hints or clues about the original Message – including its length. A digest should not grow in size as the length of the Message increases.

In our example Hashing algorithm, the longer the word, the bigger the resulting digest would be as we are adding more and more letters together. However, in an industry approved hashing algorithm, hashing the word *hello* would produce a digest the same size as hashing the entire library of congress.

4. It is infeasible to construct a message which generates a given digest.

With our example hashing algorithm, if given the digest of *52* , it would not be overly difficult to generate a list of words that might have been the original message. This is what this attribute is trying to prevent.

In a proper hashing algorithm, this should be infeasible — short of attempting every possible combination of messages until you found a match (aka, brute-forcing the algorithm). But even this becomes infeasible given a large enough digest size.

In the next article in this series, we will look at exactly *how* Hashing Algorithms are used to detect modified messages. But for now, we will continue to look at additional aspects of Hashing Algorithms.

Digest Lengths

Below is a table with commonly seen, industry recognized hashing algorithms:

Algorithm	Digest Length
MD5	128 Bits
SHA or SHA1	160 Bits

Algorithm	Digest Length
SHA384	384 Bits
SHA256	256 Bits

Each of these Hashing algorithms satisfy the four cryptography hashing algorithm properties, as described above. The primary difference between each of them is the size of the resulting digest.

As with passwords, it is typically considered that a hashing algorithm which results in a longer digest tends to be regarded as more secure.

Cryptology requirements: Mathematical Basic concepts

A. Introduction

In this lecture, we will introduce to the basic concepts in different fields in mathematics, which the cryptology fields are needed; this lecture is a collection of basic material on probability theory, number theory, abstract algebra, and finite fields that will be used throughout in our information security lectures. The following standard notation will be used throughout:

1. $\mathbb{Z}$ denotes the set of integers; that is, the set $\{\ldots,-2,-1,0,1,2,\ldots\}$.

2. $\mathbb{Q}$ denotes the set of rational numbers; that is, the set $\{\frac{a}{b} | a,b \in \mathbf{Z}, b \neq 0\}$

3. $\mathbb{R}$ denotes the set of real numbers.

4. $[a, b]$ denotes the integers x satisfying $a \leq x \leq b$.

5. $a \in A$ means that element a is a member of the set A.

6. $A \subseteq B$ means that A is a subset of B.

7. $A \subset B$ means that A is a proper subset of B; that is $A \subseteq B$ and $A \neq B$.

8. The intersection of sets A and B is the set $A \cap B = \{x | x \in A \text{ and } x \in B\}$.

9. The union of sets A and B is the set $A \cup B = \{x | x \in A \text{ or } x \in B\}$.

10. The difference of sets A and B is the set $A - B = \{x | x \in A \text{ and } x \notin B\}$.

11. The Cartesian product of sets A and B is the set $A \times B = \{(a,b) | a \in A \text{ and } b \in B\}$. For example, $\{a_1,a_2\} \times \{b_1,b_2,b_3\} = \{(a_1,b_1),(a_1,b_2),(a_1,b_3),(a_2,b_1),(a_2,b_2),(a_2,b_3)\}$.

12. $\sum_{i=1}^{n} a_i$ denotes the sum $a_1+a_2+\ldots+a_n$.

13. $\prod_{i=1}^{n} a_i$ denotes the product $a_1.a_2.\ldots.a_n$.

14. For a positive integer n, the factorial function is $n!=n(n-1)(n-2)\ldots1$. By convention, $0! = 1$.

B. Number Theory

Number theory, in mathematics, is primarily the theory of the properties of integers (whole numbers) such as parity, **divisibility**, **primality**, **additivity**, and **multiplicativity**, etc. In the next subsections we will investigate more detailed discussions about numbers.

- Primality

Definition : A positive integer n>1 that has only two distinct factors, 1 and n itself (when these are different), is called ***prime***; otherwise, it is called **composite**. The first few prime numbers are: 2,3,5,7,11,13,17,….

Note

1. It is interesting to note that primes thin out: there are eight up through 20, but only three between 80 and 100.

2. Note that 2 is the only even prime, all the rest are odd.

Note

Sieve of Eratosthenes is a method is found to specify the prime numbers. This method depends on cancelling all multiples of 2,3,5,7,…within the specified range, for example if we want to know all primes between 0 and 99:

0	1	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

The primes are (shaded numbers): 2,3,5,7,11,13,…,83,89,97.

- Multiplicativity

Theorem : (**the fundamental theorem of arithmetic**)

Any positive integer n>1 can be written uniquely in the following prime factorization form:

$$n=p_1^{\alpha_1}p_2^{\alpha_2}\cdots p_k^{\alpha_k}=\prod_{i=1}^{k}p_i^{\alpha_i}$$

where $p_1<p_2<\ldots<p_k$ are primes, and $\alpha_1,\alpha_2,\ldots,\alpha_k$ are non negative integers.

Example : The following are prime factorization of n for n=1999, 2000, …, 2010.

1999	=	1999	,	2000	=	$2^4.5^3$	,	2001	=	3.23.29
2002	=	2.7.11.13	,	2003	=	2003	,	2004	=	$2^3.3.167$
2005	=	5.401	,	2006	=	2.17.59	,	2007	=	$3^2.223$
2008	=	$2^3.251$	,	2009	=	$7^2.41$	,	2010	=	2.3.5.67

- Divisibility

Definition: Let a and b be two integers, not both zero. The largest divisor d s.t. d|a and d|b is called the **greatest common divisor** (gcd) of a and b, which is denoted by gcd(a,b).

Definition: Let a and b be two integers, not both zero. d is a common multiple of a and b, the least common multiple (lcm) of a and b, is the **smallest common multiple**, which is denoted by lcm(a,b).

Definition: Integers a and b are called **relatively prime** if gcd(a,b)=1. we say that integers $n_1,n_2,\ldots n_k$ are relatively prime if, whenever $i \neq j$, we have $\gcd(n_i,n_j)=1$, $\forall$ i,j, $1\leq i,j \leq k$.

Theorem: Suppose a and b are two positive integers.

$$\text{If } a=\prod_{i=1}^{k} p_i^{\alpha_i} \text{ and } b=\prod_{i=1}^{k} p_i^{\beta_i} \text{ , then}$$

$$\gcd(a,b)=\prod_{i=1}^{k} p_i^{\varepsilon_i} \text{ , where } \varepsilon_i=\min(\alpha_i,\beta_i),\ \forall i,\ 1\leq i\leq k.$$

$$\mathrm{lcm}(a,b)=\prod_{i=1}^{k} p_i^{\delta_i} \text{ , where } \delta_i=\max(\alpha_i,\beta_i),\ \forall i,\ 1\leq i\leq k.$$

Example: Since the prime factorization of 240 and 560 are:

$240=2^4.3.5$ and $560=2^4.5.7$, then the:

$$\gcd(240,560)=2^{\min(4,4)}.3^{\min(1,0)}.5^{\min(1,1)}.7^{\min(0,1)}=2^4.3^0.5^1.7^0=80.$$

$$\mathrm{lcm}(240,560)= 2^{\max(4,4)}.3^{\max(1,0)}.5^{\max(1,1)}.7^{\max(0,1)}=2^4.3^1.5^1.7^1=1680.$$

Theorem: Suppose a and b are two positive integers, then

$$\mathrm{lcm}(a,b)=\frac{a.b}{\gcd(a,b)}.$$

Euclidean Algorithm

The Euclidean algorithm is an efficient algorithm for computing the greatest common divisor of two integers that does not require the factorization of the integers. It is based on the following simple fact.

Fact: If a and b are positive integers with a>b, then:

gcd(a,b)=gcd(b, a mod b).

The Euclidean algorithm steps are: computing the gcd of two integers:

```
INPUT: two non-negative integers a and b with $a \geq b$.
OUTPUT: the gcd of a and b.
 WHILE $b \neq 0$ DO the following:
      Set $r \leftarrow a \bmod b$, $a \leftarrow b$, $b \leftarrow r$.
 RETURN(a).
```

Example (Euclidean algorithm):for computing gcd(4864,3458)=38

4864 = 1 . 3458 + 1406

3458 = 2 . 1406 + 646

1406 = 2 . 646 + 114

646 = 5 . 114 + 76

114 = 1 . 76 + 38

76 = 2 . 38 + 0.

The integers modulo n

Let n be a positive integer.

Definition : If a and b are integers, then a is said to be congruent to b modulo n, written: $a \equiv b \pmod{n}$, if n divides $(a-b)$. The integer n is called the modulus of the congruence.

Example

i. $24 \equiv 9 \pmod{5}$ since $24 - 9 = 3 . 5$.

ii. $-11 \equiv 17 \pmod 7$ since $-11 - 17 = -4 . 7$

Fact: (***properties of congruence's***) $\forall$ a,a1,b,b1,c $\in$ Z, the following are true.

i. $a \equiv b \pmod n$ if and only if a and b leave the same remainder when divided by n.

ii. (***reflexivity***) $a \equiv a \pmod n$.

iii. (***symmetry***) If $a \equiv b \pmod n$ then $b \equiv a \pmod n$.

iv. (***transitivity***) If $a \equiv b \pmod n$ and $b \equiv c \pmod n$, then $a \equiv c \pmod n$.

v. If $a\equiv a_1 \pmod n$ and $b\equiv b_1 \pmod n$, then $a+b\equiv a_1+b_1 \pmod n$ and $ab \equiv a_1b_1 \pmod n$.

The equivalence class of an integer a is the set of all integers congruent to a modulo n. From properties (ii), (iii), and (iv) above, it can be seen that for a fixed n the relation of congruence modulo n partitions Z into equivalence classes. Now, if $a = qn + r$, where $0\le r<n$, then $a \equiv r \pmod n$. Hence each integer a is congruent modulo n to a unique integer between 0 and n−1, called the least residue of a modulo n. Thus a and r are in the same equivalence class, and so r may simply be used to represent this equivalence class.

Definition: The integers modulo n, denoted Z_n, is the set of (equivalence classes of) integers{0,1,2,...,n−1}. Addition, subtraction, and multiplication in Zn are performed modulo n.

Example: $Z_{25} = \{0,1,2,...,24\}$.

In Z_{25}, 13+16=4, since $13+16=29\equiv 4 \pmod{25}$. Similarly, 13 . 16 = 8 in Z_{25}.

C- Arithmetic Functions

Arithmetic (or number theoretic) functions are the most fundamental functions in mathematics and computer science; for example, the computable functions studied in mathematical logic and computer science are actually arithmetic functions. In this section we shall study some basic arithmetic functions that are useful in number theory.

Definition: A **function** *f* is a rule that assigns to each element in a set D (called **Domain** of *f*) one and only one element in a set B. the set of images called the **range** (R) of *f* .

Definition:

1. The function *f* has the property of being "**one-to-one**" (or "**injective**") if no two elements in D are mapped into the same element in R.

2. The function *f* has the property of being "**onto**" (or "**surjective**") if the range R of *f* is all of B (R=B).

Definition: Given functions *f* and *g*, the **composition** of *f* with *g*, denoted by $f \circ g$ is the function by:

$(f \circ g)(x)=f(g(x))$

The domain of $f \circ g$ is defined to consists of all x in the domain of *g* for which *g*(x) is in the domain of *f*.

Definition: A function *f* is called an **arithmetic function** or a **number theoretic** function if it assigns to each positive integer n a unique real or complex number *f*(n). Typically, an arithmetic function is a real-valued function whose domain is the set of positive integer.

Example: the equation$\sqrt{n}$, n ϵ N, defines an arithmetic function *f* which assigns the real number n to each positive integer.

Definition: A real function defined on the positive integers is said to be **multiplicative** if:

$f(m)f(n)=f(mn)$, $\forall$ m,n ϵ N with gcd(m,n)=1.

Definition: Let n be a positive integer. **Euler's** (**totient**) Φ-function, Φ(n) defined to be the number of positive integer k less than n which are relatively prime to n:

$$\Phi(n)=\sum_{\substack{1\le k<n \\ \gcd(k,n)=1}} 1$$

Example: By definition above we have:

n	1	2	3	4	5	6	7	8	9	10	100	101	102	103
Φ(n)	1	1	2	2	4	2	6	4	6	4	40	100	32	102

Theorem: Let n be a positive integer, then

1. Φ(n) is multiplicative i.e. Φ(mn)= Φ(m) Φ(n).

2. if n is prime, say p, then Φ(p)=p-1, and if n is prime power p^{α}, then $\Phi(\ p^{\alpha})=p^{\alpha}-p^{\alpha-1}=p^{\alpha-1}(p-1)$.

3. if n is composite and has the standard prime factorization form, then

$$\Phi(n)=p_1^{\alpha_1-1}(p_1-1)\cdot p_2^{\alpha_2-1}(p_2-1)\cdots p_k^{\alpha_k-1}(p_k-1)=\prod_{i=1}^{k} p_i^{\alpha_i-1}(p_i-1).$$

4. Φ(n)=(p-1)(q-1) if n=pq, where p and q are prime numbers.

Definition: Let x be a positive real number ≥1, then $\pi(x)$ is defined as follows:

$$\pi(x)=\sum_{\substack{p\le x \\ p\ \text{prime}}} 1\ .$$

$\pi(x)$ is called the **prime counting** function (or the **prime distribution** function).

Example: π (1)=0, π (2)=2, π (10)=4, π (20)=8, π (30)=10, π (40)=12, π(50)=15, π (75)=21, π (100)=25.

D- Group Theory

Definition:

1. $Z_{>a}$ is the set of positive integers greater than a:

$Z_{>a}=\{a+1,a+2,\ldots\}$.

2. the set of all residue classes modulo a positive integer denoted by Z_n:

$Z_n=\{0,1,2,\ldots,n-1\}$.

Definition: A **binary operation** * on a set A is a rule that assign to each ordered pair (a,b) of elements of A a unique element of A.

Example: Ordinary addition + and multiplication • are binary operations on ℕ, ℤ, ℝ, or ℂ.

Definition: A ***group***, denoted by <G,*> (or (G,*)), or simply G, is a $G\neq\varphi$ of elements together with a binary operation *, s.t. the following axioms are satisfied:

1. ***Closure***: $a*b \in G, \forall a,b \in G$.

2. ***Associativity***: $(a*b)*c=a*(b*c), \forall a,b,c \in G$.

3. ***Existence of identity***: $\exists!$ element $e \in G$, called the identity, s.t. $e*a=a*e=a, \forall a \in G$.

4. ***Existence of inverse***: $\forall a \in G$, $\exists!$ Element $b \in G$, s.t. $a*b=b*a=e$. This b is denoted by a^{-1} and called the ***inverse*** of a.

The group <G,*> is called ***commutative*** (***abelian***) group if it satisfies further axiom:

5. ***Commutativity***: $a*b=b*a, \forall a,b \in G$.

Example: the set Z^+ with operation + is not group ($\exists$ no identity element), and it's not group with operation • ($\exists$ no inverse element in Z^+).

Definition:

1. If the binary operation of a group is +, then the identity of group is 0 and the inverse of a ϵ G is –a; this said to be an ***additive group***.

2. If the binary operation of a group is •, then the identity of a group is 1 or e, this group is said to be ***multiplicative group***.

Definition: A group is called a ***finite group*** if it has finite number of elements; otherwise it is called an ***infinite group***.

Definition: The ***order*** of the group G, denoted by |G| (or by #(G)) is the number of elements of G.

Example: the order of Z is |Z|=∞.

Definition: Let a ϵ G, where G is multiplicative group. The elements ar, where r is an integer, form a subgroup of G, called the ***subgroup*** generated by a. A group G is ***cyclic*** if ∃ a ϵ G s.t. the subgroup generated by a is the whole of G.

Note: If G is a finite cyclic group with identity element e, the set of elements G may be written $\{e,a,a^2,\ldots,a^{n-1}\}$, where an=e and n is the smallest such positive integer.

Definition: A ***field*** by <F, ⊕, ⊗> (or (F,⊕⊗)) or simply F, is abelian group w.r.t. addition, and F-{0} is abelian w.r.t. to multiplication.

Definition: A ***finite field*** is a field that has a finite number of elements in it; we call the number the order of the field.

Theorem: ∃ a field of order q iff q is ***prime power*** (i.e. $q=p^r$) with p prime and r ϵ N.

Note: A field of order q with q prime power is called ***Galois field*** and is denoted by GF(q) or just F_q.

Example: The finite field F5 has elements {0,1,2,3,4} and is described by the tables below are addition and multiplication tables.

Tables: The addition and multiplication for F_5.

⊕	**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

⊗	**1**	**2**	**3**	**4**
1	1	2	3	4
2	2	4	1	3
3	3	1	4	2
4	4	3	2	1

E- Boolean Ring and Boolean Algebra

Definition: Let $A \neq \varphi$ be a set, f be a binary operation on a set A ($f: AXA \rightarrow A$), we call the pair (A,f) as **mathematical system**.

Definition: Let X be the universal set, and let A and B be two subsets of X, then:

1. The operation + defined as A+b=A U B.

2. The operation ⊕defined on the power P(X) set of X by:

A⊕B=(A-B) U (B-A) s.t. A-B=A ∩B', B' is the ***complement*** set of B. The operation ⊕called ***Exclusive-OR*** (**XOR**) (or the ***symmetric difference***).

3. The operation • defined as A•B=A ∩ B.

Definition: Let (R,+,•) be a ring with identity element, if the **Idempotency law** be satisfied $a^2=a$, $\forall a \in R$, then the ring called **Boolean ring**.

Example: Let P(X) represents the set of all the subsets of the universal set X, then the ring (P(X), ⊕,•) is Boolean ring.

Definition: In Boolean ring (B, ⊕,•), we defined:

1. **Complement**: $\bar{a}=a\oplus 1,\ \forall a\in B.$

2. **Sum (OR)**: $a+b=a\oplus b\oplus a.b\ \forall a,b\in B.$

Definition: The ***Boolean algebra*** is the mathematical system $(B,\vee,\wedge)$ where $B\neq\varphi$, and the binary operations $\vee$ and $\wedge$ defined on B as follows:

1. The operations $\vee$ and $\wedge$ are commutative.

2. The operations $\vee$ and $\wedge$ are satisfying the distribution law for each to other.

3. ∃ two identity distinct elements 0 and 1 of the operations $\vee$ and $\wedge$ respectively s.t. $a\vee 0=a$ and $a\wedge 1=a$, $\forall a \in B$.

Example: The system $(P(X),U,\cap)$ is boolean algebra, $X\neq\varphi$, we use $\varphi=0$ and $X=1$. If B be a set of subsets of X including φ and X which is closed on U and complement then $(B,U,\cap)$ is Boolean algebra too.

Theorem: Every boolean algebra $(B,\vee,\wedge)$ is boolean ring $(B,\oplus,\bullet)$ when we defined the operations $\oplus$ and $\bullet$ as follows:

1. $a\oplus b=(a\wedge b')\vee(a'\wedge b)$.
2. $a\bullet b=a\wedge b$.

$\forall a,b\in B.$

Theorem: Every ring $(B,\oplus,\bullet)$ is Boolean algebra $(B,\vee,\wedge)$ when we defined $\vee$ and $\wedge$ as follows: $\forall a,b \in B$.

1. $a\vee b=a\oplus b\oplus a\bullet b.$
2. $a\wedge b=a\bullet b.$

Theorem: The ring $(Z_p,\oplus,\otimes)$ is field iff p is prime number s.t.

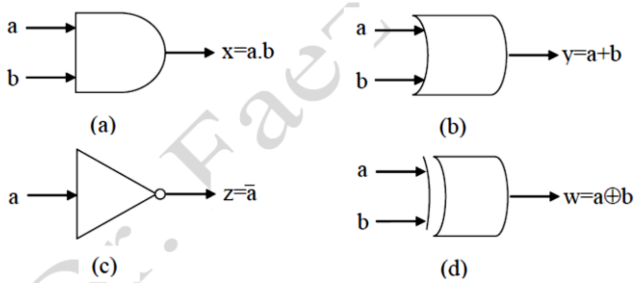
$a\oplus b=a + b \pmod p$.

$a\otimes b=a\bullet b \pmod p$.

This field is Galois field and is denoted by GF(p), ∀a,b ϵ Z_p.

F. Algebra Description of Logic Circuits

In electronically logical circuits (which are subject to the Boolean algebra), there are small circuits called Gates which are, for example, part from transistors, diodes, capacitors, and etc, these gates are shown in the figure below:



(a) (b) (c) (d)

(a).The gate AND: is multiplying the input variables.

(b).The gate OR: summing the input variables.

(c).The gate NOT: complement of the input variable.

(d).The gate XOR: summing XOR the input variables.

These gates are shown in the following table.

•	0	1
0	0	0
1	0	1

+	0	1
0	0	1
1	1	1

a	$\bar{a}$
0	1
1	0

⊕	0	1
0	0	1
1	1	0

Definition: The logical function f is called the **output function** defined f:B^n→B, where B^n is a set of n input binary data, f subject to the Boolean algebra laws and we can apply the gates concepts on it, s.t. $x=f{\bullet}g$, $y=f+g$, $z=\overline{f}$, and $w=f \oplus g$, where f and g are Boolean functions.

G. Sequences and Series

* Sequences

Definition: The **sequence** in the field F is a function *f*, whose domain is the set of non negative (or could be positive) integer, s.t. *f*:Z→F, and its denoted by S= $S=\{S_n\}_{n=0}^{+\infty}$.

Definition: The Sequence S is **periodic** when ∃p ϵ Z^+ s.t. $s_0=s_p, s_1=s_{p+1},\ldots$, the minimum p is the **period** of S.

If $Z_m=\{0,1,\ldots,m-1\}$, where m ϵ Z^+, then S is digital sequence. In special case, if m=2 then S is binary sequence.

- Series

Definition: An infinite series is an expression of the form:

$u_1+u_2+\ldots u_k+\ldots=\sum_{k=1}^{\infty} u_k$

Let Sn denotes the sum of the first n terms of the series s.t.

$S_n=\sum_{k=1}^{n} u_k$, and $\{S_n\}_{n=1}^{+\infty}$ is called the **sequence of partial sums**.

$S=\sum_{k=1}^{\infty} u_k$ is called the **sum** of the series.

Theorem: A geometric series $a+ar+ar^2+\ldots+ar^{k-1}+\ldots(a\neq 0)$ is converges if |r|<1 and the sum is $\frac{a}{1-r}=a+ar+ar^2+\ldots+ar^{k-1}+\ldots$, and diverges if |r|≥1.

H. Polynomials over Fields

Let $f(x) = a_n.x^n + a_{n-1}.x^{n-1} + a_{n-2}.x^{n-2} + \ldots + a_1.x + a_0$

be a polynomial of degree n in one variable x over a field F (namely a_n,

$a_{n-1},\ldots, a_1, a_0 \in F$).

Theorem: The equation f(x)=0 has at most n solutions in F.

Irreducible Polynomials

Definition: A polynomial is irreducible in GF(p) if it does not factor over GF(p). Otherwise it is reducible.

Example:

The polynomial $x^5+x^4+x^3+x+1$ is ***reducible*** in Z_5 but ***irreducible*** in Z_2.

- ## Implementing GF(pk) Arithmetic

Theorem: Let f(x) be an irreducible polynomial of degree k over Z_p.

The finite field $GF(p^k)$ can be realized as the set of degree k-1 polynomials over Z_p, with addition and multiplication done modulo f(x).

Example: **(Implementing $GF(2^k)$)**

By the theorem the finite field $GF(2^5)$ can be realized as the set of degree 4 polynomials over Z_2, with addition and multiplication done modulo the irreducible polynomial: $f(x)=x^5+x^4+x^3+x+1$.

The coefficients of polynomials over Z_2 are 0 or 1.

So a degree k polynomial can be written down by k+1 bits.

For example, with k=4:

x^3+x+1 (0,1,0,1,1)

$x^4+ x^3+x+1$ (1,1,0,1,1).

- ## Implementing $GF(2^k)$

Addition: bit-wise XOR (since 1+1=0)

x^3+x+1 (0,1,0,1,1)

+

x^4+x^3+x+1 (1,1,0,1,1)

----

x^4 (1,0,0,0,0)

Multiplication: $(x^2+x+1).(x^3+x+1)$ in $GF(2^5)$.

(1,1,1).(1,0,1,1)

1 0 1 1

  1 0 1 1

    1 0 1 1

----

1 1 0 0 0 1 = x^5+x^4+1

The Number of Primitive Polynomials

The function μ : $Z^+ \rightarrow Z^+$ defined by:

$$\mu(n)=\begin{cases}1 \text{ if } n=1;\\ (-1)^r \text{ if } n=p_1p_2\ldots p_r, \text{ where the } p_i \text{ are distinct primes;}\\ 0 \text{ if } n \text{ has a squared factor}\end{cases}$$

is called the ***Möbius Function***.

The number of monic irreducible polynomials of degree k over F_q is given by:

$$\psi_q(k)=\frac{1}{k}\sum_{d|k}\mu(\frac{k}{d})q^d$$

where this sum is over all positive divisors d of k.

Clearly, not every monic irreducible polynomial in $F_q[x]$ is necessarily a primitive polynomial over F_q. In fact, the number of primitive polynomials of degree k over F_q is: $\lambda_q(k) = \frac{\phi(q^k - 1)}{k}$

Example: Consider (monic) irreducible polynomials of degree 8 over $F_2=Z_2$. The positive divisors of 8 are d = 1, 2, 4, 8 so that 8/d = 8, 4, 2, 1 and $\mu(8/d)= 0, 0, -1, 1$.

Therefore, the number of monic irreducible polynomials of degree 8 in $F_2[x]$ is: $\psi_2(8) = \frac{1}{8}\sum_{d|8}\mu(\frac{8}{d})2^d = (0 + 0 - 16 + 256)/8= 30.$

Furthermore, the number of primitive polynomials of degree 8 in $F_2[x]$ is:

$$\lambda_2(8) = \frac{\phi(2^8 - 1)}{8} = \frac{\phi(255)}{8} = \frac{\phi(3.5.17)}{8} = \frac{2.4.16}{8} = 16.$$

Hence, just over half the irreducible polynomials of degree 8 in $Z_2[x]$ are primitive.

However, if $2^k - 1$ is prime then $\psi_2(k) = \lambda_2(k) =(2^k - 2)/k$ so that every irreducible polynomial of degree k is in fact a primitive polynomial in $Z_2[x]$. It is therefore beneficial, in the practical sense, to choose a reasonably large value of k such that $2^k - 1$ is prime.

Of course, if we have a prime p>2 then $p^k - 1$ is always even, and hence not a prime (excluding the trivial case: $3^1 - 1$ is prime). Thus, for prime's p>2, the number of primitive polynomial of degree k in $F_p[x]$ will always be less than the number of irreducible polynomials of degree k over F_p, with the exception of the above trivial case.

Consequently, determining a maximal period length shift register generator presents no special problem in comparison to a linear recurrence generator modulo p. We simply choose k such that Mk is prime so that every irreducible polynomial over Z_2 is a primitive polynomial. Then taking any such polynomial as the characteristic polynomial for the shift register generator will yield maximal period length sequences.

I.Probability Theory

Definition: An ***experiment*** is a procedure that yields one of a given set of outcomes. The individual possible outcomes are called ***simple events***. The set of all possible outcomes is called the ***sample space***.

we only considers discrete sample spaces; that is, sample spaces with only finitely many possible outcomes. Let the simple events of a sample space S be labeled $s_1,s_2,...,s_n$.

Definition: A ***probability distribution*** P on S is a sequence of numbers $p_1,p_2,...,\ p_n$ that are all non-negative and sum to 1. The number p_i is interpreted as the probability of s_i being the outcome of the experiment.

Definition: An ***event*** E is a subset of the sample space S. The probability that event E occurs, denoted P(E), is the sum of the probabilities pi of all simple events si which belong to E. If $s_i \epsilon S$, P({si}) is simply denoted by $P(s_i)$.

Definition: If E is an event, the ***complementary event*** is the set of simple events not belonging to E, denoted $\bar{E}$.

Fact: Let $E \subseteq S$ be an event.

i. $0 \leq P(E) \leq 1$. Furthermore, P(S) = 1 and $P(\varphi) = 0$. (φ is the empty set).

ii. P(Ĕ) = 1 − P(E).

iii. If the outcomes in S are equally likely, then P(E) = |E|/|S|.

Definition: Two events E1 and E2 are called mutually exclusive if $P(E_1 \cap E_2)=0$. That is, the occurrence of one of the two events excludes the possibility that the other occurs.

Fact: Let E1 and E2 be two events:

i. If $E_1 \subseteq E_2$, then $P(E_1) \leq P(E_2)$.

ii. $P(E_1 \cup E_2)+P(E_1 \cap E_2)=P(E_1)+P(E_2)$. Hence, if E_1 and E_2 are mutually exclusive, then $P(E_1 \cup E_2)=P(E_1) + P(E_2)$.

J. Linear Equations Systems and Matrices

* Linear Equations

Let F be field, let $a_1,a_2,\dots,a_n,b \in F$ and $x_1,x_2,\dots,x_n$ be variables (unknowns), then the combination equation:

$a_1x_1+a_2x_2+\dots+a_nx_n=b$

called ***Linear Equation***, $a_1,a_2,\dots,a_n$ are ***coefficients*** and b be the ***absolute value***.

A collection of linear equations is:

$a_{11}x_1+a_{12}x_2+\dots+a_{1n}x_n=b_1$

$a_{21}x_1+a_{22}x_2+\dots+a_{2n}x_n=b_2$

:

$a_{m1}x_1+a_{m2}x_2+\dots+a_{mn}x_n=b_m$

called ***m-system of linear equations*** with n variables.

If (b1=b2=…=bm=0), then the system called ***Homogeneous Linear Equations***, otherwise its called ***Non Homogeneous Linear Equations*****.**

The values $x_1, x_2, \ldots, x_n$ which are satisfied the system of linear equations is called ***solution***.

Example: For the following system:

$2x_1+3x_2+8x_3+x_4= 6$

$x_1+x_2+3x_3-x_4= 2$

$3x_1-4x_2+8x_3-x_4=5$

(-1,0,1,0) is a solution and (-3,6,-1,2) is another solution.

- Matrices

In example (10.1), the coefficients of the linear equations can be written as follows:

$$\begin{bmatrix} 2 & 3 & 8 & 1 \\ 1 & 1 & 3 & -1 \\ 3 & -4 & 8 & -1 \end{bmatrix}$$

This model called a ***Matrix***.

The matrix is rectangular arrangement with orthogonal rows and columns, it can be put in parentheses ( ) or [ ].

Every equation in the linear equations system is row in the matrix while, the column is same variable with different coefficient.

The general form of the matrix is:

$$A=\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

The matrix A consists of m rows with n columns so it can be denoted by $(a_{ij})_{mxn}$, or can be denoted by capital letters A,B,C…

Types of Matrices

The most important kinds of matrices are:

1. **Square matrix**: the matrix is called square matrix if m=n.

2. **Zero matrix**: It’s the matrix which all it elements are 0’s, and it denoted by O.

3. **Identity matrix**: its square matrix which all elements are 0’s, accept its 1’s on the main diagonal, and it denoted by I.

4. **Transpose Matrix**: change the ever row of the matrix to column, and it denoted by At.

5. **Triangular matrix**: the square matrix which all elements under the main diagonal are 0’s called up-triangular matrix, while its called down-triangular matrix which all elements above the main diagonal are 0’s.

6. **Diagonal matrix**: it’s the matrix which all elements under and above the main diagonal are 0’s.

Operations on Matrices

We will discuss three important operations on matrices. Two matrices are equal when they were from the same degree and the corresponding elements are equal.

Multiply by Scalar

Multiply the matrix by scalar done when all its elements are multiplied by the same scalar.

Example: If the scalar is 2 then:

$$2.\begin{bmatrix} 2 & 1 \\ -1 & 3 \\ 0 & -4 \end{bmatrix} = \begin{bmatrix} 4 & 2 \\ -2 & 6 \\ 0 & -8 \end{bmatrix}$$

Addition of Matrices

We can add only the matrices from the same degree, this operation done when adding the corresponding elements of the two matrices.

$(a_{ij})_{mxn} + (b_{ij})_{mxn} = (a_{ij}+b_{ij})_{mxn}$

Example:

$$\begin{bmatrix} 2 & 0 & 5 \\ -1 & 3 & -2 \end{bmatrix} + \begin{bmatrix} -1 & 4 & -3 \\ 0 & 1 & 4 \end{bmatrix} = \begin{bmatrix} 1 & 4 & 2 \\ -1 & 4 & 2 \end{bmatrix}$$

Matrices Multiplying

We can multiply two matrices if the number of columns of the first matrix equals the number of rows of the second matrix, then the degree of the result matrix is equal to row of the first matrix by the columns of the second matrix, the general form id:

$$(a_{ij})_{mxk} \times (b_{ij})_{kxn} = \left(\sum_{t=1}^{k} a_{it} \cdot b_{tj} \right)_{mxn}$$

(

Example:

$$\begin{bmatrix} 2 & 0 & 5 \\ -1 & 3 & -2 \end{bmatrix}_{2\times3} * \begin{bmatrix} 0 & -1 \\ 1 & 0 \\ 2 & 3 \end{bmatrix}_{3\times2} = \begin{bmatrix} 10 & 13 \\ -1 & -4 \end{bmatrix}_{2\times2}$$

- Determinants

It's a function with domain is the set of all square matrices with range is the field F. the value of this function is called the determinant of the matrix and its denoted by |A|. The calculation of the matrix determinant done with respect to its degree, which as follows:

1. 1X1 matrix: if A=[a], then |A|=a.

2. 2×2 matrix: if $A=\begin{bmatrix} a & b \\ c & d \end{bmatrix}$, then $|A|=a*d-c*b$.

3. 3×3 matrix: if $A=\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$, then:

$|A|=a_{11}*a_{22}*a_{33}+a_{12}*a_{23}*a_{31}+a_{13}*a_{21}*a_{32}-a_{13}*a_{22}*a_{31}-a_{11}*a_{23}*a_{32}+$

$a_{12}*a_{21}*a_{33}$

- Inverse of Matrices

The square matrix B will be called the inverse of the square matrix A if:

A x B = B x A = I

And it’s denoted by A^{-1}. There are many methods to find the inverse of the matrix like, adjacent matrix method, elementary matrix, Jordan method triangular method…, etc.

Theorem

Let A be square matrix, then A will be invertible matrix if and only if its determinant not equal zero.

- Numerical Solutions of the Linear Equations Systems

Let’s have the following linear equations system:

$a_{11}x_1+a_{12}x_2+\ldots+a_{1n}x_n=b_1$

$a_{21}x_1+a_{22}x_2+\ldots+a_{2n}x_n=b_2$

:

$a_{m1}x_1+a_{m2}x_2+\ldots+a_{mn}x_n=b_m$

This system consists of m equations with n variables. If we use the matrix notation then the above system will be: AX = B s.t.

$$A=\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$ is the coefficient matrix,

$$X=\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$ is the unknown (variables) matrix, and $$B=\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}$$ is the absolute

Value matrix .The augment matrix is the matrix A beside it the column B.

$$[A|B]=\left[\begin{array}{cccc|c} a_{11} & a_{12} & \cdots & a_{1n} & b_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & b_2 \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} & b_m \end{array}\right]$$

The solution of the linear equations systems means find the values of the unknowns $x_1,x_2,\ldots,x_n$ which satisfy all equations of the system.

Definition:

The zero solution is the solution of the linear system if all the valus of matrix X equal 0, s.t.

$x_1= x_2 = x_3 = \ldots = x_n = 0$

Definition

The square matrix A called singular if and only if $|A|=0$.

Theorem

Let A be square matrix of degree n, then the following relation are equivalent:

1. The homogenous system AX = 0 has zero solution only.

2. The system AX = B has unique solution for every different column B.

3. The matrix A has inverse.

- ## Matrices Solving Methods

There are many methods for solving the matrices, we will describe some of them.

Cramer Rule

Theorem

If A is the coefficient matrix of linear equations system consists of n variables, and $|A| \neq 0$, then the solution is:

$x_1 = D_1/D$, $x_2 = D_2/D$, … , $x_n = D_n/D$,

where $D=|A|$, and $D_i=|A_i|$, A_i is the matrix A when change the column B with column i, s.t. $1 \leq i \leq n$.

Example

$3x_1 - 2x_2 = 6$

$2x_1 + x_2 = 0.5$

$AX = B$

$$A = \begin{bmatrix} 3 & -2 \\ 2 & 1 \end{bmatrix}, B = \begin{bmatrix} 6 \\ 0.5 \end{bmatrix}, X = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$|A| = 7$$

$$x_1 = \frac{\begin{bmatrix} 6 & -2 \\ 0.6 & 1 \end{bmatrix}}{7} = \frac{7}{7} = 1, \quad x_2 = \frac{\begin{bmatrix} 3 & 6 \\ 2 & 0.5 \end{bmatrix}}{7} = \frac{-10.5}{7} = -1.5$$

Inverse of Matrix Method

Since $AX = B$, then $X = A^{-1}B$. That if we can find the inverse of matrix A in one of the methods mentioned in a previous section, then the multiplication of A^{-1} with B give the unknowns columns X.

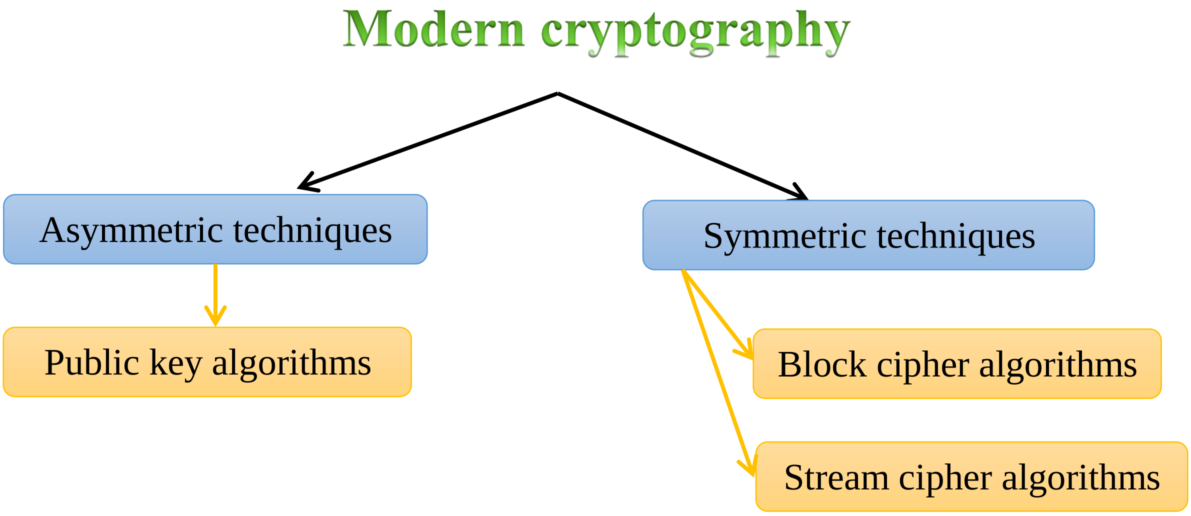
Example

$$\begin{bmatrix} -2 & 2 & -3 \\ 2 & 1 & -6 \\ -1 & -2 & 0 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 5 \\ 10 \\ -5 \end{bmatrix}, \text{ then}$$

$$\begin{bmatrix} -2 & 2 & -3 \\ 2 & 1 & -6 \\ -1 & -2 & 0 \end{bmatrix}^{-1} = \begin{bmatrix} -4/15 & 2/15 & -1/5 \\ 2/15 & -1/15 & -2/5 \\ -1/15 & -2/15 & -2/15 \end{bmatrix}$$

$$X = \begin{bmatrix} -4/15 & 2/15 & -1/5 \\ 2/15 & -1/15 & -2/5 \\ -1/15 & -2/15 & -2/15 \end{bmatrix}\begin{bmatrix} 5 \\ 10 \\ -5 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ -1 \end{bmatrix}$$

$x_1=1$, $x_2=2$ and $x_3=-1$.



Different algorithms have come up with powerful encryption mechanisms incorporated in them. It gave rise to two new ways of encryption mechanism for data security. These are:

- Symmetric key encryption
- Asymmetric key encryption

Key: It can be a number, word, phrase, or any code that will be used for encrypting and decrypting any ciphertext information to plain text and vice versa.

Symmetric and asymmetric key cryptography is based on the number of keys and the way these keys work. Let us know about both of them in details:

Symmetric Key Encryption

Symmetric key encryption technique uses a straight forward method of encryption. Hence, this is the simpler among these two practices. In the case of symmetric key encryption, the encryption is done through only one secret key, which is known as "Symmetric Key", and this key remains to both the parties. The same key is implemented for both encodings as well as decoding the information. So the key is used first by the sender prior to sending the message, and on the receiver side, that key is used to decipher the encoded message.

One of the good old examples of this encryption technique is Caesar's Cipher. Modern examples and algorithms that use the concept of symmetric key encryption are RC4, QUAD, AES, DES, Blowfish, 3DES, etc.

Asymmetric Key Encryption

Asymmetric Encryption is another encryption method that uses two keys: a new and sophisticated encryption technique. This is because it integrates two cryptographic keys for implementing data security. These keys are termed as Public Key and Private Key. The "public key", as the name implies, is accessible to all who want to send an encrypted message. The other is the "private key" that is kept secure by the owner of that public key or the one who is encrypting.

Encryption of information is done through a public key first, with the help of a particular algorithm. Then the private key, which the receiver possesses, will use to decrypt that encrypted information. The same algorithm will be used in both encodings as well as decoding. Examples of asymmetric key encryption algorithms are Diffie-Hellman and RSA algorithm.

Techniques Work in Combination With Modern Cryptography

- Encryption and it's key
- Hash functions
- Message Authentication Codes (MAC)

- Digital Signatures

Advantages and Characteristic Differences Between Classical/Traditional Encryption and Modern Encryption

Here are the marked differences between the classical as well as the modern encryption techniques:

Traditional Encryption	Modern Encryption
For making ciphertext, manipulation is done in the characters of the plain text.	For making ciphertext, operations are performed on binary bit sequence.
The whole of the ecosystem is required to communicate confidentially.	Here, only the parties who want to execute secure communication possess the secret key.
These are weaker as compared to modern encryption.	The encryption algorithm formed by this encryption technique is stronger as compared to traditional encryption algorithms.
It believes in the concept of security through obscurity.	Its security depends on the publicly known mathematical algorithm.

Symmetric cipher techniques

- Equations as number key generators

Equations was used as a random key generators for encryption secret messages sequence. First ,second and higher order equations is employed to generate random number as key sequence to add with the plain number sequence to produce the cipher number sequence .

First order equation as number generator

First formula

$C_i = a.P_i + b \mod n$,

P_i:is the plain number , C_i: is the cipher number, a,b and n represents the key.

Second formula

$C_i = a.P_i + C_{i-1} \mod n$,

when C_1, C_{i-1} will be 0. P_i:is the plain number , C_i: is the cipher number, a and n represents the key.

Example

Encryption by using first formula:

Let plain text(message): CONNECT

And values of a,b and n are 5,9 and 23 sequentially.

Then:

P1 = C = 3 → C1=5.3+33 mod 23 = 2

P2 = O = 15 → C2=5.15+33 mod 23 = 10

P3 = N = 14 → C3=5.14+33 mod 23 = 11

P4 = N = 14 → C4=5.14+33 mod 23 = 11

P5 = E = 5 → C5=5.5+33 mod 23 = 12

P6 = C = 3 → C6=5.3+33 mod 23 = 2

P7 = T = 20 → C7=5.20+33 mod 23 = 18

Then the cipher sequence is 2,10,11,11,12,2,18

Encryption by using second formula:

Values of a and n are 5 and 23 sequentially.

Then:

P1 = C = 3 → C1=5.3+0 mod 23 = 15

P2 = O = 15 → C2=5.15+15 mod 23 = 21

P3 = N = 14 → C3=5.14+21 mod 23 = 22

P4 = N = 14 → C4=5.14+22 mod 23 = 23

P5 = E = 5 → C5=5.5+23 mod 23 = 2

P6 = C = 3 → C6=5.3+2 mod 23 = 17

P7 = T = 20 → C7=5.20+17 mod 23 = 2

Then the cipher sequence is 15,21,22,23,2,17,2

Note: We can use the big number for key parts.

Question: What's the difference between two techniques in the security's respect?

• Requirements for Random Number Generators

Ideally a pseudo-random number generator would produce a stream of numbers that:

1- Are uniformly distributed. ذات توزيع منتظم

2- Are uncorrelated . لا تحقق علاقات الارتباط

3- Never repeats itself. لا تكرر نفسها او ممكن ان تتكرر بأوقات متباعدة جدا

4- Satisfy any statistical test for randomness. تحقق أي اختبار للعشوائية

5- Are reproducible (for debugging purposes). قابل لإعادة الإنتاج والاستنساخ

6- Are portable (the same on any computer). يمكن تنصيبه علي أي حاسوب (محمول)

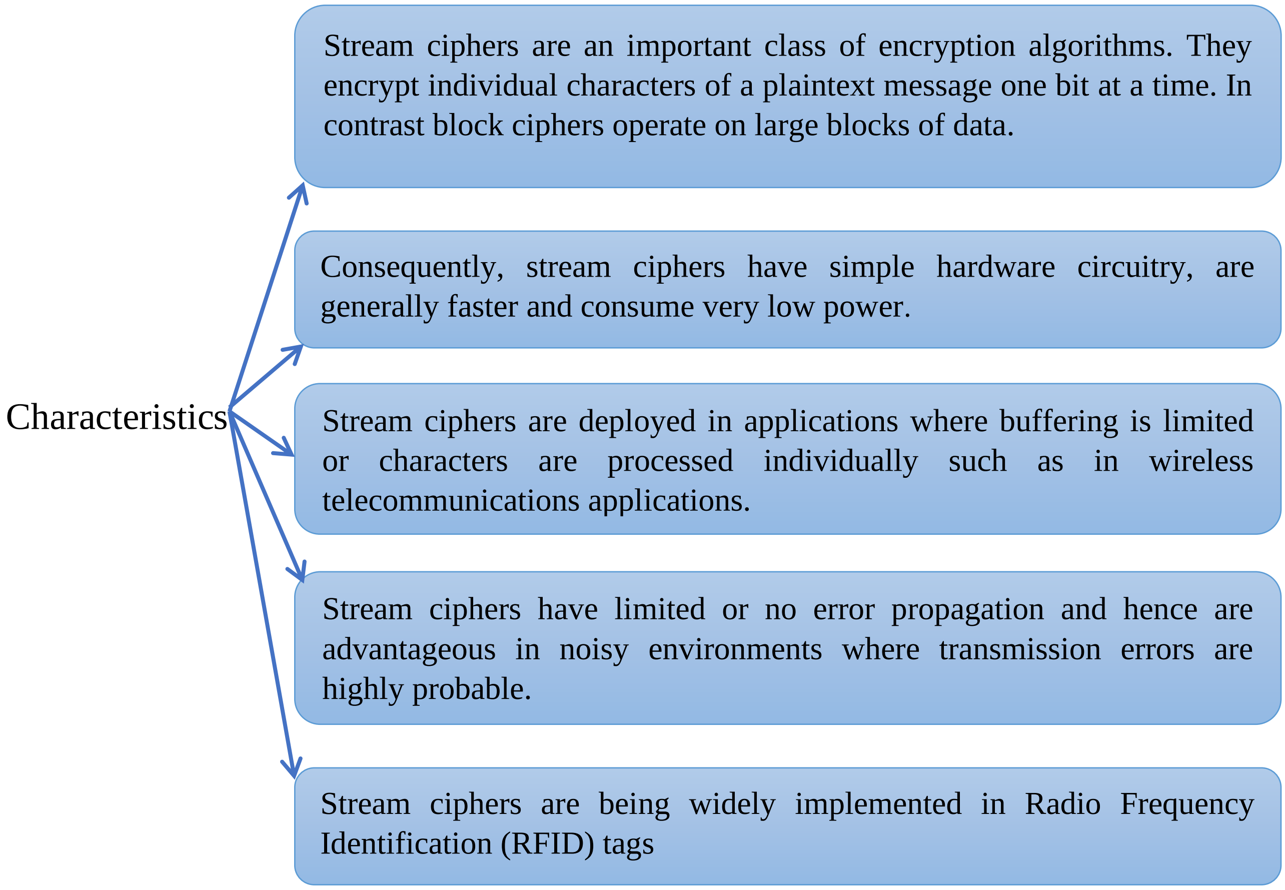
7- Can be changed by adjusting an initial "seed" value. ممكن تغييره بواسطة تعديل القيم الابتدائية

8- Can easily be split into many independent subsequences. بالإمكان بسهولة الفصل لسلاسل جزئية مستقلة

9- Can be generated rapidly using limited computer memory. يمكن توليده بسرعه باستخدام ذاكرة حاسوب محدوده

In practice it is impossible to satisfy all these requirements exactly. Since a computer uses finite precision arithmetic to store the state of the generator, after a certain period the state must match that of a previous iteration, after which the generator will repeat itself. Also, since the numbers must be reproducible, they are not truly random, but generated by a deterministic iterative process, and therefore cannot be completely uncorrelated.

- Stream cipher advantages compare with block cipher



Characteristics

Stream ciphers are an important class of encryption algorithms. They encrypt individual characters of a plaintext message one bit at a time. In contrast block ciphers operate on large blocks of data.

Consequently, stream ciphers have simple hardware circuitry, are generally faster and consume very low power.

Stream ciphers are deployed in applications where buffering is limited or characters are processed individually such as in wireless telecommunications applications.

Stream ciphers have limited or no error propagation and hence are advantageous in noisy environments where transmission errors are highly probable.

Stream ciphers are being widely implemented in Radio Frequency Identification (RFID) tags

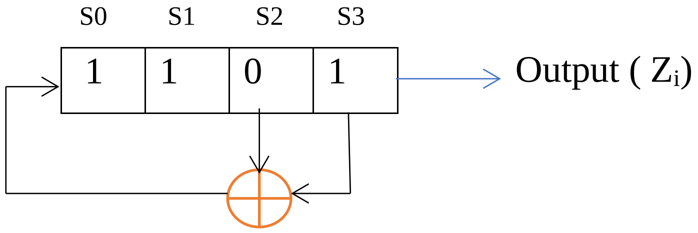
- Random numbers generators based Feedback Shift Registers FSR

The Shift Register (SR) used and still be used in many fields, like computers, communications (radar, satellite equipments,…), information theory, coding theory, protocols …etc. It’s an important part of many scientific devices design since its light, cheep, and has small size. The importance of SR raised when it's inter many modern and complex fields like communication and data security, so its inter in hardware or software of encryption devices specially the stream cipher system. These small devices are combined with each other and some Boolean functions to design an encryption algorithm to generate long binary sequences. These sequences have good randomness properties work as encryption key combined with plaintext binary digits to be encrypted before send to the receiver to be safe from intruders and attackers. The construction of the encryption algorithm must be designed with much careful. The designer must has good mathematical background before he designs the encryption algorithm to guarantee that the sequence not be estimated or calculated analytically even if the cryptanalyst has some information about the encryption algorithm or part of the encryption key.

- FSR characteristics

The hardware contains stages, each of them stores one bit, content of the FSR stages shift in one direction through an entrance from the other. FSR has a feedback calculated from connecting all or some of its constituent stages by a primitive polynomial (The connection should give a maximum period). As a software, FSR is simulated by an one dimension array(vector), the connection functions of the feedback stage are either linear or nonlinear.

EX/



Linear feedback shift register :

Length = 4 stages, Feedback stages = (s2,s3) , feedback function =XOR , input stage= s0 , output stage= s3, polynomial=1+S2+S3

FB

LFSR initial	FB bit	output key bit
1 1 0 1 → out	1	
1 1 1 1 0	1	1
2 1 1 1 1	0	0
3 0 1 1 1	0	1
4 0 0 1 1	0	1
5 0 0 0 1	1	1
6 1 0 0 0	0	1
7 0 1 0 0	0	0
8 0 0 1 0	1	0
9 1 0 0 1	1	0
10 1 1 0 0	0	1
11 0 1 1 0	1	0
12 1 0 1 1	0	0
13 0 1 0 1	1	1
14 1 0 1 0	1	1
15 1 1 0 1	sequence will repeated	0

Output key sequence:101111000100110

OKS length = 2^4-1 = 15 bits , 4 is LFSR length (this is satisfy if the connection polynomial is **primitive** which giving **maximum period of sequence**) a linear-feedback shift register (LFSR) is a shift register whose input bit is a linear function of its previous state.

The most commonly used linear function of single bits is exclusive-or (XOR). Thus, an LFSR is most often a shift register whose input bit is driven by the XOR of some bits of the overall shift register value.

The initial value of the LFSR is called the seed, and because the operation of the register is deterministic, the stream of values produced by the register is completely determined by its current (or previous) state. Likewise, because the register has a finite number of possible states, it must eventually enter a repeating cycle. However, an LFSR with a well-chosen feedback function can produce a sequence of bits that appears random and has a very long cycle. The mathematics of a cyclic redundancy check, used to provide a quick check against transmission errors, are closely related to those of an LFSR. In general, the arithmetic behind LFSRs makes them very elegant as an object to study and implement. One can produce relatively complex logics with simple building blocks.

Uses and Applications

- pseudo-noise sequences, fast digital counters, and whitening sequences. Both hardware and software implementations of LFSRs are common.
- Pseudo random number generator.
- Head and tail pointers in a FIFO.
- Program counter in a simple CPU
- In addition to the generation of pseudo-random number sequences, applications are in the field of fast digital synchronous counters,

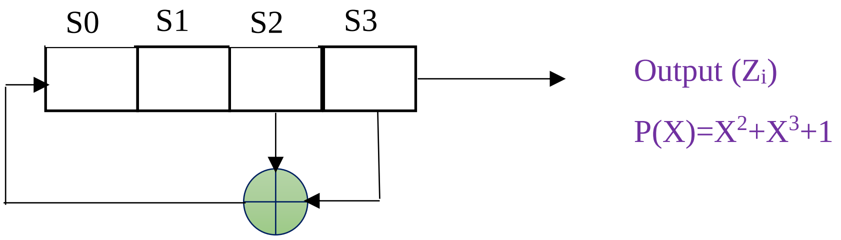
since these counters work without carryover, in the field of communications technology and cryptography in scramblers to make data sequences spectrally white , in coding theory in the coding and decoding of cyclic codes , such as in the cyclical redundancy check (CRC) or the Hamming code , and in the field of digital modulation technology in the code division multiplex method (CDMA) and in the field of steganography .

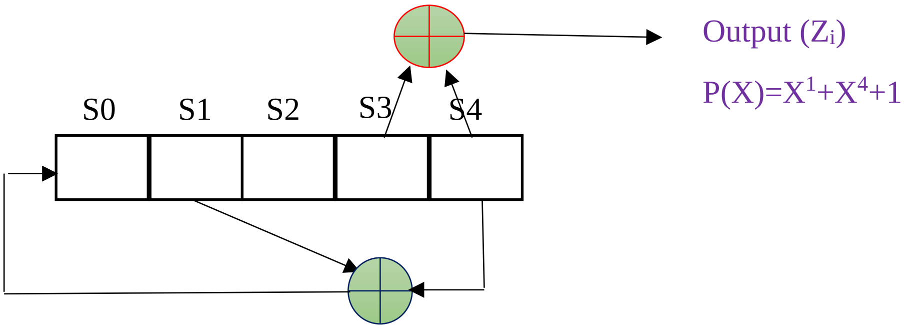
LFSR and Output-stream properties

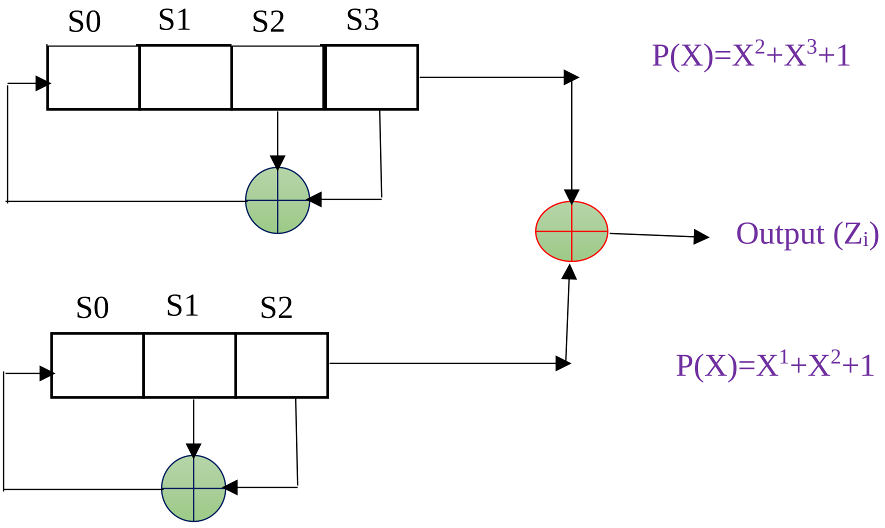
- Ones and zeroes occur in "runs". The output stream 1110010, for example, consists of four runs of lengths 3, 2, 1, 1, in order. In one period of a maximal LFSR, 2^{n-1} runs occur (in the example above, the 3-bit LFSR has 4 runs). Exactly half of these runs are one bit long, a quarter are two bits long, up to a single run of zeroes $n-1$ bits long, and a single run of ones n bits long. This distribution almost equals the statistical expectation value for a truly random sequence. However, the probability of finding exactly this distribution in a sample of a truly random sequence is rather low.
- LFSR output streams are deterministic. If the present state and the positions of the XOR gates in the LFSR are known, the next state can be predicted. This is not possible with truly random events. With maximal-length LFSRs, it is much easier to compute the next state, as there are only an easily limited number of them for each length.
- The output stream is reversible; an LFSR with mirrored taps will cycle through the output sequence in reverse order.
- The value consisting of all zeros cannot appear. Thus an LFSR of length n cannot be used to generate all 2^n values.

- Shift registers with linear feedback can be efficiently implemented both directly in hardware, such as FPGAs , and in software. In the software implementation, since most processors work with register widths larger than one bit, work is typically carried out with tables calculated in advance, which directly map the internal state of the shift register.

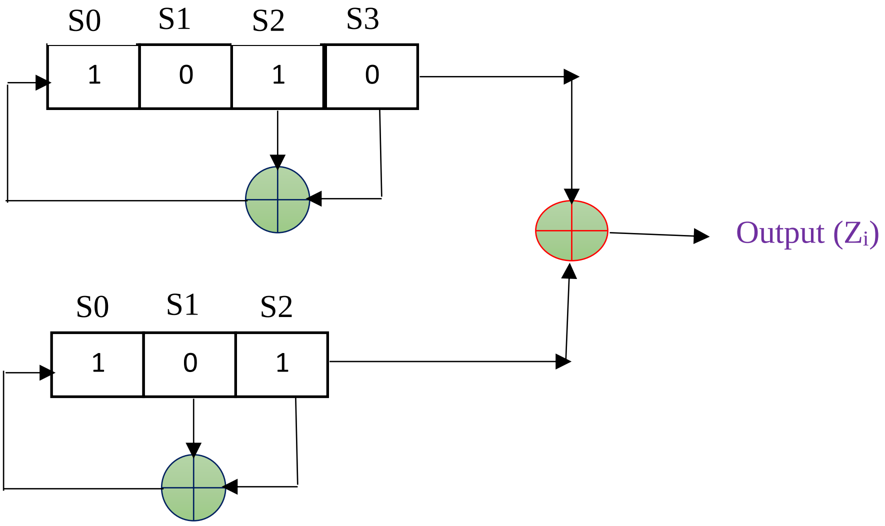
Some LFSR structures







Q/ Encrypt the plain text "CONFIDENCE" by using the generator describe in the figure below, and based on ascii-7 to convert character to binary?



- To convert characters into binary, by using ASCII-7 and prepare P:

No	Chr	ascii	binary
1	C	67	1000011
2	O	79	1001111
3	N	78	1001110
4	F	70	1000110
5	I	73	1001001
6	D	68	1000100
7	E	69	1000101
8	N	78	1001110
9	C	67	1000011
10	E	69	1000101

So, the plain text sequence is

P=10000111001111100111010001101001001100010010001011001110100000111000101

No. of plain bits=10*7=70 bits

- To produce the final key sequence (Z_i), by using generator:

Note: We need 70 bits from Z_i sequence.

M. P. of $LFSR_4=2^4-1$ = 15 bits, M. P. of $LFSR_3=2^3-1$= 7 bits

No.	LSFR$_4$	LFSR$_4$ output	LFSR$_3$	LFSR$_3$ output	Z_i=R$_4$out xor R$_3$out
0	1010	-	101	-	
1	1101	0	010	1	1
2	1110	1	001	0	1
3	1111	0	100	1	1
4	0111	1	110	0	1
5	0011	1	111	0	1
6	0001	1	011	1	0
7	1000	1	101	1	0
8	0100	0		1	1
9	0010	0		0	0
10	1001	0		1	1
11	1100	1		0	1
12	0110	0		0	0
13	1011	0		1	1
14	0101	1		1	0
15	1010	1		1	0
16		0		0	0
17		1		1	0
18		0		0	0
19		1		0	1
20		1		1	0
21		1		1	0
22		1		1	0
23		0		0	0
24		0		1	1
25		0		0	0
26		1		0	1
27		0		1	1
28		0		1	1
29		1		1	0
30		1		0	1

R_4=010111100010011010111100010011010111100010011

⊕

R_3=1010011101001110100111010011101001110100111101

=

Z_i=111110010110100000100001011101110000110001110

⊕

Pi=100001110011111001110100011010010011000100100

=

Ci=011111100101011001010101000111100011110101010

R_4=0101111000100110101111000

⊕

R_3=0011101001110100111010011

=

Z_i=0110010001010010010101011

⊕

P_i=0101100111010000111000101

=

C_i=0011110110000010101101110

Cipher text sequence

"0111111001010110010101010001111000111101010100011110110000
010101101110"

Q/ By using the following description of the stream cipher generator, encrypt the plain text “10010011001011010101”?

Generator description :

LFSR no.=**2** , LFSR1 length=**3** , LFSR2 length=**4** ,

LFSR1 polynomial = **X^1+X^3+1** , LFSR2 polynomial = **X^1+X^4+1**

LFSR1 initial=**001** , LFSR2 initial = **1100**

Problems in the symmetric cryptosystems

- *Key Distribution*

For symmetric encryption to work, the two parties to an exchange must share the same key, and that key must be protected from access by others. Furthermore, frequent key changes are usually desirable to limit the amount of data compromised if an attacker learns the key.

Therefore, the strength of any cryptographic system rests with the *key distribution technique*, a term that refers to the means of delivering a key to two parties who wish to exchange data, without allowing others to see the key. For two parties A and B, key distribution can be achieved in a number of ways, as follows:

1. A can select a key and physically deliver it to B.

2. A third party can select the key and physically deliver it to A and B.

3. If A and B have previously and recently used a key, one party can transmit the new key to the other, encrypted using the old key.

4. If A and B each has an encrypted connection to a third party C, C can deliver a key on the encrypted links to A and B.

Options 1 and 2 call for manual delivery of a key. For link encryption, this is a reasonable requirement, because each link encryption device is going to be exchanging data only with its partner on the other end of the link. However, for end-to-end encryption, manual delivery is awkward. In a distributed system, any given host or terminal may need to engage in exchanges with many other hosts and terminals over time. Thus, each device needs a number of keys supplied dynamically. ***The problem is especially difficult in a wide area distributed system.***

The scale of the problem depends on the number of communicating pairs that must be supported. If end-to-end encryption is done at a network or IP level, then a key is needed for each pair of hosts on the network that wish to communicate. **Thus, if there are *N* hosts, the number of required keys is [*N*(*N*- 1)]/2. If encryption is done at the application level, then a key is needed for every pair of users or processes that require communication. Thus, a network may have hundreds of hosts but thousands of users and processes.**

- *Digital signature*

The second problem and one that was apparently unrelated to the first was that of "digital signatures." If the use of cryptography was to become widespread, not just in military situations but for commercial and private purposes, then electronic messages and documents would need the equivalent of signatures used in paper documents. That is, could a method be devised that would stipulate, to the satisfaction of all parties, that a digital message had been sent by a particular person?

For example, suppose that John sends an authenticated message to Mary, using one of the schemes. Consider the following disputes that could arise:

1. Mary may forge a different message and claim that it came from John. Mary would simply have to create a message and append an authentication code using the key that John and Mary share.
2. John can deny sending the message. Because it is possible for Mary to forge a message, there is no way to prove that John did in fact send the message.

Asymmetric cryptography: Public key cryptosystems

The concept of public-key cryptography evolved from an attempt to attack two of the most difficult problems associated with symmetric encryption. These two problems which were examined in some detail in the previous sections.

As we have seen, first problem, key distribution under symmetric encryption, requires either (1) that two communicants already share a key, which somehow has been distributed to them; or (2) the use of a key distribution center.

Whitfield Diffie, one of the discoverers of public-key encryption (along with Martin Hellman, both at Stanford University at the time), reasoned that this second requirement negated the very essence of cryptography: the ability to maintain total secrecy over your own communication.

The second problem that Diffie pondered, and one that was apparently unrelated to the first was that of "digital signatures." If the use of cryptography was to become widespread, not just in military situations but for commercial and private purposes, then electronic messages and documents would need the equivalent of signatures used in paper documents. That is, could a method be devised that would stipulate, to the satisfaction of all parties, that a digital message had been sent by a particular person?

Diffie and Hellman achieved an astounding breakthrough in 1976 by coming up with a method that addressed both problems and that was radically different from all previous approaches to cryptography, going back over four millennia. Diffie and Hellman first *publicly* introduced the concepts of public-key cryptography in 1976. However, this is not the true beginning. Admiral Bobby Inman, while director of the National Security Agency (NSA), claimed that public-key cryptography had been discovered at NSA in the mid-1960s The first *documented* introduction of these concepts came in 1970, from the Communications-Electronics Security Group, Britain's counterpart to NSA, in a classified report by James Ellis.

Asymmetric algorithms rely on one key for encryption and a different but related key for decryption. These algorithms have the following important characteristic:

It is computationally infeasible to determine the decryption key given only knowledge of the cryptographic algorithm and the encryption key.

In addition, some algorithms, such as RSA, also exhibit the following characteristic:

Either of the two related keys can be used for encryption, with the other used for decryption.

A public-key encryption scheme has six ingredients :

- **Plaintext:** This is the readable message or data that is fed into the algorithm as input.
- **Encryption algorithm:** The encryption algorithm performs various transformations on the plaintext.

- **Public and private keys:** This is a pair of keys that have been selected so that if one is used for encryption, the other is used for decryption. The exact transformations performed by the algorithm depend on the public or private key that is provided as input.
- **Ciphertext:** This is the scrambled message produced as output. It depends on the plaintext and the key. For a given message, two different keys will produce two different ciphertexts.
- **Decryption algorithm:** This algorithm accepts the ciphertext and the matching key and produces the original plaintext.

Q/ By using the following description of the stream cipher generator, encrypt the plain text “10010011001011010101”?

Generator description :

LFSR no.=**2** , LFSR1 length=**3** , LFSR2 length=**4** ,

LFSR1 polynomial = $\mathbf{X^1+X^3+1}$, LFSR2 polynomial = $\mathbf{X^1+X^4+1}$

LFSR1 initial=**001** , LFSR2 initial = **1100**

Problems in the symmetric cryptosystems

- *Key Distribution*

For symmetric encryption to work, the two parties to an exchange must share the same key, and that key must be protected from access by others. Furthermore, frequent key changes are usually desirable to limit the amount of data compromised if an attacker learns the key.

Therefore, the strength of any cryptographic system rests with the *key distribution technique*, a term that refers to the means of delivering a key to two parties who wish to exchange data, without allowing others to see the key. For two parties A and B, key distribution can be achieved in a number of ways, as follows:

1. A can select a key and physically deliver it to B.

2. A third party can select the key and physically deliver it to A and B.

3. If A and B have previously and recently used a key, one party can transmit the new key to the other, encrypted using the old key.

4. If A and B each has an encrypted connection to a third party C, C can deliver a key on the encrypted links to A and B.

Options 1 and 2 call for manual delivery of a key. For link encryption, this is a reasonable requirement, because each link encryption device is going to be exchanging data only with its partner on the other end of the link. However, for end-to-end encryption, manual delivery is awkward. In a distributed system, any given host or terminal may need to engage in exchanges with many other hosts and terminals over time. Thus, each device needs a number of keys supplied dynamically. ***The problem is especially difficult in a wide area distributed system.***

The scale of the problem depends on the number of communicating pairs that must be supported. If end-to-end encryption is done at a network or IP level, then a key is needed for each pair of hosts on the network that wish to communicate. **Thus, if there are *N* hosts, the number of required keys is [*N*(*N*- 1)]/2. If encryption is done at the application level, then a key is needed for every pair of users or processes that require communication. Thus, a network may have hundreds of hosts but thousands of users and processes.**

- *Digital signature*

The second problem and one that was apparently unrelated to the first was that of "digital signatures." If the use of cryptography was to become widespread, not just in military situations but for commercial and private purposes, then electronic messages and documents would need the equivalent of signatures used in paper documents. That is, could a method be devised that would stipulate, to the satisfaction of all parties, that a digital message had been sent by a particular person?

For example, suppose that John sends an authenticated message to Mary, using one of the schemes. Consider the following disputes that could arise:

1. Mary may forge a different message and claim that it came from John. Mary would simply have to create a message and append an authentication code using the key that John and Mary share.
2. John can deny sending the message. Because it is possible for Mary to forge a message, there is no way to prove that John did in fact send the message.

Asymmetric cryptography: Public key cryptosystems

The concept of public-key cryptography evolved from an attempt to attack two of the most difficult problems associated with symmetric encryption. These two problems which were examined in some detail in the previous sections.

As we have seen, first problem, key distribution under symmetric encryption, requires either (1) that two communicants already share a key, which somehow has been distributed to them; or (2) the use of a key distribution center.

Whitfield Diffie, one of the discoverers of public-key encryption (along with Martin Hellman, both at Stanford University at the time), reasoned that this second requirement negated the very essence of cryptography: the ability to maintain total secrecy over your own communication.

The second problem that Diffie pondered, and one that was apparently unrelated to the first was that of "digital signatures." If the use of cryptography was to become widespread, not just in military situations but for commercial and private purposes, then electronic messages and documents would need the equivalent of signatures used in paper documents. That is, could a method be devised that would stipulate, to the satisfaction of all parties, that a digital message had been sent by a particular person?

Diffie and Hellman achieved an astounding breakthrough in 1976 by coming up with a method that addressed both problems and that was radically different from all previous approaches to cryptography, going back over four millennia. Diffie and Hellman first *publicly* introduced the concepts of public-key cryptography in 1976. However, this is not the true beginning. Admiral Bobby Inman, while director of the National Security Agency (NSA), claimed that public-key cryptography had been discovered at NSA in the mid-1960s The first *documented* introduction of these concepts came in 1970, from the Communications-Electronics Security Group, Britain's counterpart to NSA, in a classified report by James Ellis.

Asymmetric algorithms rely on one key for encryption and a different but related key for decryption. These algorithms have the following important characteristic:

It is computationally infeasible to determine the decryption key given only knowledge of the cryptographic algorithm and the encryption key.

In addition, some algorithms, such as RSA, also exhibit the following characteristic:

Either of the two related keys can be used for encryption, with the other used for decryption.

A public-key encryption scheme has six ingredients :

- **Plaintext:** This is the readable message or data that is fed into the algorithm as input.
- **Encryption algorithm:** The encryption algorithm performs various transformations on the plaintext.

- **Public and private keys:** This is a pair of keys that have been selected so that if one is used for encryption, the other is used for decryption. The exact transformations performed by the algorithm depend on the public or private key that is provided as input.
- **Ciphertext:** This is the scrambled message produced as output. It depends on the plaintext and the key. For a given message, two different keys will produce two different ciphertexts.
- **Decryption algorithm:** This algorithm accepts the ciphertext and the matching key and produces the original plaintext.

Some comparisons

A comparison between symmetric modern cryptography types: stream and block ciphers

Modern Stream cipher algorithms	Modern Block cipher algorithms
Encrypts the plain text individually, character by character or bit by bit.	Encrypts the plain text as blocks in variant sizes.
They’re suitable for online communications (because there speeds and the operations are carried out completely on the individuals before moving to the other).	They’re suitable for offline communications(because they treat blain as a block to do encrypting operations)
Often based on a generator is given IV as initial to generate key sequence that adding with plain text sequence.	Methods consist of a mathematical and logical operations , they process the plain text with the encryption key in stages
Examples: A51,A52,E0	Examples: DES,RC4

A comparison between Symmetric and Asymmetric cryptography

Symmetric cryptography	Asymmetric cryptography
Used same algorithm and key in both processes encryption and decryption.	Used same algorithm in the encryption and decryption, and deferent keys in each communication side.
Its methods based on secret key principle.	Its methods based on public key principle. ***(for this reason, and using asymmetric keys it overcame the SYM.GRYP problems)***
Suffer from some problems such as key distribution and digital signature.	Suffer from lots of used mathematical and logical operations.(increasing in mathematical complexity)
Examples: A5,DES,RC,E0.	Examples: RSA

Asymmetric cryptography algorithm
RSA public key algorithm

RSA is an **asymmetric public key** cryptographic algorithm in which two different keys are used to encrypt and decrypt the message. In the year 1978 the three inventors at MIT; Rivest, Shamir and Adleman introduced RSA public key algorithm which follows the essential steps below:

- In RSA public key cryptography each user has to generate two keys a **private key** and a **public key.**
- The public key is circulated or published to all and hence others are aware of it whereas, the private key is secretly kept with the user only.
- A sender has to encrypt the message using the intended receivers public key.
- Only the intended receiver can crack the message. In between the communication no one can harm to the confidentiality of the message as the message can only be decrypted by the intended receiver's private key which is only known to that receiver.

$$\mathbf{M' = E(PU_r, M)} \text{ ……….}\mathbf{Encryption}$$

$$\mathbf{M = D(PR_r, M')} \text{ ……….}\mathbf{Decryption}$$

M *is the original message*
M' *is encrypted message*
E *is an encryption algorithm*
D *is a decryption algorithm*
$\mathbf{PU_r}$ *is the receivers public key*
$\mathbf{PR_r}$ *is the receivers private key*
$\mathbf{PU_s}$ *is the senders public key*
$\mathbf{PR_s}$ *is the senders private key*

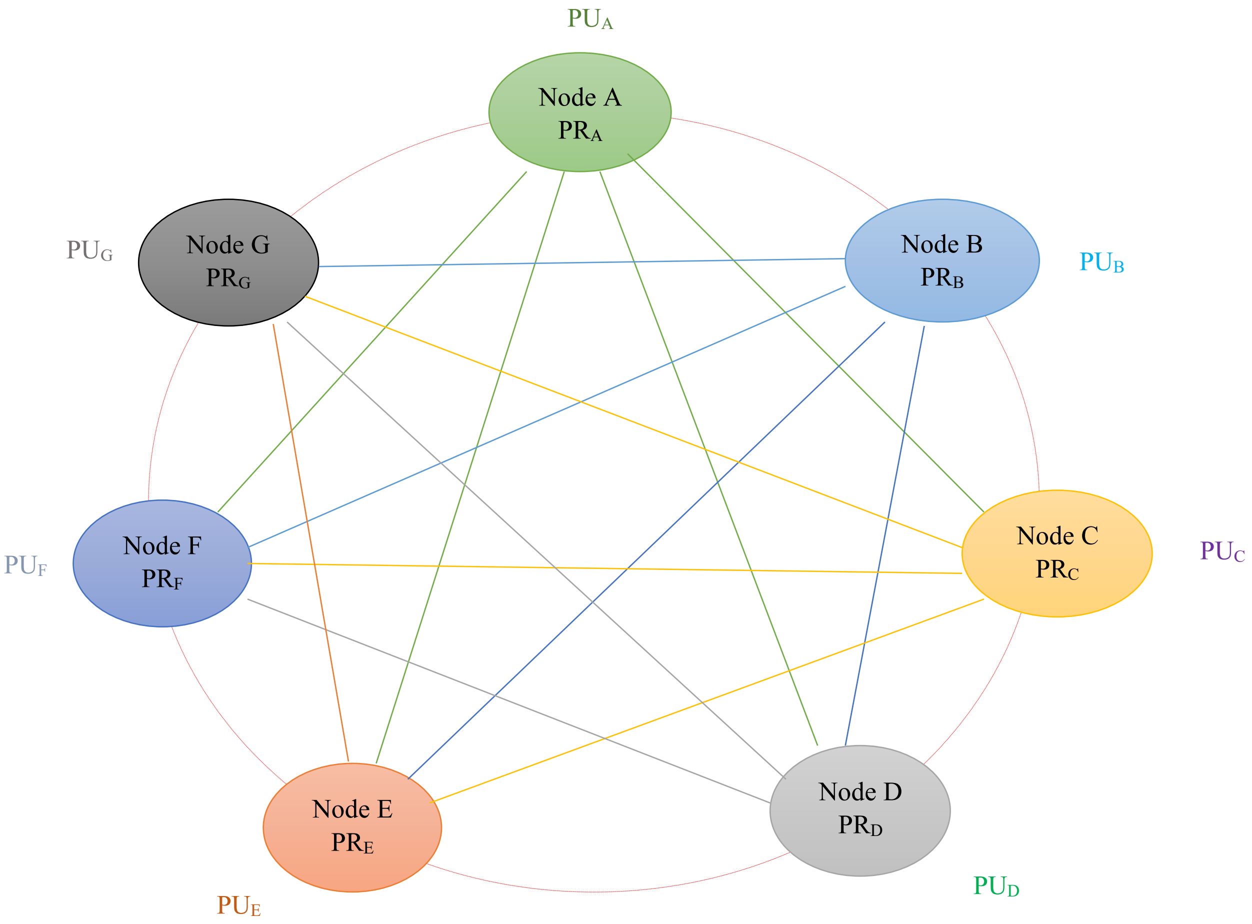


Figure: Cryptosystem of seven nodes Full connection network

Key Generation

- Select two prime numbers p and q such that $p \neq q$
- Calculate $n = p\ x\ q$
- Calculate $\phi(n) = (p-1)(q-1)$
- Select integer e such that gcd($\phi(n), e) = 1;\ 1 < e < \phi(n)$
- Calculate $d = e^{-1}$ (mod $\phi(n))$
- Public key $PU = \{e, n\}$
- Private key $PR = \{d, n\}$

An Example

Steps to generate **public key** (e, n) & **private key** (d, n)

1. First, select two prime numbers p=7 and q=11.
2. Now calculate n= p X q = 7 X 11

   n = 77
3. Calculate Ø(n)= Ø(pXq)

   = Ø(p) X Ø(q)

   = (p-1) X (q-1) ……. Ø (a) = (a-1) if **a** is a prime number.

   =(7-1) X (11-1)

   = 6 X 10

   Ø(n) = 60
4. Select e such that **1 ≤ e < Ø(n)** and also 'e' should be **coprime** to Ø(n).

   So, I select **e=7.**

   Our **Public Key** for this particular example is **(7,77)**.

5. Now we will determine the value of **d**. The value of d can be calculated from the formula given below:

$$ed = 1\ mod\text{Ø}(n)$$

In the expression above we know that and e and Ø(n) are the coprime numbers so in this case d is the multiplicative inverse of e. To calculate the value of d use the formula below:

$$d = \frac{(\text{Ø}(n)i + 1)}{e}$$

In this equation above we know the value of Ø(n), e, the value of i is unknown. First, we have to put the value of i=1.

$$d = \frac{(60 * i + 1)}{7}$$

If the result is in decimals then we have to compute the equation again but this time we have to increment the value of i by 1 so we will compute the equation with i=2. Keep on incrementing the value of i till the above equation results in a proper integer.

So, by trial and error method, for i=5 we get the result 43 i.e.

$$d = \frac{(60 * 5 + 1)}{7}$$

$$d = 43$$

Now we have generated both the private and public key.

Private Key (43, 77)

Public Key (7, 77)

RSA Encryption

Now, after generating the private and public key we will now encrypt the message. In RSA the plain text is always encrypted in **blocks.** The **binary value** of each plain text block should be **<n**. Encryption is done with the intended receiver's **public key**. The expression to calculate cipher text is as follow:

$$\mathbf{M' = M^{e} \bmod n}$$

In our example, the value of e=7 and n=77 i.e. public key (e, n) and we have to take the value of M such that **M<n**. We will take the value of M=15. So, the expression becomes

$$\mathbf{M' = 15^{7} \bmod 77}$$

$$\mathbf{M' = [\ (15^{4} \bmod 77)*(15^{2} \bmod 77)* \\ (\ 15^{1} \bmod 77)\ ]mod77}$$

$$\mathbf{M' = [(36)*(71)*(15)]\ mod77}$$

M'=71 ……… Cipher Text

RSA Decryption

Done with the encryption now its time to decrypt the message. For decryption in RSA, we require a cipher text and the private key of the corresponding public key used in encryption.

In our example the cipher text we have M'=71 and the private key we have (43, 77). The expression to calculate plain text is as follow:

$$\mathbf{M = M'^{d} \bmod n}$$

$$\mathbf{M = 71^{43} \bmod 77}$$

$$\mathbf{M = 15}$$

So, this is the method to encrypt and decrypt the message in RSA. It is very important to remember that in RSA we have to encrypt the message using the intended receiver's public key. So, the message can only be decrypted by the intended receiver private key. This provides **confidentiality** to our message.

RSA Advantages and Disadvantages

Advantages:

- **Convenience:** It solves the problem of distributing the key for encryption.
- **Provides message authentication:** Public key encryption allows the use of digital signatures which enables the recipient of a message to verify that the message is from a particular sender.
- **Detection of tampering:** The use of digital signatures in public key encryption allows the receiver to detect if the message was altered in transit. A digitally signed message cannot be modified without invalidating the signature.
- **Provides non-repudiation:** Digitally signing a message is related to physically signing a document. It is an acknowledgement of the message and thus, the sender cannot deny it.

Disadvantages:

- **Public keys should/must be authenticated :** No one can be absolutely sure that a public key belongs to the person it specifies and so everyone must verify that their public keys belong to them.
- **Slow:** Public key encryption is slow compared to symmetric encryption Not feasible for use in decrypting bulk messages.

- **Uses more computer resources:** It requires a lot more computer supplies compared to single-key encryption.
- **Widespread security compromise is possible:** If an attacker determines a person’s private key, his or her entire messages can be read.
- **Loss of private key may be irreparable:** The loss of a private key means that all received messages cannot be decrypted.

An Example

Perform encryption and decryption using RSA algorithm for p =11, q = 13, e = 7, m = 9.

$$Step\ 1: \quad p = 11, q = 13$$

$$Step\ 2: \quad n = p \times x = 11 \times 13 = 143$$

$$Step\ 3: \quad Calculate$$

$$\varphi(n) = (p-1)(q-1)$$

$$= (11-1)(13-1) = 10 \times 12 = 120$$

$$Step\ 4: \quad Determine\ d\ such\ that\ de \equiv 1(mod\ 160)$$

$$d = e^{-1} mod\ 160$$

Using extended Euclidean algorithm we calculate d

$$= -17\ mod\ 120$$

$$d = 103$$

$$Public\ key = \{7, 143\}$$

$$Private\ key = \{103, 143\}$$

$$Encryption\ (C) = M^{e}\ (mod\ n)$$

$$M = 9$$

$$C = 9^{7} mod\ 143$$

$$= [(9^{4}\ mod\ 143\ ) \times (9^{2}\ mod\ 143)(9^{1}\ mod\ 143)]mod\ 143$$

$$= (126 \times 81 \times 9)mod\ 143$$

$$= 91854\ mod\ 143$$

$$= 48$$

$$Decryption\ (M) = 13^{103} mod\ 143$$

An Example

- Choose p = 3 and q = 11
- Compute n = p * q = 3 * 11 = 33
- Compute φ(n) = (p - 1) * (q - 1) = 2 * 10 = 20
- Choose e such that 1 < e < φ(n) and e and φ (n) are coprime. Let e = 7
- Compute a value for d such that (d * e) mod φ(n) = 1. One solution is d = 3 [(3 * 7) mod 20 = 1]
- Public key is (e, n) => (7, 33)
- Private key is (d, n) => (3, 33)
- The encryption of $m = 2$ is $c = 2^7 \ mod\ 33 = 29$
- The decryption of $c = 29$ is $m = 29^3 \ mod\ 33 = 2$

How to solve Two Symmetric cryptography problems?

The two problems of symmetric key cryptography i.e. confidentiality and authentication can be overcome by the double use of public key cryptography.

1. **First**, encrypt the message by the **sender's private key** which can be decrypted by the sender's public key(known to all). This provides a digital signature to the sender's message and thus **authentication** is achieved.

$$\mathbf{E(PR_s, M)}$$

2. In the **next step**, encrypt again with the **receiver's public key**. This will allow only the intended receiver to decrypt the message, this provides the **confidentiality** to the message.

$$\mathbf{M' = E(PU_r, E(PR_s, M)}$$

The Decryption is shown by the following expression:

$$\mathbf{M = D(PU_s, E(PR_r, M')}$$

Some comparisons

A comparison between symmetric modern cryptography types: stream and block ciphers	
Modern Stream cipher algorithms	Modern Block cipher algorithms
Encrypts the plain text individually, character by character or bit by bit.	Encrypts the plain text as blocks in variant sizes.
They're suitable for online communications (because there speeds and the operations are carried out completely on the individuals before moving to the other).	They're suitable for offline communications(because they treat blain as a block to do encrypting operations)
Often based on a generator is given IV as initial to generate key sequence that adding with plain text sequence.	Methods consist of a mathematical and logical operations , they process the plain text with the encryption key in stages
Examples: A51,A52,E0	Examples: DES,RC4

A comparison between Symmetric and Asymmetric cryptography	
Symmetric cryptography	Asymmetric cryptography
Used same algorithm and key in both processes encryption and decryption.	Used same algorithm in the encryption and decryption, and deferent keys in each communication side.
Its methods based on secret key principle.	Its methods based on public key principle. ***(for this reason, and using asymmetric keys it overcame the SYM.GRYP problems)***
Suffer from some problems such as key distribution and digital signature.	Suffer from lots of used mathematical and logical operations.(increasing in mathematical complexity)
Examples: A5,DES,RC,E0.	Examples: RSA

Asymmetric cryptography algorithm

RSA public key algorithm

RSA is an **asymmetric public key** cryptographic algorithm in which two different keys are used to encrypt and decrypt the message. In the year 1978 the three inventors at MIT; Rivest, Shamir and Adleman introduced RSA public key algorithm which follows the essential steps below:

- In RSA public key cryptography each user has to generate two keys a **private key** and a **public key.**
- The public key is circulated or published to all and hence others are aware of it whereas, the private key is secretly kept with the user only.
- A sender has to encrypt the message using the intended receivers public key.
- Only the intended receiver can crack the message. In between the communication no one can harm to the confidentiality of the message as the message can only be decrypted by the intended receiver's private key which is only known to that receiver.

M'= E(PU$_r$, M) ……….Encryption

M = D(PR$_r$, M') ……….Decryption

M *is the original message*
M' *is encrypted message*
E *is an encryption algorithm*
D *is a decryption algorithm*
PU$_r$ *is the receivers public key*
PR$_r$ *is the receivers private key*
PU$_s$ *is the senders public key*
PR$_s$ *is the senders private key*

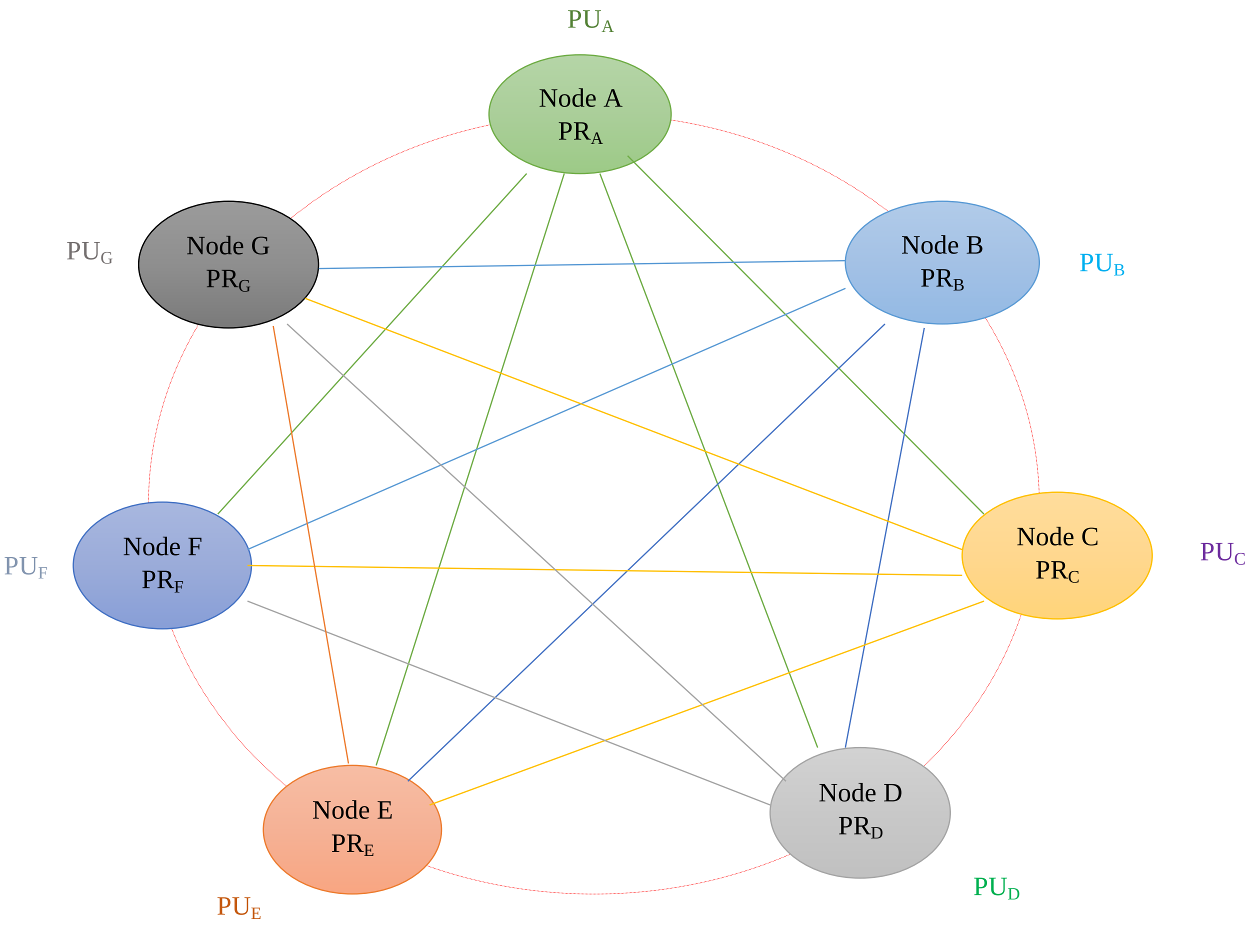


Figure: Cryptosystem of seven nodes Full connection network

Key Generation

- Select two prime numbers p and q such that $p \neq q$
- Calculate $n = p \times q$
- Calculate $\phi(n) = (p-1)(q-1)$
- Select integer e such that gcd$(\phi(n), e) = 1;\ 1<e<\phi(n)$
- Calculate $d = e^{-1}$ (mod $\phi(n)$)
- Public key $PU = \{e, n\}$
- Private key $PR = \{d, n\}$

An Example

Steps to generate **public key** (e, n) & **private key** (d, n)

1. First, select two prime numbers p=7 and q=11.
2. Now calculate n= p X q = 7 X 11

   n = 77
3. Calculate Ø(n)= Ø(pXq)

   = Ø(p) X Ø(q)

   = (p-1) X (q-1) ……. Ø (a) = (a-1) if **a** is a prime number.

   =(7-1) X (11-1)

   = 6 X 10

   Ø(n) = 60
4. Select e such that **1 ≤ e < Ø(n)** and also 'e' should be **coprime** to Ø(n).

   So, I select **e=7.**

   Our **Public Key** for this particular example is **(7,77)**.

5. Now we will determine the value of **d**. The value of d can be calculated from the formula given below:

$$ed = 1\ mod\text{Ø}(n)$$

In the expression above we know that and e and Ø(n) are the coprime numbers so in this case d is the multiplicative inverse of e. To calculate the value of d use the formula below:

$$d = \frac{(\text{Ø}(n)i + 1)}{e}$$

In this equation above we know the value of Ø(n), e, the value of i is unknown. First, we have to put the value of i=1.

$$d = \frac{(60 * i + 1)}{7}$$

If the result is in decimals then we have to compute the equation again but this time we have to increment the value of i by 1 so we will compute the equation with i=2. Keep on incrementing the value of i till the above equation results in a proper integer.

So, by trial and error method, for i=5 we get the result 43 i.e.

$$d = \frac{(60 * 5 + 1)}{7}$$

$$d = 43$$

Now we have generated both the private and public key.

Private Key (43, 77)

Public Key (7, 77)

RSA Encryption

Now, after generating the private and public key we will now encrypt the message. In RSA the plain text is always encrypted in **blocks.** The **binary value** of each plain text block should be **<n**. Encryption is done with the intended receiver's **public key**. The expression to calculate cipher text is as follow:

$$M' = M^{e} \bmod n$$

In our example, the value of e=7 and n=77 i.e. public key (e, n) and we have to take the value of M such that **M<n**. We will take the value of M=15. So, the expression becomes

$$M' = 15^{7} \bmod 77$$

$$M' = [\ (15^{4} \bmod 77)*(15^{2} \bmod 77)* \\ (\ 15^{1} \bmod 77)\ ]\bmod 77$$

$$M' = [(36)*(71)*(15)] \bmod 77$$

M’=71 ……… Cipher Text

RSA Decryption

Done with the encryption now its time to decrypt the message. For decryption in RSA, we require a cipher text and the private key of the corresponding public key used in encryption.

In our example the cipher text we have M’=71 and the private key we have (43, 77). The expression to calculate plain text is as follow:

$$M = M'^{d} \bmod n$$

$$M = 71^{43} \bmod 77$$

$$M = 15$$

So, this is the method to encrypt and decrypt the message in RSA. It is very important to remember that in RSA we have to encrypt the message using the intended receiver's public key. So, the message can only be decrypted by the intended receiver private key. This provides **confidentiality** to our message.

RSA Advantages and Disadvantages

Advantages:

- **Convenience:** It solves the problem of distributing the key for encryption.
- **Provides message authentication:** Public key encryption allows the use of digital signatures which enables the recipient of a message to verify that the message is from a particular sender.
- **Detection of tampering:** The use of digital signatures in public key encryption allows the receiver to detect if the message was altered in transit. A digitally signed message cannot be modified without invalidating the signature.
- **Provides non-repudiation:** Digitally signing a message is related to physically signing a document. It is an acknowledgement of the message and thus, the sender cannot deny it.

Disadvantages:

- **Public keys should/must be authenticated :** No one can be absolutely sure that a public key belongs to the person it specifies and so everyone must verify that their public keys belong to them.
- **Slow:** Public key encryption is slow compared to symmetric encryption Not feasible for use in decrypting bulk messages.

- **Uses more computer resources:** It requires a lot more computer supplies compared to single-key encryption.
- **Widespread security compromise is possible:** If an attacker determines a person's private key, his or her entire messages can be read.
- **Loss of private key may be irreparable:** The loss of a private key means that all received messages cannot be decrypted.

An Example

Perform encryption and decryption using RSA algorithm for p =11, q = 13, e = 7, m = 9.

$Step\ 1:\quad p = 11, q = 13$

$Step\ 2:\quad n = p \times x = 11 \times 13 = 143$

$Step\ 3:\quad Calculate$

$$\varphi(n) = (p-1)(q-1)$$
$$= (11-1)(13-1) = 10 \times 12 = 120$$

$Step\ 4:\quad Determine\ d\ such\ that\ de \equiv 1(mod\ 160)$

$$d = e^{-1} mod\ 160$$

Using extended Euclidean algorithm we calculate d

$$= -17\ mod\ 120$$
$$d = 103$$
$$Public\ key = \{7, 143\}$$
$$Private\ key = \{103, 143\}$$
$$Encryption\ (C) = M^e\ (mod\ n)$$
$$M = 9$$
$$C = 9^7 mod\ 143$$
$$= [(9^4\ mod\ 143\ ) \times (9^2\ mod\ 143)(9^1\ mod\ 143)]mod\ 143$$
$$= (126 \times 81 \times 9) mod\ 143$$
$$= 91854\ mod\ 143$$
$$= 48$$
$$Decryption\ (M) = 13^{103} mod\ 143$$

An Example

- Choose p = 3 and q = 11
- Compute n = p * q = 3 * 11 = 33
- Compute φ(n) = (p - 1) * (q - 1) = 2 * 10 = 20
- Choose e such that 1 < e < φ(n) and e and φ (n) are coprime. Let e = 7
- Compute a value for d such that (d * e) mod φ(n) = 1. One solution is d = 3 [(3 * 7) mod 20 = 1]
- Public key is (e, n) => (7, 33)
- Private key is (d, n) => (3, 33)
- The encryption of $m = 2$ is $c = 2^7 \bmod 33 = 29$
- The decryption of $c = 29$ is $m = 29^3 \bmod 33 = 2$

How to solve Two Symmetric cryptography problems?

The two problems of symmetric key cryptography i.e. confidentiality and authentication can be overcome by the double use of public key cryptography.

1. **First**, encrypt the message by the **sender's private key** which can be decrypted by the sender's public key(known to all). This provides a digital signature to the sender's message and thus **authentication** is achieved.

$$\mathbf{E(PR_s, M)}$$

2. In the **next step**, encrypt again with the **receiver's public key**. This will allow only the intended receiver to decrypt the message, this provides the **confidentiality** to the message.

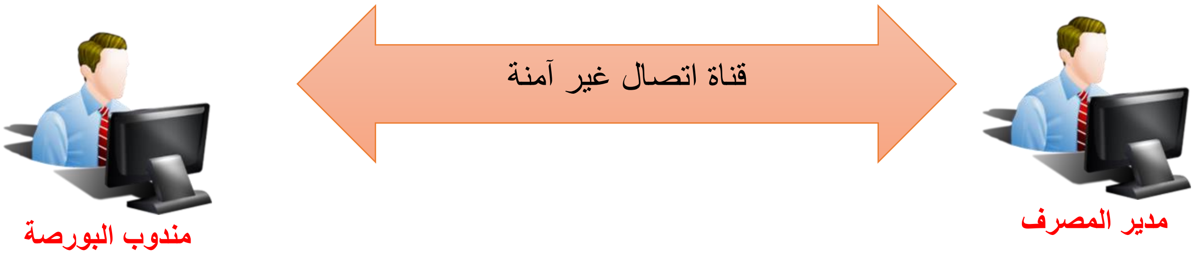
$$\mathbf{M' = E(PU_r, E(PR_s, M)}$$

The Decryption is shown by the following expression:

$$\mathbf{M = D(PU_s, E(PR_r, M')}$$

مثال :أستخدام خوارزمية RSA لحماية المعلومات المهمة و التوقيع الرقمي (الموثوقية)

شبكة معلومات مصرفية تتكون من عقدتين ، هما **مدير المصرف** و **معتمد البورصة** . تستند أمنية الشبكة على خوارزمية RSA ذات المفتاح المعلن ، لتشفير البيانات المهمة وكذلك تستخدم الخوارزمية كآلية لتحقيق الموثوقية.



القيم الابتدائية لمدير المصرف p=11,q=17,and e=23:

N=p*q=187,

Φ(187)=(p-1)*(q-1)=160,

$d= \frac{\Phi(n)*i+1}{e} = \frac{160*1+1}{23} =7$

PU=(23,187)

Pr=(7,187)

القيم الابتدائية لمندوب البورصة:

p=13, q=19,and e=17

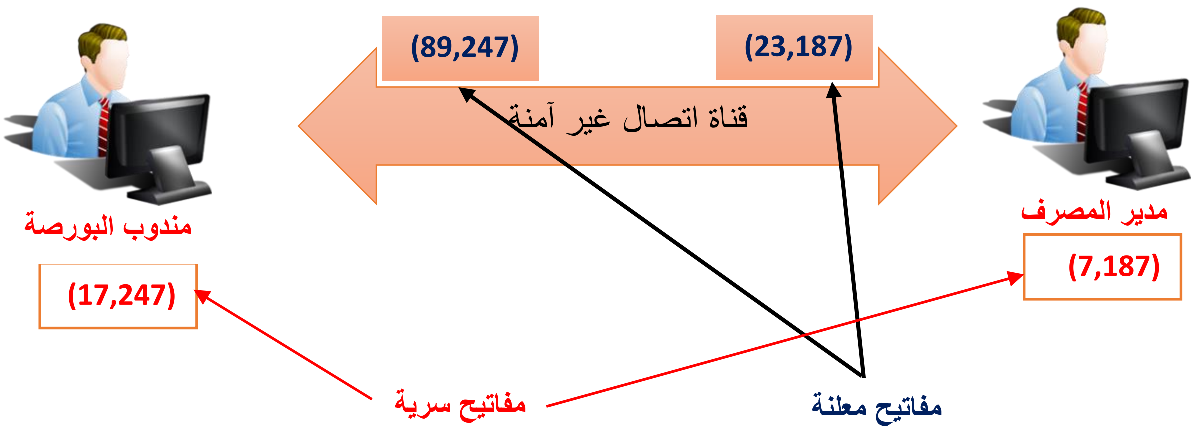
N=p*q=247,

Φ(247)=(p-1)*(q-1)=216,

$d= \frac{\Phi(n)*i+1}{e} = \frac{216*7+1}{17} =89$

PU=(17,247)

Pr=(89,247)



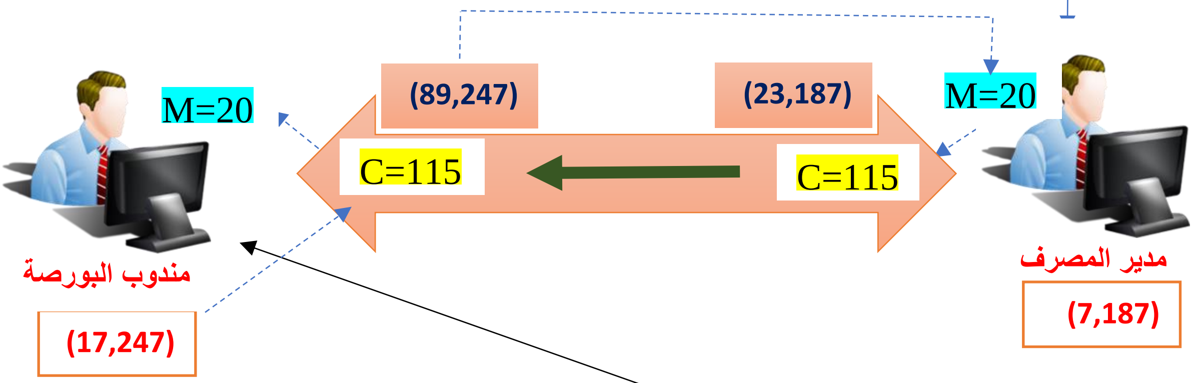
- مدير المصرف يرسل رسالة مشفرة لمندوب المصرف بالبورصة: (m=20)، فيقوم بتشفيرها بالمفتاح المعلن لمندوب المصرف بالبورصة (89,247).

$C=m^{89} \bmod 247$

$C=20^{89} \bmod 247$

$C=((191^{22} \bmod 247)*20) \bmod 247$

$C=((1^{7} \bmod 247)* 191 *20) \bmod 247 \quad = 115$



- مندوب البورصة يقوم بفتح النص المشفر المستلم (115) ، بالمفتاح السري الخاص به :(17,247)

$M=C^{17} \bmod 247$

$M=115^{17} \bmod 247$

$M =((96^{5} \bmod 247)*(115^{2} \bmod 247)) \bmod 247=20$

- مندوب البورصة يريد ان يرسل رسالة لنص واضح(m=34) موقعة من قبله ، الى مدير المصرف، فيقوم باستخدام مفاتيحه السرية(لمندوب البورصة) لتوقيع الرسالة : (17,247)

$S=m^{17} \bmod 247$

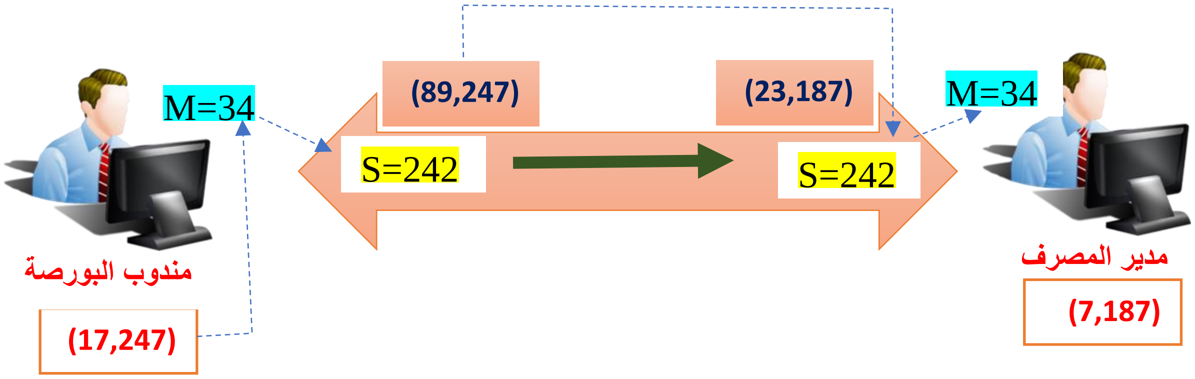
$S=34^{17} \bmod 247$

$S=((66^{4} \bmod 247)*34) \bmod 247=242$

- مدير المصرف : لكي يتأكد ان الرسالة الموقعة هي من مندوب البورصة ، يقوم بفتح توقيعها بمفتاحه المعلن(لمندوب البورصة) (89,247):

$M=S^{89} \bmod 247$

$M=242^{89} \bmod 247 = 34$



- مدير المصرف : يرسل نص مشفر و موقع من قبله الى مندوب البورصة، للرسالة الواضحة (m=53)، فيقوم بتوقيعها بمفتاحه السري(لمدير المصرف)(7,187) ، و من ثم تشفيرها بالمفتاح المعلن لمندوب البورصة(89,247):

$S=53^{7} \bmod 187 = 100$

$C=100^{89} \bmod 247$

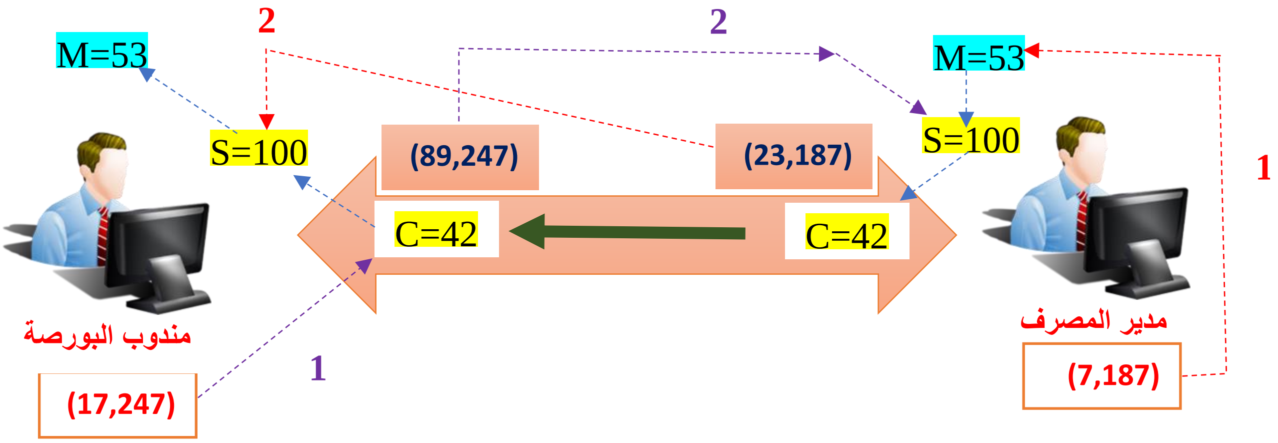
$C=((237^{17} \bmod 247)*(100^{4} \bmod 247)) \bmod 247$

$C=((120^{4} \bmod 247)*237 *74) \bmod 247 =42$

- مندوب البورصة يقوم اولا: بفتح تشفير الرسالة بمفتاحه السري (17,247)، ثم ثانيا: يقوم بفك توقيع الرسالة بالمفتاح المعلن لمدير المصرف (23,187):

(يمكن ان نجمل العمل الذي يقوم به مندوب البورصة على الرسالة المستلمة بدون تجزأة):

$M=((42^{17} \bmod 247)^{23} \bmod 187) = 53$



$M \xleftarrow{2} S \xleftarrow{1} C$ $\qquad$ $C \xleftarrow{2} S \xleftarrow{1} M$

على غرار الخوارزميات والتقنيات المستخدمة ضمن علم التشفير و التي تستند الى مبدأ بعثرة و تعويض مكونات النص الواضح للحصول على نص غير مفهوم، هناك خوارزميات و تقنيات لا تعمل تحت هذا المفهوم ، وهي لا تندرج ضمن طرق علم التشفير بل تندرج ضمن علم آخر هو علم الاخفاء Steganography .

الاخفاء Steganography

و هذا المصطلح اغريقي الاصل ، ويتكون من مقطعين stegano و تعني المخفي و Graphy و تعني الكتابة و هو بذلك يعني الكتابة المخفية. تستند جميع الخوارزميات و التقنيات المنطوية تحت مظلة هذا العلم الى مبدأ ((تضمين و اخفاء المعلومات المهمة ، بكل صورها ، بوسط آخر)) و يتنوع هذا الوسط الى:

- الاخفاء بالنص : هناك عدة طرق متبعة ضمن هذا المفهوم ، منها:

أ- الفراغات البيض white spaces: وهنا ترمز البيانات المهمة الى ثنائيات ويكون اسلوب الاخفاء بعدد الفراغات المتروكة بين الكلمات و التي تناسب مقطع البيانات المهمة(فمثلا 0 يعبر عنه بفراغ واحد و 1 يعبر عنه بفراغين)، أو قد يكون الاخفاء مستند الى الاسطر الفارغة بين الجمل ..و هكذا.

ب- الاخفاء ببعثرة الاحرف ضمن نص: كأن يتم انتاج نص ذو معنى عام و يتم اختيار كلمات متسلسلة كل منها يحتوي حرف او مجموعة احرف في مكانات ثابتة من تلك الكلمات وهي من مكونات المعلومات المهمة.

ت- الاخفاء بالمعنى: الاعتماد على مبدأ المرادفات في اللغة.

- الاخفاء بالصور:تحول الصور الى بيانات ثنائية، حسب الهيئة المخزونة بها، حيث تعطي هذه الهيئة تمثيل رقمي بعدد محدد من الثنائيات لكل خلية صورة(مثلا انواع الصور التي تعتمد 3bytes لتمثيل خلايا الصور ،تقوم بتخصيص 24bits لكل تدرج لوني من الوان الصورة ، قسم من هذه الثنائيات لا توثر على عرض الصورة ،بشكل تلاحظه العين البشرية، ان تغيرت قيمتها لذلك فهي تستغل في عملية اخفاء ثنائيات البيانات المهمة بواسطة تغيير قيم ثنائيات الصورة.
- الاخفاء بالفديو: وهو يستند الى مفهوم الاخفاء بالصور، الا انه يأخذ بنظر الاعتبار التغير بالزمن للصور المعروضة. و هذا النوع هو الاصعب من حيث اخفاء المعلومات المهمة و استرجاعها.

Cyber Security الامن السيبرانى

- **التعريف**

الامن السيبراني يشير الى جميع التقنيات ، المعالجات ،و التطبيقات المصممة لحماية الشبكات ، الاجهزة ، البرامج ، و البيانات حمايتها من المهاجمة ، الضرر ،و الوصول غير المخول.ومصطلح الامن السيبراني يمكن ان يشير ايضا لامنية تكنولوجيا المعلومات . information technology security

- **عناصر الامن السيبراني Elements of cyber security**

في المؤسسات و التنظيمات التي تتعامل بالمعلومات والبيانات سواء كانت عبارة عن حكومة ، جيش ، تنظيم مالي او مصرفي ، منظمة صحية ، او اي منظمة تتداول المعلومات والتي تحتوي هذه المعلومات في جزء او كل منها على معلومات حساسة، فللحصول على نظام سيبراني الكفوء يجب ان تصب الاهتمامات والجهود للامن السيبراني على جميع العناصر التالية:

- Network security **أمنية الشبكة**
- Application security **أمنية التطبيقات**
- Endpoint security **أمنية نقطة النهاية**
- Data security **أمنية البيانات**

- Identity management ادارة الهوية
- Database and infrastructure security أمنية البنى التحتية و قاعدة البيانات
- Cloud security أمنية الغيمة
- Mobile security أمنية النقال
- Disaster recovery/business continuity planning / تخطيط استمرارية العمل استرجاع الكارثة
- End-user education تعليم المستخدم النهائي

- أنواع التهديدات السيبرانية **Types of cyber threats**

1) الجريمة السيبرانية **Cybercrime**

و تتضمن فاعل مفرد او مجاميع تستهدف الانظمة لغرض الربح المالي أو لاحداث الاضرار.

2) الهجوم السيبراني Cyber-attack

و غالبا ما يتضمن تجميع معلومات محفزة و بسياسة.

3) الارهاب السيبراني Cyberterrorism

وهو معد لاتلاف و تدمير الانظمة الالكترونية للتسبب بالهلع و الخوف.

- **كيف يسطر افراد و جماعات التهديدات على انظمة الحاسبات**

هناك طرق شائعة تستخدم في تهديدات الامن السيبراني :

1) البرامج الضارة Malware

وهي تشمل البرامجيات الشريرة و الخبيثة. وهو واحد من التهديدات السيبرانية الاكثر شيوعا. وهو برامجية لمجرم سيبراني او مخترق hacker الغاية منه احداث عطل او ضرر بحاسوب مستخدم شرعي، وهو ينتشر من خلال البريد الالكتروني او تحميل البرامج من النت. و هناك عدد من انواع مختلفة من البرامج الضارة، تتضمن:

أ- الفيروس **Virus**

وهو برنامج تكاثر ذاتي، يلصق نفسه بملف نظيف وينتشر من خلال نظام الحاسوب، ويصيب الملفات بمقطع برمجي خبيث.

ب- أحصنة طروادة **Trojans**

نوع من البرامج الضارة يكون متنكر كبرامجية شرعية. المجرمون السيبرانين يخدعون المستخدمين لتنزيل هذه البرامج على انظمتهم الحاسوبية لاحداث ضرر او تجميع بيانات.

ت- برامج التجسس **Spyware**

برنامج يسجل بصورة سرية ماذا يفعل مستخدم الحاسوب.لذلك يمكن للمجرم السيبراني استخدام تلك المعلومات. على سبيل المثال :spyware للحصول على تفاصيل credit .card

ث- برنامج الفدية **Ransomware**

برنامج ضار يقوم بقفل ملفات المستخدم مع امكانية مسحها مالم يدفع فدية للمجرم السيبراني.

ج- برنامج الاعلان **Adware**

برنامج اعلاني يمكن ان يستخدم لنشر البرامج الضارة.

ح- الروبوت **Botnets**

يمكن ان تصاب حواسيب الشبكات ببرامج ضارة، عندما يقوم المجرم السيبراني بتجهيز مهام online بدون رخصة المستخدم.

2) حقن SQL **SQL injection**

حقن (structured language query) SQL هو نوع من انواع الهجوم السيبراني مستخدم لاعطاء سيطرة و سرقة بيانات من قاعدة البيانات.المجرم السيبراني يستغل قابلية الاصابة في تطبيقات سياقة البيانات ليضيف رمز البرنامج الضار لقاعدة البيانات من خلال عبارة استجواب SQL لبرنامج ضار ، و هذا يعطيه وصول للمعلومات الحساسة في قاعدة البيانات.

3) الخداع **Phishing**

عندما يقوم المجرمون السيبرانيون باستهداف الضحايا الهدف بسؤالهم عن معلومات حساسة، من خلال بريد الكتروني يتم ارساله من شركات شرعية.

4) هجوم رجل في الوسط **Man-in-the-middle attack**

وهو نوع من التهديد السيبراني، حيث يقوم المجرم السيبراني باعتراض الاتصال بين الافراد لسرقة البيانات.على سبيل المثال، في اتصال wifi غير الامن يقوم المهاجم باعتراض الارسال بين الضحايا و الشبكة.

5) هجوم انكار الخدمة **Denial-of-service attack**

المجرم السيبراني يمنع نظام حاسوب من انجاز طلبات شرعية، وذلك بغمر الشبكة و الخوادم بتقاطعات المسارات traffic.

Q1: Assume we use a (4 x 3) bit key of K = [1 2 3 6]. And a plaintext P = [3 0 1 2], so, compute the cipher text?

Binary of Maximum Key numbers

Solution//

The first step is to generate the stream. Initialize the state vector S and temporary vector T. S is initialized so the S[i] = i, and T is initialized so it is the key K (repeated as necessary).

S = [0 1 2 3 4 5 6 7] T = [1 2 3 6 1 2 3 6]

Now perform the initial permutation on S.

```
j = 0;
for i = 0 to 7 do
j = (j + S[i] + T[i]) mod 8
Swap(S[i],S[j]);
end
```

For i = 0: j = 1 ➔ Swap(S[0],S[1]) ➔ S = [1 0 2 3 4 5 6 7];
For i = 1: j = 3 ➔ Swap(S[1], S[3]) ➔ S = [1 3 2 0 4 5 6 7];
For i = 2: j = 0 ➔ Swap(S[2], S[0]) ➔ S = [2 3 1 0 4 5 6 7];
For i = 3: j = 6 ➔ Swap(S[3], S[6]) ➔ S = [2 3 1 6 4 5 0 7];
For i = 4: j = 3 ➔ Swap(S[4], S[3]) ➔ S = [2 3 1 4 6 5 0 7];
For i = 5: j = 2 ➔ Swap(S[5], S[2]) ➔ S = [2 3 5 4 6 1 0 7];
For i = 6: j = 5 ➔ Swap(S[6], S[4]) ➔ S = [2 3 5 4 6 0 1 7];
For i = 7: j = 2 ➔ Swap(S[7], S[2]) ➔ S = [2 3 7 4 6 0 1 5];

Hence, our **initial permutation** of **S = [2 3 7 4 6 0 1 5];** Now we generate 3 bits at a time, **k**, that we XOR with every 3 bits of **plaintext** to produce the **ciphertext**. The 3-bit k is generated by:

```
i, j = 0;
while (true)
{
i = (i + 1) mod 8;
 j = (j + S[i]) mod 8;
 Swap (S[i], S[j]);
t = (S[i] + S[j]) mod 8;
 k = S[t];
 }
```

S = [2 3 7 4 6 0 1 5] & **P = [3 0 1 2]** & i=0 & j=0

The first iteration

i = (0 + 1) mod 8 = 1 & j = (0 + S[1]) mod 8 = 3 , Swap(S[1],S[3]) ➔ S = **[2 4 7 3 6 0 1 5]**

t = (S[1] + S[3]) mod 8 = 7 , k = S[7] = 5 and p = 3

First 3-bits of ciphertext are 5 XOR 1 = 101 XOR 011 = 110 = 6

The second iteration

S = [**2 4 7 3 6 0 1 5**]

i = (1 + 1) mod 8 = 2 , j = (3 + S[2]) mod 8 = 2 , Swap(S[2],S[2]) ➔ S = [**2 4 7 3 6 0 1 5**]

t = (S[2] + S[2]) mod 8 = 6 , k = S[6] = 6 and p =0

Second 3-bits of ciphertext are 6 XOR 0 = 110 XOR 000 = 110 = 6

The third iteration

S = [**2 4 7 3 6 0 1 5**]

i = (2 + 1) mod 8 = 3 , j = (2 + S[3]) mod 8 = 5, Swap(S[3],S[5]) ➔ S = [**2 4 7 0 6 3 1 5**]

t = (S[3] + S[5]) mod 8 = 3 , k = S[3] = 0 and p =1

Third 3-bits of ciphertext are 0 XOR 1 = 000 XOR 001 = 001 = 1

The fourth (final) iteration

S = [**2 4 7 0 6 3 1 5**]

i = (1 + 3) mod 8 = 4 , j= (5 + S[4]) mod 8 = 3 , Swap(S[4],S[3]) ➔ S = [**2 4 7 6 0 3 1 5**]

t = (S[4] + S[3]) mod 8 = 6 , k = S[6] = 1 and p=2

Last 3-bits of ciphertext are 1 XOR 2 = 001 XOR 010 = 011 = 3

So, to encrypt the plaintext stream P = [3 0 1 2] with key K = [5 6 0 1] using our simplified RC4 stream cipher we get C = [6 6 1 3].

(Or in binary: P = 011000001010

$\oplus$

K = 101110000001

C = 110110001011

Q: Encrypt the plain text [4 8 2 9 1 13], by using simplified RC4 and the encryption key is (5 X 4) bit [11 5 7 6 3]?

Solution: We can clarify the steps to find the solution by utilizing the comprehensive table provided below.

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

T=[11 5 7 6 3 11 5 7 6 3 11 5 7 6 3 11]

i	**j=j + s[i] + T[i] mod 8 , initial j=0**	**S**
0	0+0+11 mod 16 = 3	3,1,2,0 ,4,5,6,7,8,9,10,11,12,13,14,15
1	3+1+5 mod 16 = 1	3,1,2,0 ,4,5,6,7,8,9,10,11,12,13,14,15
2	1+2+7 mod 16 = 10	3,1,10,0 ,4,5,6,7,8,9, 2,11,12,13,14,15
3	10+0+6 mod 16 = 0	0,1,10,3 ,4,5,6,7,8,9, 2,11,12,13,14,15
4	0+4+3 mod 16 = 7	0,1,10,3 ,7,5,6,4,8,9, 2,11,12,13,14,15
5	7+5+11 mod 16 = 7	0,1,10,3 ,7,4,6,5,8,9, 2,11,12,13,14,15
6	7+6+5 mod 16 = 2	0,6,10,3 ,7,4,1,5,8,9, 2,11,12,13,14,15
7	2+5+7 mod 16 = 14	0,6,10,3 ,7,4,1,14,8,9, 2,11,12,13,5,15
8	14+8+6 mod 16 = 12	0,6,10,3 ,7,4,1,14,12,9, 2,11,8,13,5,15
9	12+9+3 mod 16 = 8	0,6,10,3 ,7,4,1,14,9,12, 2,11,8,13,5,15
10	8+2+11 mod 16 = 5	0,6,10,3 ,7,2,1,14,9,12, 4,11,8,13,5,15
11	5+11+5 mod 16 = 5	0,6,10,3 ,7,11,1,14,9,12, 4,2,8,13,5,15
12	5+8+7 mod 16 = 4	0,6,10,3 ,8,11,1,14,9,12, 4,2,7,13,5,15
13	4+13+6 mod 16 = 7	0,6,10,3 ,8,11,1,13,9,12, 4,2,7,14,5,15
14	7+5+3 mod 16 = 15	0,6,10,3 ,8,11,1,13,9,12, 4,2,7,14,15,5
15	15+5+11 mod 16 = 15	0,6,10,3 ,8,11,1,13,9,12, 4,2,7,14,15,5

S = [0 6 10 3 8 11 1 13 9 12 4 2 7 14 15 5] & **P = [4 8 2 9 1 13]** & i=0 & j=0

Second permutation:

i=**i+1 mod 16**	j=**j+s[i] mod 16**	S (**Swap (S[i], S[j])**	**t = (S[i] + S[j]) mod 16**	Key=S[t]
0+1=1	0+6=6	0 1 10 3 8 11 6 13 9 12 4 2 7 14 15 5	7	13
1+1=2	6+10=0	10 1 0 3 8 11 6 13 9 12 4 2 7 14 15 5	10	4
2+1=3	0+3=3	10 1 0 3 8 11 6 13 9 12 4 2 7 14 15 5	3	3
3+1=4	3+8=11	10 1 0 3 2 11 6 13 9 12 4 8 7 14 15 5	10	4
4+1=5	11+11=6	10 1 0 3 2 6 11 13 9 12 4 8 7 14 15 5	1	1
5+1=6	6+11=1	10 11 0 3 2 6 1 13 9 12 4 8 7 14 15 5	12	7

We compute the ciphertext by [4 8 2 9 1 13] XOR [13 4 3 4 1 7] = [9 12 1 13 0 10] (OR by using binary ➔ P = 010010000010100100011101

$$\oplus$$

K = 110101000011010000010111

C = 100111000001110100001010