Institute of Information Technology

Jahangirnagar University

জাহাঙ্গীরনগর বিশ্ববিদ্যালয়

IT-4259: Computer Network Security

for

4th Year 2nd Semester of B.Sc (Honors) in IT (5th Batch)

Lecture: 04

Cryptography-A Security Mechanism-2

Prepared by:

K M Akkas Ali

akkas_khan@yahoo.com, akkas@juniv.edu

Associate Professor

Institute of Information Technology (IIT)

Jahangirnagar University, Dhaka-1342

Objectives of this Lecture:

- To illustrate some transposition ciphers:
  - Keyless cipher
  - Keyed cipher
  - Columnar cipher
- To differentiate between traditional and modern symmetric-key ciphers.
- To illustrate DES as a modern symmetric-key cipher.
- To illustrate RSA as a modern asymmetric-key cipher.

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

Transposition Ciphers

What is Transposition Ciphers

- A transposition cipher does not substitute one symbol for another, instead it changes the location of the symbols.
- A symbol in the first position of the plaintext may appear in the ninth position of the ciphertext. A symbol in the eighth position of the plaintext may appear in the first position of the ciphertext.

Types of Transposition Ciphers

There are three types of transposition cipher:

- Keyless Transposition Ciphers
- Keyed Transposition Ciphers
- Keyed Columnar Transposition Ciphers or Columnar Transposition Ciphers

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

Keyless Transposition Ciphers

These are simple transposition ciphers and were used in the past.

There are two methods for permutation of characters:

- **In the first method, the text is written into a table column by column and then transmitted row by row.**

- **In the second method, the text is written into a table row by row and then transmitted column by column.**

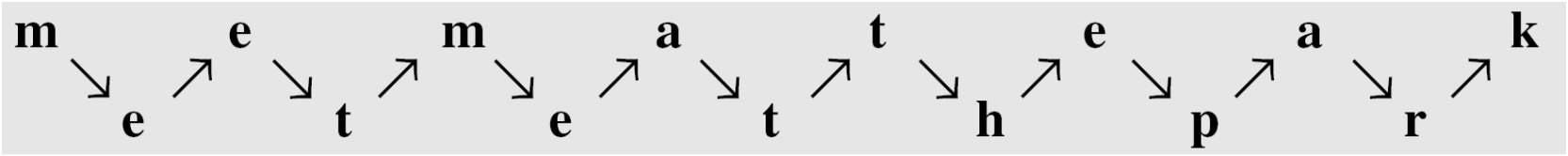
Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

Keyless Transposition Ciphers

Example:

1st Method: Written column by column and transmitted row by row

- A good example of a keyless cipher using the first method is the rail fence cipher.
- In this cipher, the plaintext is arranged in two lines as a zigzag pattern (which means column by column).
- The ciphertext is created reading the pattern row by row. For example, to send the message "Meet me at the park" to Bob, Alice writes-



By sending the first row followed by the second row, Alice then creates the ciphertext "MEMATEAKETETHPR".

Bob receives the ciphertext and divides in half (in this example, the second half has one less character). The first half forms the first row; the second half forms the second row. She reads the result in zigzag.

Because there is no key and the number of rows is fixed (2 here), the cryptanalysis of the ciphertext would be very easy for Eve.

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

Keyless Transposition Ciphers

Example:

2nd Method: Written row by row and transmitted column by column

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

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

By transmitting the characters column by column, Alice then creates the ciphertext "MMTAEEHREAEKTTP".

Bob receives the ciphertext and follows the reverse process. He writes the received message column by column and reads it row by row as the plaintext.

Eve can easily decipher the message if she knows the number of columns.

Keyless Transposition Ciphers

Example:

The cipher in the previous example is actually a transposition cipher. The following shows the permutation of each character in the plaintext into the ciphertext based on the positions.

Plaintext	:	**m**	**e**	**e**	**t**	**m**	**e**	**a**	**t**	**t**	**h**	**e**	**p**	**a**	**r**	**K**
Source Position	:	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15
		↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
Destination Position	:	01	05	09	13	02	06	10	14	03	07	11	15	04	08	12
Ciphertext	:	M	M	T	A	E	E	H	R	E	A	E	K	T	T	P

- The first character in the plaintext has not changed its position. The second character in the plaintext has moved to the fifth position in the ciphertext; the third character has moved to the ninth position; and so on.
- Although the characters are permuted, there is a pattern in the permutation: (01, 05, 09, 13), (02, 06, 10, 14), (03, 07, 11, 15), and (04, 08, 12). In each section, the difference between the two adjacent numbers is 4.

Keyed Transposition Ciphers

- The keyless ciphers permute the characters by using writing plaintext in one way (e.g. row by row) and reading it in another way (e.g. column by column). The permutation is done on the whole plaintext to create the whole ciphertext.
- In keyed transposition cipher, the plaintext is divided into groups of predetermined size, called blocks, and then use a key to permute the characters in each block separately.

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

Keyed Transposition Ciphers

Example:

- Alice needs to send the message "Enemy attacks tonight" to Bob.
- They have agreed to divide the text into groups of five characters and then permute the characters in each group.
- To make the last group the same size, a bogus character ***z*** is added at the end.

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

The key used for encryption and decryption is a permutation key, which shows how the character are permuted. For the above message, we assume that Alice and Bob used the following key:

Encryption ↓						↑ Decryption
	3	1	4	5	2	
	1	2	3	4	5	

The third character in the plaintext block becomes the first character in the ciphertext block. Similarly, the first character in the plaintext block becomes the second character in the ciphertext block , and so on.

The permutation yields:

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

Alice sends the ciphertext "EEMYNTAACTTKONSHITZG" to Bob. Bob divides the ciphertext into 5-character groups and finds the plaintext using the key in reverse order.

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

Columnar Transposition Ciphers

- This type of transposition cipher combines the keyless and keyed transposition ciphers to achieve better scrambling.
- Encryption or decryption is done in three steps:
    1. The text is written into a table row by row.
    2. The permutation is done by reordering the columns.
    3. The new table is read column by column.
- Here, the 1st and 3rd steps provide a keyless global reordering and the 2nd step provides a blockwise keyed reordering.

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

Columnar Transposition Ciphers

Example:

Encrypt the message "*enemy attacks tonight*" using Columnar transposition cipher.

Solution: The encryption and decryption is illustrated in the figure below.

- The 1st table in the figure is created by Alice writing the plaintext row by row.
- The columns are permuted using the key mentioned.
- The ciphertext is created by reading the 2nd table column by column.
- Bob does the same three steps in the reverse order.

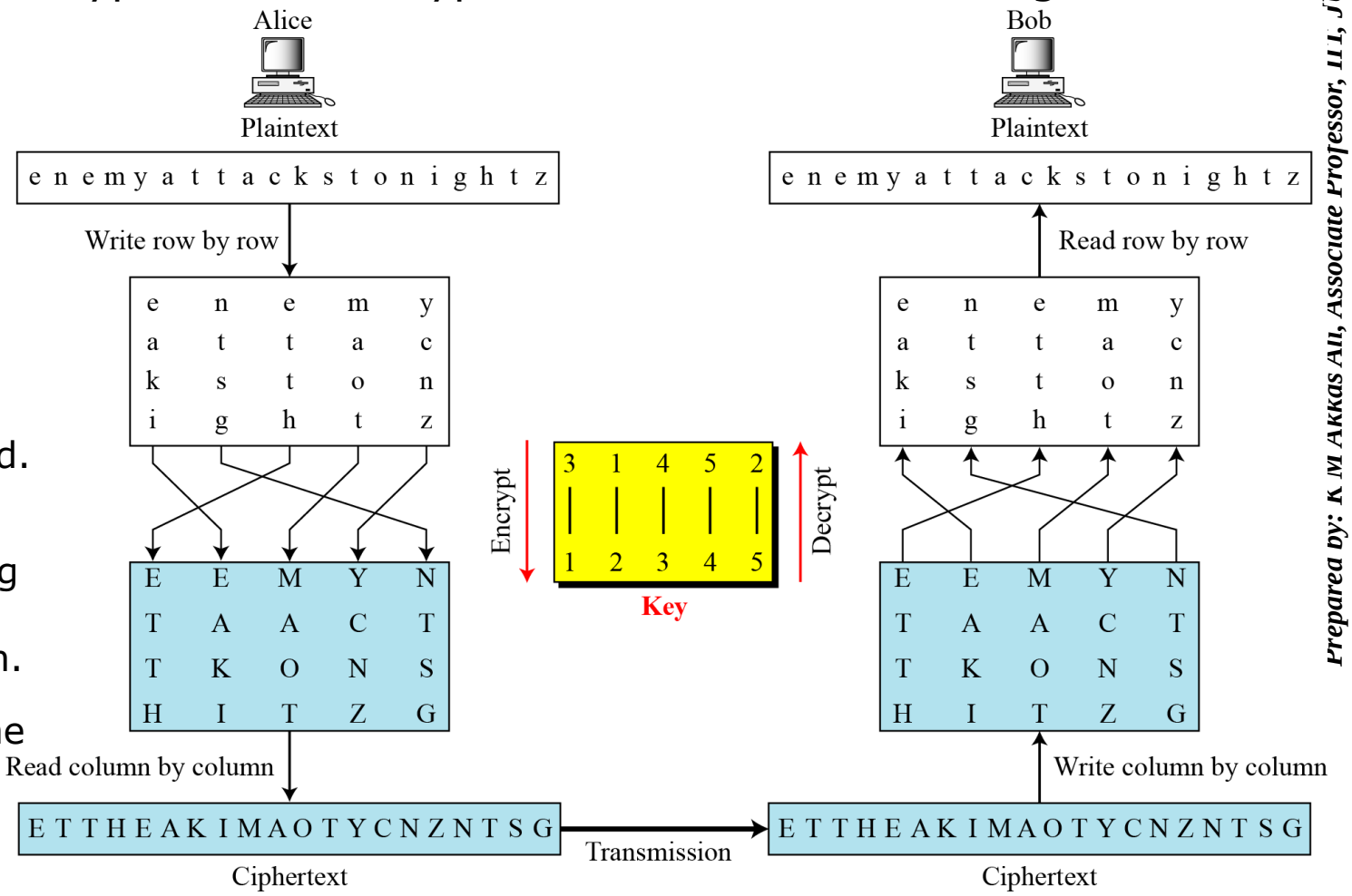


Figure: *Columnar transposition cipher*

Key inversion in a transposition cipher

- How can the inverse of a key be created if the initial or original key is given, or vice versa?
- The process can be done manually in a few steps.
- Figure below shows how to invert an encryption key, i.e. how to find the decryption key.

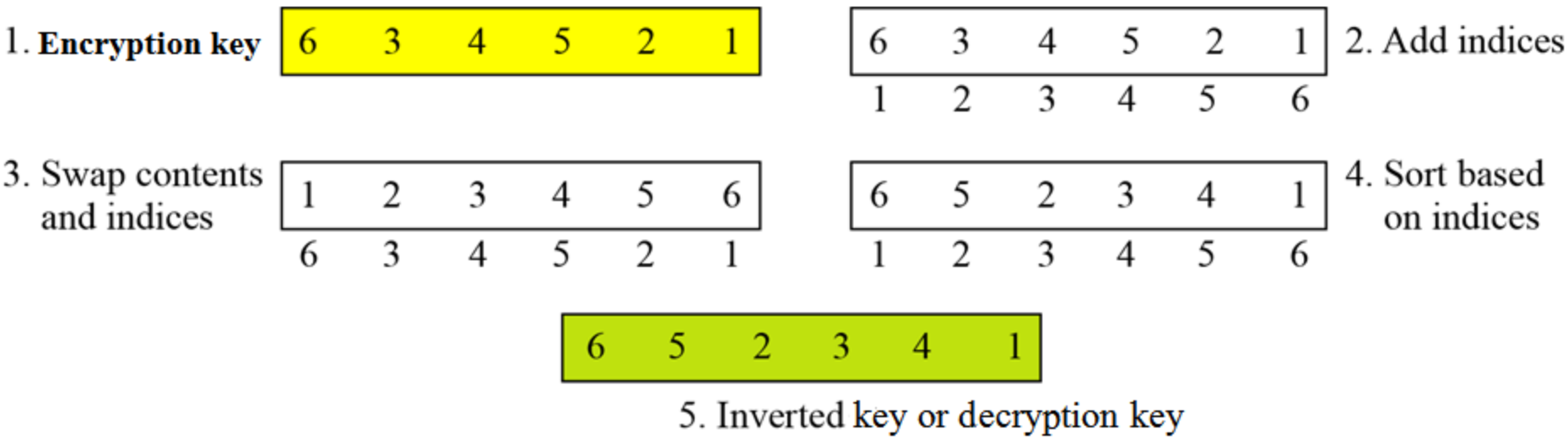


Figure: Inverting a permutation table

Traditional Vs. Modern Symmetric-key Ciphers:

- The traditional symmetric-key ciphers were used in the past. They are character-oriented ciphers.
- Traditional ciphers are simpler than modern ciphers and easier to understand. They can be easily attacked using a computer.
- Two broad categories of traditional symmetric-key ciphers are:
  - Traditional substitution ciphers
  - Traditional transposition ciphers
- Now-a-days, the information to be encrypted is not just text; it can also consist of numbers, graphics, audio, and video data. It is convenient to convert these types of data into a stream of bits, to encrypt the stream, and then to send the encrypted stream. So, we need bit-oriented ciphers.
- When data is treated as the collection of bits, it becomes larger. Mixing a larger number of symbols increases security.
- Two broad categories of modern symmetric-key ciphers are:
  - Modern stream ciphers
  - Modern block ciphers

Stream Ciphers:

- Stream cipher encrypts a single character or bit of plaintext at a time. It also decrypts a single character or bit of ciphertext at a time.

Example:

- Given plaintext: 1001101111010000 1
  Let the keystream be a stream of 1s and 0s.
- If we use an exclusive or (XOR) with the keystream and plaintext, we get ciphertext.
- This keystream is called periodic, since the sequence '10' repeats over and over.

Plaintext : 1001101111010000 1
Keystream : 1010101010101010 1
Ciphertext : 0011000101111010 0 (by XORing each plaintext bit with corresponding keystream bit)

- To decrypt this ciphertext, all we need to do is again XOR the ciphertext with the keystream:

Ciphertext : 0011000101111010 0
Keystream : 1010101010101010 1
Plaintext (XOR) : 1001101111010000 1

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

Block Ciphers:

- A symmetric-key modern block cipher encrypts an **n**-bit block of plaintext or decrypts an **n**-bit block of ciphertext together.
- The decryption algorithm must be the inverse of the encryption algorithm.
- If the message has the fewer than **n** bits, padding must be added to make it an **n**-bit block. If the message has more than **n** bits, it should be divided into **n**-bit blocks and the appropriate padding must be added to the last block if necessary.
- The common values of **n** are 64, 128, 256, or 512 bits.

Example

Plaintext : The only thing we have to fear is fear itself

Modified plaintext : Theonlythingwehavetofearisfearitself

Plaintext blocks : Theonlyt hingweha vetofear isfearit selfXend (break the plaintext into 8-character block)

Ciphertext blocks : tylnoehT ahewgnih raefotev tiraefsi dneXfles **(just reverse each plaintext block)**

Ciphertext : tylnoehTahewgnihraefotevtiraefsidneXfles

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

Data Encryption Standard (DES)

- The Data Encryption Standard (DES) is a symmetric-key block cipher designed by IBM and published by the National Institute of Standards and Technology (NIST).
  - It is a 64 bit block cipher with key length 56 bits.
  - In DES, the plaintext input bit string is divided into 64-bit blocks and each block is encrypted using the same 56-bit key. The same key is used for decryption.
  - It consists of a number of rounds where each round contains bit-shuffling, non-linear substitutions (S-boxes) and exclusive OR operations. These are done in order to increase the difficulty of performing a cryptanalysis on the cipher.
- DES had been used as a standard method of encryption until 2000, but with increase in speed in computers, it is no more considered secure since a cryptanalyst can break the code by exhaustively searching for all the keys using a fast computer.
- However, a modification of DES, called triple DES (or 3 DES), is now used which is more secure and is difficult to break.
- After 25 years of analysis, the only security problem with DES found is that its key length is too short.
- From 2001, DES has been replaced by a new standard known as the Advanced Encryption Standard (AES).

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

DES Algorithm/DES Structure:

- In DES, plaintext message is broken into blocks of length 64 bits. Each 64-bit block of plaintext is encrypted using a 56-bit key.
- A 56-bit key k is fed into a subkey generating algorithm to produce 16 round subkeys k_1, k_2, k_3, ……., k_{16} of length 48 bits each.
  - At first, an initial permutation (IP) is performed on the 64-bit block of plaintext. (The initial permutation rearranges the bits of the plaintext to form the "permuted input" based on the IP table).
  - After initial permutation, the 64-bit permuted block is divided into two 32-bit sub-blocks represented by L_0 and R_0 as the left and right sub-block respectively.
  - The encryption then proceeds through 16 rounds of identical operations using a different sub-key of length 48-bit in each round on the left and right halves of the block.
  - The output found using key k_i after i_{th} round is represented by L_i and R_i respectively where i=1, 2, 3, …..,16. Round i has input $L_{i-1}||R_{i-1}$ and output $L_i||R_i$ where
    - $L_i=R_{i-1}$
    - $R_i=L_{i-1} \oplus f(R_{i-1},k_i)$

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

DES Algorithm/DES Structure (continued…):

- In the final round, the left (L) and right (R) halves are swapped, so that the decryption algorithm has the same structure as the encryption algorithm.
- After the final round (16^{th} round), the right and left halves are joined or concatenated.
- Then, a final permutation IP^{-1} (which is the inverse of the initial permutation), is applied to the 64-bit joining block. The output of this final permutation is the 64 bit encrypted output (ciphertext).

➢ Figure on the next slide illustrates how the algorithm works.

➢ Decryption is identical to encryption, except that the subkeys are used in the opposite order. That is, subkey 16 is used in round 1, subkey 15 is used in round 2, etc., ending with subkey 1 being used in round 16.

➢ Each 64-bit block of plaintext is encrypted using a 56-bit key.

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

DES Algorithm/DES Structure (continued…):

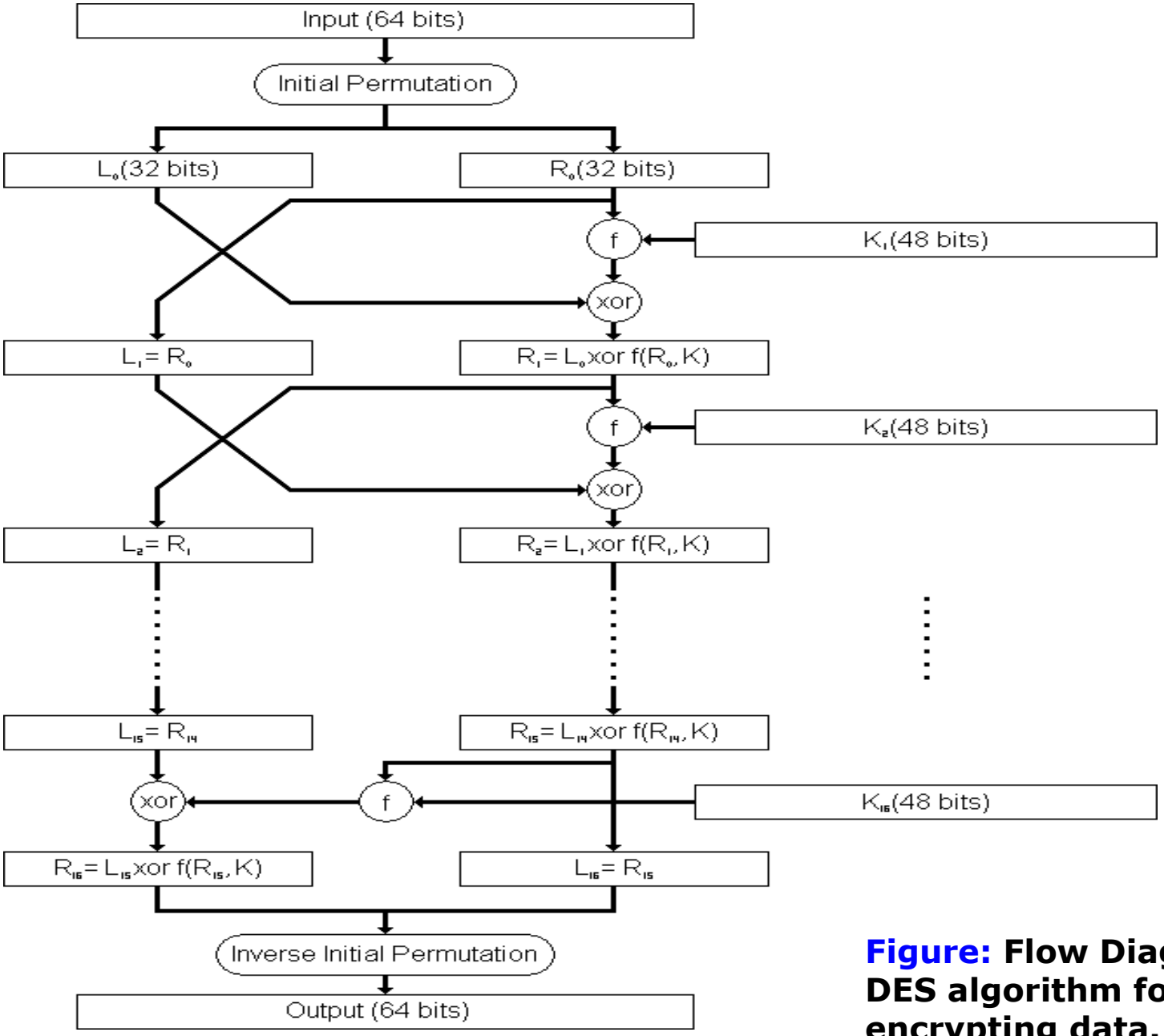


Figure: Flow Diagram of DES algorithm for encrypting data.

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

DES Round Structure /DES Round Function $f(R_{i-1},K_i)$:

- In DES, substitution and permutation are used a number of times in iterations called rounds. Generally, the more rounds there are, the more secure the algorithm is.
- The heart of DES is DES round function which mixes the bits of the right (R) portion using the subkey for the current round.
- It applies a 48-bit key to the rightmost 32 bits (R_{i-1}) to produce a 32-bit output.
- The round function is made up of four sections:
    1. An expansion P-box (E-box)
    2. A whitener (Exclusive-or that adds key)
    3. A group of S-boxes (for 48 bit to 32 bit conversion)
    4. A straight permutation P-box

DES Round Function $f(R_{i-1},K_i)$:

1. The E-box expansion permutation:

- Since R_{i-1} is a 32-bit input and K_i is a 48-bit key, we first need to expand R_{i-1} to 48 bits.
- So, the 32-bit R value is expanded to 48 bits using an expansion P-box permutation table which is necessary for combination with the 48 bit subkey.

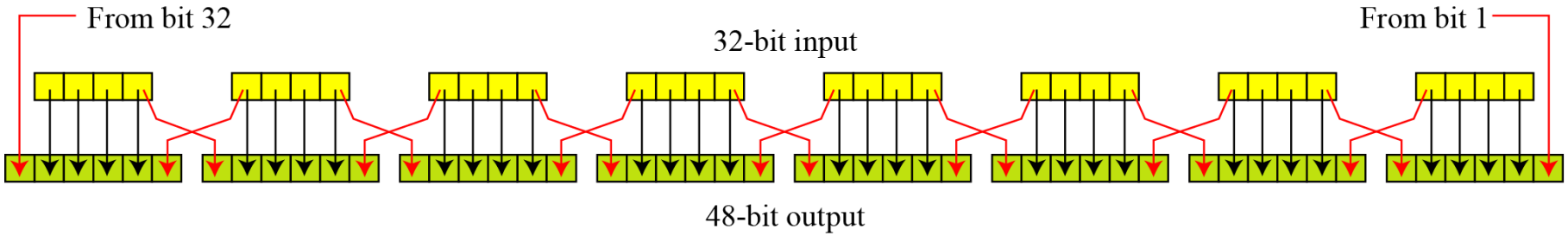


Figure: Expansion permutation

2. Whitener (Exclusive-or):

- Expanded R value found from section-1 is then exclusive-or'ed with the 48-bit subkey.

DES Round Function $f(R_{i-1}, K_i)$:

3. The S-box substitution: 48 bits to 32 bits

- The S-boxes do the real mixing (confusion). DES uses 8 S-boxes, each with a 6-bit input and a 4-bit output.
- This is a highly important substitution which accepts a 48-bit input and outputs a 32-bit number.
- The resulting 48 bits found from section-2 are divided into eight 6-bit chunks, each of which is fed into an S-Box that mixes the bits and produces a 4-bit output. Those 4-bit outputs are combined into a 32-bit value.

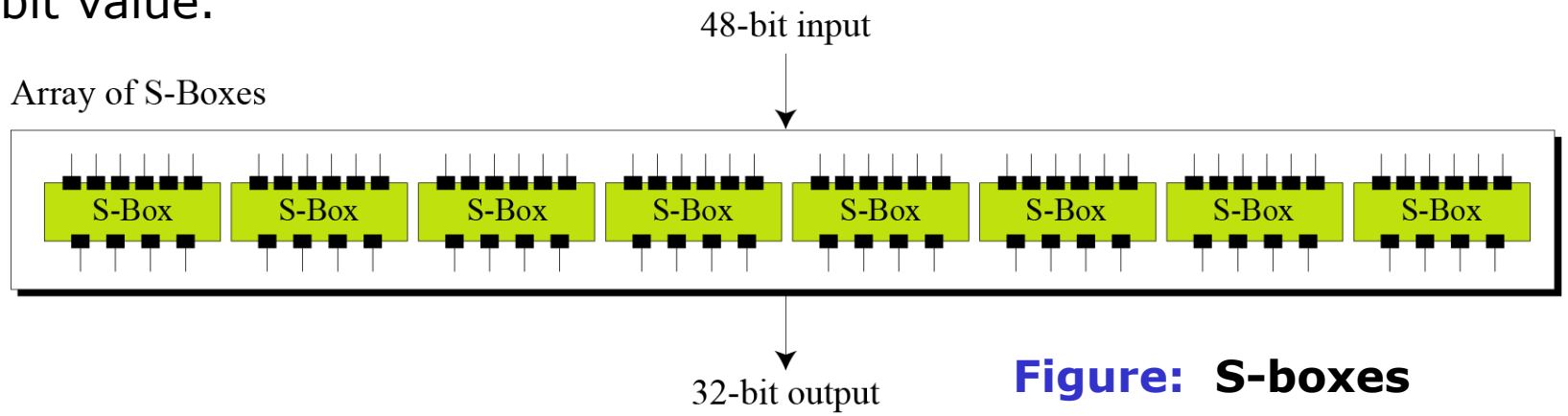


Figure: S-boxes

4. Straight Permutation (P-box):

- The combined 32 bits found from section-3 are permuted once again to produce the 32 bits output of the round function using expansion P-box table.

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

DES Round Function $f(R_{i-1},K_i)$:

- Figure below shows the DES round function f(Ri-1,Ki).

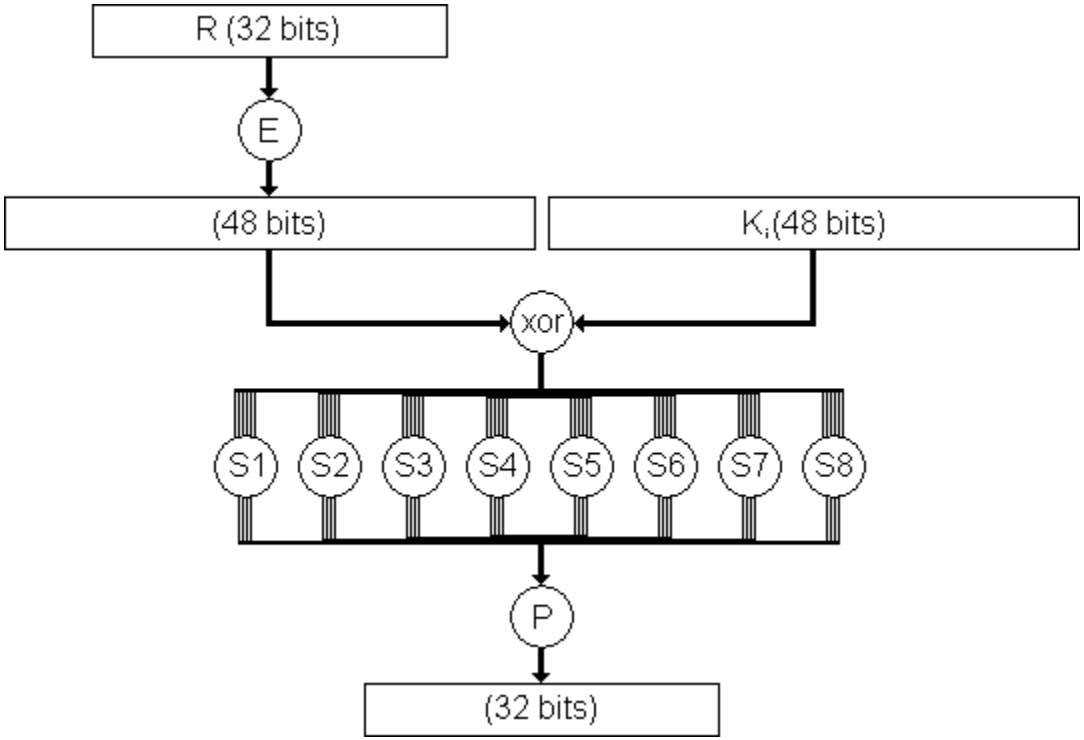


Figure: The round function $f(R_{i-1},K_i)$ of the DES algorithm.

Note:

- The s-boxes provide the "confusion" of data and key values, whilst the permutation P then spreads this as widely as possible, so each S-box output affects as many S-box inputs in the next round as possible, giving "diffusion".

RSA Cryptosystem

- RSA is the most commonly used public-key cryptography algorithm, which uses prime factorization as the trapdoor one-way function.
- It is named so after the surnames of its inventors Ron Rivest, Adi Shamir, and Leonard Adleman of the Massachusetts Institute of Technology (MIT).
- It was first published in 1978.
- This algorithm relies on one way function. A one way function is easy to compute but hard to invert. For example it is easy to take the product of two prime numbers but given the product it is difficult to split it into the original prime factors.
- This algorithm lets you choose the size of your public key.
- The 512-bit keys are considered insecure or weak, but the 768-bit keys are secure from everything but the National Security Administration (NSA).
- The 1024-bit keys are secure from everything virtually.
  - RSA is embedded in major products such as Windows, Netscape Navigator etc.
  - It is used in digital signature and other cryptosystems that often need to encrypt a small message without having access to a symmetric key. RSA is also used for authentication
  - Although RSA can be used to encrypt and decrypt actual messages, it is very slow if the message is long. Therefore, RSA is useful for short messages.

Steps in RSA Algorithm

- The RSA algorithm involves three steps:
  1. Key generation (Generating public and private key)
  2. Encryption (Encrypting the message)
  3. Decryption (Decrypting the message)
- RSA involves a public key and a private key.
  - The public key can be known by everyone and is used for encrypting messages.
  - Messages encrypted with the public key can only be decrypted in a reasonable amount of time using the private key.

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

RSA Algorithm: Key Generation

The keys for the RSA algorithm are generated by the following ways:

1. Choose two large and distinct prime numbers ***p*** and ***q***.

- For security purposes, the integers ***p*** and ***q*** should be chosen at random, and should be of similar bit-length.
- In RSA, ***p*** and ***q*** must be at least 512 bits; ***n*** must be at least 1024 bits.
- Prime integers can be efficiently found using a primality test.

2. Compute ***n*** = p * q

- ***n*** is used as the modulus for both the public and private keys. Its length, usually expressed in bits, is the key length.

3. Compute the number of integers less than ***n*** that are coprime with ***n*** (otherwise known as the totient or Euler's Phi function):

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

4. Choose an integer ***e*** such that **1 < e < φ(n)** and **gcd(e, φ(n)) = 1**; i.e. ***e*** and ***φ(n)*** are coprime.

- e is released as the public key exponent (encryption exponent).
- e having a short bit-length results in more efficient encryption– most commonly $2^{16} + 1 = 65,537$. However, much smaller values of e (such as 3) have been shown to be less secure in some settings.

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

RSA Algorithm: Key Generation

5. Determine the multiplicative inverse ***d*** of ***e***; i.e., compute a value for ***d*** such that it satisfies the relation: (***d*** * ***e***) mod φ(n) = 1

- ***d*** is kept as the private key exponent (decryption exponent).
- ***d*** is often computed using the extended Euclidean algorithm.
- ***d*** must be kept secret.
- ***p***, ***q***, and ***φ(n)*** must also be kept secret because they can be used to calculate ***d***.

6. The public key consists of the modulus ***n*** and the public key exponent ***e***; i.e., the public key is **(e, n).**

7. The private key consists of the modulus ***n*** and the private key exponent ***d***; i.e., the private key is **(d, n).**

8. To encrypt message ***m*** using the public key, use the relation:

$$\mathbf{c} = m^{\mathbf{e}} \bmod \mathbf{n}$$

9. To decrypt ***c*** using the private key, use the relation:

$$m = c^{\mathbf{d}} \bmod \mathbf{n}$$

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

RSA Algorithm: Encryption

- Bob transmits his public key (***e***, ***n***) to Alice and keeps the private key (***d***, ***n***) secret.
- Alice then wishes to send message ***M*** to Bob.
- The message is encrypted by the following ways:

1. Alice first turns message ***M*** into an integer ***m***, such that $0 \leq m < n$.

- That is, the message is represented as an integer between 0 and (n-1).
- Large messages can be broken up into a number of blocks. Each block would then be represented by an integer in the same range.

2. After turning the message into integer, Alice then computes the ciphertext ***c*** using the following relation:

$$c = m^e \bmod n$$

3. After computing ciphertext, Alice then transmits ***c*** to Bob.

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU

RSA Algorithm: Decryption

- Bob can recover ***m*** from ***c*** by using his private key exponent ***d*** using the following relation:

$$m = c^d \bmod n$$

- After having ***m***, Bob can recover the original message ***M*** by reversing the padding scheme.
- The encryption, decryption and key generation in RSA is shown in the figure below.

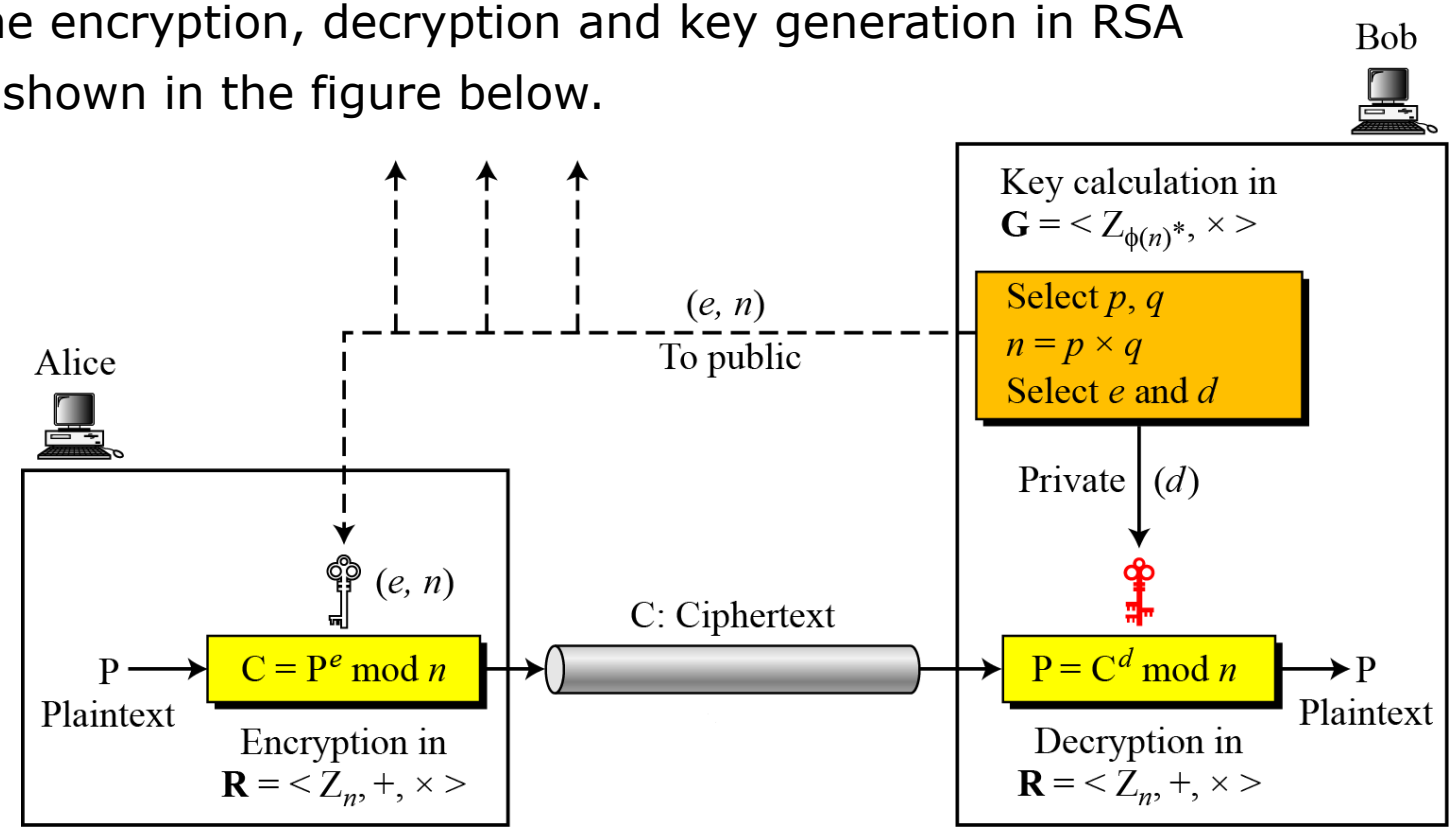


Figure: Encryption, decryption, and key generation in RSA

RSA Cryptosystem: Trivial Examples

1. Choose p = 3 and q = 11
2. Compute n = p * q = 3 * 11 = 33
3. Compute φ(n) = φ(p*q) = φ(p)* φ(q)=(p - 1) * (q - 1) = 2 * 10 = 20
4. Choose e such that 1 < e < φ(n) and e and φ(n) are coprime. We have several choices for e: 7, 11, 13, 17, 19. (We cannot use 5 as e, because 20 is divisible by 5). Let e = 7
5. Compute a value for d such that (d * e) mod φ(n) = 1. One solution is d = 3 [(3 * 7) % 20 = 1] [d is the multiplicative inverse of e]
6. Public key is (e, n) => (7, 33)
7. Private key is (d, n) => (3, 33)
8. The encryption of m = 2 is $c = m^e \bmod n = 2^7 \bmod 33 = 29$
9. The decryption of c = 29 is $m = c^d \bmod n = 29^3 \bmod 33 = 2$

Prepared by: K M Akkas Ali, Associate Professor, IIT, JU