Encryption Algorithm for RTA

Under guidance of Prof. Pramila Chauhan

By

Soham Gandhi	131070016
Pavan Birajdar	131070017
Chinmay Karmalkar	131070035
Mohil Mehta	131070038
Abhay Kamath	131070057

What is RTA?

- Real Time Applications (RTA) is an application program that functions within a time frame that user senses as an immediate action.
- It contains high volume of data, hence, classical encryption techniques are not appropriate.
- Most RTAs are implemented on the Internet so encryption and decryption techniques have to take minimal time.

What is Encryption?

- Encryption is the process of encoding messages in such a way that only authorized parties can read it.
- In encryption scheme, the intended information is encrypted using encryption algorithm generating cipher text.
- The purpose of this is to the secure the information from attackers.

Objective

- To provide Information Security for communication between client and server.
- Security goals: Confidentiality, Authentication, Integrity, Non Repudiation and Access Control.
- Main objective is to implement an algorithm to provide better security than existing algorithms.

Types of Encryption algorithm

- 1. Symmetric:
  - Same key shared between sender and receiver.
  - Example: DES.
- 2. Asymmetric:
  - Different keys for sender and receiver.
  - Example: RSA.

Goal of algorithm

- Minimise time needed to encrypt/decrypt data.
- Maximise security for protection from hackers.

Drawbacks of Existing Algorithms

- RTA are not suitable for existing algorithms because of long delay imposed on packets.
- DES (56 bit key) takes less time but can be easily broken.
- AES (192 bit key) is secure but needs more time.

Proposed Algorithm Features

- Minimum message delay which is less than 400 ms.
- Better security level than standard algorithms. Key is 1024 bit long and is randomly chosen which makes it harder to intercept.
- Different key for each packet which makes it difficult to be guessed.

Steps of the algorithm

1. Initiate the connection
2. Key generation
3. XoR Encryption Process
4. Index Generator Table Shifting
5. Index Generation Process
6. Key Insertion Process
7. Sending Packets
8. Receiver's Process
9. Key Extraction
10. Decryption Process

1. Initiate the Connection

- By sending an initial packet to communicating party
- Initial packet depends on operation performed at sender side
- Sender and Receiver both have same 16x16 table containing a value (0-9) in every cell

16*16 Table

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

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

2. Key Generation

- Sender randomly chooses 1024 bit(128 Byte) key of his choice

3. XOR Encryption

- The sender performs XoR operation over 1024 bit key and data

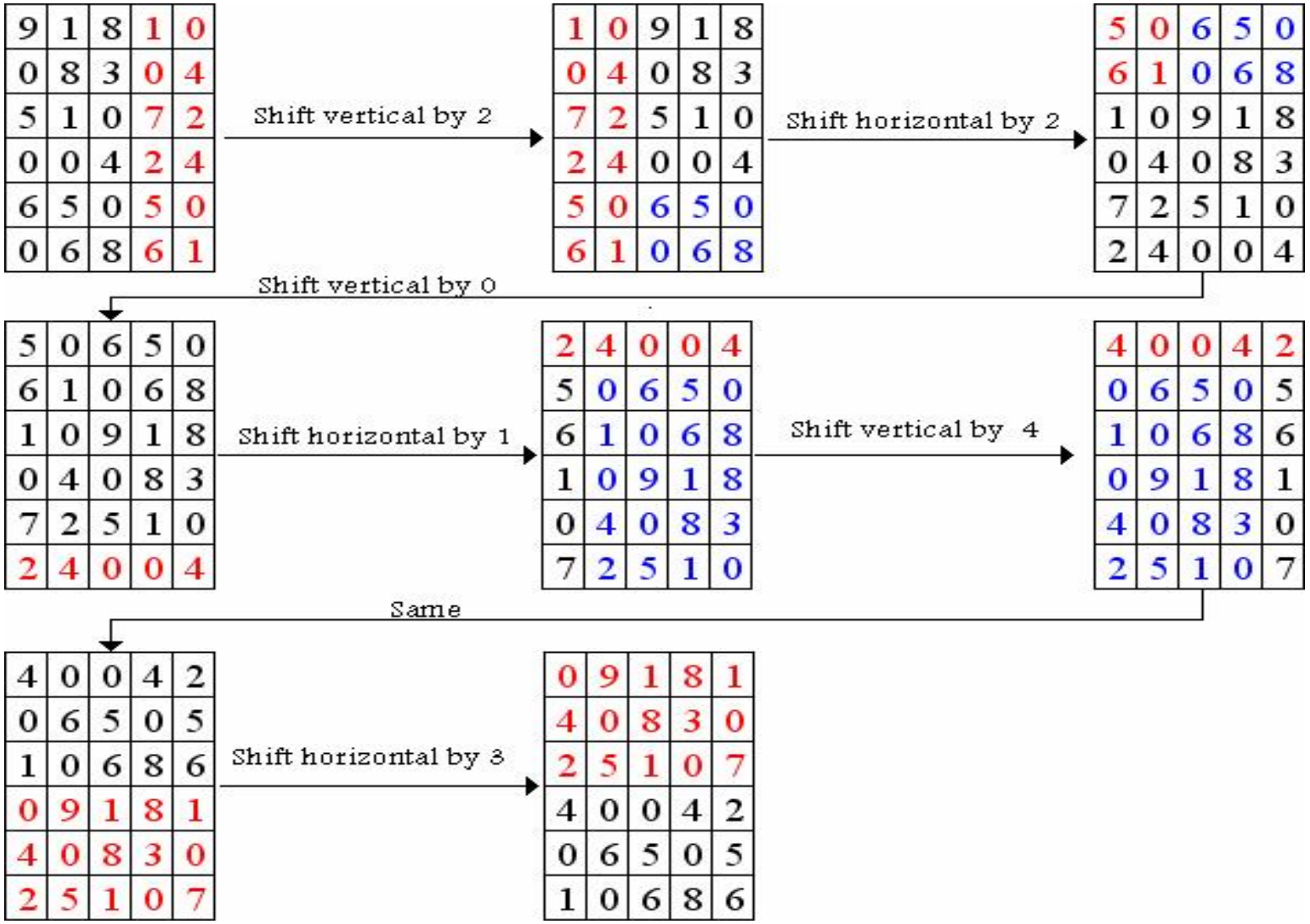
4. Index Generator Table Shifting

- Sender shifts the initial table columns circular right followed by rows circular down according to shared value between sender and receiver.

- Shared values of shifting are exchanged between sender and receiver previously.

Shifting Process Example

- Assume a small shared value like (2, 2, 0, 1, 4, and 3), and a small table size (5*6)



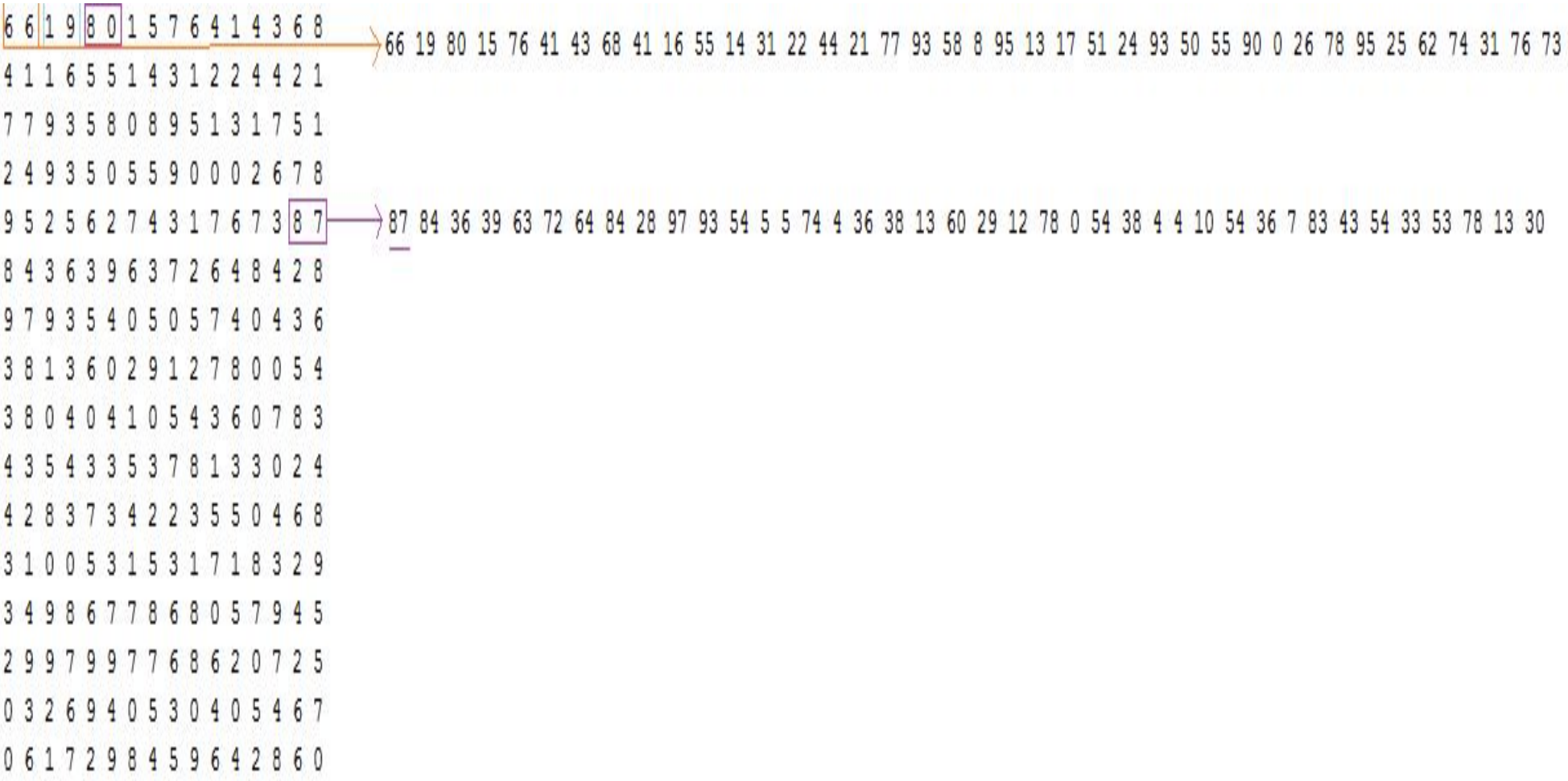
After applying column shift right 1…..

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

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

5. Index Generation

- Sender chooses a large number that represents indexes out of shifted initial table
- 2 consecutive digits are grouped together from the 256 digits to form 128 indexes


6 6 1 9 8 0 1 5 7 6 4 1 4 3 6 8
66 19 80 15 76 41 43 68 41 16 55 14 31 22 44 21 77 93 58 8 95 13 17 51 24 93 50 55 90 0 26 78 95 25 62 74 31 76 73
4 1 1 6 5 5 1 4 3 1 2 2 4 4 2 1
7 7 9 3 5 8 0 8 9 5 1 3 1 7 5 1
2 4 9 3 5 0 5 5 9 0 0 0 2 6 7 8
9 5 2 5 6 2 7 4 3 1 7 6 7 3 8 7
87 84 36 39 63 72 64 84 28 97 93 54 5 5 74 4 36 38 13 60 29 12 78 0 54 38 4 4 10 54 36 7 83 43 54 33 53 78 13 30
8 4 3 6 3 9 6 3 7 2 6 4 8 4 2 8
9 7 9 3 5 4 0 5 0 5 7 4 0 4 3 6
3 8 1 3 6 0 2 9 1 2 7 8 0 0 5 4
3 8 0 4 0 4 1 0 5 4 3 6 0 7 8 3
4 3 5 4 3 3 5 3 7 8 1 3 3 0 2 4
4 2 8 3 7 3 4 2 2 3 5 5 0 4 6 8
3 1 0 0 5 3 1 5 3 1 7 1 8 3 2 9
3 4 9 8 6 7 7 8 6 8 0 5 7 9 4 5
2 9 9 7 9 9 7 7 6 8 6 2 0 7 2 5
0 3 2 6 9 4 0 5 3 0 4 0 5 4 6 7
0 6 1 7 2 9 8 4 5 9 6 4 2 8 6 0

6. Key Insertion

- Sender inserts and shuffles the key into the XORed data according to generated indexes.
- Result output is a collection of padding and mixing of key and XORed data together.
- This provides high level “Confusion”.

7. Sending Packets

- Data is ready to be sent to receiver.
- It is sent as an ordinary network packet.

8. Receiver's Process

- Upon receiving packet, he extracts the key back from the shuffled data.
- The extraction process involves finding the relative position of each single key bit in data and put it back to right position

9. Key Extraction Process

- Receiver uses the same initial table and same shared values to shifting steps.
- Resulting output are the key and XOR sender data.

10. Decryption Process

- After getting the key, the received packet is XORed with the key to get back the original data

Timeline

Sr.No	Task	Start Date	End Date
1	Client Server Communication		
2	GUI		
3	Encryption at client side		
4	Decryption at server side		
5	Testing		
6	Reworking		

Thank You!