Enhance the Security of Extended Playfair Cipher using Extended RSA for Secure Data Transmission

A Project Report submitted in partial fulfilment of the requirements for the award of the degree of

Bachelor of Technology
in
Computer Science and Engineering

by
Name : Ayush Sharma
Roll No: 191500201

Name : Piyush Kumar Singh
Roll No: 191500541

Name : Hemant Gautam
Roll No: 191500776

Name: Yashraj Singh
Roll No: 191500940

Group No : 140

Under the Guidance of
Dr. Neeraj Varshney
(Assistant Professor & Assistant Director – IQAC)

Department of Computer Engineering & Applications
Institute of Engineering & Technology

GLA University
Mathura- 281406, INDIA

December, 2022

DECLARATION

I/we hereby declare that the work which is being presented in the Bachelor of technology. Project **"Enhance the Security of Extended Playfair Cipher using Extended RSA for Secure Data Transmission"**, in partial fulfilment of the requirements for the award of the **Bachelor of Technology (Computer Science & Engineering)** and submitted to the Department of Computer Engineering and Applicationsof GLA University, Mathura, is an authentic record of my/our own work carried underthe supervision of **Dr. Neeraj Varshney , Assistant Professor & Assistant Director - IQAC (Institute of Engineering & Technology) , GLAUniversity.**

The contents of this project report, in full or in parts, have not beensubmitted to any other Institute or University for the award of any degree.

1 – Sign : *Ayush Sharma*

Name of Candidate: Ayush Sharma

University Roll No: 191500201

2 – Sign : *Piyush Kumar Singh*

Name of Candidate: Piyush Kumar Singh

University Roll No: 191500541

3 – Sign : *Hemant Gautam*

Name of Candidate: Hemant Gautam

University Roll No: 191500776

4 – Sign : *Yashraj Singh*

Name of Candidate: Yashraj Singh

University Roll No: 191500940

ACKNOWLEDGEMENT

Presenting the ascribed project paper report in this very simple and official form, we would like to place my deep gratitude to GLA University for providing us the instructor **Dr. Neeraj Varshney , Assistant Professor & Assistant Director - IQAC (Institute of Engineering & Technology)** , our technical trainer and supervisor.

He has been helping us since Day 1 in this project. He provided us with the roadmap, the basic guidelines explaining on how to work on the project. He has been conducting regular meeting to check the progress of the project and providing us with the resources related to the project. Without his help, we wouldn't have been able to complete this project.

And at last but not the least we would like to thank our dear parents for helping us to grab this opportunity to get trained and also my colleagues who helped me find resources during the training.

Thanking You

1 – Sign : *Ayush Sharma*

Name of Candidate: Ayush Sharma

University Roll No: 191500201

2 – Sign : *Piyush Kumar Singh*

Name of Candidate: Piyush Kumar Singh

University Roll No: 191500541

3 – Sign : *Hemant Gautam*

Name of Candidate: Hemant Gautam

University Roll No: 191500776

4 – Sign : *Yashraj Singh*

Name of Candidate: Yashraj Singh

University Roll No: 191500940

CERTIFICATE

This is to certify that the project entitled **"Enhance the Security of Extended Playfair Cipher using Extended RSA for Secure Data Transmission"**, carried out in **Final Year Project** for the academic session **2022 - 23** , is a bonafide work by **Ayush Sharma , Piyush Kumar Singh , Hemant Gautam & Yashraj Singh** that is submitted in partial fulfilment of the requirements for the award of the degree **Bachelor of Technology (Computer Science & Engineering).**

Signature of Supervisor:

Name of Supervisor: Dr.Neeraj Varshney

Assistant Professor & Assistant Director - IQAC (Institute of Engineering & Technology)

Date: Dec 10 , 22

ABSTRACT

In today's digital world cryptography is used to secure information in order to provide the privacy for the intended sender and receiver by managing the message with the public key. The objective of this work is to securing the key of Playfair cipher using Extended RSA algorithm. The existing methods of Playfair cipher modified by increasing in the size of matrix, so We will use 16X16 Matrix for pick cipher characters. It makes use of alphabets both lower and uppercase characters, number and special characters for constructing the contents of the matrix. Enhance the security of Extended Playfair cipher using 16X16 matrix by implementing the Extended RSA Algorithm in the 16X16 Playfair. which will use the asymmetric key. Which is based on public and private key encryption technique . it use two different keys to encrypt and decrypt the message. The idea of RSA Algorithm is it takes only two prime number P and Q while In Extended RSA here we can take multiple prime number for Encrypt the Playfair key which will make it more secure than the RSA Algorithm. Finally, the security strength of the whole system has been analyzed and tried to fulfil the requirement of security. The Motivation of the project is that we can Send our information form one channel to another channel with fully protected manner so that no one can know the our confidential information like id , Passwords , Bank Balance etc This Algorithm can implemented in startups , individuals Growing Business for keep safe their confidential information to Any One.

CONTENTS

Chapter 1

Introduction

1.1 Overview and Motivation

The word cryptography comes from the Greek origin. It is a combination of two words Crypto and Graphy. Crypto means Secret and Graphy means Writing . Cryptography deals with creating documents that can be shared secretly over public communication channels. The present Scenario, everyone needs to encrypt the message at the sender side and decrypt it at the receiver side to preserve security and privacy. So cryptography is the study of creating and using encryption and decryption techniques. In cryptography the term plaintext is used for the original message that is to be transformed. The message which has been transformed is called Ciphertext. An encryption algorithm works with a key to transform the plaintext into ciphertext. Decryption algorithm works in the reverse order and converts the ciphertext into plaintext. The encryption /decryption algorithm is to encrypt/decrypt the message with the help of a key. The process of converting plaintext into ciphertext is called enciphering or encryption. The process of retaining the plaintext from the ciphertext is called deciphering or decryption. The following figure shows the encryption and decryption process. To secure the data, cryptography is the best choice [2]. Cryptography is a technique or a method operated to secure the data between the sender and receiver. Cryptography works by changing an ordinary-text (plain-text) into a new text that called cipher-text. This process called encryption. After a cipher-text reached by receiver, the cipher-text will be technically returned into plain-text. This process called decryption. Based on [3], one of the state-of-the-art of encryption is public-key cryptography; where it needs two separate keys. One key for locking or encrypting the plain-text, the other is for unlocking or decrypting the cipher-text. Rivest, Shamir, and Adlemen (RSA) encryption is one of public-key encryption methods. It uses private and public key as process key to encryption and decryption. For an example, if A wants to send a secret message to B, then A will encrypt the message using B's public key. After B received the message, he/she is going to decrypt the secret message using his/her private key. In transferring process, even third party has public key, he/she is still not able to decrypt the message. The Motivation of the project is that we can Send our information form one channel to another channel with fully protected manner so that no one can know the our confidential information like id , Passwords , Bank Balance etc This Algorithm can implemented in startups , individuals Growing Business for keep safe their confidential information to Any. We will use 16X16 Matrix for pick cipher characters. It makes use of alphabets both lower and uppercase characters, number and special characters for constructing the contents of the matrix. Enhance the security of Extended Playfair cipher using 16X16 matrix by implementing the Extended RSA Algorithm in the 16X16 Playfair. which will use the asymmetric key. Which is based on public and private key encryption technique.

1.2 Objective

The extended Euclidean algorithm is a security algorithm that is used in the RSA Algorithm. The extended Euclidean algorithm is a modified algorithm that is used to improve the security of the RSA Algorithm. Extended Euclidean Algorithm is a cryptographic algorithm that was designed by the National Security Agency and The National Security Agency Institute of Electrical and Electronics Engineers. The Algorithm is used to create a secure digital signature. Extended RSA will also use the asymmetric key. Which is based on public and private keyencryp on technique. The idea of RSA is the sender only needs to hold one public key that is used to encrypt the data, while In Extended RSA there can be mul ple private keys for each recipients. The Extended Version of RSA Algorithm is Called Extended RSA. Finally, the security strength of the whole system has been analyzed and tried to fulfil the requirement of security.As a child, you may have used secret messages or languages to communicate with friends or siblings, and you have likely observed the use of cryptography in various aspects of our society – maintaining the confidentiality of personal, consumer, corporate, and government data. However, on top of this, cryptography's status as an indispensable building block in digital infrastructure continues to grow with the perpetual increase in online connectivity – securing online transactions, authentication, and access to resources. The Scope of the project is that we can Send our information form one channel to another channel with fully protected manner so that no one can know the our confidential information like id , Passwords , Bank Balance etc This Algorithm can implemented in startups , individuals Growing Business for keep safe their confidential information to Any One who have not enough budget to buy a cyber security system

1.3 Issues and Challenges

1. Only Upper Case Letters Can Be Encrypt No Lower Case Letters Can Be Encrypted.
2. One letter of the alphabet (usually J) is omitted from the table (as the table can hold only 25 alphabets).
3. Symbols can't be encrypted.
4. A spare letter X is added when the plaintext word consists of odd number of character. In the decryption process this X is ignored. X is a valid character and creates confusion because it could be a part of plaintext, so we cannot simply remove X in decryption process.
5. X is used a filler letter while repeating letter falls in the same pair are separated.

Example : if we want to encrypt email ID "yash@gmail.com"

Here, "@" and "." Remain Unencrypted. Which makes it less secure.

To Overcome The Disadvantages Of PlayFair 5×5 Matrix The Extension of Playfair Cipher using 16X16 Matrix was Introduced.

THE PROPOSED PLAYFAIR CIPHER USING 16X16 MATRIX

This Playfair algorithm is based on the use of 16x16 matrix of characters constructed using a keyword. The matrix is constructed by filling the characters of keyword (minus duplicates)

from left to right and from top to bottom. Then it is filling the remaining characters in ascending order from ASCII value 0 to 255.

While repeating plaintext characters that are in the same pair, the first character is replaced by the character to the right, with the first element of the row circularly following the last. The second character is replaced by the character to the left, with the last element of the row circularly following the first. If a word consists of odd number of characters, it will add the character "Null" to complete the pairs, because "Null" character cannot affect the Plaintext at the time of decipherment.

This algorithm can accept the Plaintext containing Alphabets (capital letters and small letters), Numbers and Special characters. So the user can easily encrypt combination of alphabets, numbers and characters efficiently.

1. Should allows characters as keyword.
2. Should be considers the space between two words in plaintext as one character.
3. The user can easily encrypt and decrypt the combination of alphabets, numbers and special characters efficiently.
4. To Use Letters[A-Z , a-z], digits[0-9] and All special characters so we will use 0-255 ASCII Values to pick all the Characters.
5. Should be is case sensitive.
6. The letters I and J are considered as two different letters.
7. This algorithm cannot separate a repeating Plaintext letters with a filter letter.

1.4 Contribution

We are making the Research Based Project in this we are Enhancing the Security of Extended Playfair Cipher using Extended RSA for Secure Data Transmission. we can Send our information form one channel to another channel with fully protected manner so that no one can know the our confidential information like id , Passwords , Bank Balance. We are Making it more Secure and more strong previously we can take only two Prime number in RSA and in Extended RSA we can take 4 or 5 Prime number and due to this it is more Secure with Extended RSA no one is able to decrypt the message.

1.5 Organization of the Project Report

In First Chapter there is introduction of the project, Overview and Motivation of the Project, Objective of the Project, Issues and Challenges and Contribution.
In Second Chapter Literature Review comes in this all the related work of Project will be there after analyzing Four-Five Research Papers.
In Third Chapter there is Proposed Work in this Proposed Model , Algorithm, pseudo code ,Proposed Model and in this Every Point will be Described Properly and in last References will be there in this we give references.

Chapter 2
Literature Review

2.1 Authors and their proposed algorithm

[1]. The role of Cryptography in today's digital world is significant. It secures information mathematically by mangling message with key. The privacy of intended sender and receiver information is protected from eavesdropper. The objective of the paper is playfair cipher. The existing methods of playfair cipher are studied. The restrictions of earlier works a playfair cipher using 5X5 matrix, 7X4 matrix and 6X6 matrix are overcome in the proposed work. The proposed method plays a 16X16 matrix giving strength to playfair cipher. The proposed work is an enhancement to the existing algorithms that uses 16X16 matrix to pick cipher characters. It makes use of alphabets both lower and uppercase characters, number and special characters for constructing the contents of the matrix.
Etymologically speaking, the word cryptography comes from the Greek origin. It is a combination of two words Crypto and Graphy. Crypto means Secret and Graphy means Writing [1]. Cryptography deals with creating documents that can be shared secretly over public communication channels. The present Scenario, everyone needs to encrypt the message at the sender side and decrypt it at the receiver side to preserve security and privacy. So cryptography is the study of creating and using encryption and decryption techniques. In cryptography the term plaintext is used for the original message that is to be transformed. The message which has been transformed is called Ciphertext. An encryption algorithm works with a key to transform the plaintext into ciphertext. Decryption algorithm works in the reverse order and converts the ciphertext into plaintext [5].
The encryption /decryption algorithm is to encrypt/decrypt the message with the help of a key. The process of converting plaintext into ciphertext is called enciphering or encryption. The process of retaining the plaintext from the ciphertext iscalled deciphering or decryption. The following figure shows the encryption and decryption process. Cryptography is segmented into Symmetric key and Asymmetric key cryptography. It is further defined that same key used for encryption and decryption is called Symmetric key cryptography. Otherwise it is called Asymmetric key cryptography. This paper use substitution / replacement playfair cipher of symmetric key cryptography.

[2]. The well known multiple letter encryption cipher is the Playfair cipher. Here the digrams in the plaintext are treated as single units and converted into corresponding cipher text digrams However because of the drawbacks inherent in the 5*5 Playfair cipher which adversely affects the security we proposed an 8*8Playfair cipher. For details one can refer to [1]. This paper analyses the new proposed system. For this we have carried out cryptanalysis and through the avalanche effect we find out that the proposed cipher is a strong one.

The Playfair cipher shows a great advancement over themonoalphabetic ciphers. The identification of digrams is more difficult than individual letters. In the Monoalphabetic cipher, the attacker searches in 26 letters only. But by using the Playfair cipher, the attacker has to search in 26 x 26 = 676 diagrams. The relative frequencies of individual letters exhibit a much greater range than that of digrams , making frequency analysis much more difficult. Some of the peculiarities of Playfair cipher can be- No plaintext letter can be represented in the cipher by itself. Any given letter can be represented by 5 other letters.
Any given letter can represent 5 other letters.
Any given letter cannot represent a letter that it combines with diagonally.
It is twice as probable that the two letters of any pair are at the corners of a rectangle, than as in the same row or column. When a cipher letter has once been identified as a substitute for a plaintext letter, their is a 20% chance that it represents the same plaintext letter in each other appearance. These peculiarities make the cryptanalysis of Playfair cipher an easy task. The cryptanalysis of the Playfair cipher is also aided by the fact that a diagram and its reverse will encrypt in a similar fashion. That is, if AB encrypts to XY, then BA will encrypt to YX [2][5]. So by looking for words that begin and end in reversed diagrams, one can try to compare them with plaintext words that are similar. In recent investigation[1] we have modified the Playfair cipher by using 8*8 matrix along with LFSR for random number generation. In the present paper, we assume that the characters of the plaintext belong to the set of ASCII characters denoted by the codes 0 to 127. Here, our interest is to see that the strength of the cipher enhances significantly and no cryptanalytic attack would be possible on account of the modifications. For this we try to analyze all the drawbacks and security loopholes and provide a new cipher which is a strong one. The section II of the paper deals with the related work where details of 8*8 matrix has been depicted. Section III shows the development of the cipher. Section IV gives a brief illustration of the cipher, section V involves with the cryptanalysis, section VI involves the Avalanche effect and finally in section VII we conclude.

[3]. To make the data more secure there are several techniques. Most of the techniques have modified many times. Most authors used those modified techniques in their algorithm. Playfair technique is one of them. Playfair technique has also modified several time. Most authors used 6*6 matrix which include alphabets in lowercase, integer value as well as special symbols, most 7*7 matrix which include alphabets in uppercase and in lower case both, integer value as well as special symbols. We used 9*9 matrix which includes alphabets in lowercase or in uppercase both, integer value as well as list of operators and putting in the matrix according to their value in ASCII code. ASCII code stands for American Standard Code for Information Interchange. This work uses ASCII code to provide more security. For dual security, a substitution matrix is also used so that no one can understand it easily, even decoder have to do more and more efforts for decrypting the cipher text. Hence with the help of this technique we make our data safe.
Index Terms: Rectangular Matrix, Substitution Matrix, Playfair, Encryption and Decryption.

Cryptography is the science of text modification becoming unreadable. Cryptography has patterns like in steganography and image processing [1] [2]. Data security is required because of the large number of cybercrimes [3].It is called the encryption technique where plaintext is randomized using a key to be ciphertext [4] [5]. Compression is also a cryptographic model that compresses the message content [6] [7].If someone does not have a decryption key then the person cannot understand the content of the text [8]. Decryption is the process of returning ciphertext to plaintext. The probability of retrieving the original script by someone who has not had a decryption key is very small [9].Play fair cipher is widely used and quite useful in its era. Play fair cipher is a classical cryptographic algorithm that belongs to a Polygram cipher where play index is con-verted to a

Polygram form and a decryption encryption process perform for the poly-graph. In [10] the authors have used 5*5 matrixes. According to this the key arrangement inside the square extends by adding the sixth and sixth rows. The sixth base is the first line while the sixth columns contain the first column. In general, the key used in a series of words that are easy to understand. The use of playfair cipher method on text encoding is good enough because the key matrix used has a small possibility to be solved. Super playfair cipher is best used for symmetric cryptographic type. The bigram substitution technique on the key matrix has a small chance to solve. Each bigram change, the matrix pattern changes to the key [10].

[4]. Every enterprise stores and operates its transactions confidentially, therefore it has to

encrypt the transaction to protect it from data intruders. Technically, most encryption processes are ranging from hundreds of milliseconds. If there is a lot of encrypting process, then the performance will be getting bottleneck and the transaction's speed is being slowed down. In this paper, a promising scheme by adding GZIP compression on the former RSA encryption method to increase speed is proposed; thus every transaction is going to require less times, therefore the number of failed transactions or requests will be reduced. Hence, this way makes the transaction easier for customers to do. The result of little experimental test shows that using XOR to stream cipher-text to replace the absent of padding into specific bytes shows a lot of performance increase resulting other factors such as availability also affected without sacrificing the security level.

Security is the most valuable part in an application [1]. The main purpose is to protect data and information technologically. Data is a secretive entity, where only the permitted users are able to access. In recent years, smartphones are used to do online based transaction (e.g. shopping, buying food, money transferring, etc.). When the transaction process, there are a lot information or data sent; such as data regarding a sender, a receiver, money amount, information of the transaction, and so on. All these information/data must be hidden from third person who may steal them between sender and receiver To secure the data, cryptography is the best choice [2]. Cryptography is a technique or a method operated to secure the data between the sender and receiver. Cryptography works by changing an ordinary-text (plain-text) into a new text that called cipher-text. This process called encryption. After a cipher-text reached by receiver, the cipher-text will be technically re-turned into plain-text. This process called decryption. Based on [3], one of the state-of-the-art of encryption is public-key cryptography; where it needs two separate keys. One key for locking or encrypting the plain-text, the other is for unlocking or decrypting the cipher-text.

Rivest, Shamir, and Adlemen (RSA) encryption is one of public-key encryption methods. It uses private and public key as process key to encryption and decryption. For an example, if **A** wants to send a secret message to **B**, then **A** will encrypt the message using **B**'s public key. After **B** received the message, he/she is going to decrypt the secret message using his/her private key. In transferring process, even third party has public key, he/she is still not able to decrypt the message.
PT. XYZ operates a similar way to rationally safeguard its transaction. Every transaction occurs between server and client must be surpassed through encryption and decryption process. The current RSA method, which PT. XYZ uses, needs 400ms up to 600ms process time for encryption and 100ms up to 150ms for decryption process. This process time does not include the client's connection speed which may vary on recipient device networks. This makes clients sometime have to wait a bit longer, even for doing small transaction only. When the execution time takes so long, it is possible that the connection will be timed out, and the transaction is broken down or canceled.
Owing to this problem, we proposed a customized RSA using XOR logic that has been proven to be secure and stated "one of the most important history in cryptographic" on the patent [4] to replace the padding process on former method with GZIP to compress the text thus reducing the execution time. For the result, this method can compress the process time up to more than 70% faster for both encryption and decryption processes. This method increased a performance and availability of application programming interface (API) without give negative affect for data security. Other than the background this paper will also cover up the previous related work, proposed method, and results discussion of both former and proposed methods.

[5]. In today's digital world cryptography is used to secure information in order to provide the privacy for the intended sender and receiver by managing the message with the public key. The objective of this work is to securing the key of Playfair cipher using RSA algorithm. It is a two stage application, in first stage the existing methods of Playfair cipher modified by increasing in the size of matrix, so that the restrictions of earlier works of PF cipher using 5×5 matrix were overcome in the proposed work. The proposed method use a 12×8 matrix which contain all alphabetic, numeric and special character use in keyboard as input. This work is an enhancement to the existing algorithms that uses 5×5 matrix to pick cipher characters. It makes use of alphabets both lower and upper case characters, number and special characters for constructing the contents of the matrix. In the second stage, RSA public key encryption technique is used for sending the key of the PF ciphers securely. Finally, the security strength of the whole system has been analyzed and tried to fulfil the requirement of security.
Here , Encryption Process =ET , Decryption process=DT

The basic Playfair cipher uses a matrix of 5×5 containing a key or phrase. Memorization of the key is achieved by generating a 5×5 key table and cipher text is created by applying four simple rules on this key table [8]. To generate the key table, one would first fill in the spaces in the table with the letters of the key (dropping any duplicate letters), then fill the remaining spaces with the rest of the letters of the alphabet in order (usually omitting "Q" to reduce the alphabet

to fit; other versions put both "I" and "J" in the same space) [8]. The key can be written in the top rows of the table, from left to right, or in some other pattern, such as a spiral beginning in the upper-left-hand corner and ending in the center. The key together with the conventions for filling in the 5×5 table constitute the cipher key [8]. To encrypt a message, one would break the message into digraphs (groups of 2 letters) for example; "HELLO WORLD" becomes "HE, LL, OW, OR, LD", and maps them out on the key table. If needed, append a "Z" to complete the final digraph. The two letters of the digraph are considered as the opposite corners of a rectangle in the key table [2]. Note the relative position of the corners of this rectangle. Then apply the following 4 rules, in order, to get encrypted message for each pair of letters in the PT [8].

2.2 Conclusion

So far encryption technique adopting the concept of PLAYFAIR CIPHER MATRIX of size 5X5 has been programmed for calculating the Cipher text. But we use PLAYFAIR CIPHER MATRIX of size 16X16 so that we can use Letters[A-Z , a-z], digits[0-9] , All special characters , Keywords and Space as a Character Finally, we have pointed the merits and demerits of traditional Playfair algorithm. In order to overcome the demerits, we have proposed an extension to traditional Playfair cipher algorithm; which can be used more efficiently even for the Plaintext containing alphanumeric values and special characters. Then a public key encryption system has been designed which provides both confidentiality and authentication, but there are some limitations. Complete mathematical derivation is given to show the exact result at both sender and receiver sides the previous encryption technique is also a part of this system. After completion of the program the strength of the technique has been checked and this encryption technique can also be used for other networks. In this algorithm 16X16 Playfair matrix is used for creating the cipher text and then we will apply Extended RSA algorithm on Playfair Key for providing the More secure channel than The RSA Algorithm.

Chapter 3
Proposed Work

3.1 Proposed model

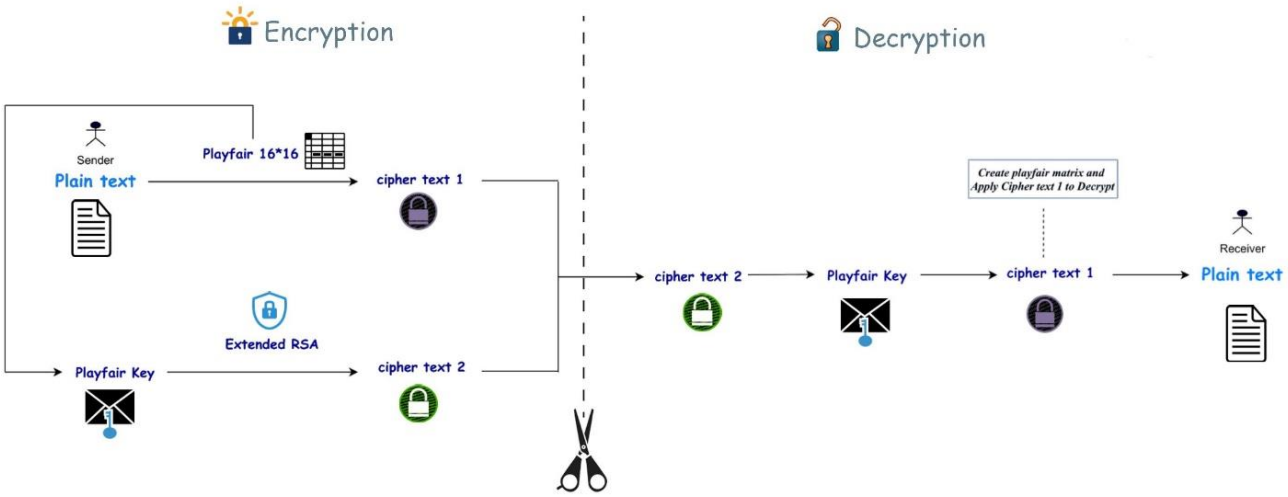

Fig 1 . The following figure shows the Encryption and Decryption Process of Our Model.

3.2 Model Overview

- Step 1 : - Sender will give a Plain Text and a Playfair Key .
- Step 2 :- Plaintext is encrypted using Playfair 16X16 and converted to ciphertext 1 (see in above figure of Proposed Model)
- Step 3 : - Playfair key is Encrypted using Extended RSA and converted to Cipher text 2(see in above figure of Proposed Model) in order to make it more secure.
- Step 4 : - Cipher text 2 (Ciphertext of key) is decrypted to get back the Playfair key.
- Step 5 : - Create Playfair matrix using Playfair key and apply ciphertext 1 to decrypt.
- Step 6 :- Cipher text 1 is Decrypted using Playfair Matrix and Plain text or message is visible at the Receiver side.

3.3 Terminologies Description

Encryption -

Encryption is the method by which information is converted into secret code that hides the information's true meaning. The science of encrypting and decrypting information is called cryptography.

Algorithm to encrypt the plain text:

The plaintext is split into pairs of two letters (digraphs).

Example:- Key : Monarchy

Plain Text : “Instruments”

After Split : ‘in’ ‘st’ ‘ru’ ‘me’ ‘nt’ ‘sz’

THEN --

With The Help Of key Square Grid and Plain Text pair We Gets The Cipher Text

Of The Plain Text.

Decryption –

The conversion of encrypted data into its original form is called Decryption. It is generally a reverse process of encryption. It decodes the encrypted information so that an authorized user can only decrypt the data because decryption requires a secret key or password.

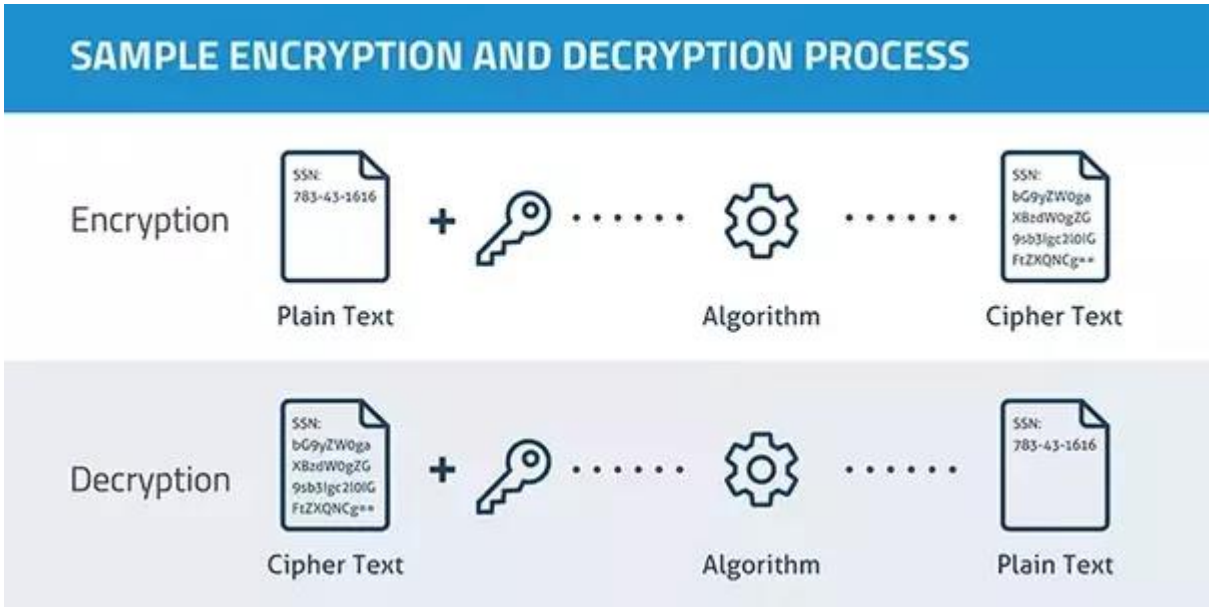

Fig . The following figure shows the Simple encryption and decryption process.

Plain text –

In cryptography, plaintext is usually ordinary readable text before it is encrypted into ciphertext, or readable text after it is decrypted.

Cipher text –

Ciphertext is encrypted text transformed from plaintext using an encryption algorithm. Ciphertext can't be read until it has been converted into plaintext (decrypted) with a key.

Playfair 16*16 –

Key : Playfair.(sample)

P	l	a	y	f	i	r	.	(	S	m	p	e	)	NUL	⇗
⇘	⇹	⇷	⇻	⇽	•	⇳	⇔	⇖	⇛	⇐	⇠	⇡	⇙	⇑	⇓
↕	!!	¶	§	⇏	↖	↑	↓	→	←	↗	↔	⇐	⇒	Space	!
“	#	$	%	&	„	*	+	,	-	/	0	1	2	3	4
5	6	7	8	9	:	;	<	=	>	?	@	A	B	C	D
E	F	G	H	I	J	K	L	M	N	O	Q	R	T	U	V
W	X	Y	Z	[	\	]	^	_	`	b	c	d	g	h	j
k	n	o	q	s	t	u	v	w	x	z	{	|	}	~	DEL
Ç	ü	é	â	ä	À	å	ç	ê	ë	è	ï	î	ì	Ä	Å
É	æ	Æ	ô	ö	Ò	û	ù	ÿ	Ö	Ü	¢	£	¥	₧	ƒ
á	í	ó	ú	ñ	Ñ	ª	º	¿	⇎	¬	½	¼	¡	«	»
⇋	⇌	⇍	⇞	⇟	↺	↻	↯	↭	↼	↪	↰	↶	↵	↴	↓
↤	↧	↦	↤	↞	↕	↷	↸	↳	↭	↿	↾	↹	↩	↑↓	⇀
⇁	↼	↾	↲	↱	↫	↬	⇄	↙	↣	↠	⇈	⇇	⇉	⇊	⇆
α	ß	Γ	π	Σ	Σ	μ	T	Φ	Θ	Ω	δ	∞	φ	ε	↘
↙	±	≥	≤	↜	↝	÷	≈	°	∙	∙	√	ⁿ	²	⇎	

To Overcome The Disadvantages Of PlayFair 5×5 Matrix The Extension of Playfair Cipher using 16X16 Matrix was Introduced.

THE PROPOSED PLAYFAIR CIPHER USING 16X16 MATRIX

This algorithm can accept the Plaintext containing Alphabets (capital letters and small letters), Numbers and Special characters. So the user can easily encrypt combination of alphabets, numbers and characters efficiently.

Advantages of 16*16 –

- It considers the space between two words in plaintext as one character.
- The user can easily encrypt and decrypt the combination of alphabets, numbers and special characters efficiently.
- Letters, digits and special characters are used to construct 16x16 matrix.
- It is case sensitive.
- The letters I and J are considered as two different letters.
- This algorithm cannot separate a repeating Plaintext letters with a filter letter.
- It allows more than 36 characters as keyword**.**

Algorithm Of PlayFair 16X16 Matrix :

- Read a keyword.
- Eliminate the repeated characters in keyword.
- Construct a matrix by filling the character of keyword from left to right and top to bottom.
- Fill the reminder of matrix with the remaining characters from ASCII values 0 to 255.
- Read a plaintext.
- Divide the plaintext into pair of characters.
- Add the character "Null" when odd number of character in the message.

Extended RSA –

RSA is a cryptographic algorithm used in secure communications. RSA is designed to ensure that two parties can agree on a secret Message without the need for a third party to help. RSA is often used in secure communications such as financial transactions and passwords.

The extended Euclidean algorithm is a security algorithm that is used in the RSA Algorithm. The extended Euclidean algorithm is a modified algorithm that is used to improve the security of the RSA Algorithm. Extended Euclidean Algorithm is a cryptographic algorithm that was designed by the National Security Agency and The National Security Agency Institute of Electrical and Electronics Engineers. The Algorithm is used to create a secure digital signature. Extended RSA will also use the asymmetric key. Which is based on public and private key encryption technique. The idea of RSA Algorithm is it takes only two prime number P and Q while In Extended RSA here we can take multiple prime number for Encrypt the Playfair key which will make it more secure than the RSA Algorithm. The Extended Version of RSA Algorithm is Called Extended RSA. Finally, the security strength of the whole system has been analyzed and tried to fulfil the requirement of security.

Extended RSA Algorithm follow these steps:

1. Generate a public key and private key
   - a. Generate by deciding n prime number as p , q , r , s , t…..etc.
   - b. Multiple p, q, r, s, t……etc. to get the value of n
   - Cal n = p x q x r x s x t…..etc.
   - c. Find out the ϕn by using the $(p-1)(q-1)$(r-1)(s - 1)(t - 1)……etc. formula
   - d. Decide the value of e, while $e = 1 < e < \phi n$, and must be coprime and the greatest common distribution of ϕn
   - e. The value of d can be calculated by the following formula $ed\ (mod\ \phi n) = 1$
   - f. Results public key and private key value will be (n, e) and (n, d) respectively

2. Encrypt; encrypt the string with $c = m\text{^}e\ mod\ n$ formula using the new generated key before

3. Send the cipher text c to recipient

4. Decrypt the cipher text in plain text m=c^d mod n.

3.4 Algorithm's source code

PlayFair 16X16 source code

```
package cryptography_practical;

import java.awt.Point;
import java.util.Scanner;
public class PlayfairCipher
{
    //length of digraph array
    private int length = 0;
    //creates a matrix for Playfair cipher
    private String [][] table;
    //main() method to test Playfair method
    public static void main(String args[])
    {
        PlayfairCipher pf = new PlayfairCipher();
    }
    //main run of the program, Playfair method
//constructor of the class
    private PlayfairCipher()
    {
        Scanner sc = new Scanner(System.in);
        //prompts user for message to be encoded
        System.out.print("Enter the plaintext to be encipher: ");
        String input = parseString(sc);
        while(input.equals("")) {
            input = parseString(sc);
        }
        //prompts user for the keyword to use for encoding &
creates tables
```

```
        System.out.print("Enter the key for playfair cipher: ");
        String key = parseString(sc);
        while(key.equals("")) {
            key = parseString(sc);
        }
        table = this.cipherTable(key);
//encodes and then decodes the encoded message
        String output = cipher(input);
        String decodedOutput = decode(output);
//output the results to user
        this.keyTable(table);
        this.printResults(output,decodedOutput);
    }
    //parses an input string to remove numbers, punctuation,

    private String parseString(Scanner sc)
    {
        String parse = sc.nextLine();
        //ascii value of space 32
//        int a=32;
//        char c=(char)a;
       // parse = parse.replaceAll("\s", "");
        return parse;
    }
    //creates the cipher table based on some input string (already
parsed)
    private String[][] cipherTable(String key)
    {
//creates a matrix of 16*16
        String[][] playfairTable = new String[16][16];
        //1-256 all char
        String str="";
        for (int i=0; i<=255; i++) {
            char c = (char) i;
            str = str+c;
        }
        String keyString = key + str;
//fill string array with empty string
        for(int i = 0; i < 16; i++)
            for(int j = 0; j < 16; j++)
                playfairTable[i][j] = "";
        for(int k = 0; k < keyString.length(); k++)
        {
            boolean repeat = false;
            boolean used = false;
            for(int i = 0; i < 16; i++)
            {
                for(int j = 0; j < 16; j++)
                {
                    if(playfairTable[i][j].equals("" +
keyString.charAt(k)))
                    {
                        repeat = true;
                    }
                    else if(playfairTable[i][j].equals("") &&
!repeat && !used)
                    {
```

```
                        playfairTable[i][j] = "" +
keyString.charAt(k);
                        used = true;
                    }
                }
            }
        }
        return playfairTable;
    }
    //cipher: takes input (all upper-case), encodes it, and returns
the output
    private String cipher(String in)
    {
        // null value to be inserted at even length or repeated
char pair
        char nullVal = (char)0;
        length = (int) in.length() / 2 + in.length() % 2;
//insert x between double-letter digraphs & redefines "length"

        for(int i = 0; i < (length - 1); i++)
        {
            if(in.charAt(2 * i) == in.charAt(2 * i + 1))
            {
                in = new StringBuffer(in).insert(2 * i + 1,
nullVal).toString();
                length = (int) in.length() / 2 + in.length() % 2;
            }
        }
//------------makes plaintext of even length--------------
//creates an array of digraphs
        String[] digraph = new String[length];
//loop iterates over the plaintext
        for(int j = 0; j < length ; j++)
        {
//checks the plaintext is of even length or not
            if(j == (length - 1) && in.length() / 2 == (length -
1))
//if not addends X at the end of the plaintext
                in = in + nullVal;
            digraph[j] = in.charAt(2 * j) +""+ in.charAt(2 * j +
1);
        }
//encodes the digraphs and returns the output
        String out = "";
        String[] encDigraphs = new String[length];
        encDigraphs = encodeDigraph(digraph);
        for(int k = 0; k < length; k++)
            out = out + encDigraphs[k];
        return out;
    }
    //---------------encryption logic-----------------
//encodes the digraph input with the cipher's specifications
    private String[] encodeDigraph(String di[])
    {
        String[] encipher = new String[length];
        for(int i = 0; i < length; i++)
        {
```

```
            char a = di[i].charAt(0);
            char b = di[i].charAt(1);
            int r1 = (int) getPoint(a).getX();
            int r2 = (int) getPoint(b).getX();
            int c1 = (int) getPoint(a).getY();
            int c2 = (int) getPoint(b).getY();
//executes if the letters of digraph appear in the same row
//in such case shift columns to right
            if(r1 == r2)
            {
                c1 = (c1 + 1) % 16;
                c2 = (c2 + 1) % 16;
            }
//executes if the letters of digraph appear in the same column
//in such case shift rows down
            else if(c1 == c2)
            {
                r1 = (r1 + 1) % 16;
                r2 = (r2 + 1) % 16;
            }
//executes if the letters of digraph appear in the different row
and different column
//in such case swap the first column with the second column
            else
            {
                int temp = c1;
                c1 = c2;
                c2 = temp;
            }
//performs the table look-up and puts those values into the encoded
array
            encipher[i] = table[r1][c1] + "" + table[r2][c2];
        }
        return encipher;
    }
    //----------------------decryption logic--------------------
// decodes the output given from the cipher and decode methods
(opp. of encoding process)
    private String decode(String out)
    {
        String decoded = "";
        for(int i = 0; i < out.length() / 2; i++)
        {
            char a = out.charAt(2*i);
            char b = out.charAt(2*i+1);
            int r1 = (int) getPoint(a).getX();
            int r2 = (int) getPoint(b).getX();
            int c1 = (int) getPoint(a).getY();
            int c2 = (int) getPoint(b).getY();
            if(r1 == r2)
            {
                c1 = (c1 + 15) % 16;
                c2 = (c2 + 15) % 16;
            }
            else if(c1 == c2)
            {
                r1 = (r1 + 15) % 16;
```

```
                r2 = (r2 + 15) % 16;
            }
            else
            {
//swapping logic
                int temp = c1;
                c1 = c2;
                c2 = temp;
            }
            decoded = decoded + table[r1][c1] + table[r2][c2];
        }
//returns the decoded message
        return decoded;
    }
    // returns a point containing the row and column of the letter
    private Point getPoint(char c)
    {
        Point pt = new Point(0,0);
        for(int i = 0; i < 16; i++)
            for(int j = 0; j < 16; j++)
                if(c == table[i][j].charAt(0))
                    pt = new Point(i,j);
        return pt;
    }
    //function prints the key-table in matrix form for playfair
cipher
    private void keyTable(String[][] printTable)
    {
        System.out.println("Playfair Cipher Key Matrix: ");
        System.out.println();
//loop iterates for rows
        for(int i = 0; i < 16; i++)
        {
//loop iterates for column
            for(int j = 0; j < 16; j++)
            {
//prints the key-table in matrix form
                System.out.print(printTable[i][j]+" ");
            }
            System.out.println();
        }
        System.out.println();
    }
    //method that prints all the results
    private void printResults(String encipher, String dec)
    {
        System.out.print("Encrypted Message: ");
//prints the encrypted message
        System.out.println(encipher);
        System.out.println();
        System.out.print("Decrypted Message: ");
//prints the decryted message
        System.out.println(dec);
    }
}
```

Extended RSA source code

```
package cryptography_practical;

import java.io.DataInputStream;
import java.io.IOException;
import java.math.BigInteger;
import java.util.Random;

public class RSA
{
    private BigInteger p; //Prime 1
    private BigInteger q; //Prime 2
    private BigInteger p2; //Prime 3
    private BigInteger q2; //Prime 4
    private BigInteger N;
    private BigInteger phi;
    private BigInteger e;
    private BigInteger d;
    private int bitlength = 1024;
    private Random     r;

    public RSA()
    {
        r = new Random();
        p = BigInteger.probablePrime(bitlength, r);
        q = BigInteger.probablePrime(bitlength, r);
        p2 = BigInteger.probablePrime(bitlength, r);
        q2 = BigInteger.probablePrime(bitlength, r);
        N = p.multiply(q).multiply(p2).multiply(q2);
        phi =
p.subtract(BigInteger.ONE).multiply(q.subtract(BigInteger.ONE)).mul
tiply(p2.subtract(BigInteger.ONE)).multiply(q2.subtract(BigInteger.
ONE));
        e = BigInteger.probablePrime(bitlength / 2, r);
        while (phi.gcd(e).compareTo(BigInteger.ONE) > 0 &&
e.compareTo(phi) < 0)
        {
            e.add(BigInteger.ONE);
        }
        d = e.modInverse(phi);
        System.out.println("P is : ---> " + p);
        System.out.println("Q is : ---> " + q);
        System.out.println("P1 is : ---> " + p2);
        System.out.println("Q1 is : ---> " + q2);
        System.out.println("N is : ---> " + N);
    }

    public RSA(BigInteger e, BigInteger d, BigInteger N)
    {
        this.e = e;
        this.d = d;
        this.N = N;
    }
```

```
    public static void main(String[] args) throws IOException
    {
        RSA rsa = new RSA();
        DataInputStream in = new DataInputStream(System.in);
        String teststring;
        System.out.println("Enter the plain text:");
        teststring = in.readLine();
        System.out.println("Encrypting String: " + teststring);
        System.out.println("String in Bytes: "
                + bytesToString(teststring.getBytes()));
        // encrypt
        byte[] encrypted = rsa.encrypt(teststring.getBytes());
        // decrypt
        byte[] decrypted = rsa.decrypt(encrypted);
        System.out.println("Decrypting Bytes: " +
bytesToString(decrypted));
        System.out.println("Decrypted String: " + new
String(decrypted));
    }

    private static String bytesToString(byte[] encrypted)
    {
        String test = "";
        for (byte b : encrypted)
        {
            test += Byte.toString(b);
        }
        return test;
    }

    // Encrypt message
    public byte[] encrypt(byte[] message)
    {
        return (new BigInteger(message)).modPow(e,
N).toByteArray();
    }

    // Decrypt message
    public byte[] decrypt(byte[] message)
    {
        return (new BigInteger(message)).modPow(d,
N).toByteArray();
    }
}
```

References

1. Dhenakaran, S.S. and Ilayaraja, M., 2012. Extension of playfair cipher using 16X16 matrix. International Journal of Computer Applications, 48(7).

2. Srivastava, S.S. and Gupta, N., 2011. A novel approach to security using extended playfair cipher. International Journal of Computer Applications, 20(6), pp.0975-8887.

3. Arroyo, J.C.T., Sison, A.M., Medina, R.P. and Delima, A.J.P., 2022. An Enhanced Playfair Algorithm with Dynamic Matrix Using the Novel Multidimensional Element-in-Grid Sequencer (MEGS). Int. J. Eng. Trends Technol, 70(3), pp.132-139.

4. Susanto, A., Putranto, M.I., Utama, D.N. and Wibowo, A., 2019, August. Extended-RSA for Encryption Process to Improve Application Server Availability. In IOP Conference Series: Materials Science and Engineering (Vol. 598, No. 1, p. 012059). IOP Publishing.

5. Chauhan, S.S., Singh, H. and Gurjar, R.N., 2014. Secure key exchange using RSA in extended Playfair cipher technique. International Journal of Computer Applications, 104(15).

6. Image Reference : https://www.venafi.com/sites/default/files/2020-12/Encrypt_Decrypt_Diff_1.png

7. Information & images Reference : https://www.geeksforgeeks.org/playfair-cipher-with-examples/

8. Information : https://www.javatpoint.com/playfair-cipher-program-in-java