

Secured Wireless Communication Using RSA

**A Project Submitted to
University of Mumbai for partial fulfillment of the degree of
Bachelor of Engineering**



Submitted By
**Pooja Kulkarni
Ritesh Kumar
Kiran Maddela
Akanksh Shetty**

Supervisor
Dr. Tusharika Sinha Banerjee



Department of Electronics & Telecommunication Engineering

MES's Pillai College of Engineering,

New Panvel – 410 206

UNIVERSITY OF MUMBAI

Academic Year 2021– 22



Department of Electronics & Telecommunication Engineering

MES's Pillai College of Engineering,

New Panvel – 410 206

ACKNOWLEDGEMENT

I would like to acknowledge the following as being idealistic channels and fresh dimensions in the completion of this project.

I take this opportunity to thank the **University of Mumbai** for giving me a chance to do this project.

I would like to thank my **Principal Dr. Sandeep Joshi**, for providing necessary facilities required for completion of this project.

I would also like to express my sincere gratitude towards my project guide **Dr. Tusharika Sinha Banerjee** for her guidance.

I would like to thank my **College Library** for having necessary reference material related to my project.

Lastly, I would like to thank each and every person who directly or indirectly helped me in the completion of the project, especially **my Parents and Peers** who supported me throughout the course of my project.



**Department of Electronics & Telecommunication Engineering
MES's Pillai College of Engineering,
New Panvel – 410 206**

CERTIFICATE

This project report entitled “**Secured Wireless Communication Using RSA**” by the following students

**Pooja Kulkarni
Ritesh Kumar
Kiran Maddela
Akanksh Shetty**

is certified for the submission for the degree of Bachelor of Electronics & Telecommunication Engineering.

Dr. Tusharika Sinha Banerjee
SUPERVISOR
Department of
Electronics & Telecommunication Engineering

Prof. Ajit Saraf
PROJECT COORDINATOR
Department of
Electronics & Telecommunication Engineering

Dr. Avinash R. Vaidya
HEAD
Department of Electronics & Telecommunication Engineering

Dr. Sandeep M. Joshi
PRINCIPAL
MES's Pillai College of Engineering

Date:

Place:



DEPARTMENT OF ELECTRONICS AND TELECOMMUNICATION ENGINEERING

Pillai College of Engineering

New Panvel – 410 206

PROJECT APPROVAL FOR B.E.

This project synopsis entitled “**Secured Wireless Communication Using RSA**” by **Pooja Kulkarni, Ritesh Kumar, Kiran Maddela and Akanksh Shetty** is approved for the degree of B.E. in **Department Of Electronics and Telecommunication Engineering**

Examiners:

1. _____

2. _____

Supervisors:

1. _____

2. _____

Chairman:

1. _____

Date:

Place:



DEPARTMENT OF ELECTRONICS AND TELECOMMUNICATION ENGINEERING

Pillai College of Engineering

New Panvel – 410 206

DECLARATION

We declare that this written submission for the B.E project entitled “**Secured Wireless Communication Using RSA**” represents our ideas in our own words and where others' ideas or words have been included. We have adequately cited and referenced the original sources. We also declared that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any ideas / data / fact / source in our submission. We understand that any violation of the above will cause disciplinary action by the institute and also evoke penal action from the sources which have thus not been properly cited or from whom paper permission has not been taken when needed.

Pooja Kulkarni

Ritesh Kumar

Kiran Maddela

Akanksh Shetty

Date:

Place:

Table of Contents

| | | |
|-----------|--|-------------|
| a) | ABSTRACT | i |
| b) | LIST OF FIGURES | ii |
| c) | LIST OF TABLES | iii |
| | | |
| 1) | INTRODUCTION | 1-2 |
| | 1.1 Fundamentals | 1 |
| | 1.2 Objectives | 1 |
| | 1.3 Scope | 2 |
| | 1.4 Organization of the Project Report | 2 |
| | | |
| 2) | LITERATURE REVIEW | 3-6 |
| | 2.1 Introduction | 3 |
| | 2.2 Literature Review | 4 |
| | 2.3 Problem Definition | 5 |
| | | |
| 3) | SYSTEM REQUIREMENT AND ANALYSIS | 7-13 |
| | 3.1 Hardware and Software Details | 7 |
| | 3.2 Introduction to Cryptography | 7 |
| | 3.3 Types of cryptography | 8 |
| | 3.4 RSA(Rivest Shamir Aldeman) | 9 |
| | 3.5 Socket programming | 11 |
| | 3.6 Graphic User Interface | 12 |

| | | |
|-----------|--|--------------|
| 4) | METHODOLOGY | 14-15 |
| | 4.1 Block Diagram | 14 |
| | 4.2 Working | 14 |
| 5) | IMPLEMENTATION OF HARDWARE AND SOFTWARE | 16 |
| | 5.1 Application | 16 |
| 6) | DISCUSSION AND RESULTS | 17-21 |
| | 6.1 Sample of Inputs/Database Used/ and Outputs/ScreenShot | 17 |
| 7) | CONCLUSION | 22 |
| | 7.1 Conclusion | 22 |
| | 7.2 Future Scope | 22 |
| | REFERENCES | 23 |
| | ACKNOWLEDGEMENT | 25 |
| | APPENDIX | 26 |

ABSTRACT

In this era of digital age tons of secret and non-secret data is transmitted over the web. Cryptography is one of the many techniques to secure data on a network. It is one of the techniques that can be used to ensure information security and data privacy. So in this project we have developed a method of cryptography, used for encryption and decryption of data to improve security during communication/uses. The technology behind this method is RSA asymmetric cryptography, which is a secure and reliable method to send and receive encrypted data.

In the proposed technique, because the number of prime count increases, divisor calculation becomes difficult. If the hacker has encryption key (e) and Product of prime numbers (N) then it is not easy to find out the prime number combinations and hence decryption key (d) will be more secure by using the proposed algorithm. This will be more difficult because given a number n , it is easy to find two numbers whose product is equal to n using Shor's algorithm and Grover's Search Algorithm but it's not very difficult and time taking to precisely determine m numbers whose product is adequate to n .

Socket programming is used to establish a connection between two different users on a network. Socket programming is a way of connecting two sockets on a network to communicate with each other. Socket is one end-point of a two-way communication link between two programs running on the network and using the Tkinter package in Python for developing GUI for the program to generate public and private keys for the encryption process.

LIST OF FIGURES

| Sr. no. | Figure no. | Name of Figure (caption) | Page no. |
|----------------|-------------------|--|-----------------|
| 1 | 1.1 | Cryptography example | 3 |
| 2 | 3.1.1 | Different types of Cryptography methods | 11 |
| 3 | 3.2 | How RSA encryption works | 12 |
| 4 | 3.5 | Socket Programming Connection Diagram | 14 |
| 5 | 4.1 | Block diagram of the proposed system | 16 |
| 6 | 6.1 | Receiver, Entering the two prime numbers | 21 |
| 7 | 6.2 | Receiver, Generation of public and private keys | 22 |
| 8 | 6.3 | Receiver, Entering the private key | 22 |
| 9 | 6.4 | Receiver, Server is listening | 22 |
| 10 | 6.5 | Sender, Sending a Message | 23 |

LIST OF TABLES

| Sr. no. | Table .no. | Name of table (caption) | Page no. |
|----------------|-------------------|---|-----------------|
| 1 | 2.2 | Literature survey review | 5 |
| 2 | 3.1.1 | Hardware Requirements | 9 |
| 3 | 3.1.2 | Software Requirements | 9 |
| 4 | 3.1.3 | Difference Between Symmetric and Asymmetric Cryptography | 11 |

Chapter 1

Introduction

In this section, the fundamentals, objectives, scope and organization of the report is discussed.

1.1 Fundamentals

Cryptography

Cryptography is the study of secure communication mechanisms that allow only the sender and receiver of a message/data to see the contents of the message/data. Cryptography means hidden and is derived from the Greek word *kryptos*. It can be further classified into two types as Symmetric key Cryptography and Asymmetric key Cryptography (more commonly known as Public key cryptography). Symmetric key cryptography requires only a single key for both the encryption and decryption process whereas in Asymmetric key cryptography two keys are required respectively for the encryption and decryption process. As a result, Asymmetric key cryptography is more secure and slower than Symmetric key cryptography. For any attacker, Asymmetric key cryptography will be much harder to decode because of the complex algorithm used.

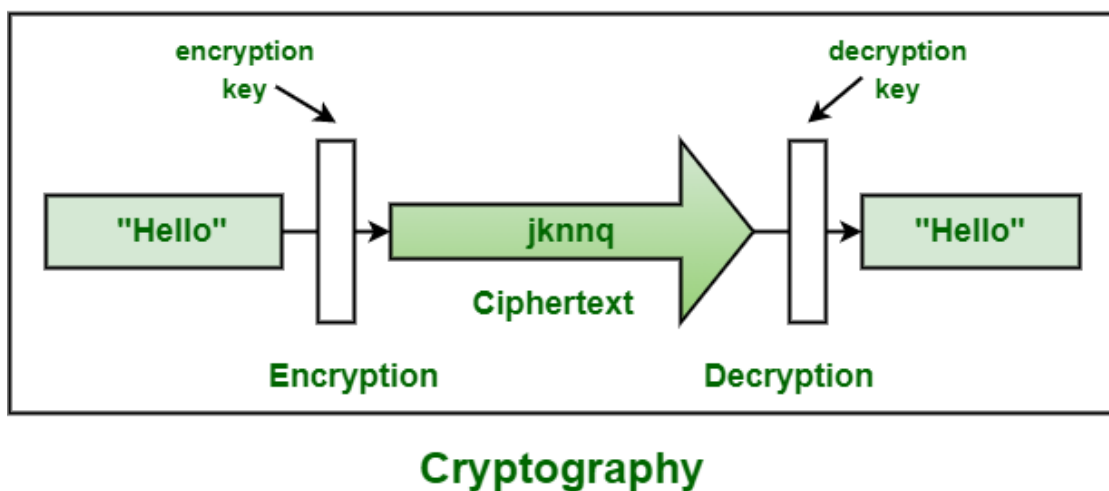


Figure 1.1 Cryptography example

1.2 Objectives

The objective is written in not more than 10 lines. The objective of this work is as follows:

1. To study the different cryptography algorithms used to encrypt and decrypt the data to be sent and received.
2. To develop a secured communication between systems present in the same network.

1.3 Scope

Creating secured wireless communication is a research project to design a system which is secure. With time the need for faster and secure communication is increasing and they're more prone to attackers if not secured properly. While doing the research, we came across the fact that AES is less secure because of it being a Symmetric cryptography method and RSA is slow as it generates 2 keys for encryption and decryption process. Our suggestion is that if we use Hybrid cryptography method which uses both AES and RSA algorithm, we'll be able to make the system more secure and faster compared to AES and RSA respectively.

1.4 Organization of report

The report is organized as follows:

Chapter 1 gives an introduction. It describes the fundamental terms used in this project. It motivates us to study and understand the different techniques used in this work. This chapter also presents the outline of the objective of the report.

Chapter 2 describes the review of the relevant various techniques in the literature systems.

Chapter 3 presents the theory and proposed work. It describes the major approaches used in this work.

Chapter 4 consists of societal and technical applications of the project.

Chapter 5 presents the summary of the proposed system and project.

Chapter 2

Literature Survey

In this chapter the relevant techniques in literature are reviewed. It describes various techniques used in the work. A summary of the literature is presented at the end of this chapter.

2.1 Introduction

Public key cryptography has been said to be the most significant new development in cryptography. In Public key cryptography, one of the keys is designated the *public key* and may be distributed as widely as the client wants. The other key is designated the *private key* and is never revealed to another party. It is straight-forward to send messages under this scheme.^[1]

2.2 Literature Review

Various methods used by various researchers, the algorithms they used, and the ways they followed for their systems, of them are described below:

| SR NO | TITLE OF THE PAPER | SOURCE | YEAR | SIGNIFICANCE |
|-------|--|--|-----------|--|
| 1 | Implementation of Advanced Encryption Standard Algorithm | International Journal of Scientific & Engineering Research Volume 3, Issue 3, March -2012 1 ISSN 2229-5518 | 2012 | Symmetric cryptography uses a single key, algorithm is based on substitution and permutation. |
| 2 | A Comparative Analysis of AES and RSA Algorithms | International Journal of Scientific & Engineering Research, Volume 7, Issue 5 | May, 2016 | As AES is a private key based algorithm that suffers from key distribution and key agreement problems however these problems are overcome in the RSA algorithm |

| | | | | |
|---|--|--|--------------|---|
| | | | | but encryption and decryption takes more time in RSA algorithm. |
| 3 | Implementation of RSA | International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT) | 2016 | Uses two keys: public and private keys. Because one of the keys can be given to anyone and the other is kept private. Uses 2 large prime numbers for a high level of security. |
| 4 | HYBRID CRYPTOGRAPHY BY THE IMPLEMENTATION OF RSA AND AES | International Journal of Current Research | April, 2011 | This work using Rijndael cryptography symmetric algorithm used for data encryption/decryption and RSA cryptography asymmetric algorithm for Rijndael key's encryption/decryption. |
| 5 | A Comprehensive Evaluation and Implementation of AES, RSA and Hybrid Cryptographic Algorithms on a | International Journal of Engineering and Advanced Technology (IJEAT) | October 2019 | The RSA takes more time for encryption and decryption but has better strength than the other two algorithms. |

| | | | | |
|---|---|---|---------------|---|
| | Portable Device | | | |
| 6 | Data Encryption and Decryption Using RSA Algorithm in a Network Environment | IJCSNS International Journal of Computer Science and Network Security | July 2013 | Made a GUI using java for Encryption and Decryption . |
| 7 | Security Enhancement of RSA Algorithm using Increased Prime Number Set | International Journal of Engineering and Advanced Technology (IJEAT) | February 2020 | We can say that when we increase the number of prime numbers in RSA algorithm then its security also improves because it's hard to find the factor of N, while there are more than two prime numbers. |
| 8 | An integrated Encryption Scheme Used in Bluetooth Communication Mechanism | International Journal of Computer Technology and Electronics Engineering | June 2012 | Encryption and Decryption in Bluetooth and how RSA is implemented for more security |
| 9 | Sockets Programming | https://docs.python.org/3/howto/sockets.html | | Sockets allow you to exchange information between processes on the same machine or across a network, distribute work to the most efficient machine, and they |

| | | | | |
|--|--|--|--|---|
| | | | | easily allow access to centralized data |
|--|--|--|--|---|

Table 2.2 Literature survey review

2.3 Problem Definition

The name of the project is “Secured Wireless Communication using RSA”. If two people want to communicate with each other without any security risks then RSA can be used for encryption and decryption of the message. Then the receiver will generate public and private keys using RSA, where the public key is for encryption and private key is for decryption. Receiver will only provide the public key to the sender and keep the private key with him. Then the sender will send a message to the receiver in the encrypted form and the receiver will decrypt the message using the private key and the original message will be retrieved.

Chapter 3

This section involves the Hardware, Software requirements, Introduction to cryptography, types of cryptography, RSA algorithm, Socket programming and GUI used in the project.

Secured Wireless Communication using RSA

3.1 Hardware and Software Required

| Hardware | Details |
|------------------|--|
| Operating System | Linux- Ubuntu 16.04 to 17.10, or Windows 7 to 10 |
| Processor | x86 64-bit CPU (Intel / AMD architecture) |
| RAM | 4 GB |
| Free disk space | 5 GB |

Table 3.1.1. Minimum Hardware Requirements

| Software | Details |
|----------------------|------------------------------|
| Operating System | Windows 11 |
| Programming language | Python 3.9 |
| Libraries | Tkinter, ast, socket, random |

Table 3.1.2. Software Requirements

3.2 Introduction to Cryptography

Cryptography is the technique of securing information and communications through use of codes so that only those persons for whom the information is intended can understand it and process it. Thus preventing unauthorized access to information. The prefix “crypt” means “hidden” and the suffix graphy means “writing”.

In Cryptography the techniques which are used to protect information are obtained from mathematical concepts and a set of rule based calculations known as algorithms to convert messages in ways that make it hard to decode it. These algorithms are used for cryptographic key generation, digital signing, verification to protect data privacy, web browsing on the internet and to protect confidential transactions such as credit card and debit card transactions.^[1]

A cryptographic algorithm, or cipher, is a mathematical function used in the encryption and decryption process. A cryptographic algorithm works in combination with a key - a word, number, or phrase - to encrypt the plaintext. The same plaintext encrypts to different ciphertext with different keys. The security of encrypted data is entirely dependent on two things: the strength of the cryptographic algorithm and the secrecy of the key. A cryptographic algorithm, plus all possible keys and all the protocols that make it work, comprise a cryptosystem.^[9]

3.3 Types of cryptography

In general there are three types Of cryptography:

1. Symmetric Key Cryptography:

It is an encryption system where the sender and receiver of a message use a single common key to encrypt and decrypt messages. Symmetric Key Systems are faster and simpler but the problem is that sender and receiver have to somehow exchange keys in a secure manner. The most popular symmetric key cryptography system is Data Encryption System(DES).

2. Hash Functions:

There is no usage of any key in this algorithm. A hash value with fixed length is calculated as per the plain text which makes it impossible for contents of plain text to be recovered. Many operating systems use hash functions to encrypt passwords.

3. Asymmetric Key Cryptography:

Under this system a pair of keys is used to encrypt and decrypt information. A public key is used for encryption and a private key is used for decryption. Public keys and Private keys are different. Even if the public key is known by everyone, the intended receiver can only decode it because he alone knows the private key.^[8]

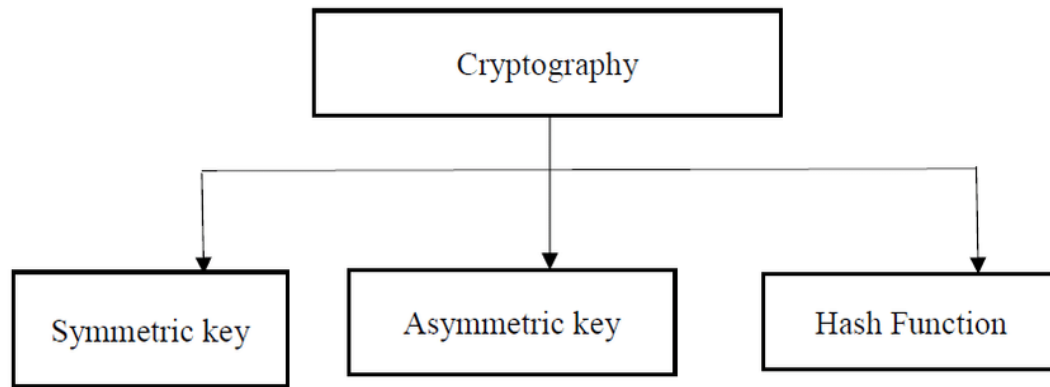


Figure 3.1.1 Different types of Cryptography methods

Difference between Symmetric and Asymmetric Cryptography

| Symmetry Key Cryptography | Asymmetric Key Cryptography |
|--|--|
| It only requires a single key for both encryption and decryption. | It requires two keys, one to encrypt and the other one to decrypt. |
| The size of cipher text is the same or smaller than the original plain text. | The size of cipher text is same or larger than the original plain text |
| The encryption process is very fast. | The encryption process is slow. |
| It only provides confidentiality. | It provides confidentiality, authenticity and non-repudiation. |
| Examples: 3DES, AES, DES and RC4 | Examples: Diffie-Hellman, ECC, El Gamal, DSA and RSA |

Table 3.3.1: Difference between Symmetric and Asymmetric Key Cryptography

3.4 RSA Algorithm

The RSA algorithm (Rivest-Shamir-Adleman) is the basis of a cryptosystem -- a suite of cryptographic algorithms that are used for specific security services or purposes -- which enables public key encryption and is widely used to secure sensitive data, particularly when it is being sent over an insecure network such as the internet.^[3]

RSA was first publicly described in 1977 by Ron Rivest, Adi Shamir and Leonard Adleman of the Massachusetts Institute of Technology, though the 1973 creation of a public key algorithm by British mathematician Clifford Cocks was kept classified by

the U.K. 's GCHQ until 1997.

Public key cryptography, also known as asymmetric cryptography, uses two different but mathematically linked keys -- one public and one private. The public key can be shared with everyone, whereas the private key must be kept secret.^[11]

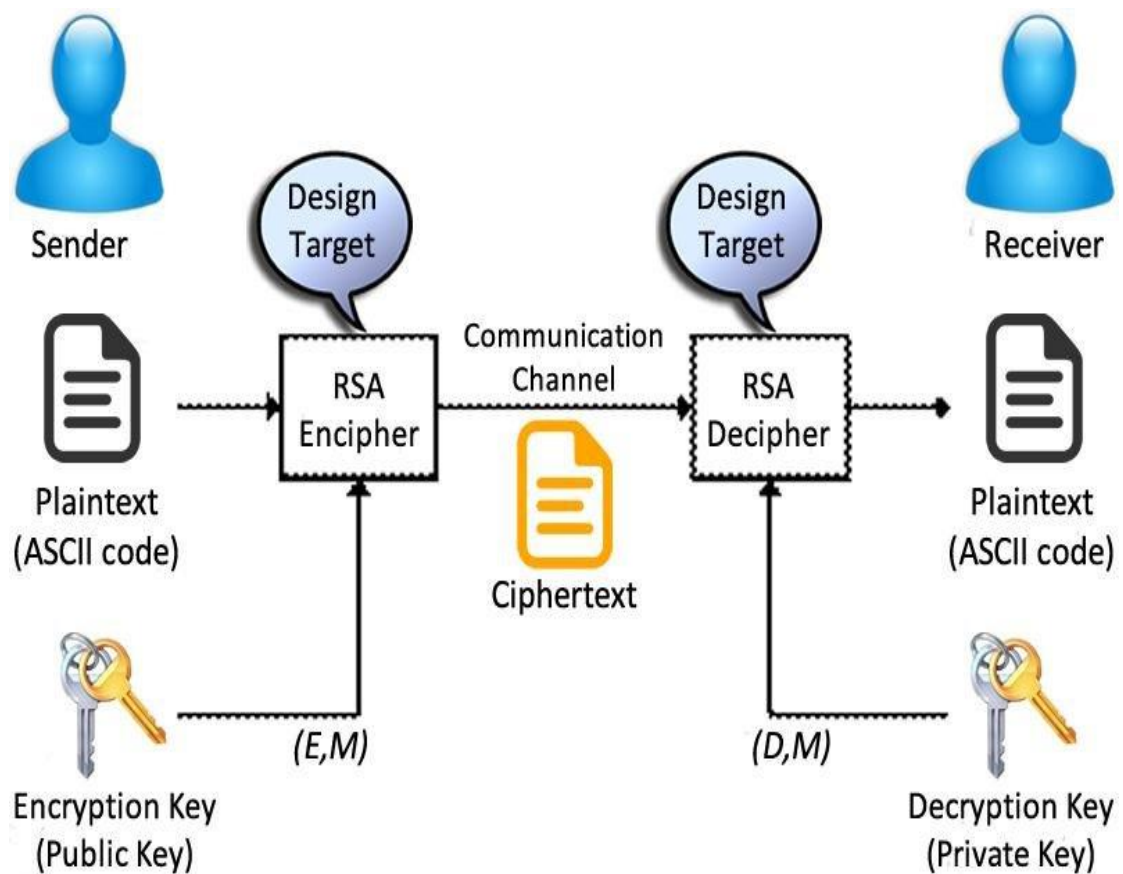


Figure 3.2 How RSA encryption work

Implementation of RSA Algorithm

Steps involved in RSA Algorithm are,

Step 1: Two random prime numbers p and q are taken such that p is not equal to q , p and q are kept secret.

Step 2: n is called modulus for both encryption and decryption and is calculated, $n = p * q$.

Step 3: $\Phi(n)=(p-1)(q-1)$ is calculated where ‘ Φ ’ is Euler’s totient function. This is to be kept private.

Step 4: An integer ‘e’ is selected for encryption such that $\gcd(\Phi(n),e)=1$ and $1<e<\Phi(n)$. Public key (e,n) is generated.

Step 5: $d=e^{-1}(\text{mod } \Phi(n))$ is calculated, where d is the modular multiplicative inverse of e. Private key (d,n) is generated here.

Step 6: $C = M^e \text{ mod } n$ is calculated, where C is the Ciphertext and M is the plaintext or the actual message.

Step 7: The cipher text is transmitted to the receiver.

Step 8: The receiver decrypts the encrypted message/ciphertext using the private key with the formula Plaintext $M = C^d \text{ mod } n$.^[4]

3.5 Socket Programming

Sockets and the socket API are used to send messages across a network. They provide a form of inter-process communication (IPC). The network can be a logical, local network to the computer, or one that’s physically connected to an external network, with its own connections to other networks. The obvious example is the Internet, which you connect to via your ISP.^[2]

TCP Sockets

To create a socket object using `socket.socket()`, specifying the socket type as `socket.SOCK_STREAM`. When you do that, the default protocol that’s used is the Transmission Control Protocol (TCP). This is a good default and probably what you want.

Why should you use TCP? The Transmission Control Protocol (TCP):

- Is reliable: Packets dropped in the network are detected and retransmitted by the sender.
- Has in-order data delivery: Data is read by your application in the order it was written by the sender.

In contrast, User Datagram Protocol (UDP) sockets created with

socket.SOCK_DGRAM aren't reliable, and data read by the receiver can be out-of-order from the sender's writes.

Why is this important? Networks are a best-effort delivery system. There's no guarantee that your data will reach its destination or that you'll receive what's been sent to you.

Network devices, such as routers and switches, have finite bandwidth available and come with their own inherent system limitations. They have CPUs, memory, buses, and interface packet buffers, just like your clients and servers. TCP relieves you from having to worry about packet loss, out-of-order data arrival, and other pitfalls that invariably happen when you're communicating across a network.^[13]

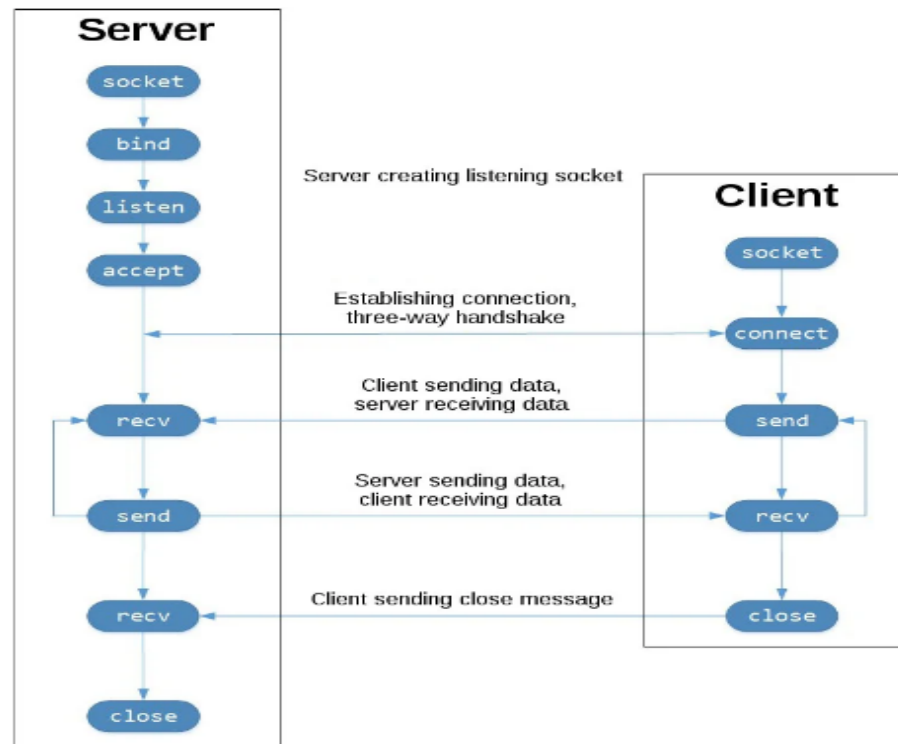


Figure 3.5 Socket Programming Connection Diagram

3.6 GUI

Tkinter

The tkinter package (“Tk interface”) is the standard Python interface to the Tcl/Tk GUI toolkit. Both Tk and tkinter are available on most Unix platforms, including macOS, as well as on Windows systems.

Running `python -m tkinter` from the command line should open a window demonstrating a simple Tk interface, letting you know that tkinter is properly installed on your system, and also showing what version of Tcl/Tk is installed, so you can read the Tcl/Tk documentation specific to that version.

Tkinter supports a range of Tcl/Tk versions, built either with or without thread support. The official Python binary release bundles Tcl/Tk 8.6 threaded. See the source code for the `_tkinter` module for more information about supported versions.

Tkinter is not a thin wrapper, but adds a fair amount of its own logic to make the experience more pythonic. This documentation will concentrate on these additions and changes, and refer to the official Tcl/Tk documentation for details that are unchanged.^[12]

CHAPTER 4

In this section we discuss about the methodology and the working of the Project.

Methodology

4.1 Block Diagram

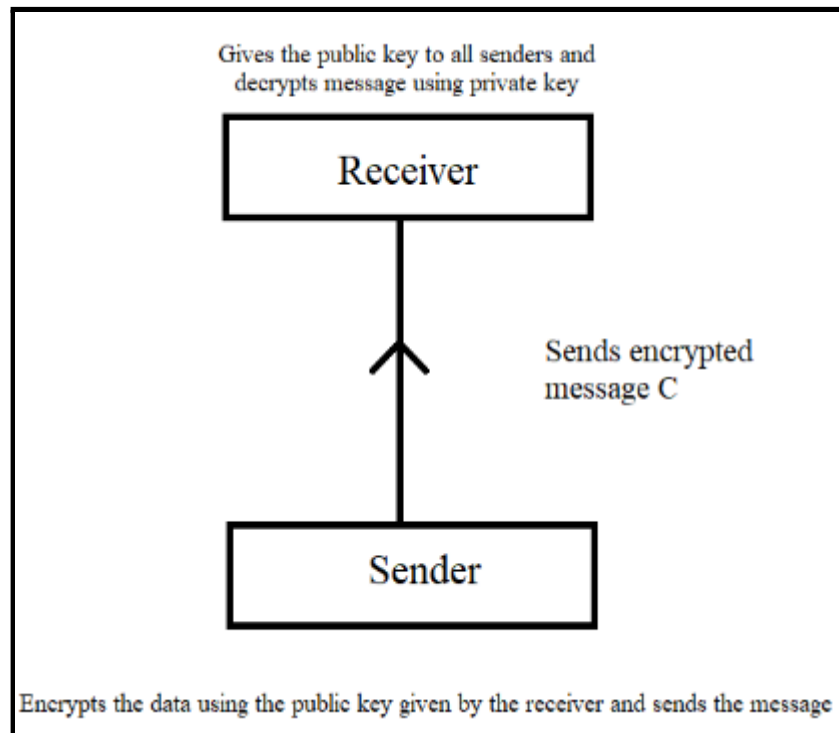


Figure 4.1: Block diagram of the proposed system

Working

In this section we will look into the working of the project "Secured Wireless Communication using RSA", which consists of socket programming, RSA algorithm and database.

Firstly, the receiver will generate the public and private keys, of which he will keep the private key a secret and the public key is given to the senders. The keys will be generated using the RSA algorithm. The RSA algorithm uses two large prime numbers as their multiplication will be difficult to factorize. The public and private keys consist of two numbers, in which the second number is the multiplication of the chosen prime numbers. If this number is factorized, the private key will be compromised. Hence, the security of the communication solely depends on how large the two prime numbers are. The generation of the public and private keys has already been explained in section 3.4.

After the generation of public and private key, the sender and receiver will be connected using socket programming. Socket Programming deals with the establishment and maintenance of the connection between the two users by using their ip addresses. Suppose the sender wants to send a message M, he will encrypt the ASCII values of the message using the public key and send it to the receiver.

The receiver will receive the encrypted message in the form of a list of numbers. When the receiver decrypts this list of numbers, he will get the original message again in ASCII values, which is later changed to their respective characters. Hence the original message M is received.

Since it is possible that there are multiple senders, the system of the receiver maintains a database of the encrypted values of the messages along with the ip addresses of the respective senders. The system prompts the receiver to enter the ip address of the sender of the message he wants to view. Even if a third person opens the database file, he will only see the list of numbers and the confidentiality of the message is ensured.

CHAPTER 5

HARDWARE AND SOFTWARE IMPLEMENTATION

This project is based on the software side and is used in the following applications:

Applications

In this section, applications of the RSA algorithm are discussed.

RSA cryptography is used in many fields for secure data transmission such as:-

Banking: In banking, RSA algorithm is used to protect their data like customer transaction records, credit and debit card details and customer personal information.

E-commerce: Used for protecting an user's transaction details and for securing communication between e-commerce sites and browsers with the help of SSL (secure) certificates.

VPN: Virtual private network uses RSA algorithm to form a secure connection between vpn clients and vpn servers.

Telecommunications: Used to encrypt the call information/data during a call for privacy and security issues.

Digital signature: RSA is used mostly in hybrid encryption schemes and digital signatures. Digital signatures are used to verify the authenticity of the message sent electronically.

CHAPTER 6

In this section we discuss about the result and discussion of the project

Result and Discussion

The complete development of the project was discussed and this system was divided into the following stages:

Problem definition stage;

Designing block diagram;

Developing algorithm for software;

Writing code for communication;

Compiling the code;

Testing and Running.

Problem definition stage

This is the very first stage to develop any project. It actually defines the aim and the concept of the project. The aim of “**Secured Wireless Communication using RSA**” is to design a device which will secure the communication between the two Clients .

Designing block diagram

At this stage we have categorized the whole system into different individual modules. These modules (block diagrams) will be helpful in understanding the concept and working of the integrated system. It also simplifies the entire debugging and testing process. So the result was the block diagram of the project.

Developing algorithm for software

To get the logical flow of the software, the development of algorithms is having a prominent role. So that we have analyzed the complete system and organized the algorithm in such a manner that one can understand the complete working of the software.

Writing Code for Communication

After the development of the algorithm we write the code for communicating between the users/client for the project. Code is written in “Python” language.

Compiling the code

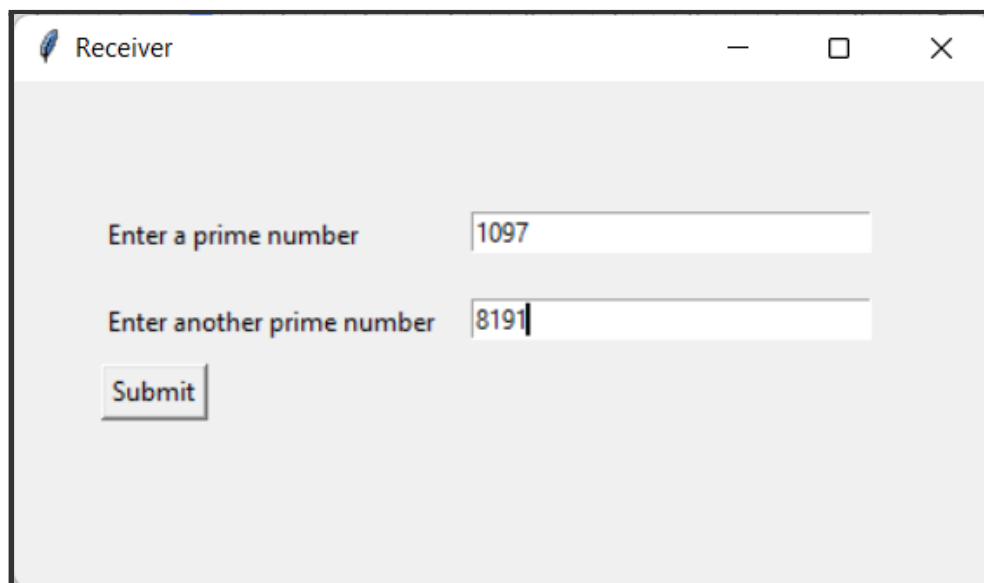
The code is implemented on the computer for which we have used a compiler named Pycharm pre-installed on PC. PyCharm is a dedicated Python Integrated Development Environment (IDE) providing a wide range of essential tools for Python developers, tightly integrated to create a convenient environment for productive Python, web, and data science development.

Testing and Running

This time we tested our project for actual working, after compiling the code . Any errors found were removed successfully. This is the last and final stage of development of our project.

Sample of Inputs/Dataset/Database Used/ and Outputs/ScreenShots

1. **Message:** My Credit Card number is 6484976
2. **Prime numbers:**1097&8191



The screenshot shows a web application window titled "Receiver". Inside the window, there are two text input fields. The first field is labeled "Enter a prime number" and contains the text "1097". The second field is labeled "Enter another prime number" and contains the text "8191". Below these two fields is a button labeled "Submit". The window has a standard title bar with minimize, maximize, and close buttons.

Figure 6.1: Receiver, Entering the two prime numbers



Figure 6.2: Receiver, Generation of public and private keys



Figure 6.3: Receiver, Entering the private key

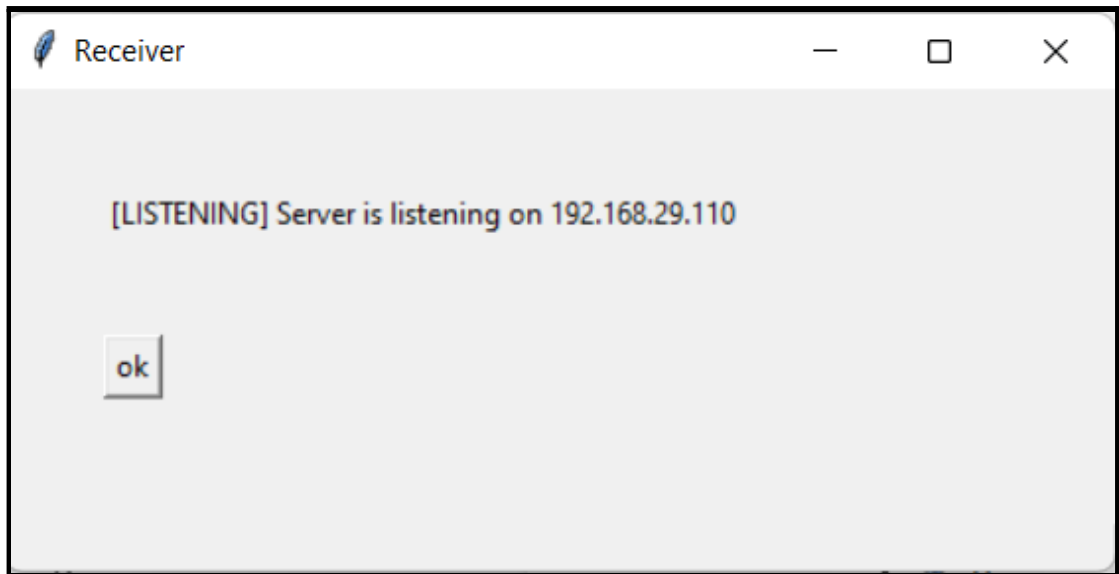


Figure 6.4: Receiver, Server is listening

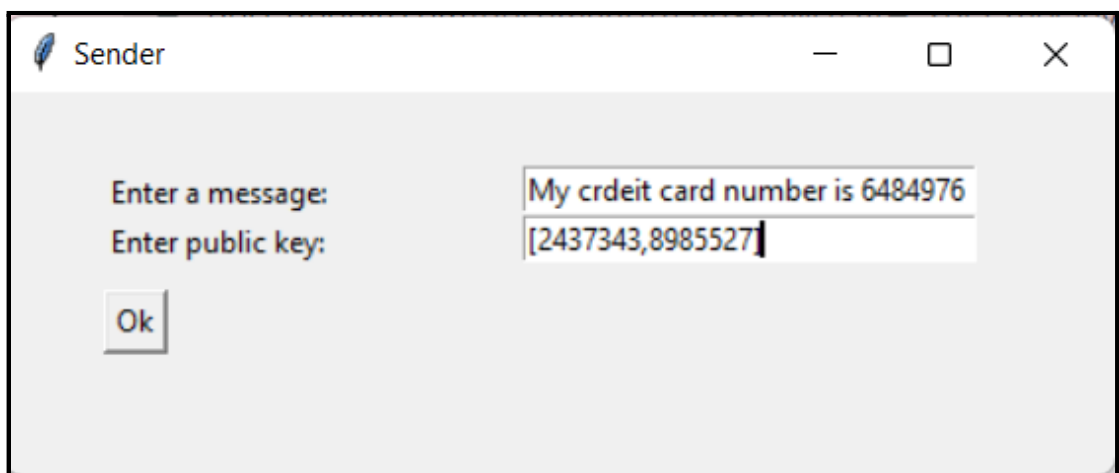


Figure 6.5: Sender, Sending a Message

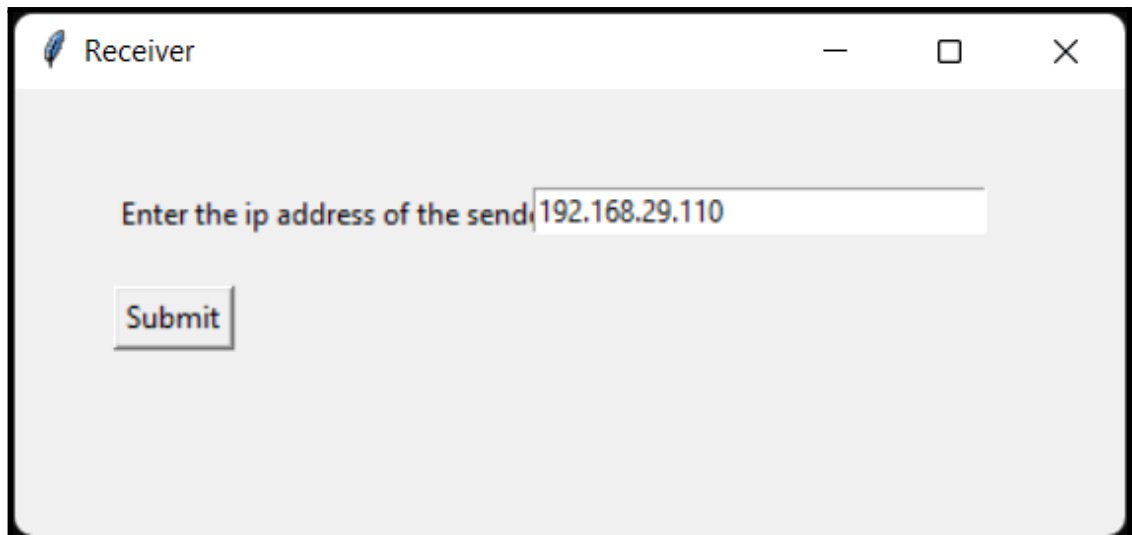


Figure 6.6: Receiver, Entering the ip address of the sender

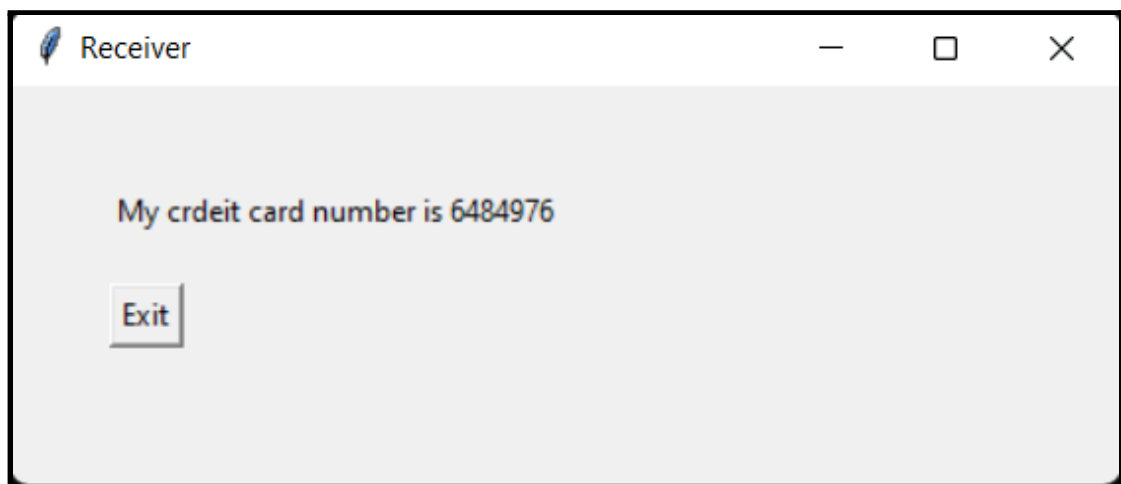


Figure 6.7: Receiver, Message received by the receiver

Chapter 7

Conclusion and Future Scope

5.1 Conclusion

In this report the study of Securing Wireless Communication using RSA and the implementation of cryptography is presented. A comparative study of the above-mentioned asymmetric cryptography method is presented in this report. The different techniques such as AES and RSA are explained. The different hybrid approaches are also described. The comparative study of various techniques mentioned above is presented. The proposed method securely transfers data between two systems present on the same network.

5.2 Future Scope

- * In this section, the improvements which can be made to the system to work better and faster has been discussed.

- * Using a hybrid version of RSA and AES makes the communication faster and more secure.

- * This can be done by encrypting the contents of a file using a symmetric key algorithm like AES, and encrypting the key generated by the AES algorithm using the RSA algorithm.

References

- [1] Dr. Niraj Singhal, Shaili Singhal, "A Comparative Analysis of AES and RSA Algorithms," *International Journal of Scientific & Engineering Research*, Volume 7, Issue 5, May-2016.
- [2] Gordon McMillan, <https://docs.python.org/3/howto/sockets.html>
- [3] Lavanya K. Galla, Venkata SreeKrishna Koganti, Nagarjuna Nuthalapati, Implementation of RSA," 2016 *International Conference on Control, Instrumentation, Communication and Computational Technologies*.
- [4] Nentawe Y. Goshwe, "Data Encryption and Decryption Using RSA Algorithm in a Network Environment," *IJCSNS International Journal of Computer Science and Network Security*, Vol.13 No.7, July 2013.
- [5] M.Pitchaiah, Philemon Daniel, Praveen, "Implementation of Advanced Encryption Standard Algorithm," *International Journal of Scientific & Engineering Research* Volume 3, Issue 3, March -2012.
- [6] Nitin Jain, Surendra Singh Chauhan, Alok Raj, "Security Enhancement of RSA Algorithm using Increased Prime Number Set," *International Journal of Engineering and Advanced Technology*, Volume-9 Issue-3, February 2020.
- [7] Palanisamy, V. and Jeneba Mary, A., "Hybrid cryptography by the implementation of RSA and AES," *International Journal of Current Research* Vol. 3, Issue, 4, pp.241-244, April, 2011.
- [8] Greeksforgeeks-<https://www.geeksforgeeks.org/cryptography-and-its-types/> -August 2016
- [9]Cs.stonybrook-
<https://www.cs.stonybrook.edu/sites/default/files/PGP70IntroToCrypto.pdf>
-September 2018
- [10] International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249-8958 (Online), Volume-9 Issue-3, February 2020
- [11] Techtarget-<https://www.techtarget.com/searchsecurity/definition/RSA> - January 2019

[12] PythonDocs-<https://docs.python.org/3/library/tkinter.html#module-tkinter> -
December 2018

[13] Geeksforgeeks-<https://www.geeksforgeeks.org/socket-programming-python/>
April 2014

ACKNOWLEDGEMENT

It gives us great pleasure and immense satisfaction to present this report on our project “**Secured Wireless Communication Using RSA**”, which became possible due to the unstinted guidance and focused direction of Prof. **Tusharika Banerjee**, Electronics & Telecommunication Department.

We express our sincere gratitude to Prof. **Dr. Avinash R. Vaidya**, HOD, Electronics & Telecommunication Department without whom it would not have been possible to successfully accomplish our project.

Furthermore, we are indebted to the Principal **Dr. Sandeep Joshi** whose constant encouragement and motivation inspired us to do our best.

Last, but not the least, we sincerely thank our family members, colleagues and all the others who directly or indirectly contributed in making our task easier.

APPENDIX

CODE OF THE PROJECT

1. p1 Package

```
import random

def multiplicative_inverse(e, phi):
    d = 0
    x1 = 0
    x2 = 1
    y1 = 1
    temp_phi = phi

    while e > 0:
        temp1 = temp_phi // e
        temp2 = temp_phi - temp1 * e
        temp_phi = e
        e = temp2

        x = x2 - temp1 * x1
        y = d - temp1 * y1

        x2 = x1
        x1 = x
        d = y1
        y1 = y

    if temp_phi == 1:
        return d + phi

def is_prime(num):
    if num == 2:
        return True
    if num < 2 or num % 2 == 0:
        return False
    for n in range(3, int(num**0.5)+2, 2):
        if num % n == 0:
            return False
    return True
```

```

def gcd(a, b):
    while b != 0:
        a, b = b, a % b
    return a

def generate_key_pair(p, q):
    if not (is_prime(p) and is_prime(q)):
        raise ValueError('Both numbers must be prime.')
    elif p == q:
        raise ValueError('p and q cannot be equal')
    # n = pq
    n = p * q

    # Phi is the totient of n
    phi = (p-1) * (q-1)

    # Choose an integer e such that e and phi(n) are coprime
    e = random.randrange(1, phi)

    # Use Euclid's Algorithm to verify that e and phi(n) are coprime
    g = gcd(e, phi)
    while g != 1:
        e = random.randrange(1, phi)
        g = gcd(e, phi)

    # Use Extended Euclid's Algorithm to generate the private key
    d = multiplicative_inverse(e, phi)

    # Return public and private key_pair
    # Public key is (e, n) and private key is (d, n)
    return ((e, n), (d, n))

def decrypt(pk, ciphertext):
    # Unpack the key into its components
    key, n = pk
    # Generate the plaintext based on the ciphertext and key using a^b mod
    m
    aux = [str(pow(char, key, n)) for char in ciphertext]
    # Return the array of bytes as a string
    plain = [chr(int(char2)) for char2 in aux]
    return ''.join(plain)

```

2. Keys

```
import p1
#rsa
import random
from tkinter import *

def submit():
    global p
    global q
    p = int(P.get())
    q = int(Q.get())
    top.destroy()

top = Tk()
top.title("Receiver")
top.geometry("450x300")

# the label for user_name
number1 = Label(top, text = "Enter a prime number").place(x = 40, y = 60)

# the label for user_password
number2 = Label(top, text = "Enter another prime number").place(x = 40, y
= 100)

submit_button = Button(top, text = "Submit", command = submit).place(x
= 40, y = 130)

P = Entry(top, width = 30)
P.place(x = 210, y = 60)

Q = Entry(top, width = 30)
Q.place(x = 210, y = 100)

top.mainloop()

frame = Tk()
frame.title("Receiver")
frame.geometry("450x300")
f1 = Label(frame, text=" - Generating your public / private key-pairs now
. . .").place(x=40, y=40)
public, private = p1.generate_key_pair(p, q)
keys = "- Your public key is "+ str(public)+ " and your private key is "+
str(private)
```

```
f2 = Label(frame, text=keys).place(x=40, y=60)
b = Button(frame, text='ok', command=frame.destroy).place(x=40,
y=100)
```

```
frame.mainloop()
```

3. Receiver

```
import sqlite3
import socket
import threading
import ast
import p1
#rsa
import random
from functools import partial
import json
```

```
#Print on screen
```

```
from tkinter import *
```

```
def read_key():
    global top
    global private1
    global inp1
    private1 = (inp1.get()).strip()
    top.destroy()
```

```
def read_msg():
    global s_name
    global addr
    temp1=""
    conn1 = sqlite3.connect('test2.db')
    mycursor=conn1.cursor()
    ip=(s_name.get()).strip()
    mycursor.execute("SELECT MESSAGE FROM MESSAGES
WHERE IP_ADDRESS=?", (ip,))
    result=mycursor.fetchone()
    for x in result:
        temp=json.loads(x)
        mesg =
p1.decrypt(tuple(ast.literal_eval(private1)),ast.literal_eval(temp))
```

```

    temp1+=mesg

window=Tk()
window.title("Receiver")
window.geometry("450x300")
ms=Label(window, text=temp1).place(x=40,y=40)
Bu=Button(window, text='Exit',
command=window.destroy).place(x=40,y=80)
window.mainloop()

def sname():
    global s_name
    frame=Tk()
    frame.title("Receiver")
    frame.geometry("450x300")
    snam= Label(frame, text='Enter the ip address of the sender:
').place(x=40, y=40)
    s_name=Entry(frame,width=30)
    s_name.place(x=210,y=40)
    enter = Button(frame, text='Submit',
command=read_msg).place(x=40,y=80)
    frame.mainloop()

#Connection
HEADER = 64
PORT = 5050
SERVER = socket.gethostbyname(socket.gethostname())
ADDR = (SERVER, PORT)
FORMAT = 'utf-8'
DISCONNECT_MESSAGE = "!DISCONNECT"

server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server.bind(ADDR)

def handle_client(conn, addr):
    global private1
    global inp1
    global msg
    global top
    global connected
    conn1 = sqlite3.connect('test2.db')

```



```

try:
    conn1.execute("CREATE TABLE MESSAGES
        (ID INT PRIMARY KEY NOT NULL,
        IP_ADDRESS TEXT NOT NULL,
        MESSAGE TEXT NOT NULL);")
except:
    pass;
print(f'[NEW CONNECTION] {addr} connected.')

connected = True
count=0
while connected:

    msg_length = conn.recv(HEADER).decode(FORMAT)
    if msg_length:
        msg_length = int(msg_length)
        msg = conn.recv(msg_length).decode(FORMAT)
        listToStr = json.dumps(msg);
        if msg == DISCONNECT_MESSAGE:
            connected = False
            count+=1;
            conn1.execute("INSERT OR REPLACE INTO MESSAGES
(ID,IP_ADDRESS,MESSAGE)
VALUES(?,?,?)",(count,addr[0],listToStr))
            conn1.commit()
            conn.send("Msg received".encode(FORMAT))

conn.close()

def start():
    server.listen()
    T = Tk()
    T.title("Receiver")
    T.geometry("450x300")
    msg_lis = f'[LISTENING] Server is listening on {SERVER}'
    msg1 = Label(T, text=msg_lis).place(x=40, y=40)
    bu = Button(T, text='ok', command=T.destroy).place(x=40, y=100)
    T.mainloop()
    while True:
        try:
            conn, addr = server.accept()

```

```

        thread = threading.Thread(target=handle_client, args=(conn,
addr))
        thread.start()
        print(f"[ACTIVE CONNECTIONS] {threading.activeCount() -
1}")
    except KeyboardInterrupt:
        print('You hit keyboard interrupt')
        sname()
        break

top = Tk()
top.title("Receiver")
top.geometry("450x300")
inp= Label(top, text= 'Enter private key: ').place(x=40, y=40)
inp1 = Entry(top,width = 30)
inp1.place(x = 210,y = 40)
ok = Button(top, text = 'Ok', command = read_key).place(x = 40,y = 80)
top.mainloop()
start()

```

4. **Sender**

```

import socket
import ast
from tkinter import *

HEADER = 64
PORT = 5050
FORMAT = 'utf-8'
DISCONNECT_MESSAGE = "!DISCONNECT"
SERVER = "192.168.128.102" #Write the IP address of the
receiver
ADDR = (SERVER, PORT)

def encrypt(pk, plaintext):
    # Unpack the key into its components
    key, n = pk
    # Convert each letter in the plaintext to numbers based on the character
using a^b mod m
    cipher = [pow(ord(char), key, n) for char in plaintext]
    # Return the array of bytes
    return cipher
#communication

```

```

client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
client.connect(ADDR)

def send(msg):
    message = msg.encode()
    msg_length = len(message)
    send_length = str(msg_length).encode(FORMAT)
    send_length += b' ' * (HEADER - len(send_length))
    client.send(send_length)
    client.send(message)
    print(client.recv(2048).decode(FORMAT))

def read_msg():
    global mesg
    global public
    global n
    mesg = m2.get()
    public = p2.get()
    encrypted_msg = encrypt(tuple(ast.literal_eval(public)),mesg)
    send(str(encrypted_msg))
    m2.delete(0,END)
    p2.delete(0,END)
    top.destroy()

top = Tk()
top.title("Sender")
top.geometry("450x300")
m1 = Label(top, text='Enter a message: ').place(x=40,y=30)
m2 = Entry(top, width = 30)
m2.place(x = 210,y=30)
ok_button = Button(top, text='Ok', command = read_msg).place(x = 40,y
= 80)
p1 = Label(top, text='Enter public key: ').place(x=40,y=50)
p2 = Entry(top, width = 30)
p2.place(x = 210,y=50)
top.mainloop()

```