SECURE VOICE-DATA TRANSMISSION USING MODIFIED RSA

RITIKA SINGH, SAURABH SINGH, and PARTH BADKUL

***Abstract*—As the prevalence of Voice assistants surges, companies which collect speech data to improve their assistants, put millions of consumers' personal data at risk to attacks by hackers or crackers, who may use the data for nefarious purposes such as selling of private information to third party companies or forging voices to create fake calls or messages in our name.**

To help combat this threat, we propose a system which converts the speech commands into text, encrypted using Modified RSA, and hashed with salt to be stored anonymously so that the most hackers can get would be random texts of customer preferences.

The Google Speech API and Socket connections which emulate the communication between assistants and servers, send encrypted data over to the client in text form so that the voice data is not stored and the assistant gets the data it needs to train itself.

Keywords-

GOOGLE SPEECH API, CRYPTOGRAPHY, ENCRYTPTION, DECRYPTION, PUBLIC AND PRIVATE KEY, SPEECH RECOGNITION, SOCKET PROGRAMMING , RSA, MODIFIED RSA, HASHING AND SALTING

I. INTRODUCTION

This project on **Secure Voice Transmission** overcomes the threats faced by people who are excessively using Voice Assistants with the comprehended knowledge and concepts of the currently flourishing and expanding field of **Cryptography** and **Network S ecurity**.

Voice assistants are a very convenient feature increasingly being found on most smartphones and other home appliances. The main concept of these voice assistants is itself listening to its users conversations and storing the data and learning from it. Also, as the companies behind voice assistants admit that our voice commands are being recorded for quality assurance purposes, the personal data of millions of consumers is at risk.

This leaves the possibility of hackers breaking into companies' databases and steal our personal information, preferences and the structure of our voices which can be forged to sound like us.

Thus to mitigate the potential damage posed by this intrusion, **we propose a system** which

- converts the speech commands into text,
- encrypted using double key RSA,
- and hashed and salted to be stored anonymously

so that the most hackers can get would be random texts of customer preferences.

This model can also help to send secure voice messages from onedevice to another. Using the speech to text function and the secured encryption and authentication system, voice messages can be used for convenience or for the visually impaired.

To implement this model, we use **socket connection** and the **Google Speech to Text API** to emulate the voice assistants.

It is then encrypted by a **modified RSA** encryption system for encrypting the text and the key to decrypt the text. The ciphertext is then **hashed with a salt** to prevent easy decryption.

II. RELATED METHODS/LITERATURE SURVEY

[1]**Socket Programming - M.Tech Student, Department of Computer Science and Engineering, Assam Down Town University, Guwahati, India.** According to this paper, the four major operations of a socket are to connect to a remote machine, Send data, Receive data and Close the connection**.** A network is composed of computers which is either a client or a server. A server is a program that is offering some service whereas a client is a program that is requesting some service. Sockets are end-points of a connection between two separate systems identified by unique IP Address and a port number. It also states that when a network application is created, the developer's main task is to write the code for both the client and server programs.

[2]**Speech to text and text to speech recognition systems-Areview Ayushi Trivedi,Navya Pant, Pinal Shah,Simran Sonik and Supriya Agrawal DepartmentofComputerScience, NMIMS University,Mumbai,India**. Corresponding Author:Navya Pant. . According to this paper, a path-breaking innovation has recently come to play in the SMS technology using the speech recognition technology, where voice messages are being converted to7 text messages. Over the past few years, Cell Phones have become an indispensable source of communication for the modern society. It is known that verbal communication is the most appropriate modem of passing on and conceiving the correct information, avoiding misquotations. To fulfil the gap over a long distance, verbal communication can take place easily on phone calls Quite a few applications used to assist the disabled make use of TTS, STT, and translation. They can also be used for other applications, taking an example: Siri. Speech Recognition is the ability of machine/program to identify words and phrases in spoken language and convert them into machine-readable format.

[3]**R . L. Rivest, A. Shamir and L. Adleman, "On Digital Signatures and Public Key Cryptosystems", Technical Memo 82, Laboratory for Computer Science, Mas-**

sachusetts Institute of Technology, April 1970. Public key cryptography is the basis for many modern cryptosystems. The public key cryptosystem proposed by Rivest, Shamir and Adleman in this paper, commonly called the RSA algorithm, is one of the earliest and most effective of its kind. It is a one-way trap-door permutation. The algorithm proposes the use of 2 large distinct prime numbers to calculate a pair of private and public keys for decryption and encryption respectively. The prime numbers, p and q are used to calculate the product, n=p*q. The Euler's function of n, phi(n)=(p-1)*(q-1) is also computed. An integer e is chosen such that 1<e.

[4]**Ms. Ritu Patidar, Mrs. Rupali Bhartiya, "Modified RSA Cryptosystem Based on Offline Storage and Prime Number".** The authors of this paper acknowledge the significance of the RSA algorithm while also addressing its shortcomings such as its lack of security and long computation time. Thus to tackle these issues, a modified model of RSA was proposed in this paper. Using a third prime number r to calculate n along with p and q can make the modulus n not easily decomposable by possible hackers. The key parameters of the algorithm are stored in an offline database before the algorithm starts to help8 speed up the computational process. The key generation process, encryption and decryption methods of the RSA algorithm remain relatively unchanged. It would be extremely difficult to guess the values of p, q and r simultaneously from the offline databases. Thus the proposed algorithm in this paper was compared with the original algorithm and it showed marked improvements.

[5] Survey Paper on Different Type of Hashing Algorithm - Mahesh A. Kale, Prof. Shrikant Dhamdhere. Hash Functions plays an important role in data security over the web. The hash capacities that are used as a part of different security related applications are called cryptographic hash capacities. Message correctness is given through this, i.e. on the off chance that the message has been changed after transmission from sender and before it might be gotten by the comparing receiver, can be followed by the accumulator, and along these lines, such an altered message can be disposed of. According to the research done in this paper, the vast majority of the hash capacities depend on Merkle-Damgard development, for example, MD-5, SHA-1, SHA2, SHA3 etc, which are definitely not hundred percent safe from assaults. This paper examines a few of the assaults that are conceivable on this development, and subsequently on these hash capacities likewise confront same assaults.

[6] Overview Of Web Password Hashing Using Salt Technique-Diksha.S.Borde, Poonam.A.Hebare, Priyanka.D.Dhanedhar. Using simple hashing to protect data has become vulnerable these days due to the increase in large of hacking tools and high performance computing . So, a process of salting was introduced to add more security to the hash generated. Salt provides security. From the overview provided by this paper Dictionary Attack, Birthday Attack can be prevented by using salting as attackers will now have to recalculate their entire dictionary for every individual account they're attempting to crack. But this paper points out that a salt can only help against prebuilt dictionaries. If an intruder gets access to the system and if he/she uses brute force attack, then salt will not provide adequate security.

III. PROBLEM STATEMENT

As we are experiencing a significant change where gadgets are becoming smarter to perform tasks only on voice commands. People communicate with one another through voice in a common network. This can be problematic as any third party could intercept their calls and be hear or modify the contents of the speaker and receiver.

Unpatched Security vulnerabilities or unknown assets in the network makes it dangerous for consumers to communicate. A third party could abuse the users account privileges this way.So, the main problem statement of this project is making data transfer and communication more secure by adding triple layers of cryptographic security.

The novelty of our project lies in the fact that we are using socket communication to demonstrate the secure transfer of a voice command between client and server, and all that any middle man or a hacker will hear will be a desired gibberish.

IV. PROPOSED METHODOLOGY

The methodology undertaken by us was to provide an implementation of Secure Voice Data Transmission using various cryptographic encryption techniques like Modified RSA, hashing and salting.

The methodologies included in our proposed model are explained below-

1- ***GOOGLE SPEECH API :***

- It is an open source API which can convert spoken text through a microphone into text.
- Speech Recognition is the ability of machine/program to identify words and phrases in spoken language and convert them into machine-readable format.
- Speech to text conversion is the process of converting spoken words into written texts.

2- ***SOCKET PROGRAMMING :***

- Communication between computers or nodes is important to allow us to interact with devices and for devices to interact with each other.
- One of the ways we can establish a connection between the 2 nodes is through Socket programming.
- To establish a connection, one of the sockets listens in for an active connection request (server), while the other actively sends the request (client) on the same IP and port.
- A socket can have multiple clients and servers and can work efficiently using threads.

3-***RSA* :**

Proposed by Rivest, Samir and Adleman, the RSA algorithm is one of the most popular public key asymmetric encryption systems in use today.

The algorithm consists of Key generation, Encryption and Decryption.

Algorithm:

- Select two unique and large prime numbers p and q.
- Compute n=p*q.
- Calculate phi(n)=(q-1)(p-1) (Euler's formula)

• An integer e is chosen, such that 1<e<f(n) and GCD(e,f(n))=1 **i.e** e and f(n) are co-primes of each other.

• 5.Compute d as the multiplicative inverse of e mod (f(n))(e*d) mod f(n)=1.

4- ***MODIFIED RSA USING KEY ENCRYPTION*:**

• To improve the security of the generated key, the use of a third prime number 'r' to help calculate the value of 'n' was first proposed in one of the reference papers we referred.

• This can help bolster the security of the cryptosystem and can confuse hackers who are already familiar with the RSA encryption system.

Fig. 1. Original vs Modified RSA

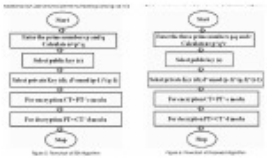

Fig. 2. Algorithm of RSA vs Modified RSA

5-SALTING AND HASHING:

Hashing basically means putting the plaintext through a hash function (which is applied by the hashing algorithm) into a meaningless sequence of aplha-numeric characters which provides Compression, Collision Resistance and Pre-image Resistance.

• **Hashing-Algorithm:** A set of procedure or instructions to apply the hash function to the given data.

• **Hash Function:** A mathematical function that converts some numeric value of a given size to a numeric value of different or same size.

Salting is a process of adding a fixed length cryptographically strong random value to an already generated hash to increase its complexity without increasing user requirements, making the hash unique and to prevent password attacks like Rainbow Tables.

• Hence, for different users, if they have same passwords then also their hashes generated are different making it almost impossible to crack.

V. ALGORITHM – STEPS AND EXPLANATION

1) The voice data is received through the microphone and converted to text using the Google Speech API.
2) The public and private keys are generated and encrypted according to the modified double encrypted RSA algorithm.
3) To add an extra layer of security, the obtained text is encrypted and is hashed and salted to be stored in a database.
4) The encrypted text is sent to the client/receiver using socket communication and is decrypted accordingly.
5) The decrypted text is displayed and is read aloud by the system using the API.

Fig. 3. Algorithm of Secure data transmission using Google speech APU, Modified RSA and hashing and salting.

VI. CODE EXPLANATION

[1] Here we are *importing the socket library* and other necessary modules for RSA encryption, hashing and salting . Then we are creating a socket.

[1] Here we are importing the socket library and other necessary modules for RSA encryption, hashing and salting . Then we are creating a socket.

Also we are importing Google speech recognition API.[1] Here we are importing the socket library and other necessary modules for RSA encryption, hashing and salting . Then we are creating a socket.

Also we are importing Google speech recognition API.

```
#Socket programming is started by
    importing the socket library and
    making a simple socket.
import socket
s=socket.socket()
#importing the RSA,hashing and salting
    function to integrate with our
    socket connection
import speech_recognition as sr
import RSA_cryptoproj  as cry
import HashingAndSalting as hs
import hashlib
```

Code Snippet 1.

[2] This is the code for a *normal speech recognition*.

We set up the microphone using Recognizer and we also make a loop which will continuously listen to our speech. The speech is heard using the listen function Once the speech or words are heard the api interprets it and converts it to text using the recognize google function.

We have also included a condition where the api is not able to receive proper input so we just display could not understand audio

```
r = sr.Recognizer()
#use the microphone as source for input
    .
with sr.Microphone() as source:
    print("Say something!")
 # putting the socket into listening
    mode for any audio from microphone
```

```
 #listens for the user's input
    audio = r.listen(source)
    print ("socket is listening")
    try:
  # for testing purposes, we're just
      using the default API key
  # to use another API key, use `r.
      recognize_google(audio,   key="
      GOOGLE_SPEECH_RECOGNITION_API_KEY
      ")`
    # instead of `r.recognize_google(
      audio)`
        print("You said: " + r.
            recognize_google(audio))
    except sr.UnknownValueError:
        print("Google Speech
            Recognition could not
            understand audio")
    text=r.recognize_google(audio)
    print(text)
```

Code Snippet 2.

[3] This is how *we are integrating speech recognition with sockets*.

We have an ip address and a port number with which our socket connects with The google api is integrated such that the audio is heard, and it is converted to text.

This text is passed securely through the socket to the listener.

```
# reserving a port on our computer in
    our case it is 1234 but it can be #
    anything
port=1234
ip="192.168.42.134"
# creating a socket object
client = socket.socket(socket.AF_INET,
    socket.SOCK_STREAM)
print ("Socket successfully created")
# Next binding to the port to let the
    server listen to requests
client.connect((ip, port))
#Initialize the recognizer
r = sr.Recognizer()
#use the microphone as source for input
    .
with sr.Microphone() as source:
    print("Say something!")
    # putting the socket into listening
        mode for any audio from #
       microphone
    #listens for the user's input
    audio = r.listen(source)
    print ("socket is listening")
    try:
    # for testing purposes, we're just
       using the default API key
    # to use another API key, use `r.
        recognize_google(audio, key="
        GOOGLE_SPEECH_RECOGNITION_API_KEY
        ")`
    # instead of `r.recognize_google(
        audio)`
        print("You said: " + r.
            recognize_google(audio))
    except sr.UnknownValueError:
        print("Google Speech
            Recognition could not
            understand audio")
    text=r.recognize_google(audio)
    print(text)
    enc=cry.encrypt_string(text)
    encrypted_salted_text = hs.salt(enc
        )
    print(enc)
    print(encrypted_salted_text)
#hashing done through SHA-256 algorithm
    Sender_HASH = hashlib.sha256(
        encrypted_salted_text.encode())
        .hexdigest()
    #encoding string to bytes
    client.send(enc.encode())
    client.send(Sender_HASH.encode())
```

Code Snippet 3.

[4] Code for Listener-

In this code *we are connecting to the socket of the sender*. The listen function sets up the socket to receive any request.

The accept function connects with the senders socket and this is the function responsible for receiving the oncoming speech converted to text.

The text comes in encrypted format which is received by the socket using recv function The text is decoded and passed along to the **pyttsx3 module**. This module helps convert the text to voice.

```
#Socket programming is started by
    importing the socket library and #
    making a simple socket.
import socket
import subprocess as sp
import pyttsx3
s=socket.socket()
# reserving a port on our computer in
    our case it is 1234 but it can be
    anything
port= 1234
ip= "192.168.42.134"
# Next binding to the port to let the
    server listen to requests
s.bind((ip,port))
# putting the socket into listening
    mode
s.listen()
cl,addr= s.accept()
voice=pyttsx3.init()
```

```
voice.setProperty('rate',150)
while True:
    data=cl.recv(1024)
    talk=data.decode()
    voice.say(talk)
    voice.runAndWait()
    print(talk)
```

Code Snippet 4.

Hence between two remote systems we can have secure voice conversations or message sending using this application.

Project link: https://github.com/ritikaxx/SecureVoiceTransmission/tree/main

Video link: https://www.youtube.com/watch?v=fI5ti3o7GSQ

VII. RESULTS

The speech to text application worked as expected with minimal errors and the encrypted text was decrypted successfully post sending it through the socket layer and the salted hash was recalculated and verified successfully.

This model can be implemented in a large scale to help protect consumer privacy. On comparing the decryption time taken by Original RSA and the modified RSA, both showed similar trends with increasing key size.

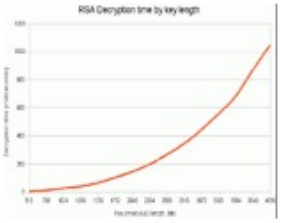

Fig. 4. X-Axis - Key length in bits; Y-Axis - Decryption time in milliseconds

However, there is a marked improvement in the security of the cryptosystem, as the complexity to break up the large number 'n' in the modified RSA is much higher than that in the original RSA.

COMPLEXITY TO BREAK 'N' VS KEY SIZE COMPARISON

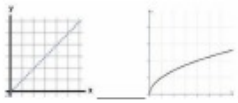

Fig. 5. X axis -Size of the encryption key in bits Y axis –Complexity

OUTPUT SCREENSHOTS

Fig. 6. Encryption and Authentication

Fig. 7. Speech Recognition and socket communication (Sender)

VIII. CONCLUSION AND FUTURE WORK

• The proposed algorithm for secure transmission of voice data has been implemented successfully using modified RSA as an encryption method, which shows marked improvements in security over normal RSA.

• While the decryption time for both approaches are similar, the complexity to break the product of primes 'n' is higher in modified RSA, thus increasing security.

• This is a novel application of the concepts used and can help secure transmission of voice data for multiple applications.

This project has a huge Future Scope including-

• Expanding this Secure Voice Data Transmission to audio and video calls will prove to be great work in future.

• As significant AI adoption is driving the move to **voice applications,** this model can be expanded accordingly to provide user security.

• Additionally, IoT devices such as thermostats, speakers, and smart appliances are making **voice assistants** ever more useful in the lives of everyday users.

REFERENCES

[1] N. Cryptography, W. Security, and Stallings.

[2] R. L. Rivest, A. Shamir, and L. Adleman, "On Digital Signatures and Public Key Cryptosystems," *Technical Memo*, vol. 82, 1970.

[3] Ms, M. R. Patidar, and Bhartiya.

[4] [Online]. Available: https://cloud.google.com/speech-to-text/docs/reference/libraries

[5] [Online]. Available: https://www.geeksforgeeks.org/socket-programming-cc/

[6] P. N. Patel and J. Patel, Eds., *A cryptography application using salt hash technique*.