DEMOCRATIC AND PEOPLE'S REPUBLIC OF ALGERIA MINISTRY

OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH

MOHAMED BOUDIAF UNIVERSITY - M'SILA

FACULTY: Mathematics and Informatics

DEPARTMENT: Computer Science

N°: ……

DOMAIN: Mathematics and Informatics

BRANCH: Computer Science

OPTION: Information and Communication Networks and Technologies

A Dissertation in Fulfillment

For the Requirements of the Degree of Master

By : - Hanane Seghiri
- Rima Belguidoum

Subject

Development of a Secure Messaging App for Android Mobile

Evaluated by the jury composed of:

Ms. Lalia Saoudi	President	University of M'sila
Mr. Noureddine Chikouche	Supervisor	University of M'sila
Mr. Fares Mezrag	Examiner	University of M'sila

Academic Year 2020/2021

Dedications

This modest work is dedicated to my beloved mother, who has been a source of inspiration and encouragement to me throughout my life. To my beloved father who gave everything they ever had, so I could pursue my dream; To my sisters and brothers who supported me financially, morally and psychologically and gave me encouragement throughout my life to all my family; To all my friends without exception.

Acknowledgement

First of all, I thank Allah for giving me strength and ability to complete this study. Furthermore, I would like to express my deepest gratitude to my advisors Mr. Noureddine Chikouche for his guidance and suggestions throughout this research. I thank all my friends for sharing their experiences and knowledge. And a special thanks to all my sisters and brothers, without exception, for their financial and moral support and for motivating me throughout my academic career. Finally, I am deeply grateful to my parents for their trust everlasting love, care, and indefectible support morally and materially during all my years of studying.

Abstract

Nowadays the easiest way to connect with people is chatting by using Mobile Chatting Applications, which offers us a lot of helpful facilities. However, different Chat Applications offers different security to users but leads to increase in vulnerabilities & risks of attack on data. To overcome this kind of vulnerabilities and risks of attacks on data we need apply security mechanisms for a secured conversation. This project suggests a novel encryption scheme for instant messaging (IM) communication that uses multiple rounds of encryption to provide better privacy to the transmitted message against unauthorized access. An AES encryption algorithm with CBC (Cipher block chaining) mode is used to encrypt the message while the message is being sent. In this system, user only can decrypt the software-encrypted message in time of accessing the conversation by introduction of the encrypted user's define key. Through this approach of encryption, it can create more powerful security for data transmission.

Key words: Cryptography; Encrypted Messaging; AES-256 Encryption; CBC mod;

Résumé

De nos jours, le moyen le plus simple de se connecter avec les gens est de discuter en utilisant les applications de chat mobile, qui nous offrent de nombreuses fonctionnalités utiles. Cependant, différentes applications de chat offrent une sécurité différente aux utilisateurs, mais entraînent une augmentation des vulnérabilités et des risques d'attaque des données. Pour surmonter ce type de vulnérabilités et de risques d'attaques sur les données, nous devons appliquer des mécanismes de sécurité pour une conversation sécurisée. Ce projet suggère un nouveau schéma de cryptage pour la communication par messagerie instantanée (IM) qui utilise plusieurs cycles de cryptage pour offrir une meilleure confidentialité au message transmis contre tout accès non autorisé. Un algorithme de cryptage AES avec mode CBC (Cipher block chaining) est utilisé pour crypter le message pendant son envoi. Dans ce système, l'utilisateur ne peut déchiffrer le message chiffré par logiciel qu'au moment d'accéder à la conversation en introduisant la clé de définition de l'utilisateur chiffré. Grâce à cette approche de cryptage, il peut créer une sécurité plus puissante pour la transmission de données.

Mots clés : Cryptographie ; Messagerie cryptée ; Cryptage AES-256 ; mode CBC ;

الملخص

أسهل طريقة للتواصل مع الأشخاص في الوقت الحاضر هي الدردشة باستخدام تطبيقات الدردشة عبر الهاتف المحمول، والتي توفر لنا الكثير من المرافق المفيدة. ومع ذلك، توفر تطبيقات الدردشة المختلفة أمانًا مختلفًا للمستخدمين ولكنها تؤدي إلى زيادة نقاط الضعف ومخاطر الهجوم على البيانات. للتغلب على هذا النوع من نقاط الضعف ومخاطر الهجمات على

البيانات، نحتاج إلى تطبيق آليات أمان لإجراء محادثة آمنة. يقترح هذا المشروع نظام تشفير جديدًا لاتصالات المراسلة الفورية (IM) التي تستخدم جولات متعددة من التشفير لتوفير خصوصية أفضل للرسالة المرسلة ضد الوصول غير المصرح به. يتم استخدام خوارزمية تشفير AES مع وضع CBC (تسلسل كتلة التشفير) لتشفير الرسالة أثناء إرسال الرسالة. في هذا النظام، يمكن للمستخدم فقط فك تشفير الرسالة المشفرة بالبرمجيات في وقت الوصول إلى المحادثة عن طريق إدخال مفتاح تعريف المستخدم المشفر. من خلال نهج التشفير هذا، يمكنه إنشاء أمان أكثر قوة لنقل البيانات .
الكلمات الأساسية: التشفير؛ مراسلة مشفرة تشفيرAES-256 ؛ وضعCBC ؛

TABLE OF CONTENTS

Liste des tableaux

Liste des Figures

List of Acronyms

AES: Advanced Encryption Standard.

API: application program interface.

BFA: Brute Force Attack.

CBC: Cipher block chaining.

CFB: Cipher text feedback.

CTSS: Compatible Time-Sharing System

DES : Data Encryption Standard.

3DES : Triple Data Encryption Standard.

DH: Diffie-Hellman Key Exchange.

DSS: Digital Signature Standard.

DUKPT: Derived Unique Key Per Transaction.

E2EE: End-to-end encryption.

ECB: Electronic code book.

ECC: Elliptic Curve Cryptosystem.

ECDH: elliptic curve Diffie–Hellman.

HMAC: hashed message authentication code.

HTTP: Hypertext Transfer Protocol.

HTTPS: HTTP Secure.

ID: Identifier.

IM: instant messaging.

IP: Internet Protocol.

KDF: key derivation function.

MAC : message Authentication code.

MD : message digest.

MIT: Massachusetts Institute of Technology.

MMS: Multimedia Messaging Service.

NIST: National Institute of Standards and Technology.

NSA: National Security Agency.

OS: operating system.

OTR: Out of Record Messaging.

P2PP: Peer-to-Peer Protocol.

PGP: Pretty Good Privacy.

RSA: Rivest–Shamir–Adleman.

SHA: Secure Hash Algorithm.

SMS: Short Message Service.

SSH: Secure Shell.

SSL: Secure Socket Layer.

TLS: Transport Layer Security.

VoIP: Voice over IP.

X3DH: extended triple Diffie-Hellman.

XMPP: Extensible Message and Status Protocol.

XOR: exclusive OR.

GENERAL INTRODUCTION

Today, the Internet and smart phones have become an important part of daily life. It helps us keep in touch with others and manage business projects more easily. Instant Messaging (IM) immediately adapts to the possibilities and human needs of the digital realm. The security of chat application messages is very important, so that the distributed messages are always safe for others who do not have access rights. However, not all chat applications currently have tools for protecting messages. This is still often overlooked, thus providing an easy space for attackers to hack the distributed messages.

New secure mobile applications are easy to use by end-users, where they only need a phone number to create an account, while the applications generate and exchange the cryptographic keys in the background, without the user having to do any interaction. The users would never know about the different cryptographic keys in use, and this is useful for the layman because they get seamlessly end-to-end encryption without having to understand the background on how the keys work.

Statement of the Problem

- The search for ways to protect the security of instant messages is still under study. Currently, most chat applications still lack the ability to protect their messages.
- Nowadays, the methods and techniques used by attackers to invade and eavesdrop on conversations are evolving because most personal information is at risk and threatened by theft.
- In addition to the threat of malicious code, the use of public instant messaging at work also creates many very dangerous risks.

Objectives

- Our main objective is to create a more secure chat application in which private communications and conversations are encrypted by applying a special encryption system. Prevent unwelcome people from eavesdropping on news.
- We also aim to develop this application and customize it as a company or institution that has its own secure chat application.

Organization of the manuscript

This manuscript divided in three chapters:

We will discuss some important concepts in message encryption in the first chapter.

These concepts are expressed in encryption and encryption algorithms, such as symmetric and asymmetric encryption, as well as the security requirements of messages and some basic encryption protocols.

In Chapter 2, we will discuss some concepts and basics of different instant messaging applications, as well as the methods and techniques used to make these applications more secure.

In Chapter 3, we will introduce our work to secure chat applications, explain the methods and techniques used to protect messages, and analyze and evaluate their performance and security.

Finally, the conclusion assesses the work carried out and its prospects.

CHAPITRE 1 :
CRYPTOGRAPHY

1.1. Introduction :

Cryptography has an exciting history, and many changes have taken place in the past few centuries. Throughout the age of civilization, it seems important to keep secrets for one reason or another. Keeping secrets can enable individuals or groups to conceal their true intentions, gain a competitive advantage and reduce vulnerability. The changes that cryptography has undergone throughout history have closely followed technological developments. The cryptographic method starts when someone carves a message into wood or stone, and then passes it to the target individual who has the coercive means to decrypt the message. This can be an extension of the encryption method used today. Now, the cryptography that will not be engraved into the material is inserted into the code stream transmitted through network lines, Internet communication paths and radio waves.

In today's era, we cannot do without the Internet communication system. There are various messaging applications on the Internet. Such as WhatsApp, Telegram, messenger, SMS transceiver etc.

The chat application is one of the most commonly used applications for users to distribute messages today. The distributed messages are usually secret messages, or they may not. Currently, not all chat applications have message security tools, so other people without access rights will often attack and abuse messages. The use of message security technology in very chat applications is essential to ensure access rights and the originality of messages. Encryption technology is one of the commonly used technologies to protect data or message security by encoding the initial message into other characters, and other characters will hide the style and meaning of the initial message. The algorithm key supporting cryptographic technology consists of two parts: symmetric cryptography and asymmetric cryptography.

In this chapter, we will discuss some important concepts in message encryption. These concepts are expressed in terms of encryption and encryption algorithms, such as symmetric and asymmetric encryption, as well as the security requirements for messages and some basic encryption protocols.

1.2. Concepts

1.2.1. Cryptography

Cryptology is the study of various mathematical methods and techniques related to information security to ensure the confidentiality and reliability of messages. It allows sensitive information to be stored or transmitted over insecure networks. (Such as the Internet) to ensure that it is only read by an authorized recipient. It includes converting understandable messages

(texts, pictures, numbers) into encrypted messages, which are messages that are incomprehensible to everyone except for holders of decoding code. [1]

1.2.2. Cryptography Algorithms

Nowadays computer technology, encryption technology is usually related to scrambling plain text into cipher text. This process is called encryption and then reverts to plain text. This process is called decryption.

There are many algorithms used to perform encryption and decryption. The most successful algorithms use keys. The key is just a parameter of the algorithm, which allows the encryption and decryption process. The modern field of key-based cryptographic algorithms can be divided into two categories, such as symmetric key cryptography and asymmetric cryptography or public key cryptography. Symmetric key encryption refers to an encryption method in which the sender and receiver share the same key. Public key encryption is an encryption that uses a pair of keys to encrypt and decrypt a message to make it arrive safely. Another encryption algorithm is the cryptographic hash function, which uses mathematical transformations to irreversibly "encrypt" information. [2]

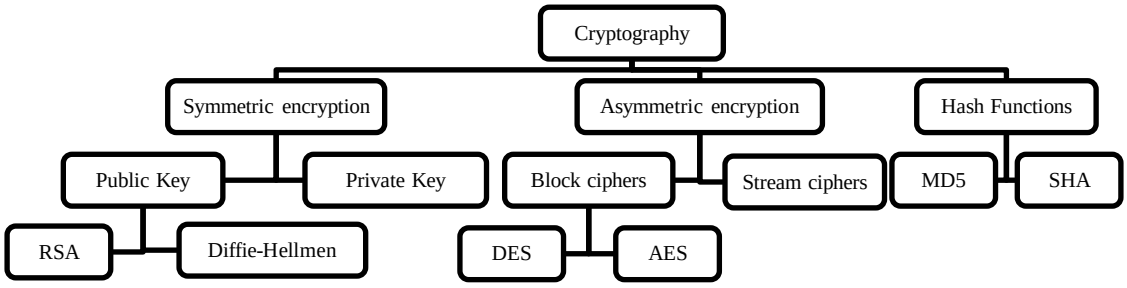


Figure 1.1: Cryptography Algorithms

1.3. Security requirements

In modern cryptography, there are four core problems or pillars to solve: confidentiality, integrity, authentication, and non-repudiation. These four frameworks that are commonly applied in network services are described as follows:

1.3.1. Confidentiality

It is a service that retains the content of all information, but does not include those who have the right to view or understand the information. There are many ways to provide confidentiality, ranging from physical protection to mathematical algorithms that make data difficult to understand. [3]

1.3.2. Integrity

Data integrity is related to maintaining and proving the accuracy and consistency of the data sent between the two parties. This means that if someone sends data to a third party, that person should be able to detect whether the data has been corrupted or tampered with in any way. Data operations include operations such as insertion, deletion, and replacement. some different cryptographic primitives presented in that can be used to help achieve data integrity, including hashing algorithms such as MD5 and secure hashing algorithms such as SHA-1, SHA-256, and SHA-512. [4]

1.3.3. Authentication

Identity verification is about determining the identity of the person or system sending the message. This service applies to sender and receiver entities. For the sake of clarity, the two parties that initiate a secure communication should first identify each other. A good example is the use of SSL certificates on the web server to prove the identity of the server you are connecting to. The identity is verified through the use of an encryption key. Less secure keys mean less trust between the two parties. [3], [4]

1.3.4. Non-repudiation

Consists of preventing communicators from subsequently denying their actions: the sender denies having sent a message and the receiver denies having received a message. [1]

1.4. Ciphering Techniques

1.4.1. Symmetric Encryption

The modern cryptographic algorithms discussed so far are all examples of symmetric algorithms, which means that you use the same key for encryption and decryption. Current symmetric algorithms (such as DES, Triple DES, and AES) are very effective in encrypting large amounts of data, but the main disadvantage of these algorithms is that it is difficult to share keys between multiple parties. [4]

By using a symmetric encryption algorithm, the data will be transformed into a form that anyone who does not possess the key can understand. Once the intended recipient with the key receives the message, the algorithm reverses its operation so that the message returns to its original and easy-to-understand form.

There are two types of symmetric encryption algorithms:

1. **Block algorithms.** Use a specific key to encrypt the set bit length in the electronic data block. As the data is being encrypted, the system holds the data in its memory as it waits for complete blocks.

2. **Stream algorithms**. Data is encrypted as it streams instead of being retained in the system's memory.

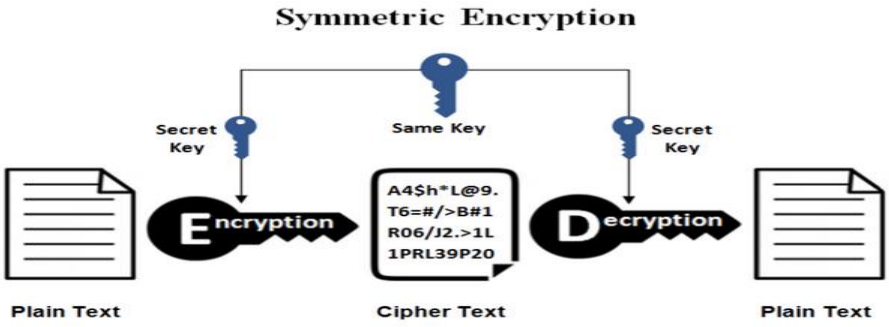


Figure 1.2: Symmetric-key Cryptography. [3]

1.4.2. Asymmetric Encryption:

In the case of symmetric encryption, this method can no longer ensure the required security, because this key also makes decryption possible. On the other hand, there is no requirement that the encryption key also allows decryption. Therefore, one can imagine an "asymmetric" system in which the two keys are different, and the issuance of the encryption key does not compromise security [1].

In a seminal paper in 1976, Whitfield Diffie and Martin Hellman proposed the concept of public key cryptography, in which two different but mathematically related keys were used. Public and private keys [4].

The public key is announced to all members, while the private key is only known to the recipient. The sender uses the public key to encrypt, and the receiver uses the private key to decrypt the message [5]. Instead, both keys are generated secretly, as an interrelated pair. Moreover, it is virtually impossible to deduce the private key if you know the public key [2].

Each key type can be used to encrypt and decrypt, so do not get confused and think the public key is only for encryption and the private key is only for decryption. They both have the capability to encrypt and decrypt data.

Asymmetric cryptosystems run much slower than symmetric systems, but can provide confidentiality, authentication and non-repudiation based on their configuration and use. Compared with symmetric systems, asymmetric systems also provide easier and more manageable key distribution, and there is no scalability issue of symmetric systems.[3]

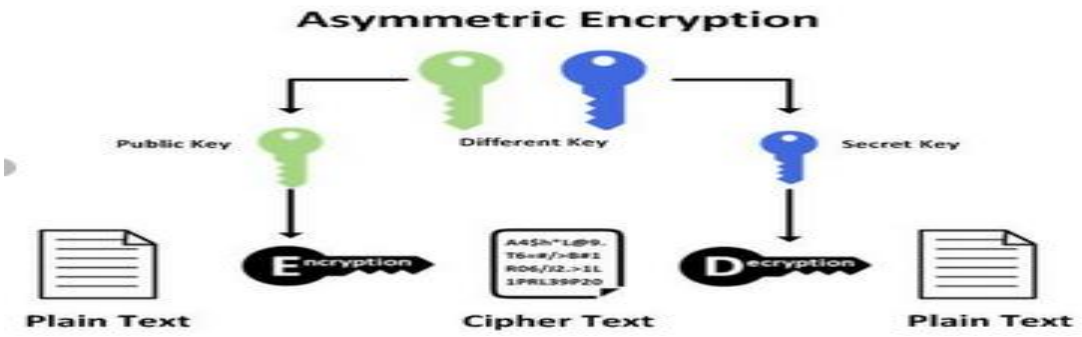


Figure 1.3: Asymmetric-key Cryptography. [3]

Attributes	Symmetric Key Encryption	Asymmetric Key Encryption
Symmetric Key vs Asymmetric key	Only one key (symmetric key) is used, and the same key is used to encrypt and decrypt the message.	Two different cryptographic keys (asymmetric keys), called the public and the private keys, are used for encryption and decryption.
Complexity and Speed of Execution	It's a simple technique, and because of this, the encryption process can be carried out quickly.	It's a much more complicated process than symmetric key encryption, and the process is slower.
Length of Keys	The length of the keys used is typically 128 or 256 bits, based on the security requirement.	The length of the keys is much larger, e.g., the recommended RSA key size is 2048 bits or higher.
Usage	It's mostly used when large chunks of data need to be transferred.	It's used in smaller transactions, primarily to authenticate and establish a secure communication channel prior to the actual data transfer.
Security	The secret key is shared. Consequently, the risk of compromise is higher.	The private key is not shared, and the overall process is more secure as compared to symmetric encryption.
Examples of Algorithms	Example include RC4, AES, DES, 3DES, etc.	Example include RSA, Diffie-Hellman, ECC, etc.

Table 1.1: Symmetric Encryption vs Asymmetric Encryption.

1.4.3. Hybrid Encryption

Hybrid cryptosystems use the advantages of asymmetric and symmetric cryptography. Basically, all data to be transmitted is encrypted using symmetric key (AES), and the symmetric key itself is encrypted using asymmetric (RSA) encryption. Both the encrypted data and the encrypted key are passed to the receiver.

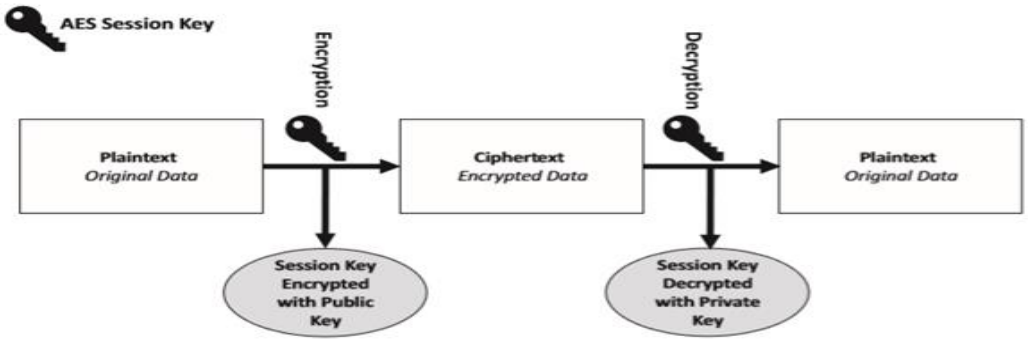


Figure 1.4: Using RSA to protect our AES session key [5].

The combination of encryption methods has various advantages. One is to establish a connection channel between the devices of two users. Then, users can use this hybrid encryption technology to communicate. One disadvantage of asymmetric encryption is that it can slow down the encryption process, but when used with symmetric encryption, both of them can play their respective advantages, the efficiency of symmetric encryption and the split encryption of asymmetric keys. As a result, security is added to the message sending process and overall system performance is improved. [4]

1.5. Hash function:

Hash functions are just simple functions, they accept a certain length of input and compress it into a shorter fixed-length output. Any changes to the original input data will cause the hash code to change. The value returned by a hash function is called a hash value, hash code, digest or hash. These values are usually used to index a fixed-size table called a hash table. [4], [6]

The hash function provides a simple and widely used way to implement password storage protection. The purpose of this is to store the password in a "disguised" form in the password file so that it can be checked, but anyone with access to the password file (including system administrators) cannot recover the password by themselves. [7]

Generating a hash code for a piece of data is a one-way operation, which means that once the hash code of a piece of data is calculated, the hash code cannot be restored back to the original data. There is no reverse process to return the hash code to the original data.

The properties of hashing (for example, hashing can only be done in one direction, and the hash code is unique for a piece of data) make hashing an ideal mechanism for checking data

integrity. Integrity checking means that when you send data to others over the network, you can use hashing to determine whether the original data has been tampered with or damaged. The two most common hashing methods are MD5 and SHA series hashes (SHA-1, SHA-256 and SHA-512), and .NET all support these hashing methods.

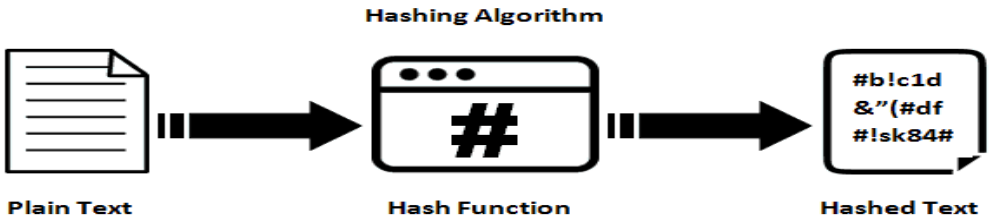


Figure 1.5: Hash function (one-way cryptography).[5]

1.6. Cryptographic Attacks

he basic intention of an attacker is to break a cryptosystem and to find the plaintext from the ciphertext. To obtain the plaintext, the attacker only needs to find out the secret decryption key.

attacks on cryptosystems are categorized as follows:

1.6.1. Brute Force Attack (BFA)

In this method, the attacker tries to determine the key by trying all possible keys. If the length of the key is 8 bits, the number of possible keys is 28=256. The attacker knows the ciphertext and algorithm, and now, he tries all 256 keys one by one to decrypt. If the key is very long, the time to complete the attack will be very long. [8]

1.6.2. Dictionary Attack

There are multiple variants of this attack, all of which involve writing a "dictionary." The easiest way to use this attack is for the attacker to construct a dictionary of the ciphertext and corresponding plaintext he has learned over a period of time. In the future, when the attacker gets the ciphertext, he will refer to the dictionary to find the corresponding plaintext.[8]

1.7. Cryptography protocol

Encryption protocols are widely used in secure application-level data transmission to protect transmitted messages. Encryption protocols usually have at least some of the following functions:

1.7.1. Key Exchange

The key exchange protocol achieves secure communication on untrusted networks by

deriving and distributing shared keys between two or more participants. The key exchange protocol is essential to enable shared key encryption technology to protect the transmitted data in a secure network. [9]

1.7.2. Diffie–Hellman

An agreement that allows two parties to establish shared secrets by exchanging information visible to eavesdroppers. This protocol is now called the Diffie-Hellman (DH) protocol. Before Diffie-Hellman, establishing shared secrets required tedious processes, such as manually exchanging sealed envelopes. Once the communicating party has established a shared secret value through the DH protocol, the secret can be used to establish a secure channel by converting the secret to one or more symmetric keys, and then use the symmetric key to encrypt and authenticate subsequent Communication.[10]

1.7.3. Authenticated Key Exchange

Authentication key exchange (AKE) (or authentication key agreement) is the exchange of session keys in the key exchange protocol, which also authenticates the identities of the parties participating in the key exchange. [11]

1.7.4. X3DH Protocol

The extended triple Diffie-Hellman (X3DH) supplements the standard Diffie-Hellman key agreement protocol to provide mutual authentication of all parties. In addition, X3DH provides forward secrecy and encryption denial. X3DH establishes a shared key between two parties that mutually authenticate each other based on a public key. X3DH is designed for asynchronous setup, where one user ("Bob") is offline, but has posted some information to the server. Another user ("Alice") wants to use this information to send encrypted data to Bob and also establish a shared secret key for future communication. X3DH also allows key exchanges where one party is "offline" and will be exchanged through a third-party server. [12]

1.8. How do we store passwords safely?

Passwords are still the most common method of authenticating users, but it is easy to get yourself into a situation where the system is insecure and vulnerable to attacks. When you create a system that requires the use of passwords to authenticate users, you need to store these passwords somewhere so that the user can log in again in the future. Many techniques can be used to store passwords, but by far the biggest mistake you can make is storing passwords in the database in clear text. This first negates the benefits of using a password, which is to use a password that only the user knows to authenticate and grant access to the system.

1.8.1. Using Hashes to store passwords

It is not a good idea to store passwords in clear text in a database using hash storage. Hashing is a one-way function, which means that once the password is hashed, the hash cannot be restored to the original password. Although using algorithms like MD5, SHA-1 or SHA-2 to hash passwords seems to be a better solution, it does have some disadvantages.

Hash passwords are vulnerable to two different attacks. They are brute force attacks and rainbow table attacks (see Figure 1-6). A brute force attack is when an attacker tries to use different password combinations until he obtains a password that matches the password hash; it will be easier if the attacker has destroyed your hashed password table from the database. They can have an extensive password dictionary that they have prepared, as well as leaked passwords that can be used through that dictionary, until they find a match with the hash in your password table. [4]

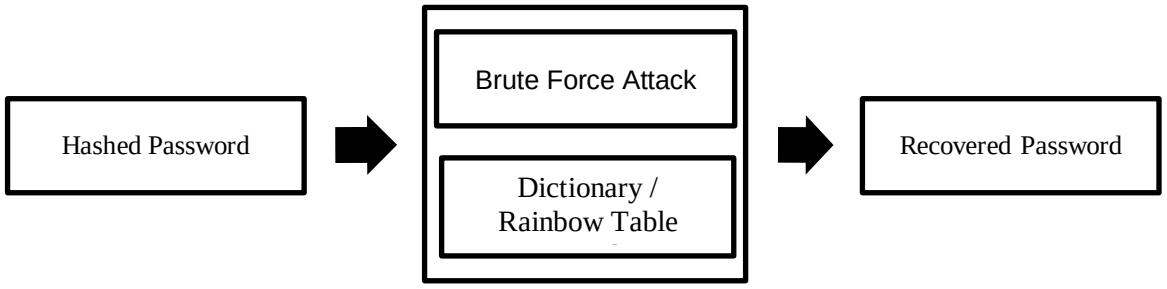


Figure 1.6: Performing attacks against a hash's password.

1.9. Digital signatures

Digital signature is a method to check the integrity of data using asymmetric cryptography. It works similarly to the message authentication code MAC. For the recipient of a message, a digital signature enables the recipient to believe that the correct sender has sent the message, which can be considered equivalent to a letter signature. In addition to digital signatures, it is more difficult to forge. Digital signatures provide you with authentication and non-repudiation. Perform identity verification, because the signature must be created by the user with a valid private key, and it is undeniable, because the receiver can trust the known sender to sign the message only if it knows the private key. The signature process uses a hash function to generate a fixed-size hash digest. The signature function uses the private key to encrypt the hash value into a signature, and anyone can use the sender's public key to verify the signature.[3]

For example, Bob is sending a message to Alice, and the message has been signed with a digital signature. First, Bob encrypts some data to be sent to Alice. After encrypting this data, Bob will SHA-256 hash the data and sign the information with his own dedicated signature key;

this will create a digital signature. Then, Bob sends the encrypted data and signature to Alice. After Alice receives the encrypted data and digital signature, she must first recalculate the hash value of the encrypted data. Alice uses the calculated hash value and Bob's public signature key to verify the digital signature, which tells Alice whether the signature is valid. If it works, Alice can be sure that Bob will send her a message, because the message can only be signed with Bob's private signature key that Bob knows. If the signature is invalid, Alice should not trust the source and authenticity of the message, but should discard it completely.

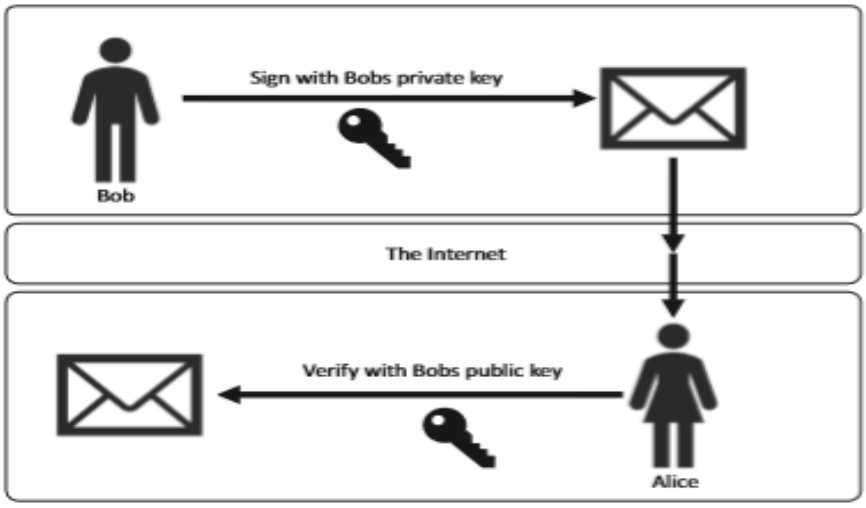


Figure 1.7: The digital signing process from sender to receiver. [4]

1.10. Message cryptography

Today, chat applications are still one of the most commonly used communication applications, but not all chat applications have message security tools to protect messages from unauthorized actions by others. Utilizing encryption algorithms is a solution that can be used to minimize the abuse of messages distributed through chat applications. The security of the chat application certainly makes it easy for users to distribute messages. By adding encryption and decryption process tools to the chat application, the AES algorithm used to protect the message security in the chat application can be implemented. This additional tool will be enabled when the user performs the process of sending and receiving messages.[13]

1.10.1. AES:

The Advanced Encryption Standard (AES) is that the foremost recent encryption standard adopted by the National Institute of Standards and Technology (NIST) in 2001 for the symmetric encryption of messages. The AES algorithm was selected as part of a contest to hunt out a re-placement for DES. The AES algorithm relies on the Rijndael cipher that was developed by two Belgian cryptographers, Joan Daemen and Vincent Rijmen. [4]

1.10.1.1. How AES works?

AES is a variant of the Rijndael cipher that uses a fixed-block size of 128 bits and a variable sized key of 128, 192, or 256 bits. AES is based on a design principle that is known as a substitution-permutation network; this is a combination of both substitution and permutation and is fast in both software and hardware (see Figure 1-8).

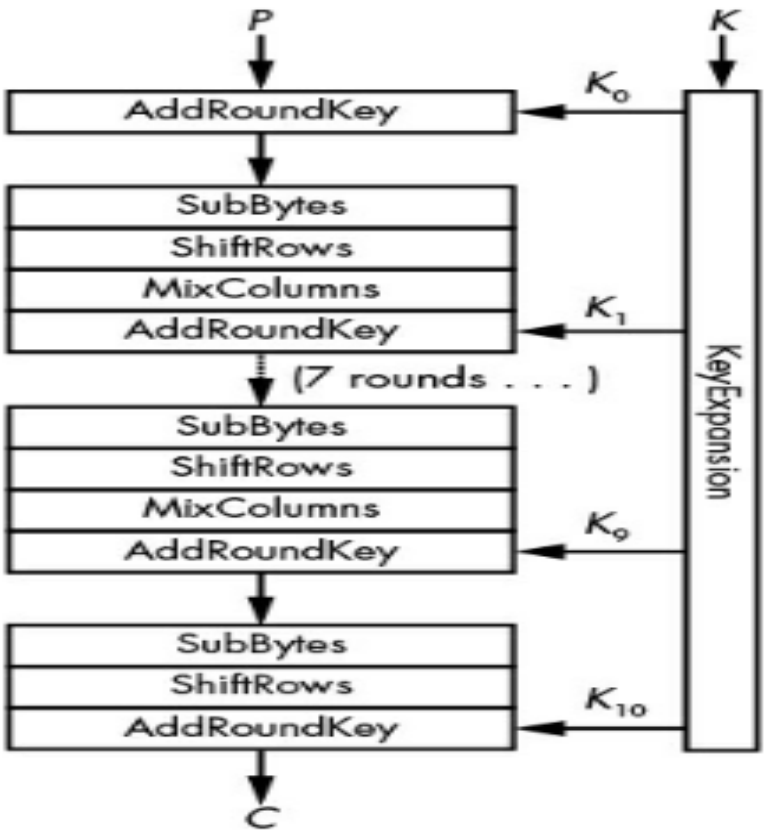


Figure 1.8: The internal structure of AES.[5]

Some ciphers can only be used for single-bit or 64-bit words, while AES is used for processing bytes. It treats 16-byte plain text as a two-dimensional array of bytes (s = s0, s1, ..., s15), as shown in Figure 1-9. (The letter s is used because this array is called the internal state, or state for short.) AES converts the bytes, columns, and rows of this array to produce the final value, the ciphertext.[14]

AES works by repeating the same definition steps (called rounds) multiple times. Each round of encryption includes several processing steps, including one processing step using an encryption/decryption subkey generated from a shared key. Each key is expanded so that a separate subkey can be used for each round. The number of AES rounds usually depends on the length of the key. There are 10 rounds for a 128-bit key, 12 rounds for a 192-bit key, and 14 rounds for a 256-bit key. [4], [14]

Example for AES 256-bit, the encryption process of the key size starts from keyExpension.

After IntialRound is executed, there is only one stage named AddRoundKey. Then perform thirteen repetitions of the regular round consisting of four different stages, named SubBytes, ShiftRows, MixCoulmns, and AddRoundKey. Round 14 is the final round, just like the regular round, but without the MixColumns stage. Only has SubBytes, ShiftRows, AddRoundKey.

AES 256 bits Decryption process is same as encryption. But all process will be inversed and SubBytes, ShiftRows and MixCoulmns function will be replaced with InvSubBytes, InvShiftRows and InvMixCoulmns. [15]

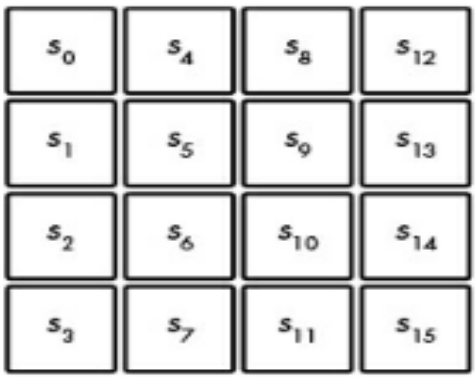


Figure 1.9: The internal state of AES viewed as a 4 × 4 array of 16 bytes.

1.10.1.2. Encryption Mode

In cryptography, the block cipher mode of operation is an algorithm that uses block ciphers to provide information security (such as confidentiality or authenticity). Block ciphers use the same algorithm for each block processed. Therefore, when using the same key and algorithm for encryption, the plain text block always returns the same ciphertext.

1.10.1.2.1. Electronic code book (ECB):

The simplest encryption mode is the Electronic Code Book (ECB) mode. The message is divided into multiple blocks, and each block is encrypted separately. The disadvantage of the electronic codebook mode is that the same plaintext block is encrypted into the same block, and the data mode cannot be well hidden.

1.10.1.2.2. Cipher block chaining (CBC):

The mode introduces feedback into the encryption process. Before encrypting each plaintext block, perform a bitwise XOR operation with the ciphertext of the previous block. This ensures that even if the plaintext contains many identical blocks, they are each encrypted into different ciphertext blocks. Before encrypting the block, the initialization vector (IV) is combined with the first plaintext block through a bitwise XOR operation. [4]

1.10.1.2.3. Ciphertext feedback (CFB):

Another way to provide message dependency is to use the cryptographic feedback (CFB)

mode. It has roughly similar properties to the CBC mode, but slightly different in the way of operation [8]. Sometimes it is desirable to encrypt and transmit some plain text values one at a time. Like cipher block chaining, cipher feedback also uses initialization vectors (IV). In the ciphertext feedback mode, the previous ciphertext block is encrypted, and the output is XORed with the current plain text block to create the current ciphertext block. The XOR operation hides the plain text mode. Unless the block can be retrieved from the beginning or end of the ciphertext, the plain text cannot be processed directly. [2], [4]

1.10.1.3. Advantage and Disadvantage of Encryption mode

Mode	Advantage	Disadvantage
Electronic CodeBook (ECB)	• Simple • Fast • Support for parallel (encryption/decryption)	• Duplicate data in plaintext will be reflected in the ciphertext • The plaintext can be operated by deleting or replacing the ciphertext. • If the ciphertext packed is damaged, it will affect the plaintext. • Can't resist replay attacks. • Should not be used.
Cipher Block chaining (CBC)	• Support for parallel computing (decryption). • Ability to decrypt any ciphertext packet. • Duplicate data in plaintext will not be reflected in the ciphertext.	• Do support for parallel computing (Encryption) • Wrong blocks affect all following blocks.

Cipher feedback (CFB)	• No padding. • Support for parallel computing (decryption). • Ability to decrypt any ciphertext packet. • Can be prepared for encryption and decryption first.	• Do not support for parallel computing (Encryption). • Can't resist replay attacks. • Wrong blocks affect all following blocks.

Table 1.4: Advantage and Disadvantage of Encryption Mode

1.10.1.4. Advantage and Disadvantage of AES

1. Advantage of AES:

- Can be implemented in hardware and software.
- Longer key sizes (128, 192, and 256 bits) enable AES to more reliably prevent hackers from intruding.
- Three keys provide users with choices based on speed and safety.
- So far, no known cryptographic analysis attacks against AES have been discovered.

2. Disadvantage of AES:

- The key used in AES if not employed properly it can cause a cryptanalytic attack. Therefore, key scheduling should be done carefully.
- Though it can be implemented in both hardware and software, it is complex to implement in software.[16]

1.10.2. RSA

The RSA algorithm is the basis of a cryptographic system. The cryptographic system is a set of cryptographic algorithms used for specific security services or purposes. This algorithm can implement public key encryption and is widely used to protect sensitive data.

Public key cryptography, also known as asymmetric cryptography, uses two different but mathematically linked keys one is a public key and the other is a private key. The public key can be shared with everyone, while the private key must be kept secret.

In RSA encryption, both public and private keys can encrypt messages. The key opposite to the key used to encrypt the message is used to decrypt the message. RSA usually uses three key sizes: 1024 bits, 2048 bits and 4096 bits. According to today's standards, you should use

at least 2048-bit keys, because 1024-bit keys are now considered weak [4].

1.10.2.1. Advantages and disadvantages of RSA Algorithm

1. advantages of RSA

- RSA algorithm is safe and secure for its users through the use of complex mathematics.
- RSA algorithm is hard to crack since it involves factorization of prime numbers which are difficult to factorize.
- Moreover, RSA algorithm uses the public key to encrypt data and the key is known to everyone, therefore, it is easy to share the public key.

2. disadvantages of RSA

- RSA algorithm can be very slow in cases where large data needs to be encrypted by the same computer.
- It requires a third party to verify the reliability of public keys.

1.11. Encryption End to End System

End-to-end encryption (E2EE) is a secure communication method that prevents third parties from accessing data when it is transmitted from one end system or device to the other end. In E2EE, data is encrypted on the sender's system or device, and only the receiver can decrypt it. Anyone in between, whether it is an Internet service provider, an application service provider, or a hacker, cannot read or tamper with it.[17]

Encryption key:

In E2EE encryption, the encryption keys should be known only between the two parties to the communication, and in order to achieve this, the data is encrypted using a series of pre-shared codes, and this is done using Pretty Good Privacy (PGP) or through a secret code that is generated once using Derived Unique Key. Per Transaction, or what is known as (DUKPT), and encryption keys can be exchanged using (DH) Diffie - Hellman key exchange, where we can define them more as follows:

- Pretty Good Privacy (PGP): It is an encryption and decryption program that can protect privacy through encryption and authentication. It is usually used for text and email encryption and decryption.
- Derived Unique Key Per Transaction (DUKPT): This is the method used to manage encryption keys in encryption. In each transmission, a unique key deduced from a fixed value is sent. In this way, if an attacker can reveal the value of the key used in the

encryption process, he will be able to know the data. Only the current data, he will not be able to know the previous data or the old data.

- Diffie-Hellman Key Exchange (DH): is a dedicated method used to exchange encryption keys securely through a public communication channel.

1.12.Summary

In this chapter we begin to discover the secrecy pillar in our quest to use cipher. Our primary focus is looking at symmetric and asymmetric cipher. Symmetric encryption is where you use the same key to encrypt and decrypt data.

Symmetric encryption is fast and efficient due to the nature of its algorithm, but it has a major downside in that sharing the encryption key between the parties is difficult to do securely.

Asymmetric encryption involves the use of an algorithm such as RSA that uses the public key and split private key to encrypt the data. The recipient of the message has two public keys and a private key. The private key can be shared with anyone and the private key that they keep secret. If you want to send a message to the recipient, you get his public key, encrypt your message and send it to him and then use their private keys to retrieve the message.

Encrypting all of your data using an asymmetric encryption system such as RSA can be very inefficient, which means it is a good idea to use a blended approach called hybrid cryptography. A hybrid cryptosystem uses the best of both asymmetric and symmetric cryptography.

We also introduced several secure messaging protocols, which are encrypted end-to-end messaging. The out-of-record protocol only supports a single concurrent session, which means that users need to be connected to the Internet to chat with each other, while Signal provides simultaneous chat sessions. In addition, X3DH also provides confidential routing and rejection encryption. X3DH is designed for asynchronous settings.

We also discussed end-to-end encryption approach which provides a secure IM application design which protects its users with better integrity, confidentiality and privacy. Encryption doesn't allow a replica of the message to be stored anywhere, including on the server.

CHAPITRE 2 :

INSTANT MESSAGING SECURITY

2.1. Introduction

The Internet and other communication media such as telephones have changed the way information is disseminated, processed, and influenced. The Internet makes possible new models of human interaction through instant messaging, Internet forums and social networks. In any technology that will control data access and use, there is no centralized and unified management agency like the government.

The overall Internet usage has grown tremendously. In recent years, Internet technology has made great progress. When people use messages to meet friends around the world and keep in touch with them.

Mobile chat applications are very popular among Internet users and smartphone owners. Hundreds of millions of smartphone owners use chat apps every day. These chat applications provide communication for free, and most of them are installed for free, so they are very attractive to potential customers. These chat applications provide users with different services and built-in functions, but in most cases, they ignore the security aspects of usage and messages.

Where social networking sites and services, such as Facebook, Myspace, Tumblr, Twitter, etc., provide new interaction models. There are dedicated websites and social networks. For example, LinkedIn is used for business contacts, YouTube is used for video sharing, and Flickr and Instagram are used for photo sharing. Instant messaging provides people with hearing impaired or hearing or speech impaired a casual, simple and advanced way of communication. This is an effective form that can give equal opportunities for communication.

In this chapter, we will discuss some concepts and basics about the various instant messaging applications and the methods and techniques used to make these applications more secure.

2.2. History of messaging

Among its many uses and benefits, the Internet has changed and simplified the way people communicate with each other on a global scale. By this time, near-instant communication has had a huge impact on the communication culture. In addition to e-mail, instant messaging also plays an important role in bringing people together. From ICQ to AIM, from Google chat to Facebook chat, Internet users have been able to send messages to each other instantly for many years.

2.2.1. Instant Messaging in the 1970 to 2000s

The term "instant messaging" became popular in the early 1990s, but the concept actually dates back to the mid-1960s. Multi-user operating systems such as the Compatible Time-Sharing System (CTSS) created at the Massachusetts Institute of Technology (MIT) Computing Center in 1961, allow up to 30 users to log in at the same time and send messages to each other. By 1965, the system might be closer

to what we now think of as e-mail. The system has hundreds of registered users from MIT and other New England colleges.

In the 1970s, programmers were engaged in the research of peer-to-peer protocols, allowing universities and research laboratories to establish simple communications between users of the same computer.

In 1996, the Israeli company Mirabilis launched ICQ, a text-based messenger, which was the first messenger to truly enter the broad online user market. ICQ allows multi-user chat, file transfer, searchable user directory, etc. The latest version of ICQ includes Facebook integration, mobile synchronization and further updates.

However, the real turning point occurred in 1997, when AOL launched AIM, which attracted a group of tech-savvy Internet users. When you think of AIM, you may hear the sound of opening and closing doors when your friends appear in your friends list or disappear. Like the previous service, AIM allows users to send messages to each other, and includes user profiles, leave messages and icons to increase engagement. By 2005, AIM had occupied the instant messaging market with 53 million users. Chat rooms, in which multiple people could IM with each other, were another popular AOL feature. The application is shown in Figure 2-1.

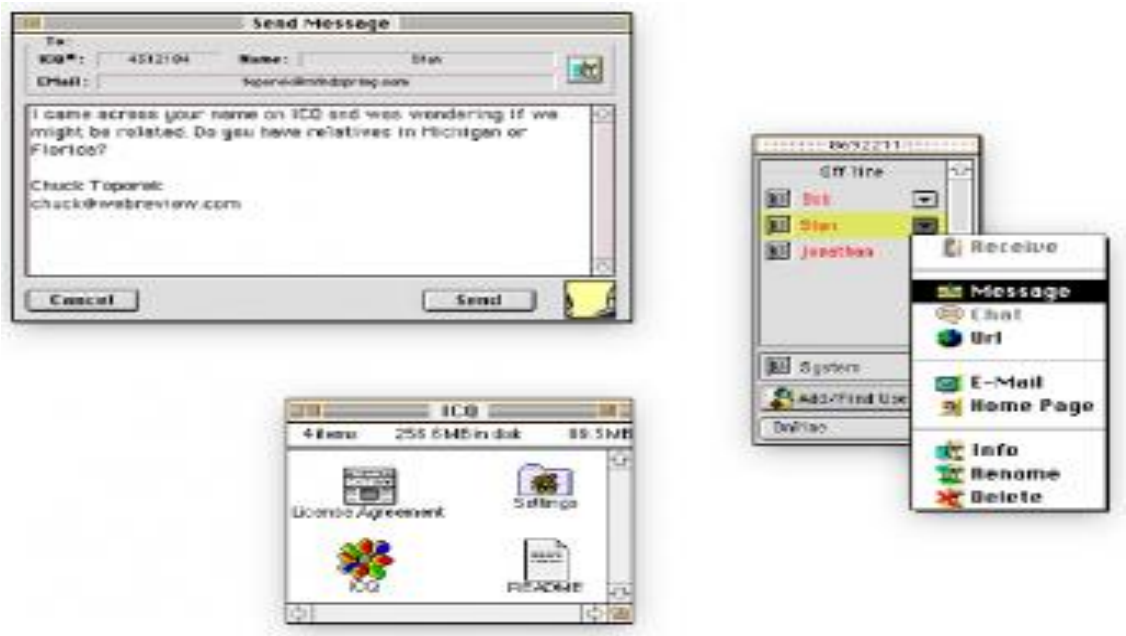


Figure 2.1: One of the very first versions of ICQ.

Yahoo launched Yahoo! Messenger in 1998, originally under the name Yahoo! Pager. Used with a user's Yahoo! ID, Yahoo! Messenger included customized "IMVironments," address book integration and custom status messages. Like AOL, Yahoo had a chat room service.

Microsoft released MSN Messenger in 1999. A press release from its launch read, "MSN Messenger Service tells consumers when their friends, family and colleagues are online and enables them to exchange online messages and email with the more than 40 million users of the MSN Hotmail TM Web-based email service as well as with people using AOL Instant Messenger." Microsoft renamed the service Windows Live Messenger in 2005, adding photo sharing capabilities, social network integration and games. In 2009, the company announced more than 330 million active users every month.[18]

2.2.2. Instant Messaging in the 2000s

In 2000, Internet users noticed Jabber, a multi-protocol instant messaging program that acts as a single gateway for users to chat with friends and simultaneously access friends lists on all large networks (AIM, Yahoo, and MSN). Jabber.org is the original IM service based on the Extensible Message and Status Protocol (XMPP). Recently, in August 2012, Jabber disabled new registrations due to user abuse and denial of service attacks.

Skype was established in 2003 to allow Internet users to communicate with others through video, voice and instant messaging. Although the instant messaging aspect of the service may not be the most popular feature compared to video conferencing, it is used by many people. In July 2011, Skype announced its integration with Facebook, so users can see Facebook friends on Skype and Facebook Chat through these two services.

2.2.3. Social Media Chat

Myspace developed MySpaceIM in 2006 as a supplement to its social platform, which was the first social network to do so. Since 2009, users can instant message with friends on the desktop and online through MySpaceIM for Web. Later, the service was integrated with Skype.

Facebook released Facebook Chat in 2008, allowing users to instantly send messages to a friend or

multiple people through the group function when logging in to a social network. In 2011, Facebook announced the integration of video into the chat integrated with Skype, and also released the mobile application Facebook Messenger.

WhatsApp and Facebook Messenger are the most downloaded apps in global history. Facebook Messenger is integrated with Facebook's chat function, which is part of a series of mobile applications that breaks down the most famous social network. There are 900 million active users in a month. WhatsApp has a database of one billion users, with 315 million daily active users, and was acquired by Facebook Inc. on February 19, 2014. The application is shown in Figure 2-2 and Figure 2-3.

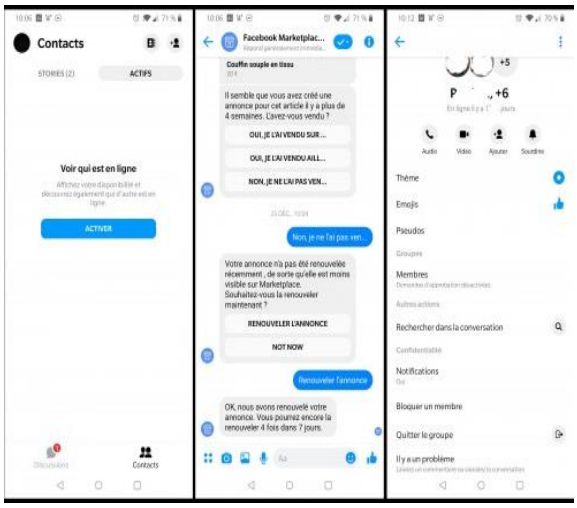


Figure 2.2: Facebook Messenger.

Figure 1.3: WhatsApp.

Google Hangouts is a communication platform launched on May 15, 2013. It is currently the default application for sending and receiving SMS on Android devices. It also replaced the previous Google communication services, Google Talk, Huddle, and Google + video chat. Hangouts provides IM, VoIP and video conferencing, as well as optional SMS and voice calls with other phones. The application is shown in Figure 2-4. [19]

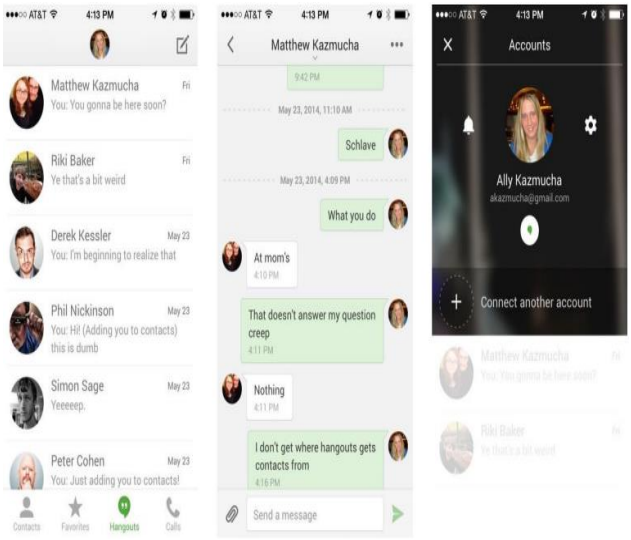


Figure 2.4: Google Hangouts.

2.3. Secure open-source applications

Open Whisper Systems is an open-source software organization whose mission is to make private communications easy (Open Whisper Systems, 2016). The organization has developed a set of end-to-end encryption applications: RedPhone and TextSecure for Android and Signal for iOS. Open Whisper Systems merged the aforementioned applications into an open-source IM software called Signal.

RedPhone is a VoIP encryption application for Android OS. TextSecure is a free and open-source

application for SMS and MMS on Android, with encryption functions between application users.

TextSecure provides a user-friendly service for end-to-end encrypted IM in private chats and group chats. However, when the user receives a message from another TextSecure user and recently entered a passphrase in the application, the notification message will be displayed in plain text. Although it makes the use of the app easier, it also means that the message can be fully visible on the smartphone screen without warning. It is important to set an appropriate time limit because the application can keep the passphrase for how long, because the messenger of the passphrase is not secure.[19]

Signal is Android and iOS IM software with private chat, encrypted calls, secure file exchange and it works over TOR functionality. There is verification of encryption keys to prevent man-in-the-middle-attack: during the call, the application shows on screen two words from PGP word list if words match on both ends of the call, the call is secure. [20]

2.4. The most secure instant messengers

There are a multitude of viable, available options out there when choosing a secure messaging program that emphasizes data privacy. Here is our take on the best secure messaging apps on the market today:

2.4.1. Signal Private Messenger

Signal is a cross-platform encrypted messaging service dedicated to end-to-end encrypted voice calls and encrypted text messages. Encryption has always been the core of the company's products, and when it was acquired by Twitter, its purpose was to further improve its privacy protection platform. To improve security, users can choose a different disappearing message interval for each conversation they save in the chat history. Messages sent through the Signal app can only be viewed by the sender and receiver. Open Whisper Systems, the company behind the app, couldn't even decrypt the message. In addition to instant messaging, you can also make voice calls, group messaging, and encrypted video calls.[21]

2.4.2. Wire

Wire is one of the very few applications that sets end-to-end encryption as the default to protect your messages, images, and files. Wire's encryption works transparently in the background and does not need to be activated because it is always enabled. In addition, each message uses a new encryption key, thereby reducing the impact of a single compromised key.[21], [22]

2.4.3. Telegram

Telegram provides a feature called "secret chat" that allows you to use end-to-end encryption to

protect your messages. After sending and receiving messages, files, photos and videos for a certain period of time, you can choose to self-destruct them. It even has self-destructing messages. For a more secure messaging experience, users can open the secret chat through the application's advanced settings. If you choose to do so, you can force the application on the other side of the secret chat to delete messages.[23]

2.4.4. WhatsApp

When end-to-end encryption is not applied to a specific chat, WhatsApp is the only application that explicitly warns users. WhatsApp is one of the first chat applications to enable and implement end-to-end encryption for more secure communication. WhatsApp does not store messages on its servers, so if cybercriminals invade the platform, they will not be able to decrypt any messages. In addition, WhatsApp does not have a key to view encrypted emails. By default, the way WhatsApp stores messages allows them to be backed up to the cloud via iOS or Android. However, WhatsApp allows users to delete chats from these cloud backups.[24]

2.4.5. iMessage

iMessage has always been the basis for providing end-to-end encryption between users. In addition, this secure messaging application allows users to control how long a message is retained and how many times the recipient can view the message. The main security issue that users complain about is the option to back up iMessage to iCloud. Messages stored in the cloud are encrypted by keys controlled by Apple, so if your iCloud is hacked, these messages may be leaked. However, the solution is to avoid storing private messages on web-based platforms (such as iCloud) to improve security.[25]

2.5. Comparison of messaging applications

	Threema	Telegram	WhatsApp	Snapchat	Viber	Facebook Messenger
End-to-End Encryption	✓	✓	✓	✗	✗	✗
Offline key verification	✓	✓	✓	✗	✗	✗
Encrypted local	✓	✓	✗	Weak	✗	✗

storage						
App-level password lock	✓	✗	✗	✗	✗	✗
Encrypted Group chats	✓	✗	✓	✗	✓	✗
Self-destructed messages	✗	✓	✗	✓	✗	✗
Encryption file transfer	Media	Media	✗	Media	Media	Media
Sends address book to servers	✗	✗	✓	✓	✓	✓
Open source	✗	✓	✗	✗	✗	✗
Server location	DE	CH	RU, CA, UK, SI	Google		
Supported platforms	iOS, Android, Windows phone	iOS, Android	iOS, Android, WP	iOS, Android	iOS, Android	iOS, Android, Windows

Table 2.1: Threema vs Telegram vs WhatsApp vs snapchat vs Viber vs Facebook Messenger.[2][19][26]

2.6. Security Messaging

Today, most social networks provide some form of real-time online messaging, the users are very demanding. IMs are required to have both mobile and web access with more advanced capabilities, such as media, voice, or location sharing, and ability to see if a contact is online and available for a chat. Often preference is given to an open-source project. Internet privacy should include protecting private data from eavesdropping on third parties such as marketers and advertisers, or politics and the military.

The necessity for instant messenger is, therefore, to prevent third parties' access to the communication between users. At the same time, it is important to control that the service provider does not collect sensitive data and information about the individuals who use these messengers. Different aspect underlies in control how many personal data can be gathered from contact lists and conditions to

be added as a new friend. It is vital to stop attempts of using the social network for reconnaissance attacks. Because of the nature of these applications and the more private nature data exchanged between users in terms of the vast amount of data and in terms of inappropriate for publication to the general public as in other types of social networks, hence the topics of security and privacy are important in the evaluation of IM Applications. The Electronic Frontier Foundation EFF maintains a scoreboard of messaging applications' security; it evaluates messengers based on the following criteria:

- Is the message encrypted in transit? User communication must be encrypted.
- Is the message encrypted in such a way cannot be accessed by the provider? End-To-End encrypted, from the moment message sends by the user to the moment it received by the other party. No re-encrypting or decrypting may occur during that process, only the two parties of communication should be able to decrypt messages.
- Can the user verify contacts' identities? A verification mechanism requires to opposite side's identity to prevent Man-in-the-Middle attacks.
- If the keys are stolen, are the past communications secure? The messages must be encrypted with routinely changing keys. This requires end-to-end encryption as in the second criterion.
- Is the cryptography design well documented? The cryptography behind the messaging application and its explaining should be placed on record in the detailed documentation, and can answer the following, which algorithms and parameters (such as key sizes) are used, how keys are generated, and how exchanged between the users and the keys life-cycle and the process for users to revoke or change their key.[27], [28]

2.7. Security technologies in messaging applications

2.7.1. Signal

Signal is a free and open-source messaging app.

Encryption technology: Provides end-to-end encryption for text messages. Additional layer, uses Curve25519, AES-256, and HMAC-SHA256.There is an advanced end-to-end encryption protocol in the application. The app only stores the time users connect to their servers.

2.7.2. Threema

Encryption Technology: End-to-end encryption. Threema encrypts end-to-end encryption for all messages, group conversations, and medial files. This encrypted messaging app leaves little information on the server, and provides maximum protection to the end user. The secure texting app uses NaCl encryption library, open-source encryption technology, and encryption keys stored on user devices to prevent access to the back door. All messages passing through the Threema app are deleted immediately

after they are delivered and the encrypted local files are stored, this makes the maximum security for your connection.

2.7.3. The WhatsApp

Encryption technology: Message key using AES256 in CBC mode and HMAC-SHA256 for encryption and authentication. WhatsApp has provided end-to-end encryption to its users. WhatsApp's end-to-end encryption technology ensures that only the sender and message receiver can read what is being sent, and no one in between, not even WhatsApp. Each message has a unique lock and key. Because of this, messages are only viewable to you. User can check whether calls and conversation are protected end-to-end encrypted or not.

2.7.4. Viber

Encryption technology: A one-time 128-bit symmetric key is generated and used to encrypt the message body with the Salsa20 encryption algorithm. This encrypted messaging app is free and dedicated to user security and privacy. Users of the app are marked with an icon of purple color, which upon click provides the option to make calls.

2.7.5. Telegram

Encryption technology: Symmetric AES 256-bit encryption, RSA 2048 encryption, and Diffie-Hellman secure key exchange. Secret chats use an additional layer of customer-customer encryption
The encryption of the application is fast and efficient. The encryption key changes every week or immediately after 100 messages to redirect confidentiality. If your phone is undergoing the hacking process or the encryption keys are in the wrong hands. Non-existent messages cannot be decrypted.

2.7.6. Wickr

Encryption technology: Application uses AES-256, RSA 4096 (old application), ECDH521, TLS, and SHA-256 to protect data. Wickr meets encryption standards stated by the US National Security Agency (NSA)
The Wickr secure chat app uses standard AES256 encryption to secure users' data. It forensically erases all data on their servers and the user ID is unknown to everyone. Users can set their expiration time on all messaging data.

2.8. Related Work

It is worth noting that the security at the beginning of electronic communication is not important, because most of the information stored in it is different from today, and its sensitivity is not high, because the value of electronic communication is getting higher and higher. The information stored in the electronic computer has a greater attempt to some people, that is, in order to destroy or sell it to an

interested party in order to obtain illegitimate benefits and try to access the information from it. Therefore, the security of this information Sex has become very important. Importance. The first research paper I have seen is based on Dr. Mijanur Rahman's paper, which is the Development of Cryptography-Based Secure Messaging System [2]. This research is related to the development of a secure messaging system based on encryption algorithms. They say it is faster, less attacked, more complex, easier to encrypt, and includes many advanced security features. This project is designed and developed for secure messaging in the Web and Android platforms. But in this article, the encryption and decryption of the message is done by using keyword mono-alphabetic substitution algorithm. This is less secure than the AES cipher scheme and the public key cipher scheme.

Secondly, I read the research paper titled " An Application for End-to-End Secure Messaging Service on Android Supported Device " [15]. This is a paper by Somen Nayak by Surajit Das et al. In this article, an end-to-end encryption framework is developed to operate through the HTTP protocol. The connection protocol is related to the traffic management protocol and is a key component of the framework, which can make the framework safer and more convenient. The connection protocol provides two modules that only allow authorized devices to communicate with the server. The flow management protocol controls the encryption and decryption of messages. The user's message is encrypted using AES-256 encryption.

The third paper is titled "Message Security of Chat Application Based on Messi Omura Algorithm" [13]. This is a paper written by Taronisokhi Zebua, Rivalri Kristianto Hondro, Eferoni Ndruru. This article describes how to use the massey-omura algorithm to protect text messages in chat applications when distributing messages. A secure chat application based on the omura algorithm is effective for protecting messages because the messages will be encrypted using multiple layers of encryption. However, depending on the time of the process, this algorithm is time-consuming because the process must pass three protocols. The key generation process used in the encryption and decryption process is executed by the parties involved in the message distribution based on the concept of a three-pass protocol.

The main disadvantage of all the mentioned articles is that they do not describe the structure of the chat application, nor do they recommend designing proper security in the IM application. Even in research cases where researchers put more energy into chat application structure, they did not propose any solutions or specific designs for a secure chat application system.

2.9. Summary

This chapter introduces mobile IM applications. In order to study the current security features, they provide to users, different security issues and background information have been explained, and different chat applications have been studied.

CHAPITER 3:

PROPOSED APPROACH AND IMPLEMENTATION

3.1. Introduction

The chat application is one of the most commonly used applications for users to distribute messages today. The distributed messages are usually secret messages, or they may not. Currently, not all chat applications have message security tools, so people without access rights often attack and abuse messages.

In this chapter, we will propose a new and more secure chat application to ensure the confidentiality and reliability of instant messages.

3.2. Proposed Application System

The proposed application is an attempt to find useful solutions to the problems of security and privacy in the field of instant messaging. Our secure instant messaging application will implement a range of encryption techniques and models. Where the latter contains the basics of encryption: AES with CBC mode. As shown in the figure 3-1

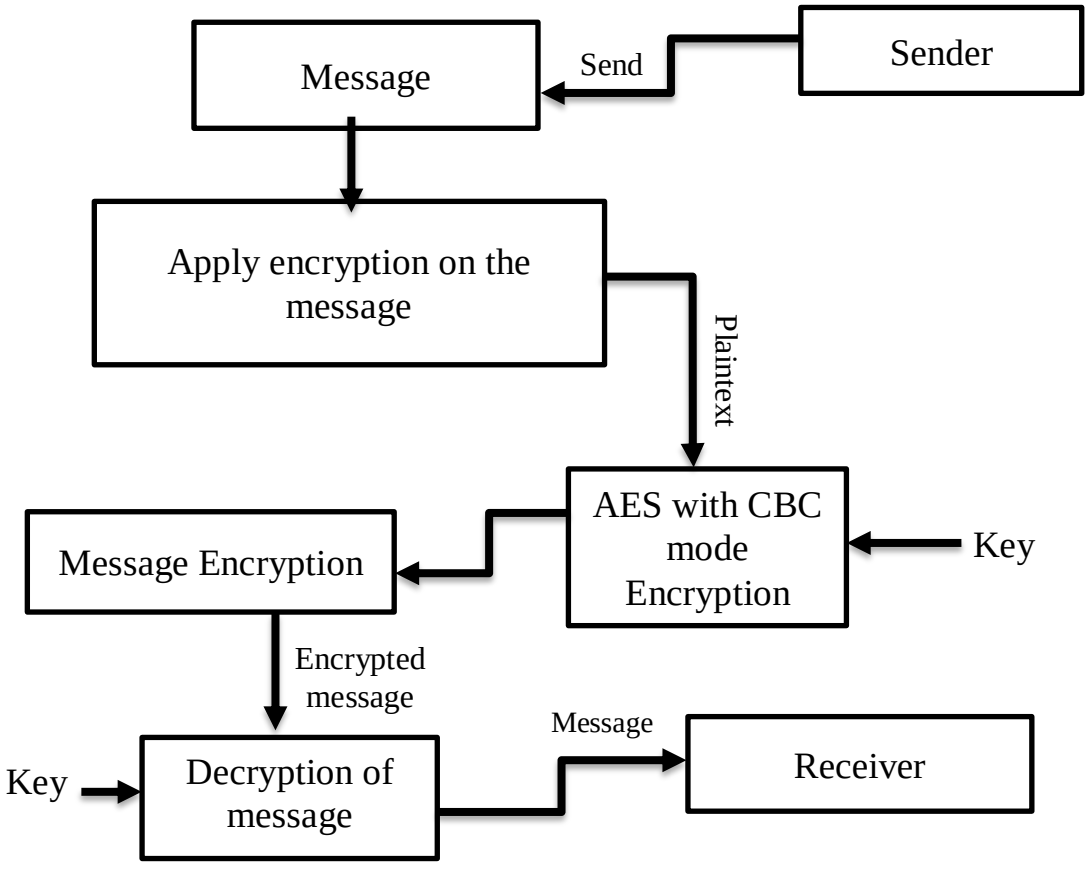


Figure 3.1: General structure of the cryptographic system of proposed application.

3.3. Application Development Methodology

3.3.1. Chat Application Use Case Diagram

This Use Case Diagram is a graphic depiction of the interactions among the elements of Chat Application. It represents the methodology used in system analysis to identify, clarify, and organize system requirements of Chat Application. The main actors of Chat Application in this Use Case Diagram are: Super Admin, Admin, User, Groups, who perform the different type of use cases such as Manage Chat, User Profile, Manage Chat History, Manage Group Chat, Manage Notification, Manage Delete Chat, Manage Users and Full Chat Application Operations. Major Manage elements of the UML use case diagram of Chat Application are shown on the Figure 3.2 below.

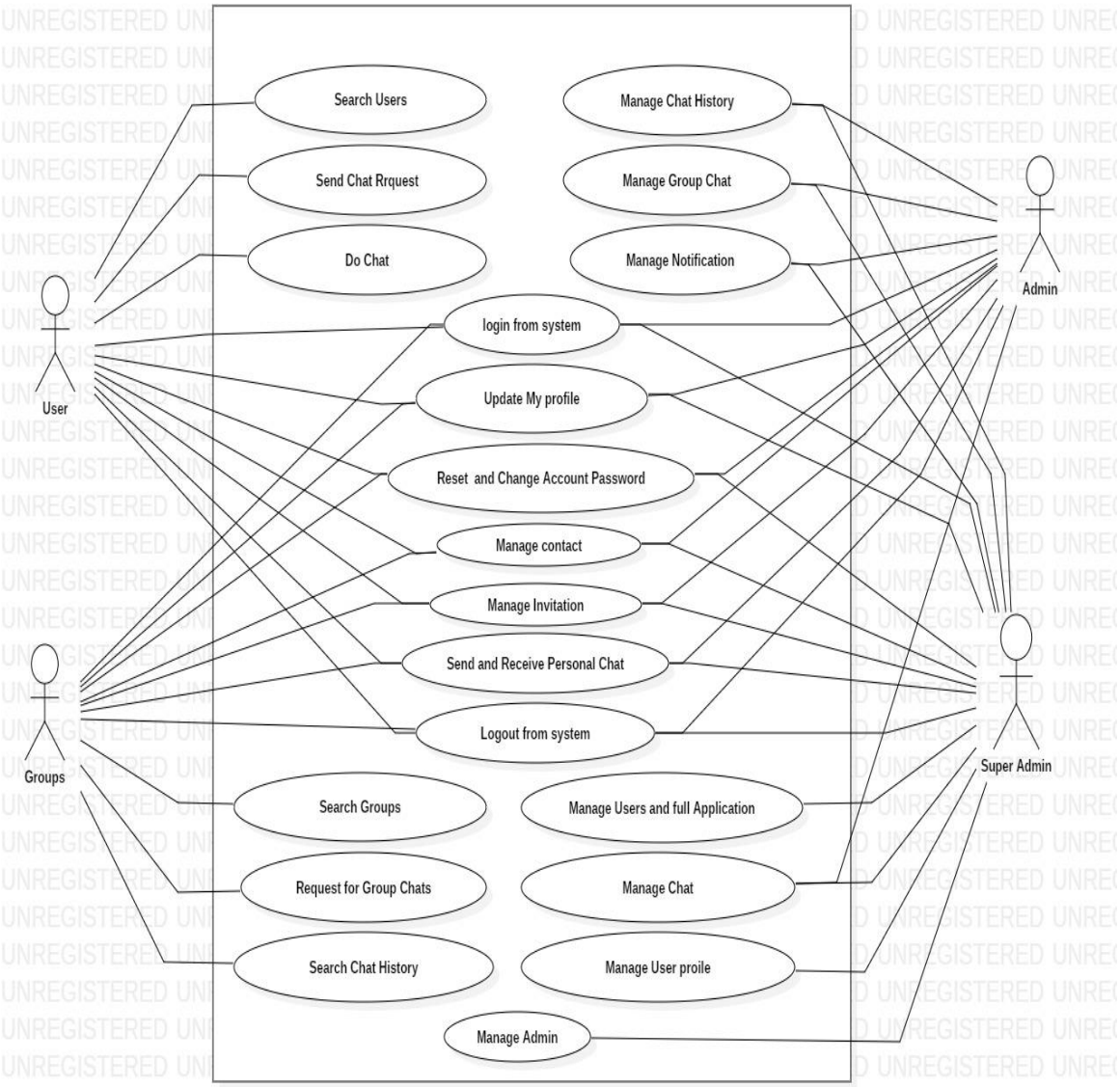


Figure 3.2: Use Case Diagram of Chat Application

1. **The relationships between and among the actors and the use cases of Chat Application:**
   - **Super Admin Entity**: Use cases of Super Admin are Manage Chat, Manage User Profile, Manage Chat History, Manage Group Chat, Manage Notification, Manage Delete Chat, Manage Users and Full Chat Application Operations.
   - **Admin Entity:** Use cases of System User are Manage Chat, Manage User Profile, Manage Chat History, Manage Group Chat, Manage Notification, Manage Delete Chat.
   - **User Entity:** Use cases of User are Search Users, Send Chat Request, Do Chats, Check Chat History.
   - **Groups Entity:** Use cases of Groups are Search Groups, Search Chat History, Request for Group Chats, Create Group Chats, Add Members.

3.3.2. Chat Application System Sequence Diagram

This is the UML sequence diagram of Chat Application System which shows the interaction between the objects of Chat Profile, Chat, Notification, Delete Chat.

1. Login Sequence Diagram of Chat Application System:

This is the Login Sequence Diagram of Chat Application System, where admin will be able to login in their account using their credentials. After login user can manage all the operations on Notification, Chat Profile, Chat, Delete Chat All the pages such as Chat, Delete Chat are secure and user can access these pages after login. The diagram below helps bemonstrate how the login page works in a Chat Application System. The various objects in the Delete Chat, Notification and Chat Profile page interact over the course of the sequence, and user will not be able to access this page without verifying their identity are shown on the Figure 3.3 below.

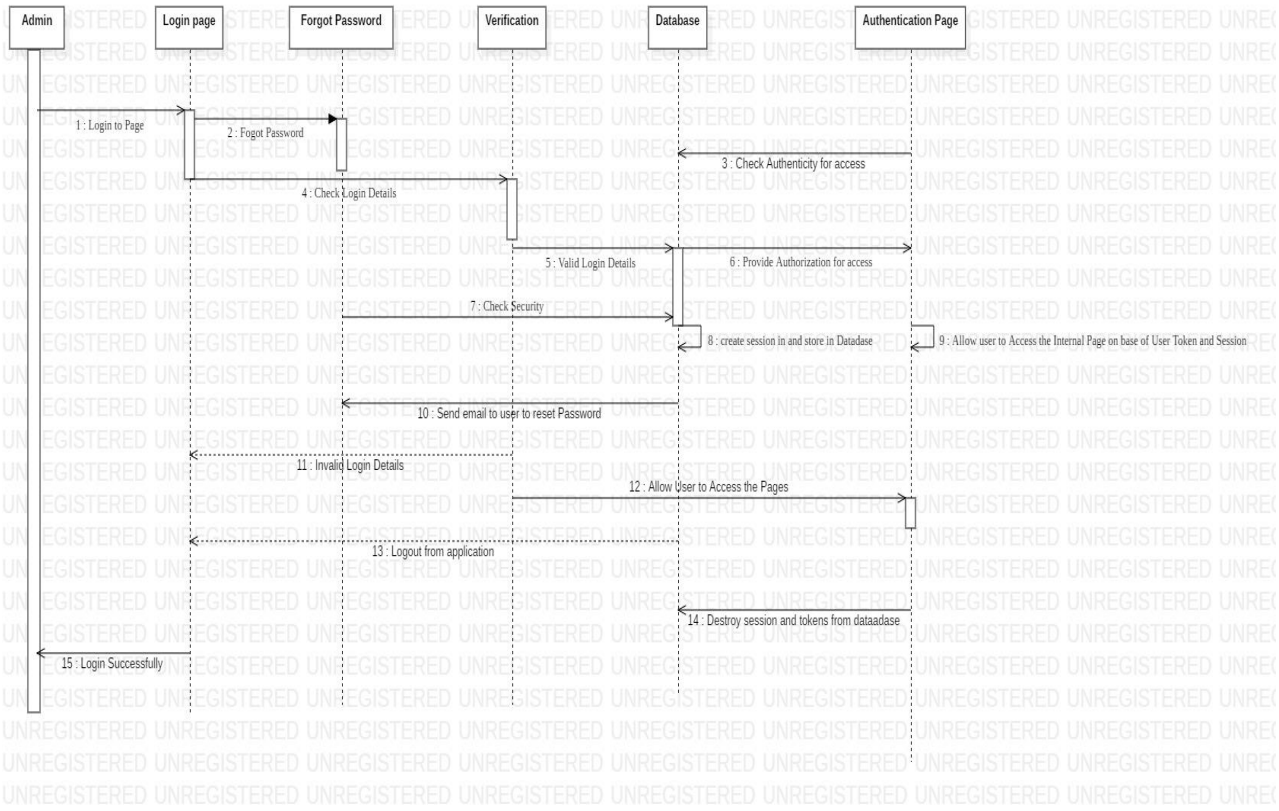


Figure 3.3: Login Sequence Diagram of Chat Application System.

2. Sequence Diagram of Chat Application System

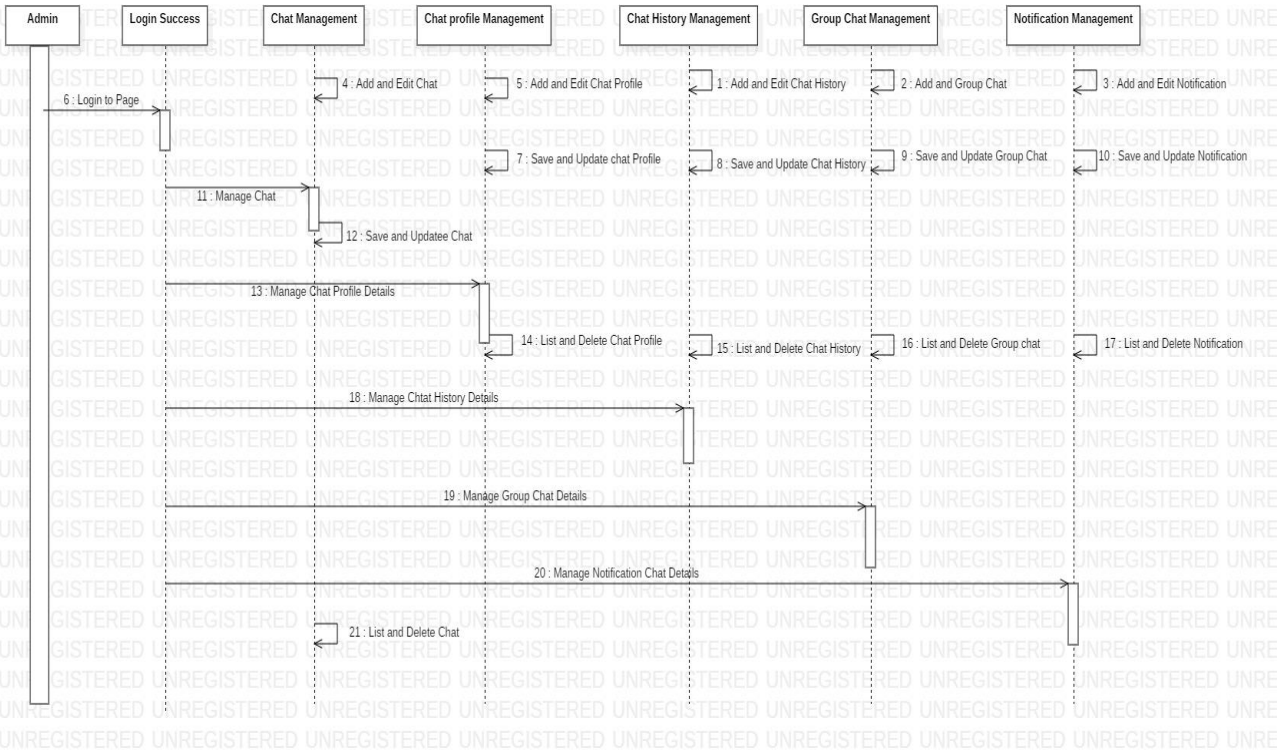


Figure 3.4: Sequence Diagram of Chat Application System.

3. Sequence Diagram of Message

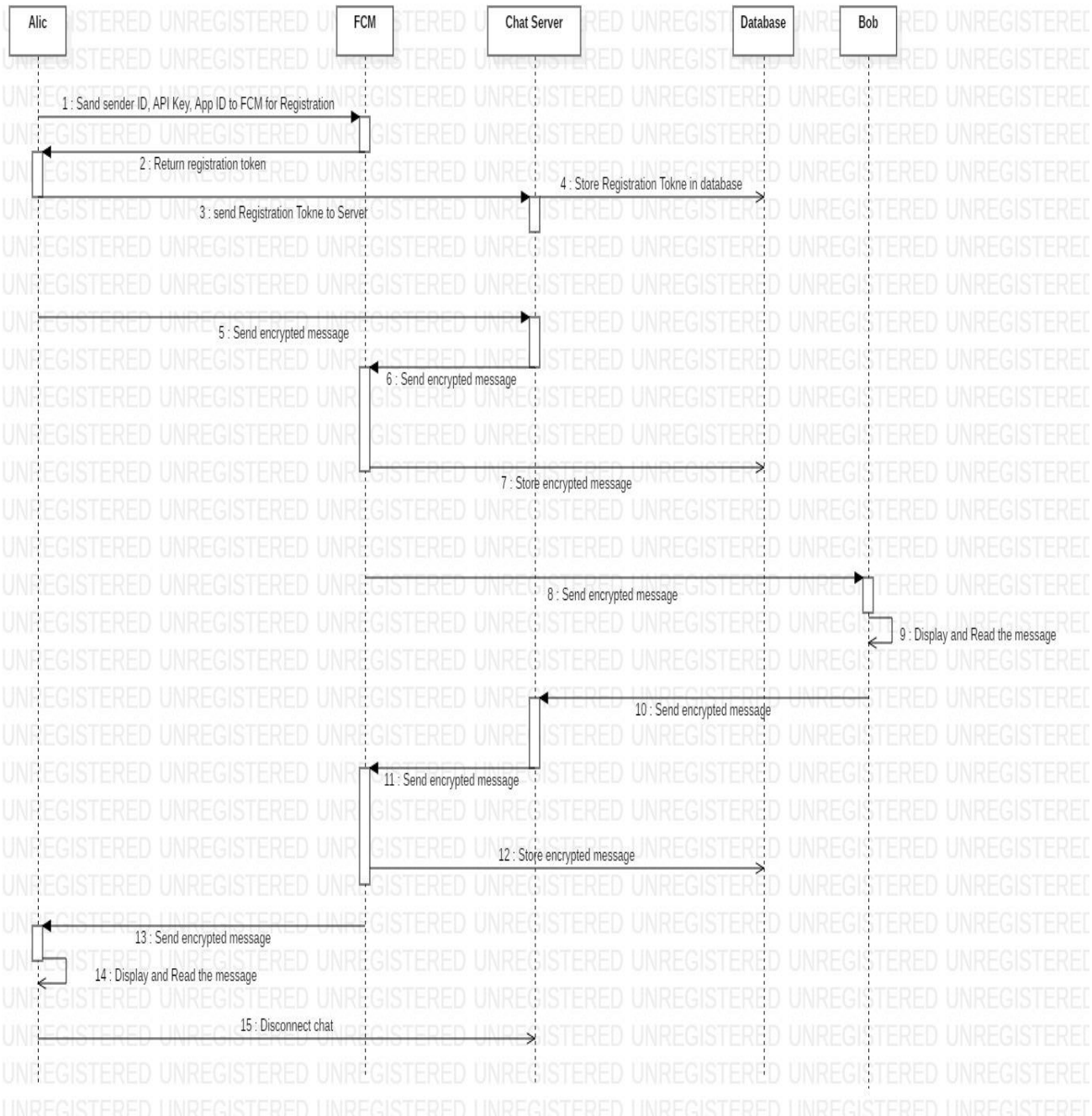


Figure 3.5: Sequence Diagram of Message.

3.4. Proposed Architecture Model

The proposed application attempts to find a useful solution to the problem of the secure chat application. The main architecture of the proposed system is shown in Figure 3.6. The system is a mobile application developed in the Android Studio environment using JAVA and XML languages, MySQL database, and the interface design adopts the Java server programming

platform. All messages exchanged through this proposed instant messaging tool are AES encrypted in CBC mode using encryption and decryption algorithms to accommodate the limitations and diversity of users' mobile devices. Message server handles messages between users by using Firebase Cloud Messaging (FCM). If the recipient is offline, the messages will be stored temporarily on the FCM queue for a specific period of time, and when recipient becomes online these messages are forwarded to him then deleted from the queue. The generic architecture is shown in Figure 3.6.

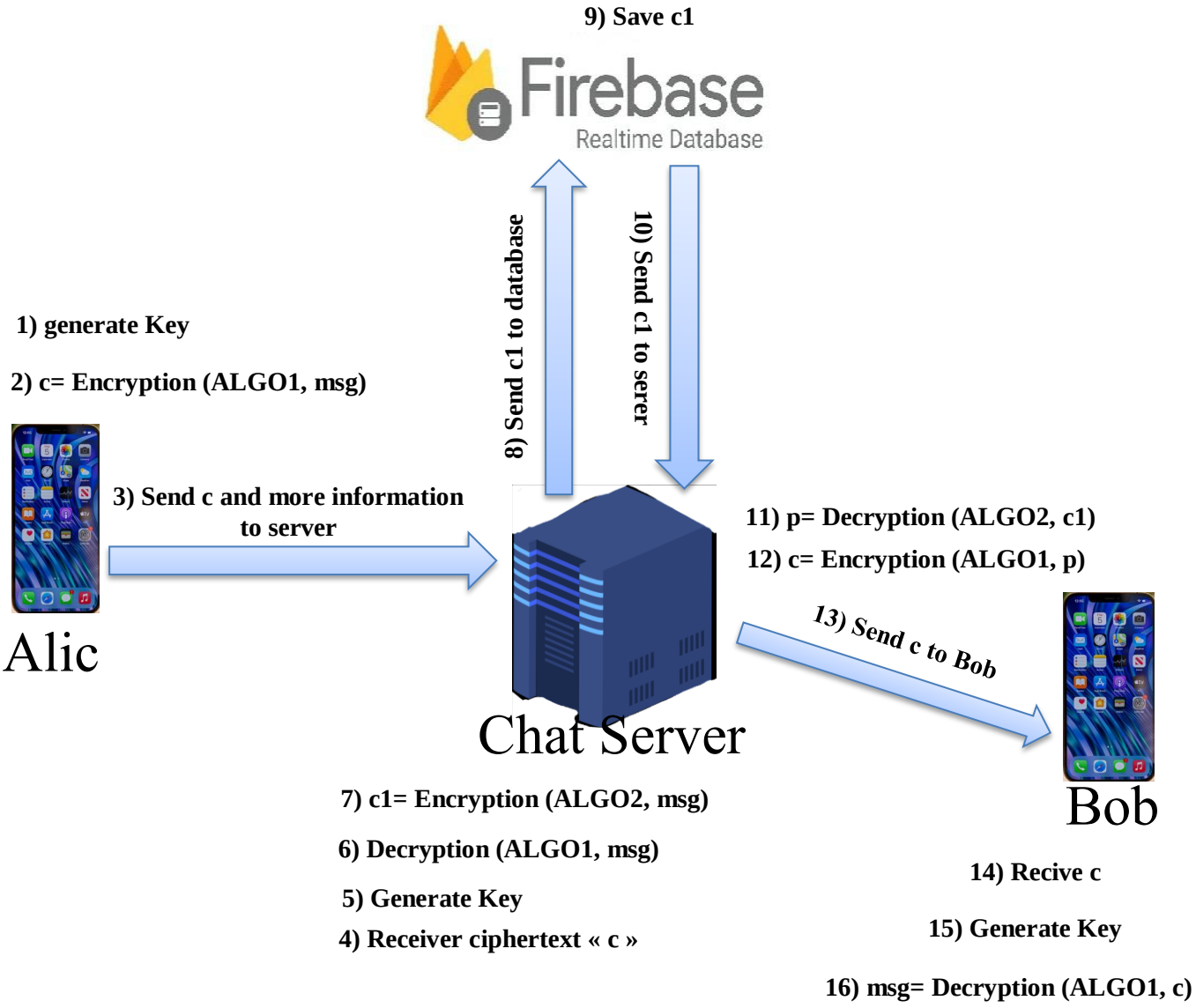


Figure 3.6: Generic Architecture of Proposed Chat application.

Part 1: Basic Functionalities of chat application

The proposed application provides the major functionality required for the IM. The application design needs to build an encrypted database to contain user information, conversations and some important cases. The database is designed to store and process encrypted messages. FCM has also been used to facilitate messaging between mobile applications and server applications. The main tasks are presented as follows:

A. On first launch of the application, it gets the following: The application connects to the FCM server and registers itself. Upon successful registration, FCM provides the registration token for the device. This registration code uniquely identifies each device. The application sends the registration code to the server for storage in the MySQL database.
B. Registration and Login: Once the user opens the application, a set of operations will be performed to check the databases if this phone number is already registered to facilitate user login and open the chat friends interface list, otherwise, a new registration process is performed to obtain the new participant's phone number and send SMS to verify identity authentication.

Part 2: Privacy Techniques

This part is to overview the privacy techniques and features that used to reinforcement the security and privacy aspects in the proposed application accompaniment to the problems of user awareness, these techniques are:

- Session Destruction: An optional choice allows the user to destroy the conversation. This means that all conversation data and messages will be erased from the databases and cannot be retrieved.

Part 3: Cryptographic Techniques

One of the major problems that the proposed IM application faced is how to make the user's private information and exchange data (messages) between them are more secure, so that the encrypted message cannot be accessed. Where the message is encrypted on the sender's device, it can only be decrypted by the receiver. The general structure of the cryptographic techniques of the proposed system is depicted in figure 3.1. The idea of using the AES algorithm with CBC mode is to ensure that messages are protected, encrypted and transmitted securely between sender and receiver. Figure 3.7 also represents the following major events:

A. The sender sends the encrypted message to the chat server
B. The chat server sends the encrypted message to Firebase Cloud Messaging (FCM) and

stores it in the database.

C. Firebase Cloud Messaging (FCM) sends the encrypted message to the receiver

D. The receiver decrypts the message and reads it.

3.5. Implementation

We used Java in the Android Studio editor to implement our application and also used to encrypt AES messages with CBC mode as well as FCM.

- **AES Encryption:** We use AES for database encryption for ease of overall implementation, strong security, fast execution and use of the Block encryption implementation. This robust security algorithm may be implemented in both hardware and software. It is resilient against hacking attempts, thanks to its higher-length key sizes (128, 192, and 256 bits). It is an open-source solution.
- **CBC mode encryption:** We used CBC mode. Since the direct encryption and output of each input plaintext block are in the form of encrypted ciphertext blocks, it is easier. Generally, if the size of the message is larger than b bits, you can break it into a bunch of chunks and repeat the process. Because the XOR process hides the plain text mode. Even if the first plaintext block and the third plaintext block are exactly the same plaintext segment, the first ciphertext block and the third ciphertext block are unlikely to be the same.
- **Firebase Cloud Messaging (FCM):** Firebase Cloud Messaging (FCM) is a service that facilitates messaging between mobile applications and server applications. It's built on Google Play Services that supports cross-platform (iOS, Android & Web). It works on the principle of down streaming messages from FCM servers to user's app and upstream messages from user's apps to FCM servers. Firebase comes with a lot of new features along with the GCM infrastructure. It also helps to deliver the content rapidly anywhere. FCM provides a reliable, low-battery-consumption connection between your server and devices, allowing you to send and receive messages and notifications for free on iOS and Android. At the beginning of running the application for the first time gets the following:

1) The application connects to FCM server and registers itself.
2) When successful registration, FCM provides registration token to the device. This registration token uniquely identifies each device.
3) The application sends the registration token to the server to store it in MySQL

database. The above steps are shown in Figure 3.7.

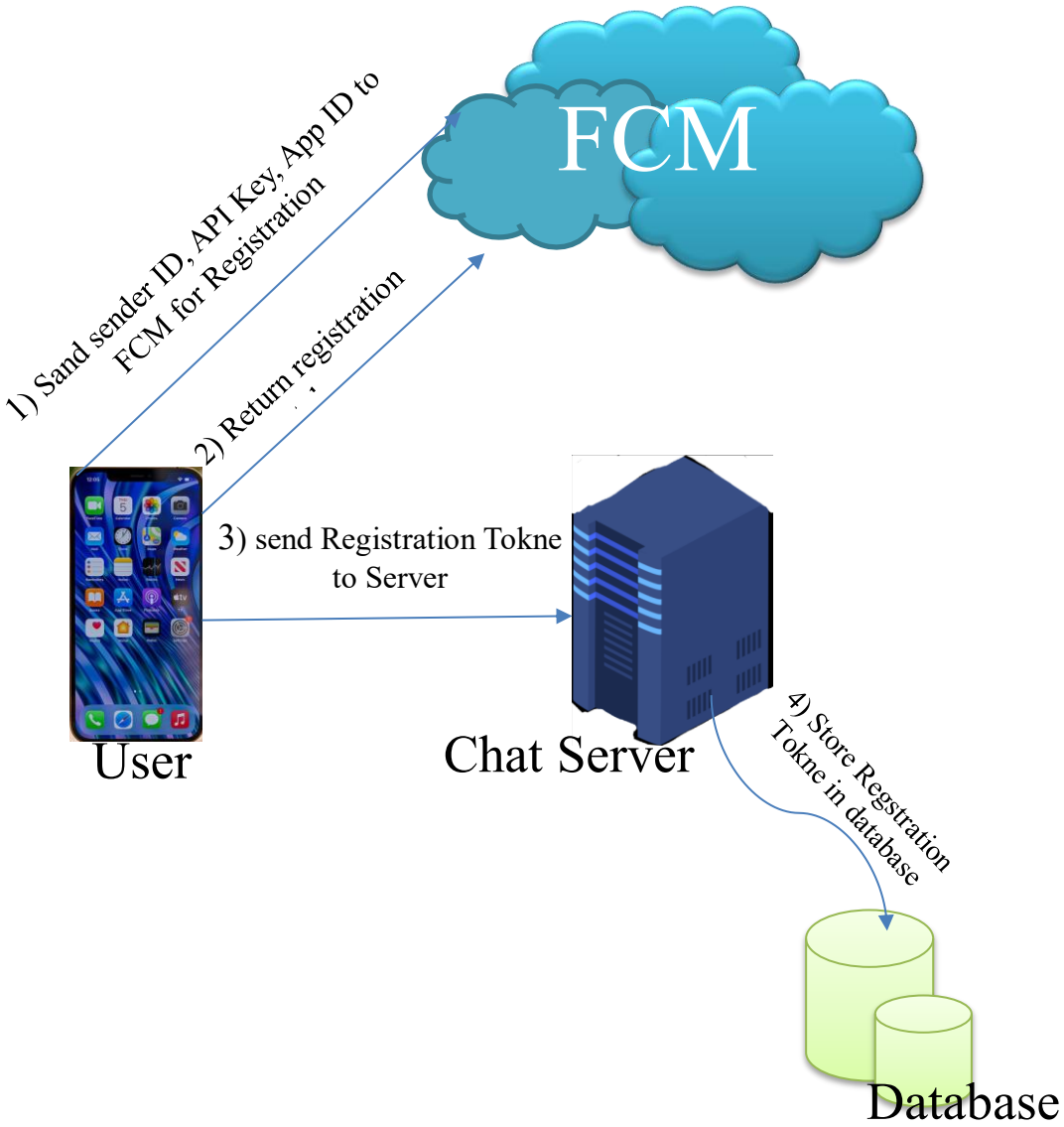


Figure 3.7: Firebase Cloud Messaging

- **Firebase:** Firebase is considered as web application platform. It helps developers‟ builds high-quality apps. It stores the data in JavaScript Object Notation (JSON) format which doesn't use query for inserting, updating, deleting or adding data to it. It is the backend of a system that is used as a database for storing data[29]. The services available are:

  - **Firebase Authentication:**

    Most apps need to know the identity of a user. Knowing a user's identity allows an app to securely save user data in the cloud and provide the same

personalized experience across all of the user's devices.

Firebase Authentication provides backend services, easy-to-use SDKs, and ready-made UI libraries to authenticate users to your app. It supports authentication using passwords, phone numbers, popular federated identity providers like Google, Facebook and Twitter, and more.

Firebase Authentication integrates tightly with other Firebase services, and it leverages industry standards like OAuth 2.0 and OpenID Connect, so it can be easily integrated with your custom backend.

To sign a user into your app, you first get authentication credentials from the user. These credentials can be the user's email address and password, or an OAuth token from a federated identity provider. Then, you pass these credentials to the Firebase Authentication SDK. Our backend services will then verify those credentials and return a response to the client.

After a successful sign in, you can access the user's basic profile information, and you can control the user's access to data stored in other Firebase products. You can also use the provided authentication token to verify the identity of users in your own backend services.[30]

- **Firebase Realtime Database:**

Store and sync data with our NoSQL cloud database. Data is synced across all clients in Realtime, and remains available when your app goes offline.

The Firebase Realtime Database is a cloud-hosted database. Data is stored as JSON and synchronized in Realtime to every connected client. When you build cross-platform apps with our iOS, Android, and JavaScript SDKs, all of your clients share one Realtime Database instance and automatically receive updates with the newest data.

The Firebase Realtime Database lets you build rich, collaborative applications by allowing secure access to the database directly from client-side code. Data is persisted locally, and even while offline, Realtime events continue to fire, giving the end user a responsive experience. When the device regains connection, the Realtime Database synchronizes the local data changes with the remote updates that occurred while the client was offline, merging any conflicts automatically.

The Realtime Database provides a flexible, expression-based rules language, called Firebase Realtime Database Security Rules, to define how your data should be structured and when data can be read from or written to. When integrated with Firebase Authentication, developers can define who has access to what data, and how they can access it.

The Realtime Database is a NoSQL database and as such has different optimizations and functionality compared to a relational database. The Realtime Database API is designed to only allow operations that can be executed quickly. This enables you to build a great Realtime experience that can serve millions of users without compromising on responsiveness. Because of this, it is important to think about how users need to access your data and then structure it accordingly.[31]

- **Cloud Storage for Firebase:**

  Cloud Storage for Firebase is built for app developers who need to store and serve user-generated content, such as photos or videos.

  Cloud Storage for Firebase is a powerful, simple, and cost-effective object storage service built for Google scale. The Firebase SDKs for Cloud Storage add Google security to file uploads and downloads for your Firebase apps, regardless of network quality.

  You can use our SDKs to store images, audio, video, or other user-generated content. On the server, you can use Google Cloud Storage APIs to access the same files.

  Developers use the Firebase SDKs for Cloud Storage to upload and download files directly from clients. If the network connection is poor, the client is able to retry the operation right where it left off, saving your users time and bandwidth.

  Cloud Storage for Firebase stores your files in a Google Cloud Storage bucket, making them accessible through both Firebase and Google Cloud. This allows you the flexibility to upload and download files from mobile clients via the Firebase SDKs for Cloud Storage. In addition, you can do server-side processing such as image filtering or video transcoding using the Google Cloud Storage APIs. Cloud Storage scales automatically, meaning

that there's no need to migrate to any other provider. Learn more about all the benefits of our integration with Google Cloud.

The Firebase SDKs for Cloud Storage integrate seamlessly with Firebase Authentication to identify users, and we provide a declarative security language that lets you set access controls on individual files or groups of files, so you can make files as public or private as you want.[32]

- **WebStorm:** WebStorm is an integrated development environment for coding in JavaScript and its related technologies. Just like IntelliJ IDEA and other Jet Brains IDEs, WebStorm will make your development experience more enjoyable, automating routine work and helping you handle complex tasks with ease.

  Here are some key features you get with WebStorm:

  Out-of-the-box support for JavaScript, TypeScript, React, Vue, Angular, Node.js, HTML, style sheets, and others.

  Smart editor with code completion, on-the-fly error detection, safe code refactoring's, and fast navigation across the entire code base.

  A variety of built-in developer tools that allow you to debug and test your client-side and Node.js apps as well as to work with version control, linters, build tools, terminal, and HTTP client.

  Tools for efficient teamwork, including a service for remote collaborative development and pair programming and the ability to share your project configuration with others.

  The ability to customize your work environment by experimenting with things like themes and plugins.

  Postman is the collaboration platform for API development. Postman simplifies each step of building an API and streamlines collaboration so you can create better APIs—faster.[33][34]

- **Node.js:**

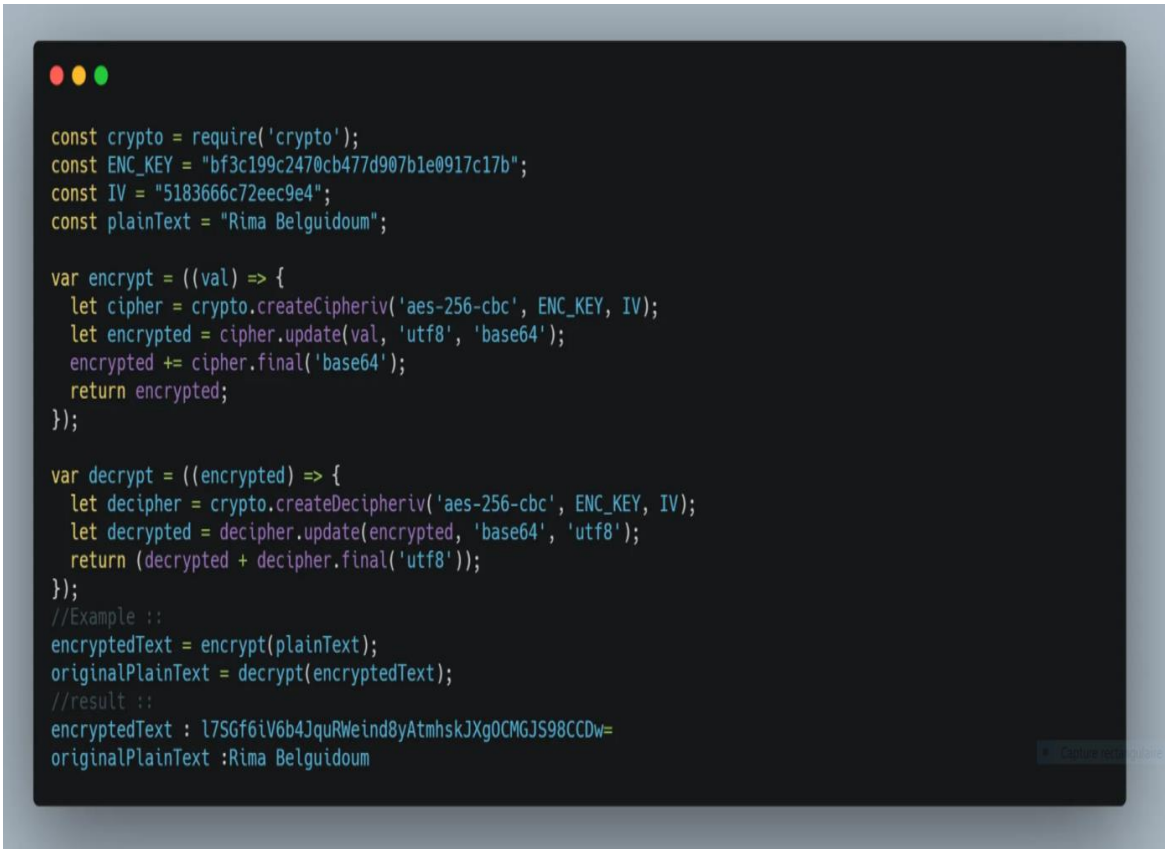
Node.js is similar in design to, and influenced by, systems like Ruby's Event Machine and Python's Twisted. Node.js takes the event model a bit further. It presents an event loop as a run time construct instead of as a library. In other systems, there is always a blocking call to start the event-loop. Typically, behavior is defined through

callbacks at the beginning of a script, and at the end a server is started through a blocking call like Event Machine::run(). In Node.js, there is no such start-the-event-loop call. Node.js simply enters the event loop after executing the input script. Node.js exits the event loop when there are no more callbacks to perform. This behavior is like browser JavaScript — the event loop is hidden from the user.

HTTP is a first-class citizen in Node.js, designed with streaming and low latency in mind. This makes Node.js well suited for the foundation of a web library or framework.

Node.js being designed without threads doesn't mean you can't take advantage of multiple cores in your environment. Child processes can be spawned by using our child process fork () API, and are designed to be easy to communicate with. Built upon that same interface is the cluster module, which allows you to share sockets between processes to enable load balancing over your cores.[35]

- **Example:** cbc algo use node.js



```
const crypto = require('crypto');
const ENC_KEY = "bf3c199c2470cb477d907b1e0917c17b";
const IV = "5183666c72eec9e4";
const plainText = "Rima Belguidoum";

var encrypt = ((val) => {
  let cipher = crypto.createCipheriv('aes-256-cbc', ENC_KEY, IV);
  let encrypted = cipher.update(val, 'utf8', 'base64');
  encrypted += cipher.final('base64');
  return encrypted;
});

var decrypt = ((encrypted) => {
  let decipher = crypto.createDecipheriv('aes-256-cbc', ENC_KEY, IV);
  let decrypted = decipher.update(encrypted, 'base64', 'utf8');
  return (decrypted + decipher.final('utf8'));
});
//Example ::
encryptedText = encrypt(plainText);
originalPlainText = decrypt(encryptedText);
//result ::
encryptedText : l7SGf6iV6b4JquRWeind8yAtmhskJXgOCMGJS98CCDw=
originalPlainText :Rima Belguidoum
```

Figure 3.8: cbc algo use node.js.

- **REST API**: **(Representational state transfer)**

  Also known as **RESTful API** is an application programming interface (API or web API) that conforms to the constraints of REST architectural style and allows for interaction with RESTful web services. REST stands for representational state transfer and was created by computer scientist Roy Fielding.

  An API is a set of definitions and protocols for building and integrating application software. It's sometimes referred to as a contract between an information provider and an information user—establishing the content required from the consumer (the call) and the content required by the producer (the response). For example, the API design for a weather service could specify that the user supply a zip code and that the producer reply with a 2-part answer, the first being the high temperature, and the second being the low.

  In other words, if you want to interact with a computer or system to retrieve information or perform a function, an API helps you communicate what you want to that system so it can understand and fulfill the request.

  You can think of an API as a mediator between the users or clients and the resources or web services they want to get. It's also a way for an organization to share resources and information while maintaining security, control, and authentication—determining who gets access to what.

  Another advantage of an API is that you don't have to know the specifics of caching—how your resource is retrieved or where it comes from.

- **Retrofit**

  Retrofit is a REST Client for Java and Android. It makes it relatively easy to retrieve and upload JSON (or other structured data) via a REST based web service. In Retrofit you configure which converter is used for the data serialization. Typically for JSON you use GSon, but you can add custom converters to process XML or other protocols. Retrofit uses the Ok Http library for HTTP requests.[36]

  Example: In this example, we will try to explain how to add a user

```
@FormUrlEncoded
   @POST("users/create")
   Call<user> createUser(@FieldMap Map<String, String> fields);

 private void createUser() {
       Map<String, String> fields = new HashMap<>();
       fields.put("username", "Rima");
       fields.put("email", "Rima92@gmail.com");
       fields.put("state", "true");
       //... any fields
       Call<user> call = usersApi.createUser(fields);
       call.enqueue(new Callback<user>() {
           @Override
           public void onResponse(Call<user> call, Response<user> response) {
               if (!response.isSuccessful()) {
                   textViewResult.setText("Code: " + response.code());
                   return;
               }
               // print result
               System.out.println("get users -> response code : "+response.code());
                   String content = "";
                   content += "ID: " + response.body().getId() + "\n";
                   content += "Uersname: " + response.body().getUsername() + "\n";
                   content += "email: " + response.body().getEmail() + "\n";
                   content += "state: " + response.body().isState() + "\n";
                   textViewResult.append(content);
                 System.out.println("print data\n"+response.body());
           }
           @Override
           public void onFailure(Call<user> call, Throwable t) {
               textViewResult.setText(t.getMessage());
               System.out.println("Throwable ::"+t.getMessage());
           }
       });
   }
```

Figure 3.9: explain how to add a user.

- **JAVA:** is a popular programming language that is widely used for application development worldwide. It has the advantages of multithreading, scalability, memory management, high security, and community support.

- **Android Studio**: is the best Integrated Development Environment for android application development projects. We have been using it for a while because it facilitates various features, which helped us to develop applications in a better and easier way.

3.6. Implementation of the application and the result

In this section, we present the various experimental results of the application.

3.6.1. Description of the work carried out

This part is devoted to the description of the realization and implementation phase of this project, so we will show the different modules of our application in order to illustrate more clearly the various uses of the application. We start the presentation of our application with the home interface.

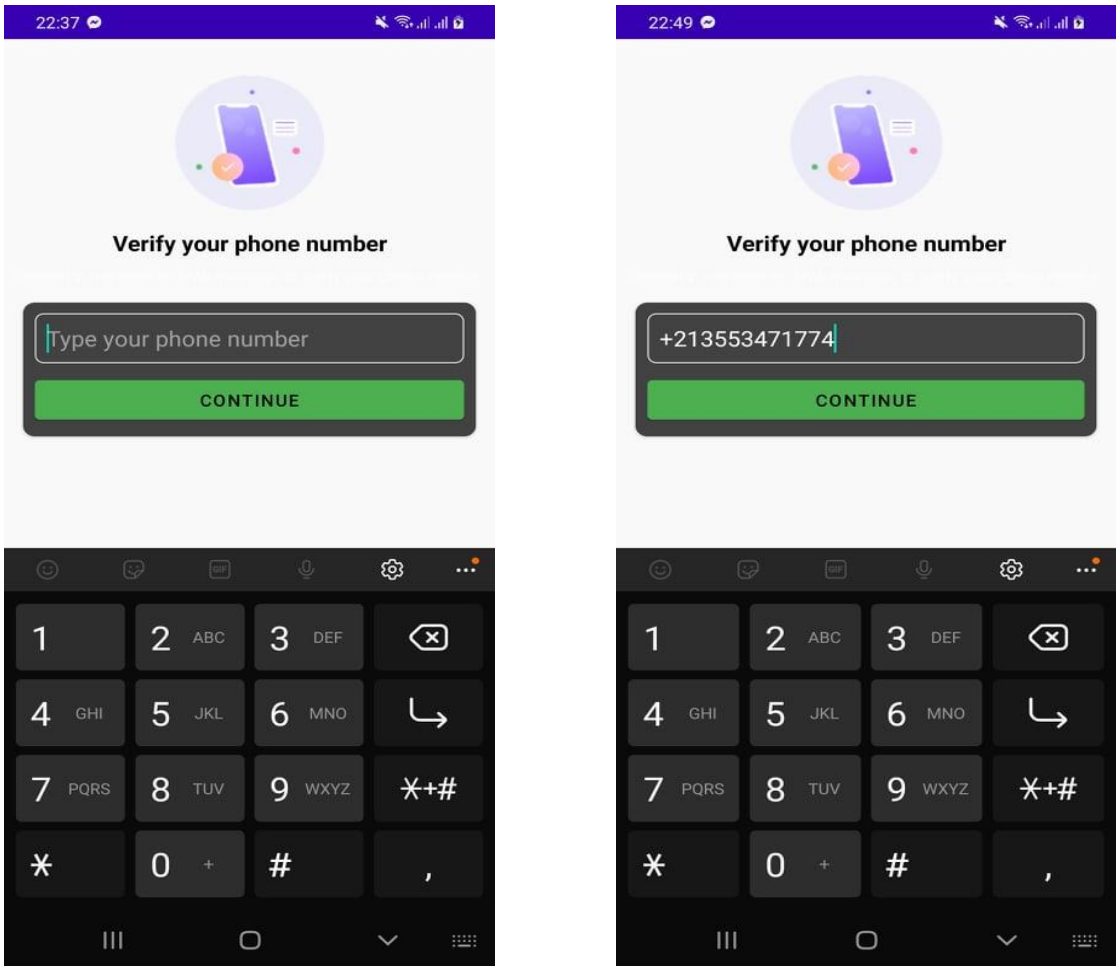


Figure 3.10: Activity phone number.

After installing the application, user needs to turn on the internet connection for registering as a new user by providing telephone number. As shown in Figure 3.10 above.

After entering the phone number and pressing Continue, an SMS will be sent containing an OTP to verify the identity authentication. As shown in the figures below.

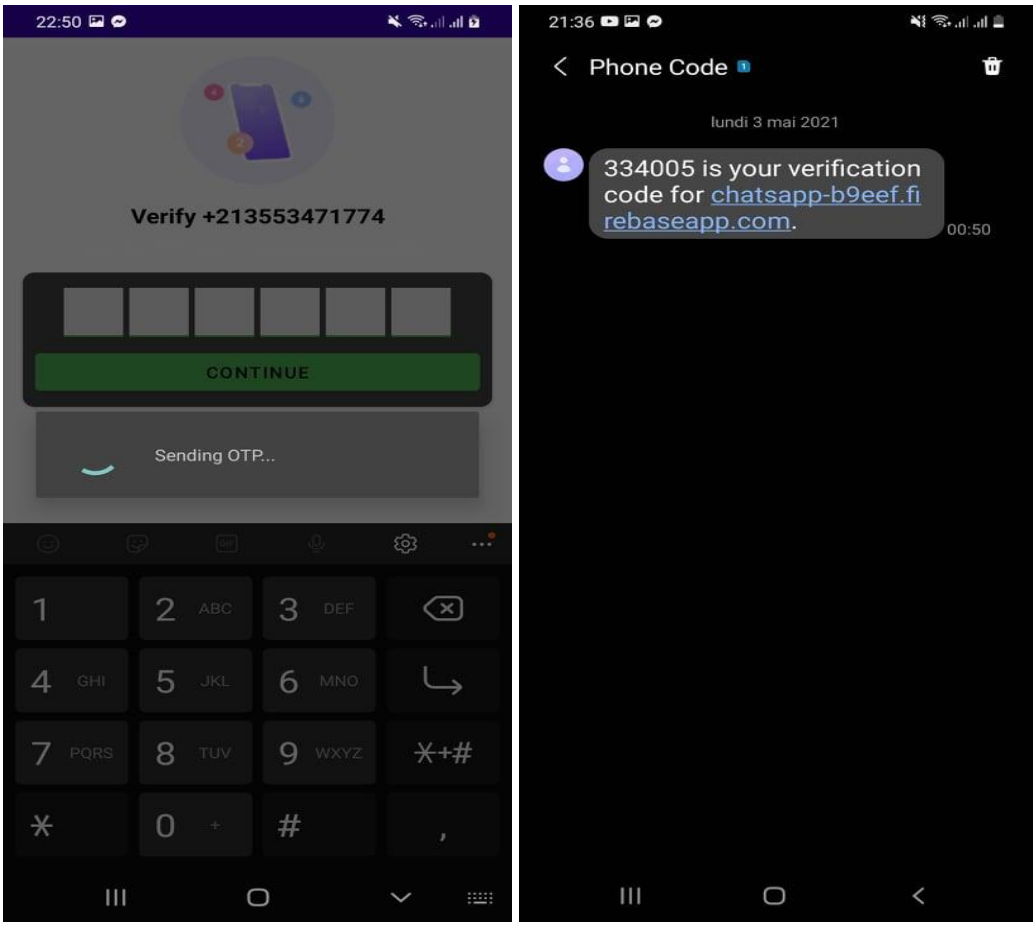


Figure 3.11: OTP code.

Note

Since OTP is not free, we cannot purchase it for economic reasons, so we enter it manually and will not send it to the relevant phone number via SMS. As shown in the figure 3.12 below.

Figure 3.12: Sign in phone number and OTP code.

Write OTP token to verify user authentication. If the code is incorrect, a "Failed" notification will appear. As shown in the figure 3.13.

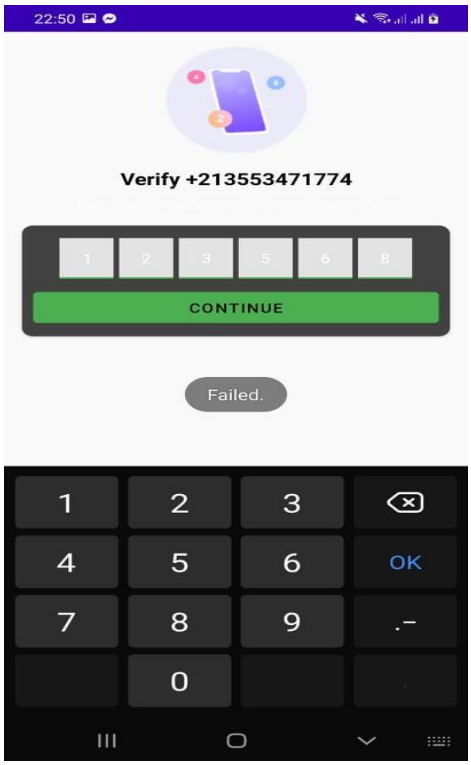


Figure 3.13: failed OTP code.

After typing the OTP token and verifying the authentication it will be sorted into the user list. As shown in the figure 3.14 below.

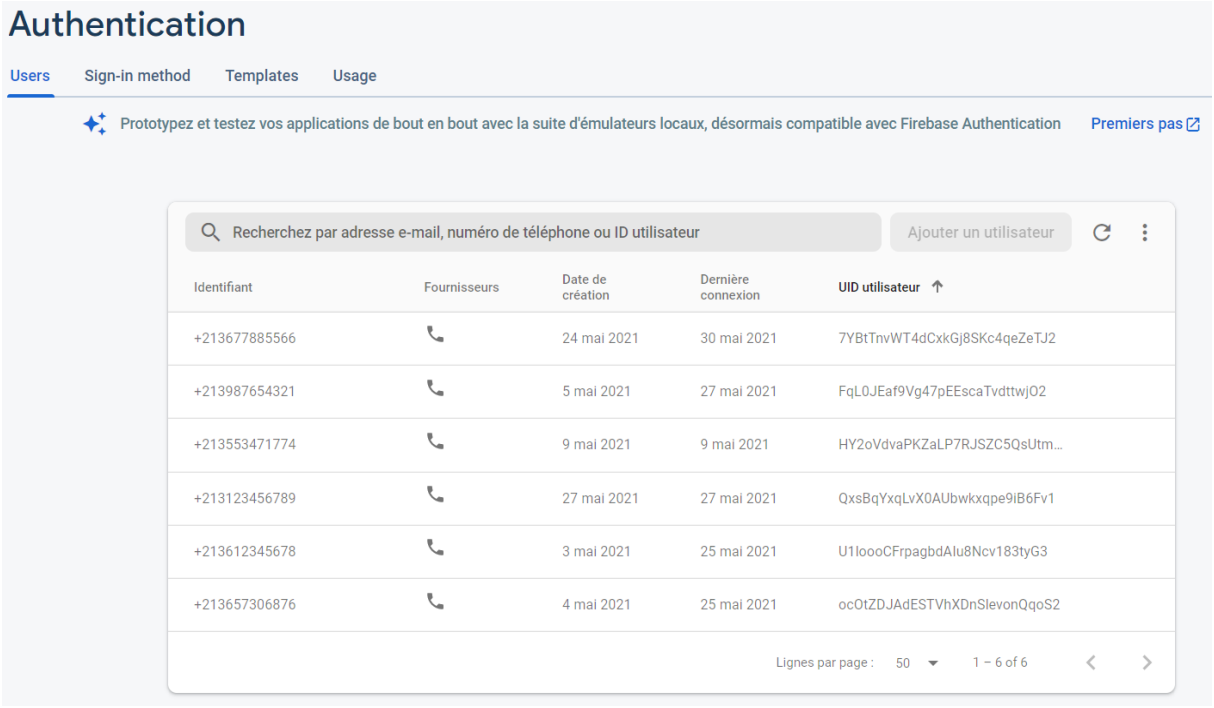


Figure 3.14: Registered numbers.

After entering the correct OTP code and passing the verification, you will enter the personal information page where you can enter the user's name and photo as shown in Figure 3.15. Since entering the user's name is mandatory, he cannot enter the next page without writing the name, but the image is optional and not required. As shown in the figure 3.16.

Figure 3.15: Setup profile.

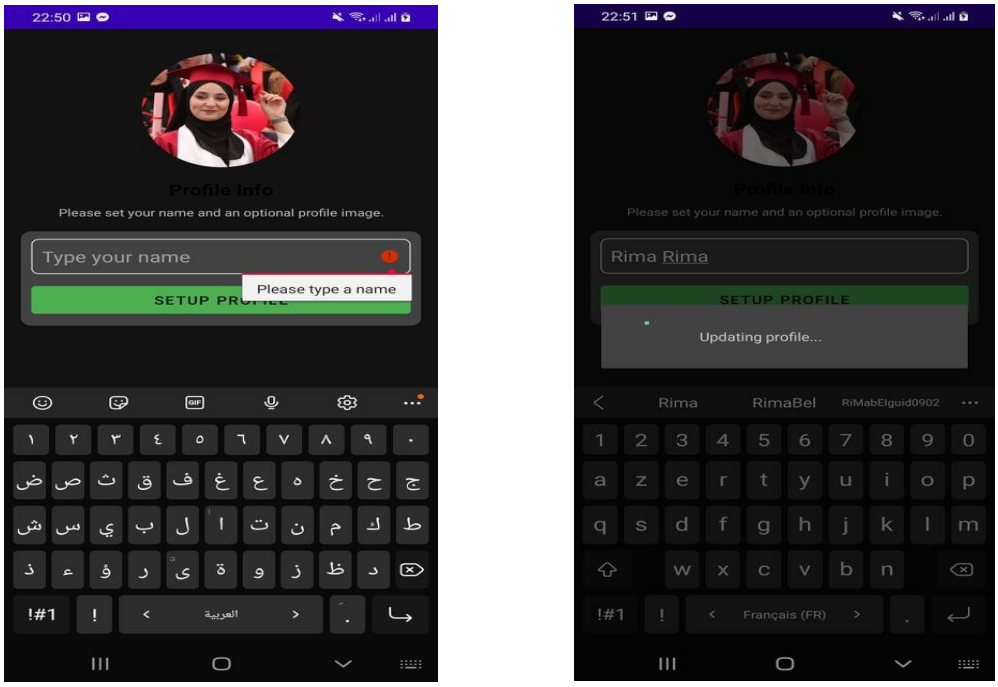


Figure 3.16: Verification of the setup profile.

Now after login, the list of friends comes up with the name of other users for exchanging data (messages). As shown in the figure 3.16

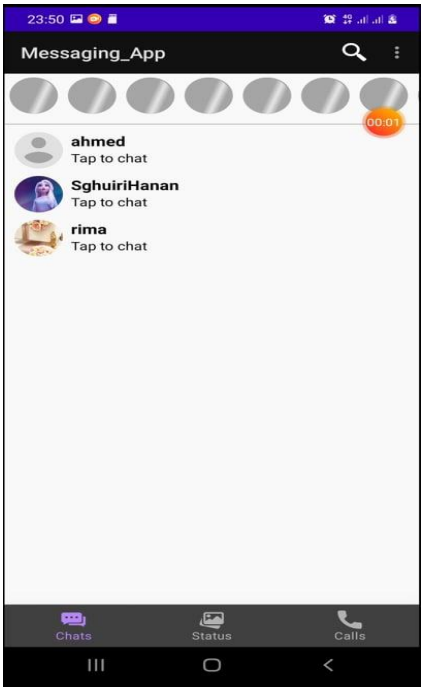


Figure 3.17: Row conversation.

When the user enters the chat list for the first time, the chat page shown in the figure above will appear, the location of other users and the friend search box.

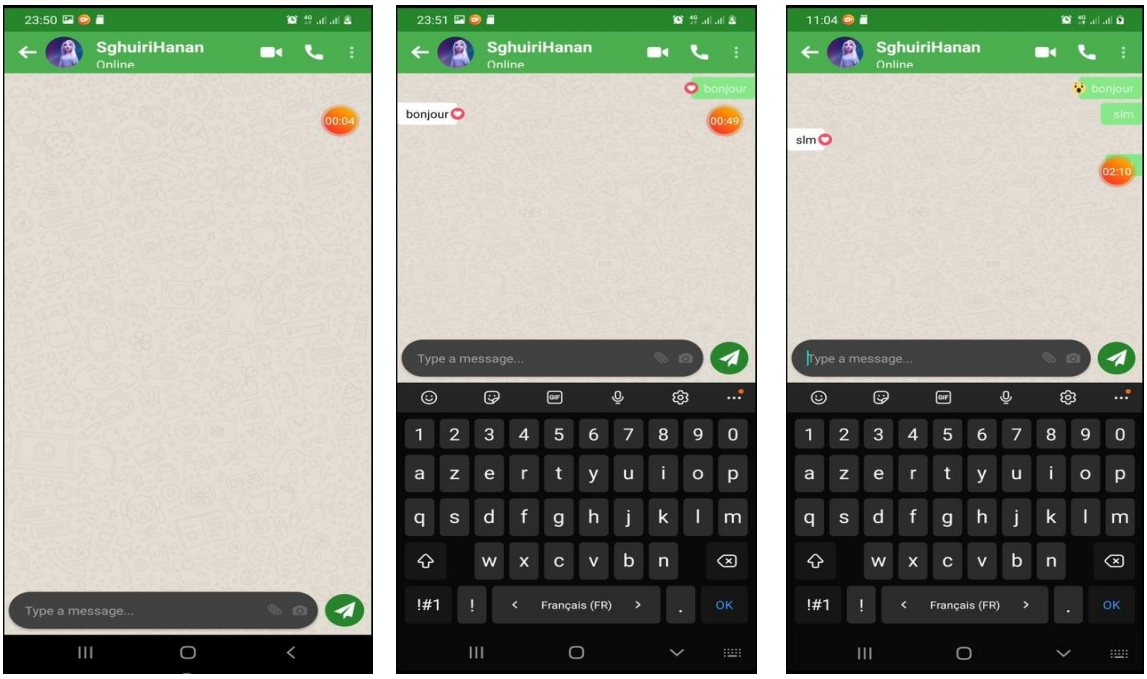


Figure 3.18: Chat activity and the different reactions of the messages.

When you log in to your friend's account to chat with him, the user interface will appear as shown in Figure 3.18. The boxes that appear at the bottom of the page are used to write messages and send, and if the user is online, he seems to be connected to the Internet, nothing is displayed when offline. Users can also interact with the messages and also change the interactive as they wish. As shown above.

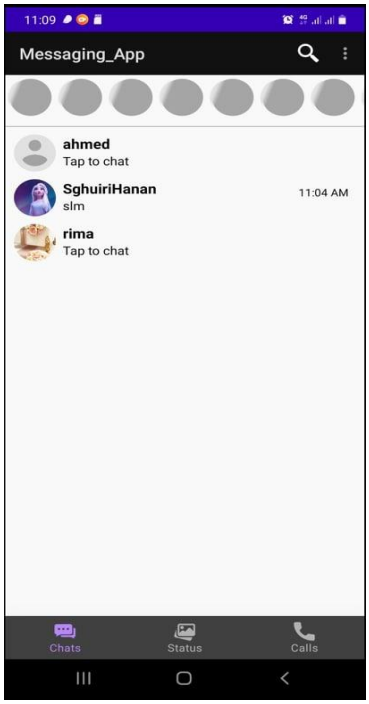


Figure 3.19: The last message and the time of received.

In Figure 3.19 above, the last message of the friends you were chatting with and the time it was sent is shown.

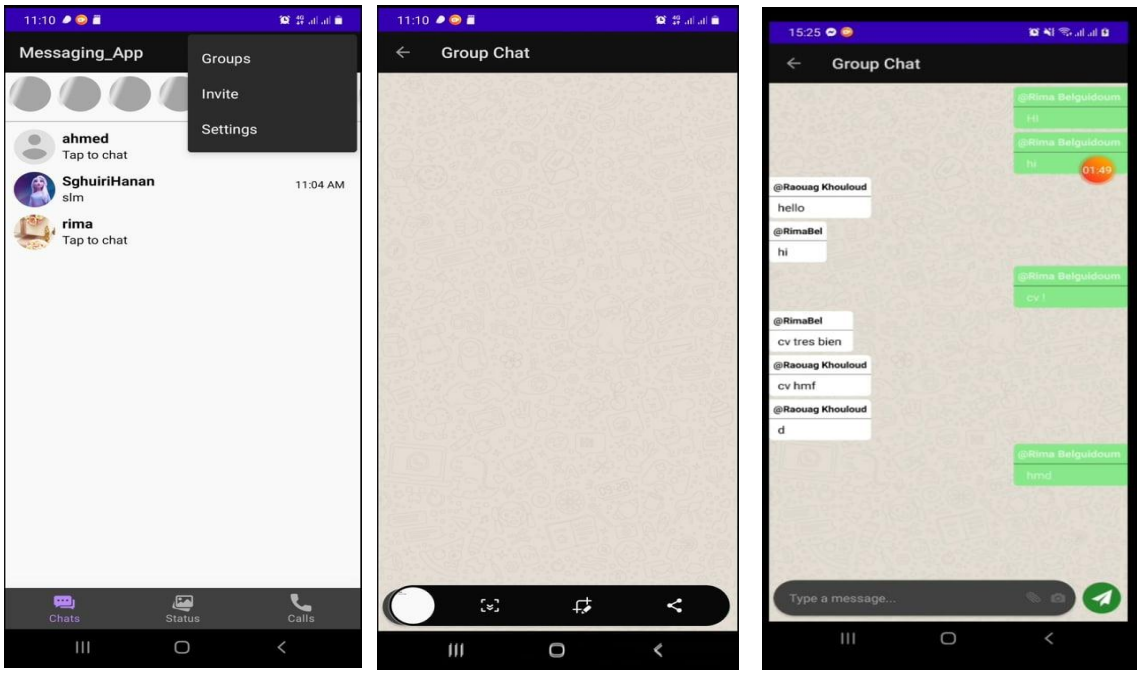


Figure 3.20: Group chat

Then we enter the group chat interface. Where the message will be displayed with the name of the user who sent it. It can also interact with any message. As shown in Figure 3.20.

- **Database**

The database consists of four important components, which are:

- **chats**: which contain encrypted messages between the sender and the receiver.
- **presence:** which contains all users who use the application, as well as whether the users are online or not.
- **public:** Contains group chat messages.
- **users:** belonging to the presence and contains the users' information.

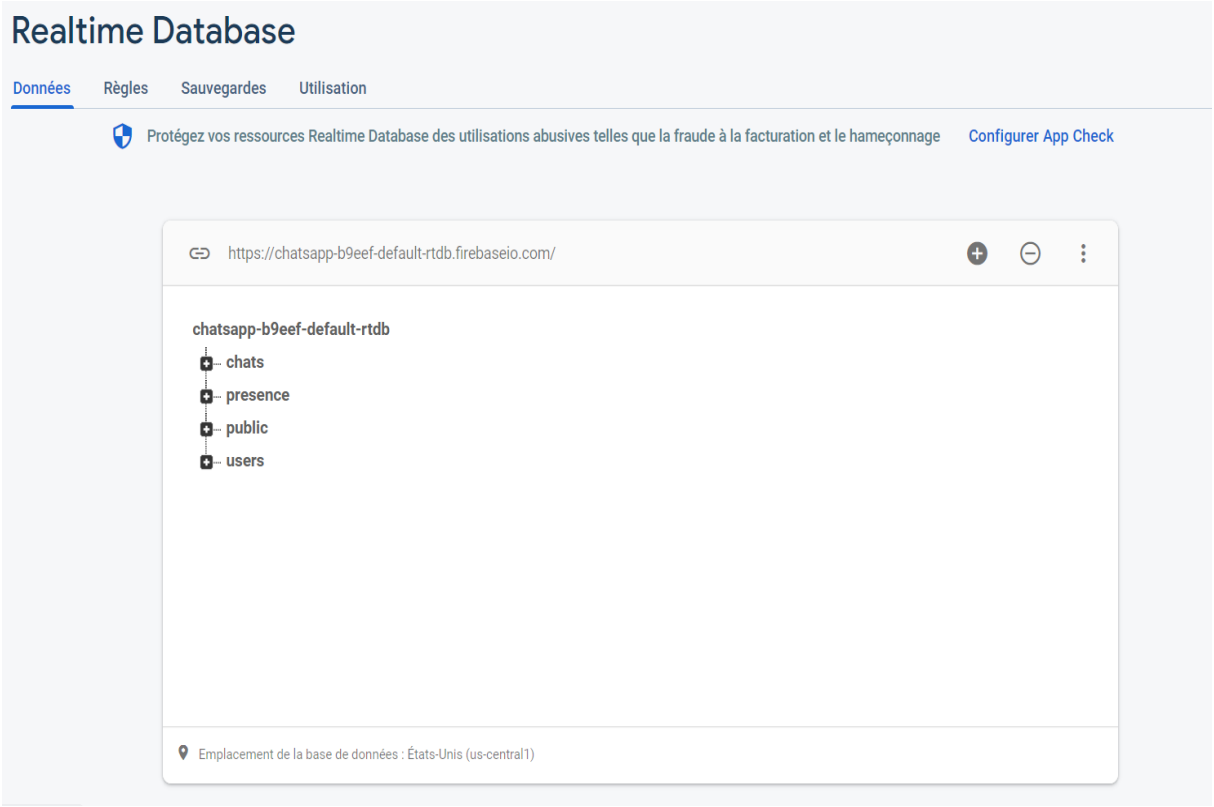


Figure 3.21: Real-time Database.

The following two pictures are messages in the chat. So that it gives all the characteristics of the messages sent, these characteristics are:

- **Feeling**: the state of expressing the reaction.
- **Message:** Under normal circumstances, the message is normal, but because we encrypt the message, it will appear encrypted in the database.
- **senderId:** ID of the sender.
- **Timestamp**: the time when the message was sent.
- **lastMesg:** The last message sent or received.
- **lastMsgTime**: the time when the last message was sent.
  - Now, when you start the chat process, the encryption process will start. When a message is sent from one user to another, the encrypted message will be sent and stored in an encrypted database. As shown in Figure 3.22 and Figure 3.23.

Figure 3.22: The different information's of message.

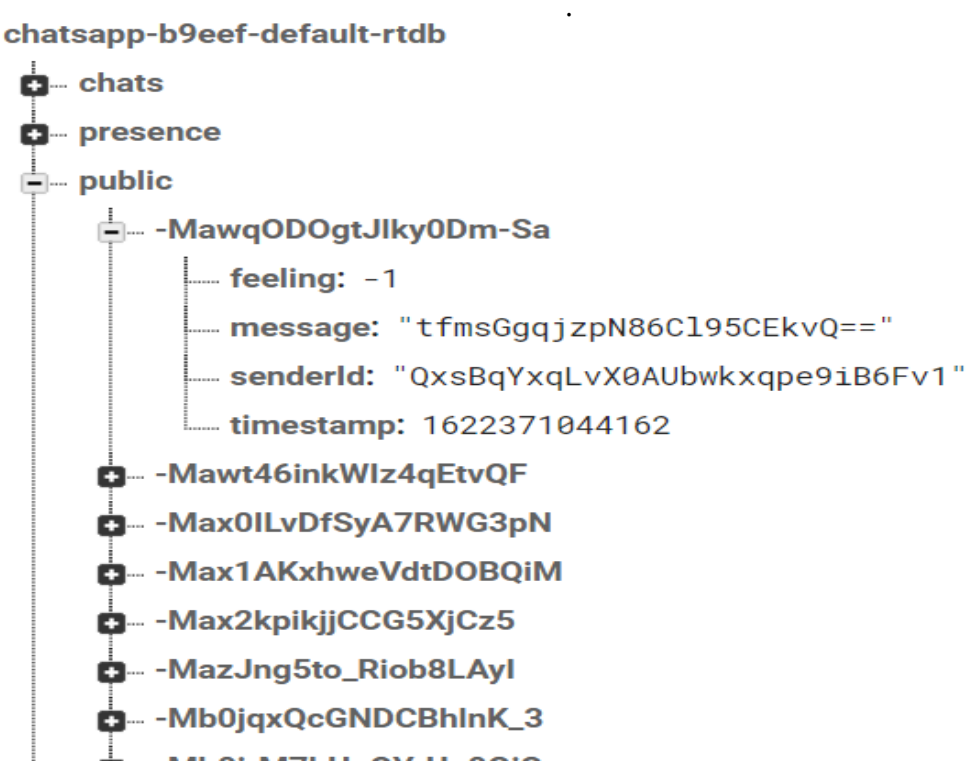


Figure 3.23: The different information's of message Group.

The user can also delete the message by clicking the message until the delete list appears, and choose to delete it for me or delete it for everyone or cancel, as shown in Figure 3.24. When the message is deleted, the following notice appears “This message is removed”.

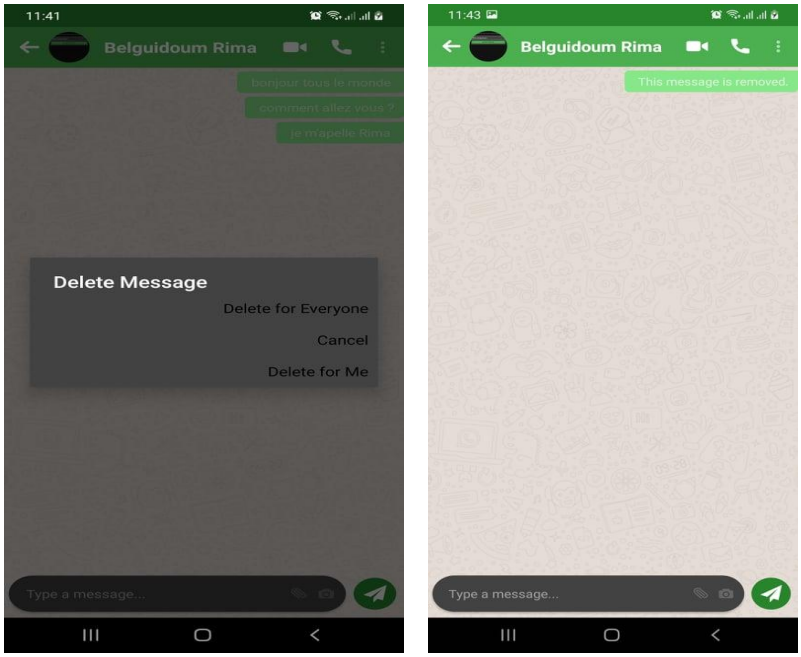


Figure 3.24: delete messages.

In addition, you can add and delete images in the conversation. As shown in Figure 3.25 and Figure 3.26 below.

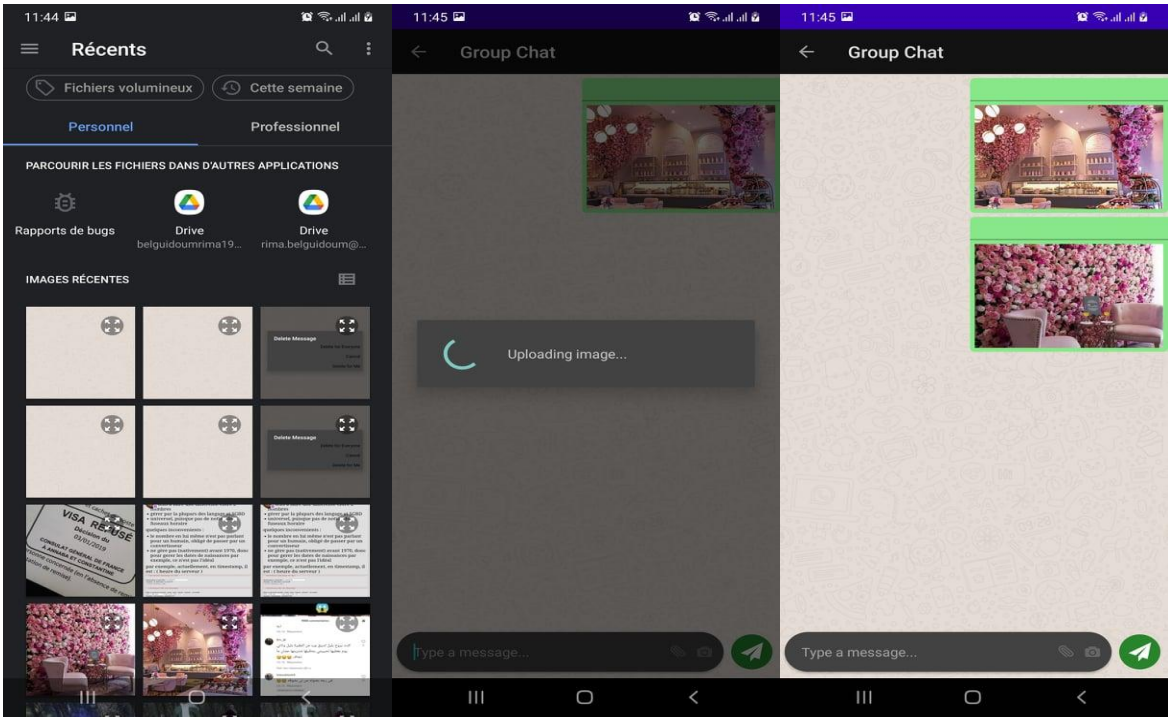


Figure 3.25: uploading images.

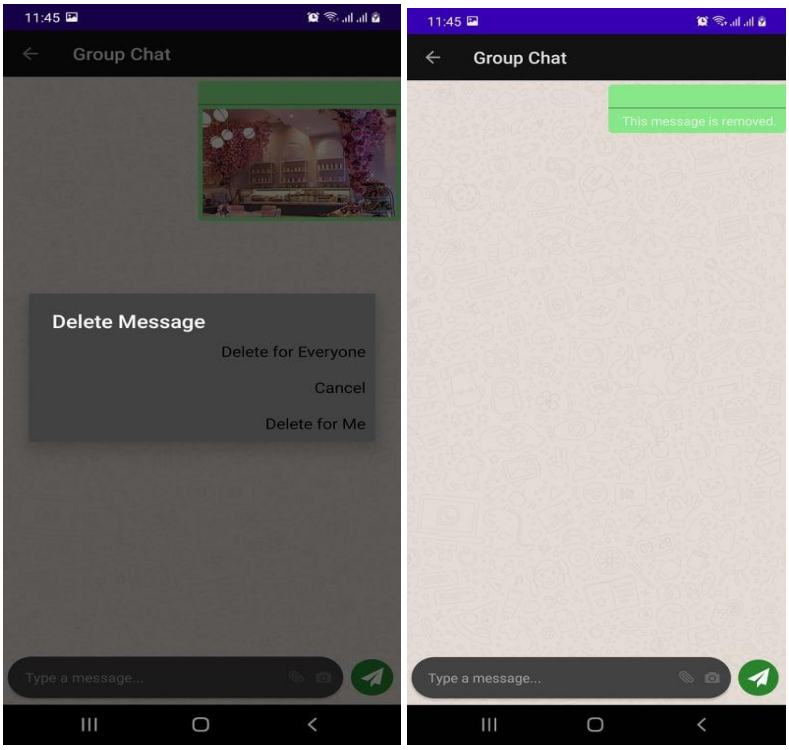


Figure 3.26: delete images.

Users can also add one or more stories, as shown in the figure 3.27 below.

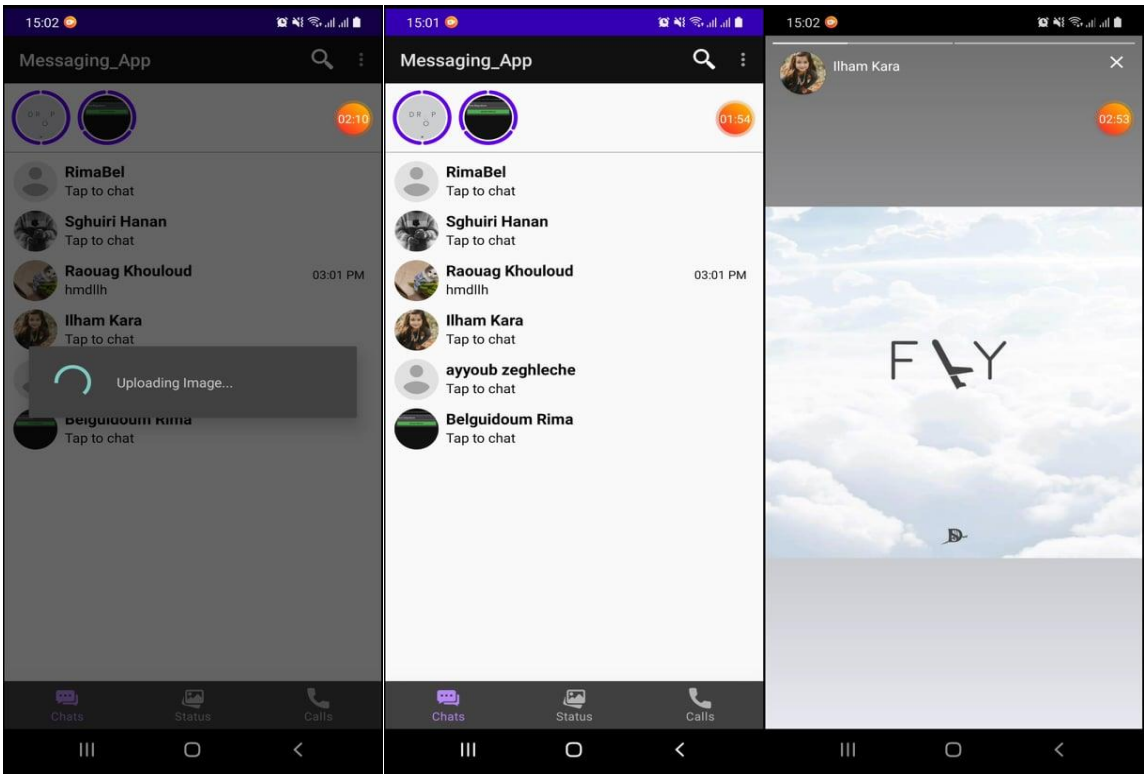
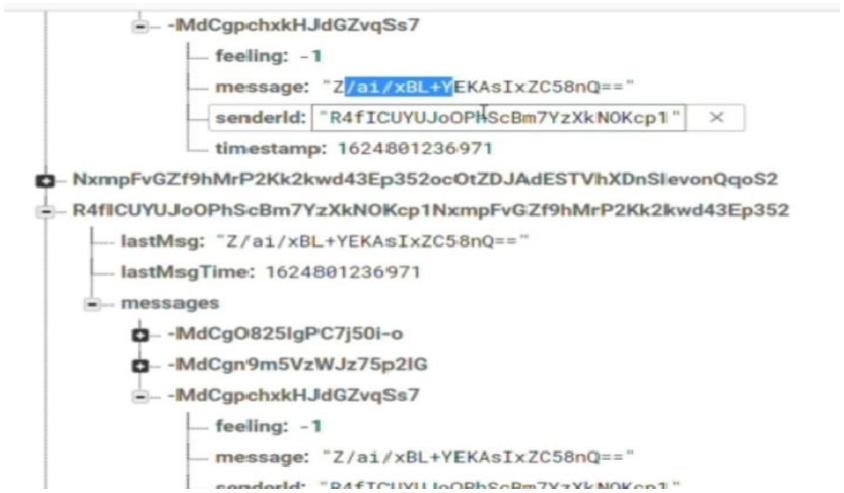


Figure 3.27: Add stories.

3.7. The Results

Message Encryption	
	**Original message**
	**Encrypted message**

3.8. Development environment

- **Material resources**
  - Intel(R) Core (TM) i5-5300 CPU @ 2.30GHz (4 CPUs), ~2.3GHz
  - Installed memory (RAM): 8.00G.
- **Software resources**
  - Operating system: Windows 10 professional 64 bits.
  - Editor used is Android Studio.

3.9. Discussion

Here is the end of our work that focused on encrypting instant messages to secure the transmission of information. Throughout this work, we have shown how messages can be better protected. We are looking for the best message encryption algorithms in order to protect them from eavesdropping and theft.

We have introduced a method based on a set of modern algorithms to maximize the protection of our information. Network-level and database-level protection, that is, use the AES algorithm with CBC mode to encrypt messages. This is used to encrypt messages at the database level. As for the network, we rely on CBC using Node.js and API, which depends on changing the key to change the encryption. Each message has a variable encryption every time, which increases the strength of the protection and makes it difficult for hackers. This is due to the change of the key, making the information transmitted over the network inaccessible.

Therefore, the results we obtained from the realization of the symmetric cryptographic algorithm:

The encryption used is safe because we did not assign a value to the key, it is random, initialized by the program, and the execution time is very fast. Another advantage is that the encryption and decryption is done automatically and does not need to enter the keys by the users. The most important thing is not to lose information; Therefore, we hope that we have provided an integrated complementary work and that the results obtained are mortgaged. We are also satisfied with the work and the results that we have provided.

3.10. Summary

Based on the test results provided by the application, we can say that the proposed method meets the needs and works perfectly in terms of sending messages and execution time, and also encrypting / decrypting without data loss, and at the same time the security is enhanced because everything is implemented in the application, since we only send encrypted messages, without giving the key, everything is intertwined and hidden. Therefore, we conclude that this method is suitable for sending messages secure.

GENERAL CONCLUSION

In this paper, we discussed the issue of instant messaging protection. With the development of the Internet and switching networks, this issue has become more and more important. We have implemented a special encryption system to solve this problem. Adopt AES-256 symmetric encryption algorithm and CBC mode to encrypt the database. End-to-end encryption is also implemented to encrypt message communication. This makes messages and communications secure, so far almost impossible to crack.

The main purpose of this encryption is to ensure the confidentiality of the information contained in the message after it is sent.

Experimental results show that our application has high security and efficiency.

From the perspective of this work, we will use a hybrid encryption system based on two algorithms, AES and RSA, to improve our application to ensure the three pillars of security: confidentiality, authentication, and integrity. We will also implement a system for encryption of voice and video messages.

BIBLIOGRAPHY

[1] S. Amina and G. Denia, “Sécurisation des Régions de l’Image Satellite par la Cryptographie,” 2016.

[2] R. MM, A. T, and R. A, “Development of Cryptography-Based Secure Messaging System,” *J. Telecommun. Syst. Manag.*, vol. 05, no. 03, 2016, doi: 10.4172/2167-0919.1000142.

[3] A. Brahim, “Implementation of IP Cores Dedicated To Cryptography,” 2012.

[4] S. Haunts, *Applied Cryptography in .NET and Azure Key Vault*. 2019.

[5] A. Malla, P. Sahu, and M. Mishra, “A novel encryption scheme for secure SMS communication,” *Lect. Notes Electr. Eng.*, vol. 443, pp. 467–478, 2018, doi: 10.1007/978-981-10-4765-7_50.

[6] K. Jonathan and L. Yehuda, *Introduction to Modern Cryptography, Second Edition*, vol. 1. 2014.

[7] KEITH Martin, *Everyday Cryptography*, vol. 53, no. 9. 2017.

[8] “Attacks On Cryptosystems - Tutorialspoint.” https://www.tutorialspoint.com/cryptography/attacks_on_cryptosystems.htm (accessed May 27, 2021).

[9] R. Canetti and H. Krawczyk, “Analysis of Key-Exchange protocols and their use for building secure channels,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 2045, pp. 453–474, 2001, doi: 10.1007/3-540-44987-6_28.

[10] P. Jorgensen, “Applied cryptography: Protocols, algorithm, and source code in C,” *Gov. Inf. Q.*, vol. 13, no. 3, p. 336, 1996, doi: 10.1016/s0740-624x(96)90083-0.

[11] R. Mokhtarnameh, N. Muthuvelu, S. B. Ho, and I. Chai, “A Comparison Study on Key Exchange-Authentication protocol,” *Int. J. Comput. Appl.*, vol. 7, no. 5, pp. 5–11, 2010, doi: 10.5120/1161-1459.

[12] M. Marlinspike and T. Perrin, “The X3DH Key Agreement Protocol,” *Signal*, p. 11, 2016, [Online]. Available: https://www.whispersystems.org/docs/specifications/x3dh/.

[13] T. Zebua, R. K. Hondro, and E. Ndruru, “Message Security on Chat App based on Massey Omura Algorithm,” *IJISTECH (International J. Inf. Syst. Technol.*, vol. 1, no. 2, p. 16, 2018, doi: 10.30645/ijistech.v1i2.11.

[14] J.-P. Aumasson, *Serious cryptography : a practical introduction to modern encryption*. No Starch Press, Inc., 2018.

[15] S. Nayak *et al.*, “An application for end to end secure messaging service on Android supported device,” *2017 8th IEEE Annu. Inf. Technol. Electron. Mob. Commun. Conf. IEMCON 2017*, pp. 290–294, 2017, doi: 10.1109/IEMCON.2017.8117222.

[16] “Information Security Notes: More about Data Encryption Standard (DES) and Advanced Encryption Standard (AES).” https://www.cryptographynotes.com/2019/07/encryption-standard-des-and-aes-differences-advantages-disadvantages.html (accessed May 27, 2021).

[17] “Developing a real-time secure chat application like WhatsApp & Signal with end-to-end encryption | QED42.” https://www.qed42.com/blog/developing-real-time-secure-chat-application-like-whatsapp-or-signal-with-end-end-encryption (accessed May 27, 2021).

[18] “A Brief History of Instant Messaging.” https://mashable.com/2012/10/25/instant-messaging-history/ (accessed May 27, 2021).

[19] D. Nurtdinova, “Security in Mobile Messaging,” 2016.

[20] “What is Advanced Encryption Standard (AES)? Definition, Encrption, Decryption, Advantages and Disadvantages - Binary Terms.” https://binaryterms.com/advanced-encryption-standard-aes.html (accessed May 27, 2021).

[21] “10 Most Secure Messaging Apps - Best Encrypted Chat App Solutions.” https://getstream.io/blog/most-secure-messaging-apps/ (accessed May 27, 2021).

[22] J. Daernen and V. Rijrnen, *The Design of Rijndael: AES - The Advanced Encryption Standard*. 2002.

[23] “The Best Encrypted Messaging Apps You Should Use Today [Updated 2019].” https://heimdalsecurity.com/blog/the-best-encrypted-messaging-apps/ (accessed May 27, 2021).

[24] “Cryptanalysis - an overview | ScienceDirect Topics.” https://www.sciencedirect.com/topics/computer-science/cryptanalysis (accessed May 27, 2021).

[25] “The Most Secure Messaging Apps for Android & iOS 2021 | AVG.” https://www.avg.com/en/signal/secure-message-apps (accessed May 27, 2021).

[26] A. Mujaj, “A Comparison of Secure Messaging Protocols and Implementations,” pp. 1–129, 2017, [Online]. Available: https://www.duo.uio.no/bitstream/handle/10852/56906/1/master_thesis.pdf.

[27] J. Rittinghouse and J. Ransome, *IM Instant Messaging Security*. 2005.

[28] R. M. Ali and S. N. Alsaad, “Instant messaging security and privacy secure instant messenger design,” *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 881, no. 1, 2020, doi: 10.1088/1757-899X/881/1/012117.

[29] C. Khawas and P. Shah, “Application of Firebase in Android App Development-A Study,” *Artic. Int. J. Comput. Appl.*, vol. 179, no. 46, pp. 975–8887, 2018, doi: 10.5120/ijca2018917200.

[30] “Firebase Authentication.” https://firebase.google.com/docs/auth?authuser=0 (accessed Jun. 28, 2021).

[31] “Firebase Realtime Database.” https://firebase.google.com/docs/database?authuser=0 (accessed Jun. 28, 2021).

[32] “Stockage cloud pour Firebase.” https://firebase.google.com/docs/storage?authuser=0 (accessed Jun. 28, 2021).

[33] “Meet WebStorm | WebStorm.” https://www.jetbrains.com/help/webstorm/meet-webstorm.html (accessed Jun. 28, 2021).

[34] “About Us | Postman.” https://www.postman.com/company/about-postman/ (accessed Jun. 28, 2021).

[35] “About | Node.js.” https://nodejs.org/en/about/ (accessed Jun. 28, 2021).

[36] “Retrofit.” https://square.github.io/retrofit/ (accessed Jun. 28, 2021).