Journal of Analysis and Computation (JAC)
(An International Peer Reviewed Journal), www.ijaconline.com, ISSN 0973-2861
Volume XVII, Issue I, Jan-June 2023

SPECIFICATION FOR PRESERVING THE SECURITY AND PRIVACY OF THE CHAT APPLICATION

B.Umamaheswari, Priyanka Mitra, Anju Rajput, Somya Agrawal

Assistant Professor, Jaipur Engineering College and Research Centre, Jaipur, India

ABSTRACT

One of the most significant and well-liked smartphone applications today is chat. It allows for the free sharing of text messages, photographs, and files, making it possible for users to stay in touch. Every message needs to be secure. The purpose of the study is to suggest a chat application that offers End-to-End security and enables secure data flow between users without concern. In addition to storage's protection. This article presents a list of specifications for a secure chat application, and the program was created in accordance with these specifications. Based on those requirements, the suggested chat application was compared to other well-known programs and tested as evidence for delivering End-to-End security.

Keywords-Secure chat application, Security, Android, Secure session, Secure storage

[1] INTRODUCTION

Mobile devices have assimilated into daily activities due to the rapid development of mobile phones. Because of their special qualities that draw users, chat applications have developed and significantly changed social media in recent years [1]. It offers a variety of services, such as the exchange of text messages, photographs, files, and other things, as well as real-time messaging. Additionally, it supports cross-platform devices like iOS and Android. Currently, chat applications are used on smartphones by 100 million users each month [2].Peer-to-peer networks and client-server architecture are the two forms of architecture used in those applications. There is no central server and each user has their own data storage in a peer-to-peer network. In contrast, a client-server network has dedicated servers and clients, and the information is kept on a central server [3].
Though they are of utmost importance, security and privacy in chat applications are rarely taken seriously. The majority of the widely used messaging apps failed to meet the majority of security standards in a test conducted by the Electronic Frontier Foundation. The discussions may be used by these programmes as information for specific objectives. Furthermore, it is unacceptably intrusive to read the private talks. The service provider has full access to every

message sent and received through most applications because they solely utilise Transport Layer Security (TLS) to secure channels construction [4]. As a result, intruders can access these communications. In order to guarantee security and privacy, communications should be encrypted from sender to receiver so that no one, not even the service provider, can read them [5]. This also protects the device's local storage.
In this work, we propose end-to-end security, which ensures that only the sender and recipient can access messages without a third party, with a focus on security, privacy, and speed. Additionally, messages between the parties are transferred quickly and with storage safety.

[2] MOBILE CHAT APPLICATIONS

In this section, we briefly introduce many of popular chat applications in the mobile market according to security and privacy concerns. Unfortunately, some chat applications are not public or open source makes it difficult for evaluated by the developer's community, security experts or researcher academic.

Viber:

Viber is a Voice over IP (VoIP) and instant messaging programme for smartphones created by Viber Media. Users can send and receive photos, videos, and audio messages in addition to instant chats. End-to-end encryption has recently been offered by Viber for their service, but only for one-on-one and group chats in which all participants are using the most recent Viber 6.0 for Android, iOS, or Windows 10. End-to-end encryption is not yet supported in the Viber iOS app for the iPhone and iPad when sending attachments like photographs and movies via the iOS Share Extension [6]. Viber has privacy issues, such as adding a user to a friend list or group without their knowledge or consent. Local storage is also not secured.

WhatsApp:

One of the most widely used messaging apps, WhatsApp, just made end-to-end encryption available to all of its 1 billion users across all platforms. To verify that the discussion is encrypted, WhatsApp uses a portion of a security protocol created by Open Whisper System and offers a security-verification code that can be shared with a contact [7]. Because WhatsApp is closed source, it is difficult to check its operation and compare it to the performance of the recently released encryption protocol. This makes it difficult to fully trust the app.

Telegram:

Users can communicate messages, photographs, videos, stickers, and files using the open source instant messaging programme Telegram [8]. Regular chat and secret chat are the two communications options offered by Telegram. Regular chat uses cloud-based messaging that is client-server based, does not offer end-to-end encryption, keeps all messages on its servers, and synchronises with all user devices [9]. Additionally, local storage is not by design encrypted. Client-client communication for secret chats offers end-to-end encryption. Unlike ordinary chat messages, secret chat messages are only accessible on the devices that launched the secret chat and the devices that accepted it. They cannot be accessed on other devices. Secret chat messages can be erased at any moment and optionally have them self-destruct [8].A sizeable portion of the cryptographic community has attacked Telegram for the security of their own cryptographic protocol, MTProto[9].
SMS is a requirement for Telegram, Viber, and WhatsApp's signup processes. The Signaling

System 7 (SS7) protocol is used to transmit SMS. The problem is with SS7 [10]. By intercepting SMS communications, attackers used the SS7 protocol to gain access to the victim's account [11]. Because Telegram is cloud-based, the attacker can take full control of the victim's account and prohibit him from accessing it by taking advantage of this vulnerability. To make the account more secure should activate two-factor authentication [12].

Facebook Messenger:

Popular chat app Facebook Messenger is accessible on both iOS and Android devices. It offers both open chat and private talks as two different messaging options. Regular chat only offers secure communication via TLS and does not offer end-to-end encryption. It also retains all communications on its servers. The concept of hidden chat in Telegram is the same as in secret talks [13].

[3] PROPOSED ARCHITECTURE

The proposed architecture is designed to be Client-Server chat application. In client side, when a user sets up the application, the user either selects registration or log-in. In server side, the chat server consists of users' server and a message server. User's server that manages user's credentials. 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 Fig. 1.

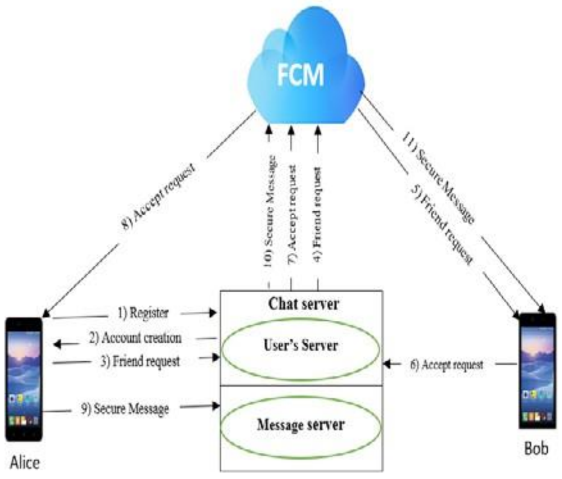

Fig. 1. Generic Architecture of Proposed Chat

Registration an account

Before starting the application, there must have a lock screen to configure the Keystore that provides a secure container to store the local storage key to make moredifficult for extraction it from the device by unauthorized persons or other applications [14].
Each account has only one device and it is distinguishedby device id. In addition, Email and username are unique. Name, email and password are required to register a newaccount. After

typing the registration information, thepassword is encrypted by using XSalsa20 algorithm [15]then the user credentials are sent to the server. Afterverification, the server generates a unique identifier thatacts as the user ID. After that, the acknowledgementmessage is received for successful registration to the clientapplication and the client information is stored in localstorage.

The application generates a set of keys:

(a) Key for encrypting the password.
(b) A public key pair for calculating session key.
(c) Symmetric storage key for encrypting/decrypting localstorage contains contact list, chat history and key store.

Login

Email and password are required for user authentication. After typing the authentication information, the password is encrypted then the user credentials are sent to the server. The server checks if the email and password are valid. After validation, JSON Web Token (JWT)[16] is created and sends to the client to store it. When a client makes a request at the later time, JWT is passed with the request. The server verifies of the JWT, if it is valid, the request is processed (Fig.2).

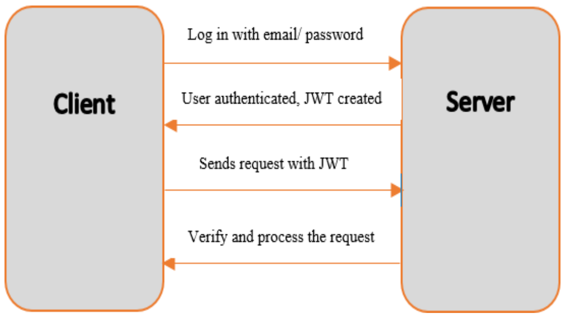
Client
Log in with email/ password
User authenticated, JWT created
Sends request with JWT
Verify and process the request
Server

Fig. 2 Login process

Firebase Cloud Messaging

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 is a free service that allows sending lightweight messages from the server to the devices whenever there is new data available[17]. This saves a lot of user’s battery by avoiding requesting to the server for new messages. It provides TLS for securing channel.

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 MongoDB database as in Fig 3.

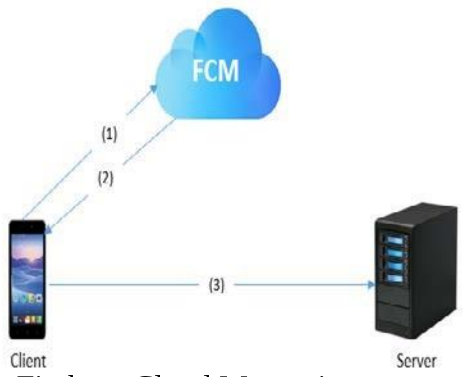

Fig. 3. Firebase Cloud Messaging.

When the server sends a push notification, it sends a request to FCM sending the push message along with the registration token. FCM identifies the target device by using registration token then starts to push data.

Session key Setup

To add users to contact list either by username or by email address.For sending a request to a friend on the assumption that the first user knows the username or email of the second user due to the username and email are a unique for each user and the second user should have already registered in the server. Presumably, the first user is called Alice and the second is called Bob.
When the send request, Bob name is typed by Alice and her public key is fetched from the local storage then the request is sent to the server.
When a request is received, it appears as a notification (Fig. 4). If the friendship request is accepted by Bob, his private key is fetched with Alice's public key to calculate the session key by using Elliptic Curve Diffie-Hellman (ECDH) over the curve Curve25519 [18] and hashes the result with HSalsa20 [15] then the session key is stored in local storage. In the end, the acceptance is sentwith his public key to the server to be delivered to Alice. Upon receipt of the acceptance of the request, the samesteps on the above are taken. The session key is calculatedby using Alice private key and Bob public key then it isstored in the local storage for later use.

The session key is the same for both parties and this is the strength of the Elliptic Curve Diffie-Hellman (ECDH) andthus it is difficult to attack by the man-in-the-middle. In addition to, the weakness of the traditional Diffie-Hellmanhas been eliminated.

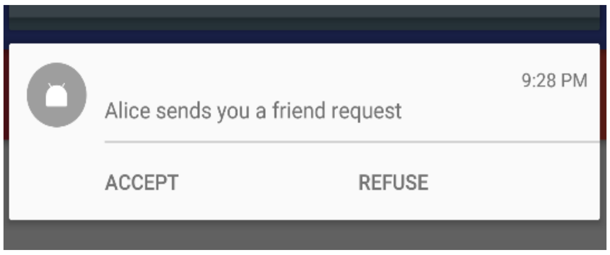
9:28 PM
Alice sends you a friend request
ACCEPT
REFUSE

Fig. 4. Friend request notification.

Exchanging Messages

When a message is typed, the application encrypts the message using XSalsa20 encryption algorithm to encrypt the message body and Poly1305 to compute a Message Authentication Code (MAC) [19]. Each message has its own separate key and nonce which brings better security for each single message in such discovering one of the keys cannot decrypt previous messages. After encrypting the message, it is encrypted again using the recipient's session key then it is sent to the server (Fig. 5).

After the message is received from FCM, the MAC of the encrypted message is calculated and compares it with the received MAC to verify the integrity of the message. If theresults are not the same, it is rejected and does not show tothe user otherwise it is decrypted by the sender session key. Next, the message body is verified in the same steps above. Now the key and nonce to decrypt the message are known. The message is then decrypted and stored in the local storage and displayed to the recipient. If the application is in the background the message will bedisplayed as a notification while if the recipient uses the application it will be displayed in the chat window.

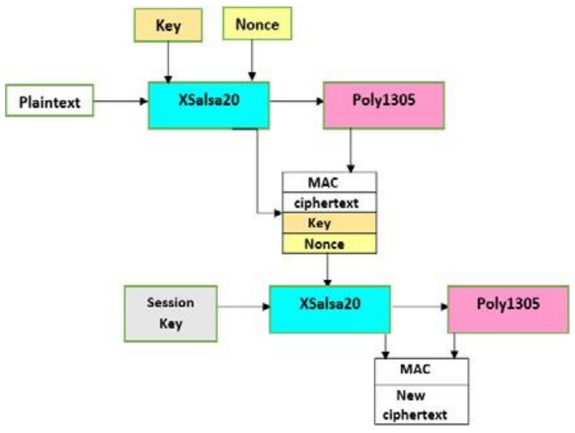

Fig. 5. Procedure to encrypt a message.

Local Storage

The data is stored locally in the application by using Realm database. Realm is a lightweight mobile database that supports cross-platform. It's easy to use and fast. More, it has lots of modern features such as JavaScript Object Notation (JSON) support, a fluent API, data change notifications and encryption support [20]. Encrypted data is protected from unauthorized access and is accessible only if have been a right encryption key. Realm uses AES-256+SHA2 algorithm and 64-byte key for encrypting storage [21]. To prepare Realm storage passes through several steps that are:
Step 1: The application checks whether the lock screen is present or not. If it exists, the following steps are completed.
Step 2: Generate Realm Key that is used for encrypting storage.
Step 3: Generate key from Keystore.
Step 4: Realm key is encrypted with the key generated in step 3 by using AES in CBC mode.
Step 5: Save the encrypted key in shared preferences in private mode so that other applications cannot access this data directory.
Three files are stored in the local storage. UserInfo file that stores all information pertaining to the user. While Friends file stores all information pertaining to the friends. Finally,

Messages file stores all information pertaining to messages.

Server Side Implementation

Server-side has relied on Node JS[21] and MongoDB database. Node JS is fast, capable of handling a large number of simultaneous connections with high throughput, which is equivalent to high scalability. MongoDB and Node JS have often used together because of their using

JSON so no need to spend time for transforming the data between them making it easy to deal with each other. In addition, MongoDB provides TLS that makes a secure connection (Fig. 6). To perform a client request passes through several steps that are:
Step 1: Initially, must run the MongoDB connection then run the Node JS from Command Prompt. At this stage, the server is ready to receive the client's request.
Step 2: When the client sends a request, the server receives the HTTP request in JSON format. The request then parsed.
Step 3: The HTTP request is compared with the base path if it is matched, it is handed to Express framework.
Step 4: The Express receives the HTTP request and routes it to the specific endpoint that matched it. In case of not matched with any of the routes will display error in Command Prompt. Otherwise, it will be forwarded to the controller which handles the required function.
Step 5: Make a request to MongoDB database by mongoose for processing function.
Step 6: When the data is fetched from MongoDB database and the required operations are done, Node JS receives the response then sends to the client.

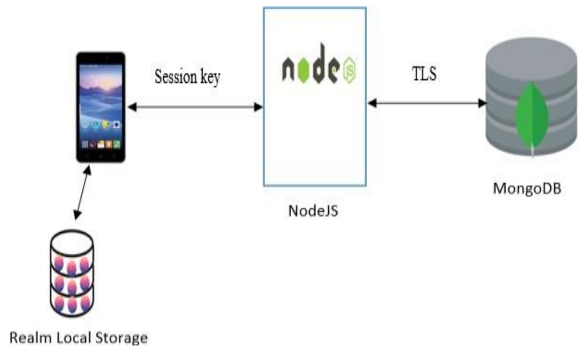

Fig. 6. The Specific Architecture of Proposed Chat.

[4] Analysis of the Proposed Chat

In section 3, we listed a set of requirements for securing chat. To analyze and evaluate proposed chat we have compared proposed chat with popular applications discussed in section 2. The comparison is based on the requirements listed in Table 1

Table 1. Comparison with Popular Chat Applications

Criteria	Whats App	Viber	Telegram	Facebook Messenger	Propose d Chat
Req1	N	N	N	N	Y
Req2	Y	Y	Y	Y	Y
Req3	Y	Y	P	P	Y
Req4	Y	N	N	P	Y
Req5	Y	Y	N	N	Y
Req6	N	N	N	N	Y

Note:”Y” it means that it meets the requirement. ”N“does not support the requirement. ”P”only the secret part supports it.

[5] SUMMARY

In this paper, we introduced a specification for preserving the security and privacy of the chat application. We described a set of requirements for making secure chat and implement it by using modern methods and lightweight for providing speed and good protection to its clients. XSalsa20 algorithm ideal for mobile devices because of its high security, high performance and maintains battery life. Clients can be confident that nobody can read their messages, even if the mobile phone reaches wrong hands cannot enter to the application and cannot access the data stored locally.

REFERENCES

[1] Ash Read, "How Messaging Apps Are Changing SocialMedia," 2016. [Online]. Available:https://blog.bufferapp.com/messaging-apps.

[2] Most popular messaging apps 2017 | Statista," 2017. [Online]. Available: https://www.statista.com/statistics/258749/most-popular- global-mobile-messenger-apps/.

[3] D. Moltchanov, "Client/server and peer-to-peer models: basic concepts," 2013.

[4] Martin Kleppmann, "The Investigatory Powers Bill would increase cybercrime — Martin Kleppmann's blog," 2015. [Online]. Available: https://martin.kleppmann.com/2015/11/10/investigatory- powers-bill.html.

[5] D. P. Roel Hartman, Christian Rokitta, Oracle Application Express for Mobile Web Applications - Roel Hartman, Christian Rokitta, David Peake - Google Books. 2013.

[6] Viber Encryption Overview." [Online]. Available: https://www.viber.com/security-overview/.

[7] WhatsApp inc, "WhatsApp security whitepaper," p. 10, 2017.

[8] "Telegram F.A.Q." [Online]. Available:https://telegram.org/faq.

[9] anka, "Security Analysis of the Telegram IM," p. 70, 2016.

[10] B. O. B. Kamwendo, "Vulnerabilities of signaling system number 7 (ss7) to cyber attacks and how to mitigate against these vulnerabilities. bob kamwendo," vol. 7, no. 7, 2015.

[11] John Leyden, "SS7 spookery on the cheap allows hackers to impersonate mobile chat subscribers • The Register," 2016. [Online]. Available: https://www.theregister.co.uk/2016/05/10/ss7_mobile_chat_ hack/.

[12] "Active Sessions and Two-Step Verification." [Online]. Available: https://telegram.org/blog/sessions-and-2-step- verification.

[13] T. Whitepaper, "Messenger Secret Conversations," 2016.

[14] "Android Keystore System | Android Developers." [Online]. Available: https://developer.android.com/training/articles/keystore.html.

[15] D. J. Bernstein, "Extending the Salsa20 nonce," no. Mc 152, pp. 1–14, 2011.

[16] M. B. Jones, "The Emerging JSON-Based Identity Protocol Suite," 2011.

[17] “Firebase Cloud Messaging | Firebase.” [Online]. Available: https://firebase.google.com/docs/cloud-messaging/.

[18] D. J. Bernstein, “Curve25519 : new Diffie-Hellman speed records,” vol. 25519, 2006.

[19] D. J. Bernstein., “Poly1305.” [Online]. Available: https://en.wikipedia.org/wiki/Poly1305.

[20] “Realm: Create reactive mobile apps in a fraction of the time.” [Online]. Available: https://realm.io/.

[21] “Realm Swift 2.10.2.” [Online].Available: https://realm.io/docs/swift/latest/